Civil Engineering Study
Structural Series 89-31

INRESB-3D-SUP

USER'S MANUAL

A COMPUTER PROGRAM FOR INELASTIC ANALYSIS OF 3-DIMENSIONAL REINFORCED-CONCRETE AND STEEL SEISMIC BUILDINGS

by

Franklin Y. Cheng
Curators' Professor

Gregory E. Mertz
Graduate Assistant

Department of Civil Engineering
University of Missouri-Rolla
Rolla, MO 65401-0249

Report Series
Prepared for the National Science Foundation under Grant
NSF ECE 8513852

Civil Engineering Study
Structural Series 89-31

INRESB-3D-SUP

USER's MANUAL

A COMPUTER PROGRAM FOR INELASTIC ANALYSIS OF 3-DIMENSIONAL REINFORCED-CONCRETE AND STEEL SEISMIC BUILDINGS

by Franklin Y. Cheng
Curators' Professor

Gregory E. Mertz
Graduate Assistant

Department of Civil Engineering
University of Missouri-Rolla
Rolla, MO 65401-0249

Report Series
Prepared for the National Science Foundation under Grant
NSF ECE 8513852

ABSTRACT

This report has been prepared as a user's guide for the computer program, INRESB-3D-SUP, for analyzing elastic and inelastic building systems subjected to static loadings, multi-component earthquake motions, and psudo-static cyclic loadings. Additionally, the program is capable of calculating the elastic natural frequency and buckling load.

A joint based system is used to define the geometry of the structure. The structural members may be elastic 3D prismatic beams, nonlinear reinforced concrete shear walls, and nonlinear springs. For the shear wall elements, the hysteresis models are for bending, shear and axial force-deformation relationships for low-rise walls. The Takeda model may be used for beams, columns and slender walls. The bilinear hysteresis model is also included. The geometric matrix for large deformations is available for both the elastic 3D prismatic beam and reinforced concrete shear wall elements.

The system formulation has the following attributes: 1) joint based degrees of freedom, 2) rigid body and planar constraints, 3) incremental nonlinear static solution, 4) unbalanced load correction for overshooting, 5) incremental nonlinear dynamic solution, 6) mass and stiffness proportional damping, 7) condensation to reduce the size of a dynamic problem, 8) energy balance, 9) damage index, and 10) ductility and excursion ratio for various definitions of displacement, constant strain energy, and variable strain energy. The computer program has been developed for achieving efficiency in both the computation and data preparation. The output solutions include the static results of member forces and joint displacements as well as the dynamic results of member forces, joint displacements, ductility factors, excursion ratios,

damage indicies, seismic input energies, dissipated energies, and the option of saving data for plotting.

The main features of the report include the program listing, a description of the program, instructions for data preparation, and a guide to modify the programs dynamic storage.

ACKNOWLEDGMENTS

This is one of a series of reports on response analysis and optimum design of building systems with multicomponent seismic input and code provisions studied at UMR. This report is based on a joint research project undertaken by a team at NCKU (National Cheng-Kung University, Taiwan) and UMR. NCKU performed experimental work and UMR carried out analytical studies. UMR work was supported by the National Science Foundation under the grant NSF ECE 8513852. The authors gratefully acknowledge the NSF support and the experimental data provided by NCKU team, particularly Dr. S.C. Liu of NSF, Drs. M.S. Sheu, W.M. Liao, and Y.T. Kuo of NCKU. Dr. Sheu is the principal investigator of the NCKU team.

The authors also received some experimental data from the Los Alamos National Laboratory and deeply appreciated the cooperation of Drs. C.A. Anderson, J.G. Bennett, R.C. Dove, E.G. Endebrock, and C.R. Farrar for their cooperation.

TABLE OF CONTENTS

Page

ABSTRACT ii

ACKNOWLEDGMENTS iv

LIST OF ILLUSTRATIONS ix

LIST OF TABLES xi

I. INTRODUCTION 1

II. DESCRIPTION OF INRESB-3D-SUP COMPUTER PROGRAM 2

A. STRUCT - DEFINITION OF THE STRUCTURAL MODEL . 2

1. Material Library 4

a. Elastic 3D Prismatic Beam Material 4

b. R/C Axial Hysteresis Model 4

c. Cheng-Mertz B1 Bending Hysteresis Model 4

d. Cheng-Mertz S1 Shear Hysteresis Model 8

e. Takeda Hysteresis Model 8

f. Bilinear Hysteresis Model 8

2. Element Library 8

a. Elastic 3D Prismatic Beam Element 8

b. Spring Element 13

c. Reinforced Concrete Shear Wall Element 18

B. SOL01 - ELASTIC STATIC ANALYSIS WITH MULTIPLE LOAD CASES 20

C. SOL02 - ELASTIC / NONLINEAR SEISMIC TIME HISTORY RESPONSE 20

D. SOL03 - ELASTIC NATURAL FREQUENCY / ELASTIC BUCKLING LOAD 24

E. SOL04 - NONLINEAR STATIC CYCLIC RESPONSE 26

III. DESCRIPTION OF PROGRAM 30
A. MAIN PROGRAM AND COMMON BLOCKS 30
B. DESCRIPTION OF ROUTINES 33
C. DESCRIPTION OF MISCELLANEOUS ROUTINES 41
D. DYNAMIC MEMORY MANAGEMENT 42
1. Dynamic Memory Allocation 42
2. Compact Matrix Storage 43
3. Capacity of the Dynamic Memory 45
E. ADDITION OF MATERIALS, ELEMENTS, AND SOLUTIONS 47
1. Addition of a Material to the Material Library 47
2. Addition of an Element to the Element Library 48
3. Addition of a Solution 49
IV. INPUT DATA FOR PROGRAM INRESB-3D-SUP 50
A. STRUCTURE - DEFINE THE STRUCTURAL MODEL 52
1. Joints and Degrees of Freedom 53
a. Joint Coordinates 53
b. Joint Direction Cosines 55
c. Joint Restraints 55
d. Joint Condensation 56
e. Joint Constraints 57
2. Materials and Hysteresis Model Information 58
a. TYPE='3D-BEAM' 59
b. TYPE='AXLMOD' 59
c. TYPE='BEND1' 60
d. TYPE='SHEAR1' 60
e. TYPE='TAKEDA' 61
f. TYPE='BILINEAR' 62

3. Geometric Stiffness Data 62
4. Element Data 63
a. TYPE='3D-BEAM' 64
b. TYPE='SPRING' 65
c. TYPE='SHEAR WALL' 67
5. Mass 68
6. Damping 69
B. SOL01 - ELASTIC STATIC SOLUTION 70
1. Joint Loads 71
2. Element Loads 72
C. SOL02 - DYNAMIC SOLUTION BY NUMERICAL INTEGRATION 74
1. Special Loading Mass 76
2. Output Data to Disk Files 76
3. Initial Displacements, Velocities, Accelerations and Loads . 78
4. Ground Acceleration Record 80
D. SOL03 - EIGENVALUE SOLUTION 83
E. SOL04 - INCREMENTAL STATIC SOLUTION 84
1. Output Data to Disk Files 84
2. Joint Loads 85
3. Element Loads 85
4. Load Factors 85
F. BUG - SET BUG OPTIONS 86
G. READ - READ OUTPUT FILES 87
H. NOECHO - INHIBIT INPUT ECHO 88
I. DUMP - PRINT MEMORY 88
J. RELEASE - RELEASE MEMORY 88
K. STOP - TERMINATE EXECUTION 89

V. RUNNING PROGRAM INRESB-3D-SUP 91
A. GENERATING THE COMMON BLOCK MACRO LIBRARY 91
B. COMPILING THE PROGRAM 91
C. RUNNING THE PROGRAM 92
VI. EXAMPLE PROBLEMS 94
A. CANTILEVER BEAM 94
1. Description of Input Information 94
2. Input Data 95
3. Output 96
B. ONE STORY R/C BOX TYPE BUILDING 99
1. Description of Input Information 99
2. Input Data 101
3. Output 103
C. TWO STORY UNSYMMETRIC BUILDING WITH ISOLATED SHEAR WALLS 107
1. Description of Input Information 107
2. Input Data 110
3. Output 112
BIBLIOGRAPHY 120
APPENDICES
A. Ductility and Excursion Ratio 122
B. Damage Index 125
C. Listing of Program INRESB-3D-SUP 127

LIST OF ILLUSTRATIONS

Figure		Page
1	Block STRUCT - Define the Structural Model	3
2	Axial Hysteresis Model Before Tensile Yield	5
3	Axial Hysteresis Model After Tensile Yield	5
4	Low-Rise Shear Wall Cheng-Mertz Bending Hysteresis Model	6
5	Multiple Segment Backbone Curve	7
6	Low-Rise Shear Wall Shear Cheng-Mertz Hysteresis Model	9
7	Takeda Hysteresis Model	10
8	Bilinear Hysteresis Model	11
9	3D Prismatic Beam Element	12
10	Spring Element	14
11	Diagonal Bracing Member Modeled With an Axial Spring	15
12	Two DOF Model With Shear Springs	16
13	Member End Rotation of an Elastic 3D Prismatic Beam Modeled with a Rotational Spring	17
14	Shear Wall Element	19
15	Block SOL01 - Static Analysis	21
16	Block SOL02 - Elastic / Nonlinear Dynamic Analysis	22
17	Block SOL03 - Natural Frequency / Elastic Buckling Load	25
18	Block SOL04 - Nonlinear Static Cyclic Response	27
19	Flow Chart for Program INRESB-3D-SUP: Part A	31
20	Flow Chart for Program INRESB-3D-SUP: Part B	32
21	Dynamic Memory Example	44
22	Cantilever Beam	95
23	One Story R/C Box Type Building	100

24 Two-story Unsymmetric building: Dimensions, Joints and Walls 108
25 Two-story Unsymmetric building: Degrees of Freedom 109
26 Displacement Definition of Ductility 123
27 Variable Strain Energy Definition of Ductility 123
28 Constant Strain Energy Definition of Ductility 123

LIST OF TABLES

Table		Page
I	DEGREES OF FREEDOM FOR UT BOX B6	102

I. INTRODUCTION

This publication is to serve as the user's manual of the computer program, INRESB-3D-SUP (INelastic Analysis of REinforced Concrete and Steel Building Systems for 3-Dimensional Ground Motions). The numerical procedures of the program are described in Chapter II. In Chapter III, the description of the subroutines, the programs capacity and the addition of materials, elements and solutions are discussed. Chapter IV contains the detailed input instructions. Instructions for running the program under the CMS environment for an IBM computer are included in Chapter V. In Chapter VI, three examples are used to illustrate the preparation of the input data and the output of solutions. The ductility and excursion ratios are derived in Appendix A. The damage index is discussed in Appendix B. The program listing is given in Appendix C.

II. DESCRIPTION OF INRESB-3D-SUP COMPUTER PROGRAM

The computer program INRESB-3D-SUP is capable of analyzing elastic and nonlinear 3D structures subject to static and seismic loadings. The nonlinear behavior of R/C shear walls is represented by a shear wall element. A nonlinear spring element is also included that may be used to represent steel and reinforced concrete members.

INRESB-3D-SUP is a modular computer program consisting of five primary blocks. The first block (STRUCT) defines the structural model. The remaining four blocks (SOL01, SOL02, SOL03 and SOL04) are independent solutions for static loading, seismic loading, natural frequency or buckling load, and static cyclic loading, respectively. The numerical procedures for each of the five program blocks are presented in this chapter.

A. STRUCT - DEFINITION OF THE STRUCTURAL MODEL

The structural model consists of an assemblage of elements. The point where two or more elements connect is a called a joint. A structure is modelled by first defining the location and orientation of each joint; then materials that describe the behavior of the elements, the elements that connect the joints, and the orientations of the elements are then defined. For dynamic analysis, the lumped mass at each joint is also defined. All of these definitions are in the program block STRUCT. The flow chart for STRUCT is shown in Figure 1.

Step 1. --- Define Joints and Determine the DOF's. The coordinates of the joints and their orientation are defined by the user. The coordinates are defined in the global coordinate system (GCS). The orientation of each joint defines its joint coordinate system (JCS). Each joint initially has six degrees of freedom (dof) in the JCS. The user also defines the joint's dof that are restrained, constrained and condensed out. The program generates the structural degrees of freedom (17).

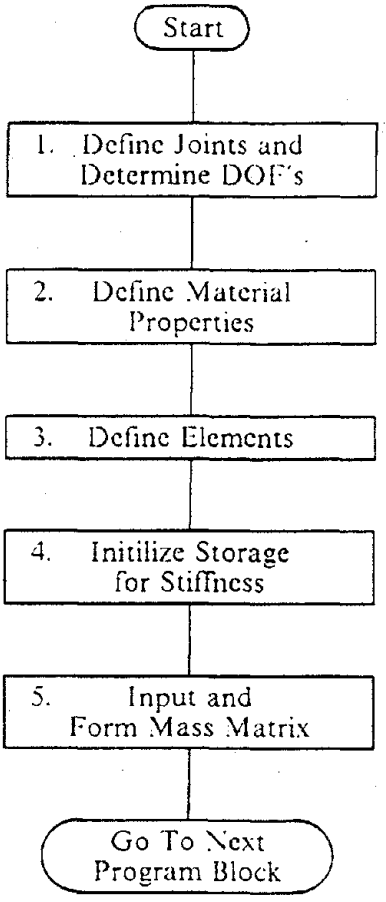


Figure 1. Block STRUCT - Define the Structural Model

Step 2. --- Define Material Properties. The material properties are input and initialized. There are six different material behaviors available, which constitute the material library, and are discussed later.

Step 3. --- Define Elements. The element data is input. The transformation matrices, initial element structural stiffness and the initial element geometric stiffness are calculated. There are three different elements available in the program, which constitute the element library, and are discussed later.

Step 4. --- Initialize Storage for Stiffness. The storage for the structural stiffness and geometric matrices is initialized.

Step 5. --- Input and Store Mass. The lumped mass at each joint is input. The structures mass matrix is stored.

1. Material Library.

a. Elastic 3D Prismatic Beam Material. This material consists of the elastic section properties of a 3D prismatic beam, A_X, J, I_y, I_Z, and the material's moduli E and G.

b. R/C Axial Hysteresis Model. An axial hysteresis model developed for the reinforced concrete boundary columns of a shear wall (16). This hysteresis model is sketched in Figures 2 and 3 and discussed in Reference 17.

c. Cheng-Mertz B1 Bending Hysteresis Model. A bending hysteresis model developed to model the bending deformations in low-rise reinforced concrete shear walls (17). This hysteresis model is sketched in Figure 4 and uses the multiple segment backbone curve shown in Figure 5.

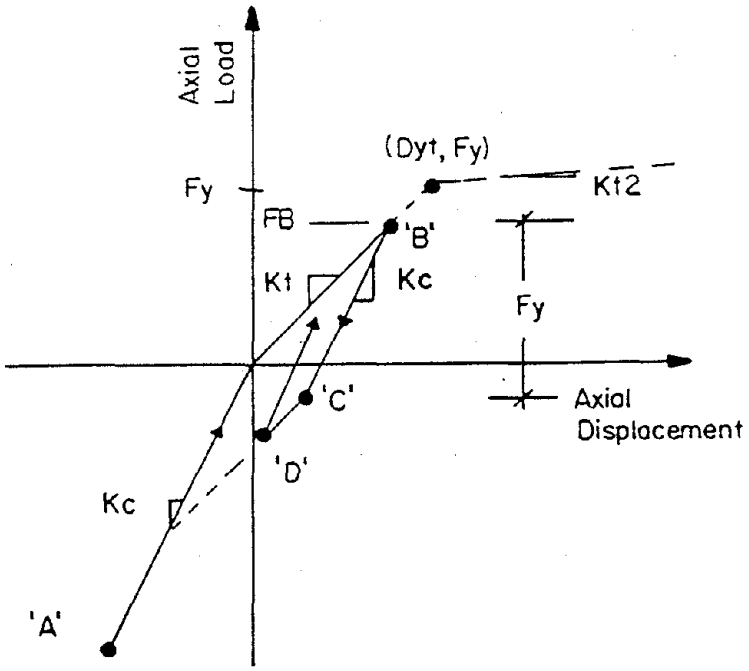
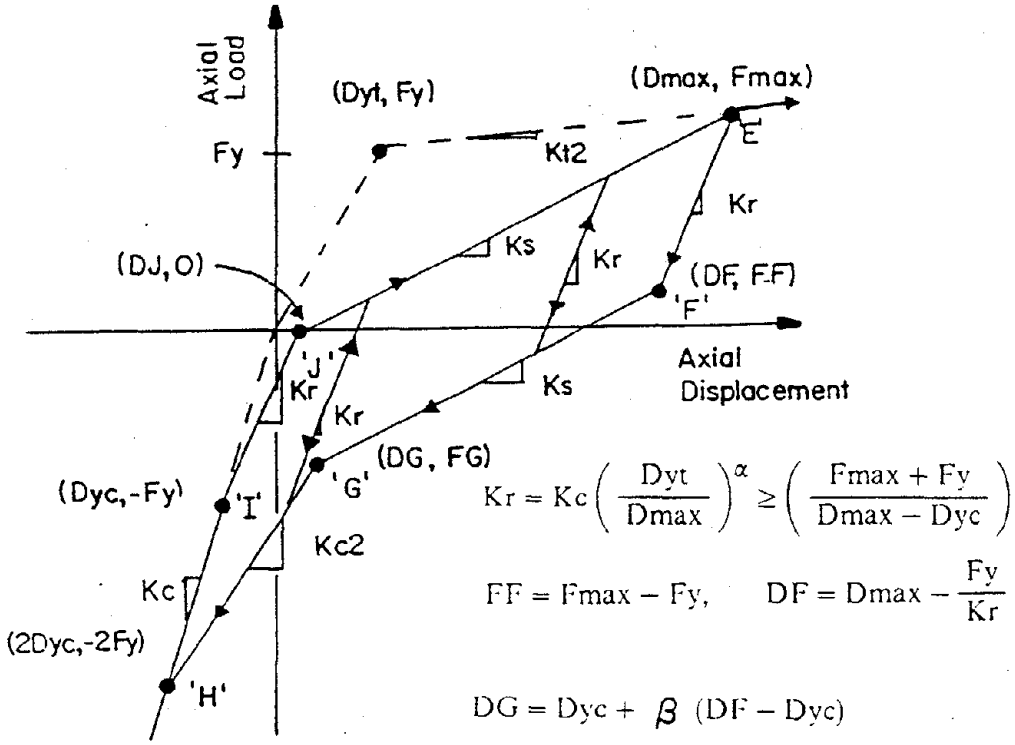


Figure 2. Axial Hysteresis Model Before Tensile Yield



$$Kr = Kc\left(\frac{Dyt}{Dmax}\right)^{\alpha} \geq \left(\frac{Fmax + Fy}{Dmax - Dyc}\right)$$

$$FF = Fmax - Fy, \qquad DF = Dmax - \frac{Fy}{Kr}$$

$$DG = Dyc + \beta\ (DF - Dyc)$$

Figure 3. Axial Hysteresis Model After Tensile Yield

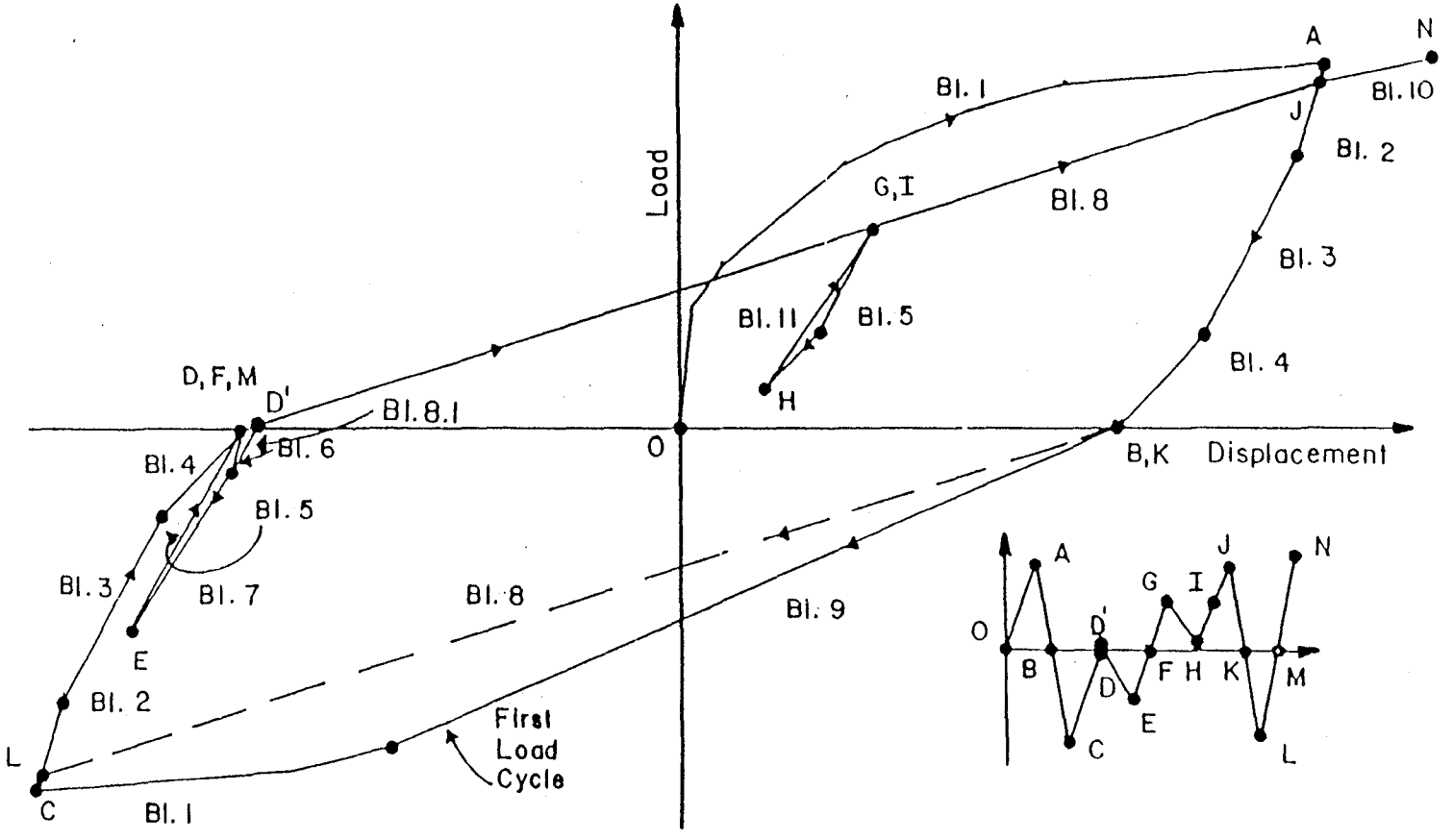


Figure 4. Low-Rise Shear Wall Cheng-Mertz Bending Hysteresis Model

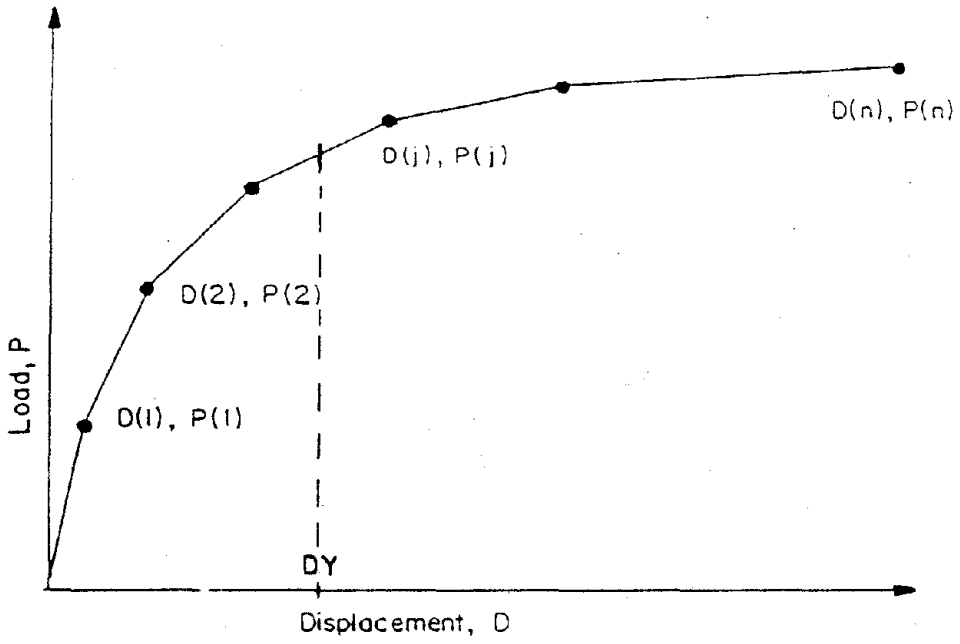


Figure 5. Multiple Segment Backbone Curve

d. Cheng-Mertz S1 Shear Hysteresis Model. A shear hysteresis model developed to model the shear deformations in low-rise reinforced concrete shear walls (17). This hysteresis model is sketched in Figure 6 and uses the multiple segment backbone curve shown in Figure 5.

e. Takeda Hysteresis Model. A bending hysteresis model developed to model the bending deformations of reinforced concrete members (21). This hysteresis model is sketched in Figure 7.

f. Bilinear Hysteresis Model. A hysteresis model that has a bilinear backbone curve and an elastic unloading and reloading curve. The model may also represent the elasto-plastic model by setting the post yielding stiffness to zero. This hysteresis model is sketched in Figure 8.

2. Element Library.

a. Elastic 3D Prismatic Beam Element. The elastic 3D prismatic beam element is shown in Figure 9. This beam element connects a start and end joint. At the start end of the element, a rigid body of length XS is used to model the structural joint. A similar rigid body, of length XE is used at the end joint. The beam's element coordinate system (ECS) X_e axis goes from end 'A' towards end 'B'. The orientation of the ECS Y_e axis is defined by a vector, $\vec{V}$, which lies on the ECS XY-plane. The ECS Z_e axis is perpendicular to the X_e and Y_e axes, right hand rule. There are six internal forces F_X, F_Y, F_Z, M_X, M_Y and F_Z at end 'A' in the ECS. Similarly, six internal forces also exist at end 'B'. All of the internal forces are positive in the direction of the ECS.

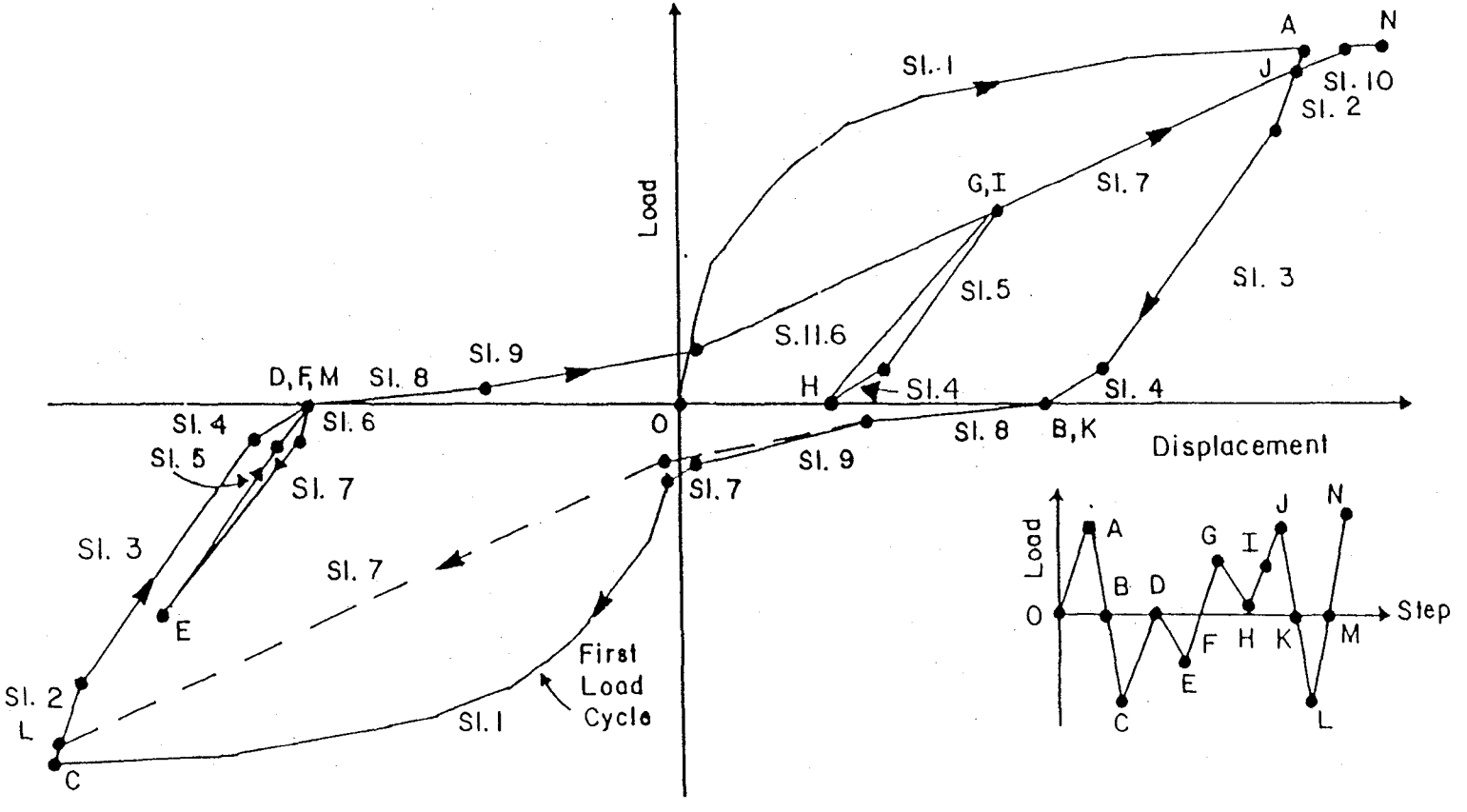


Figure 6. Low-Rise Shear Wall Shear Cheng-Mertz Hysteresis Model

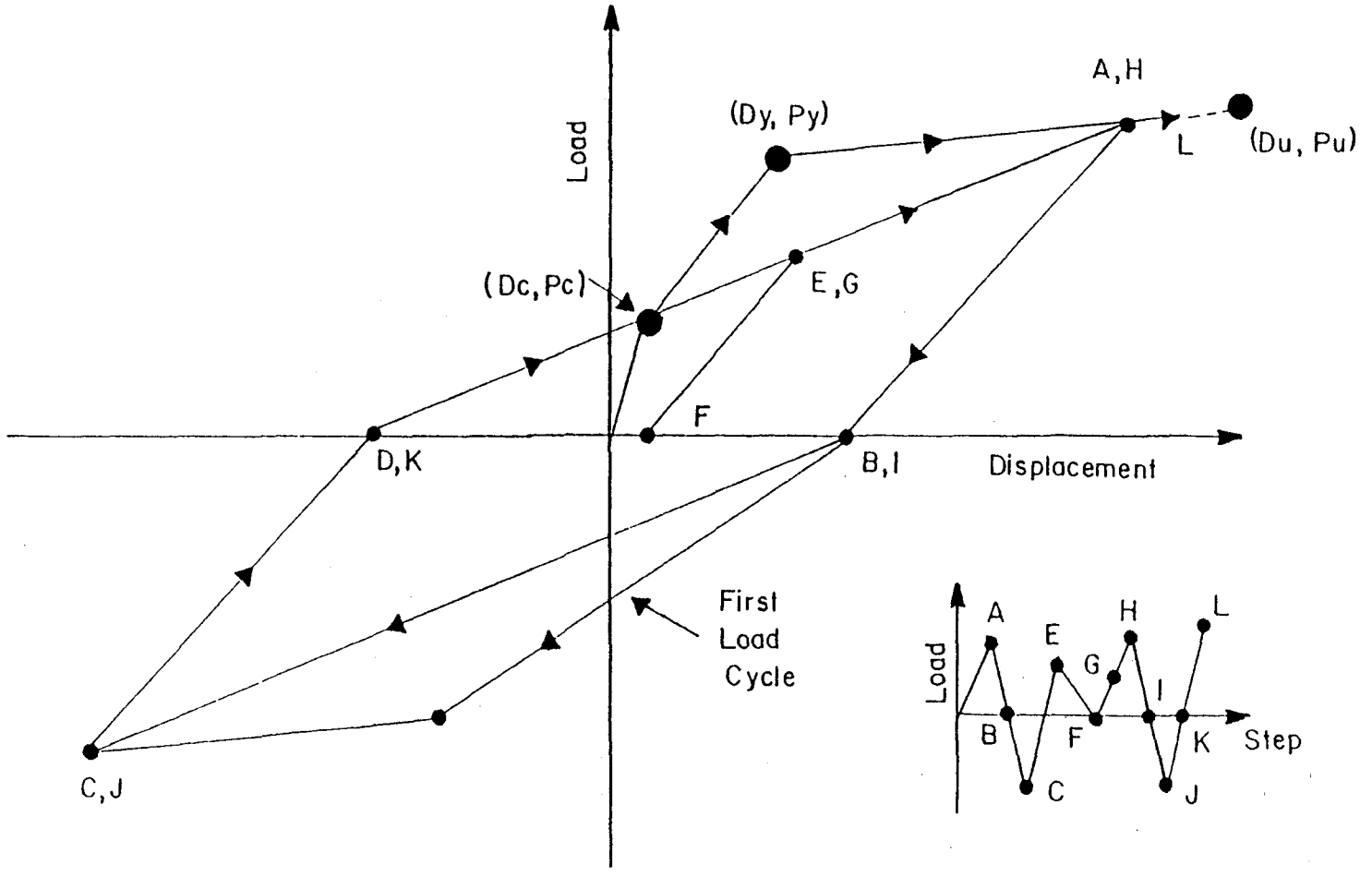


Figure 7. Takeda Hysteresis Model

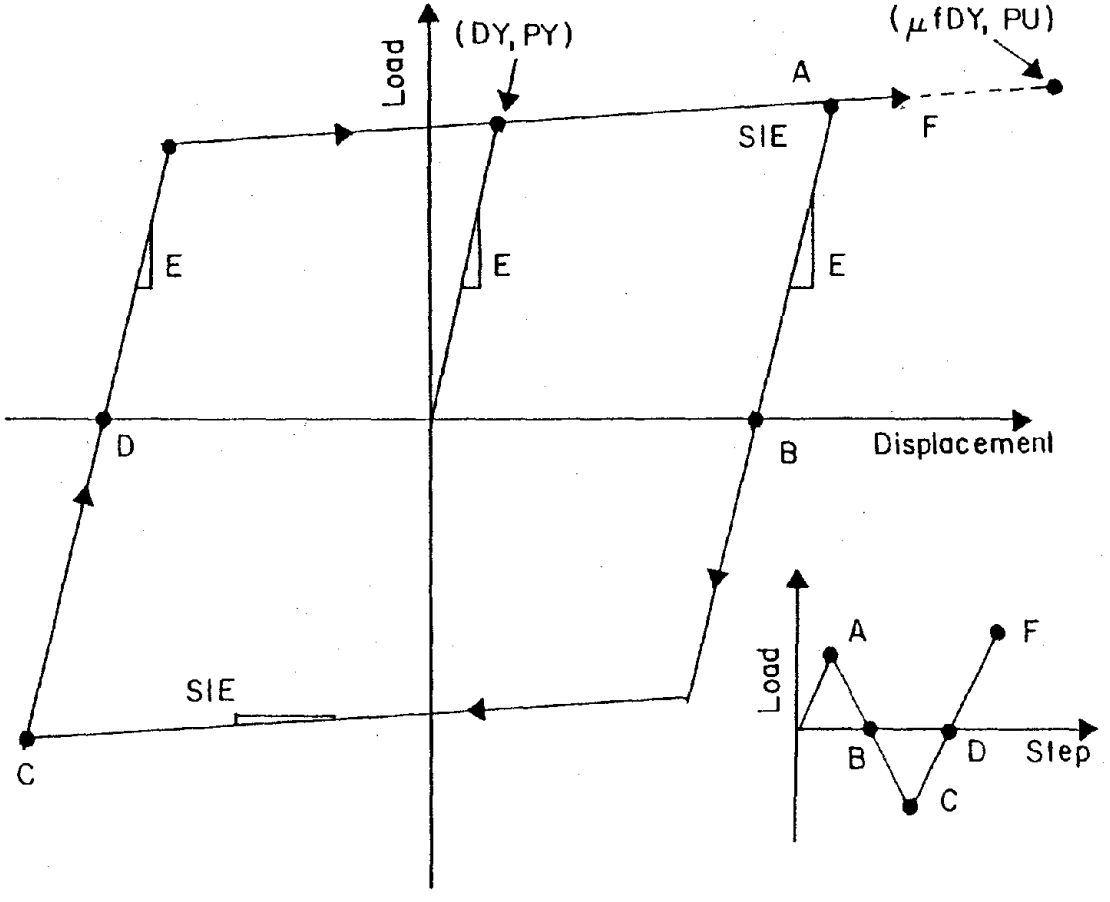


Figure 8. Bilinear Hysteresis Model

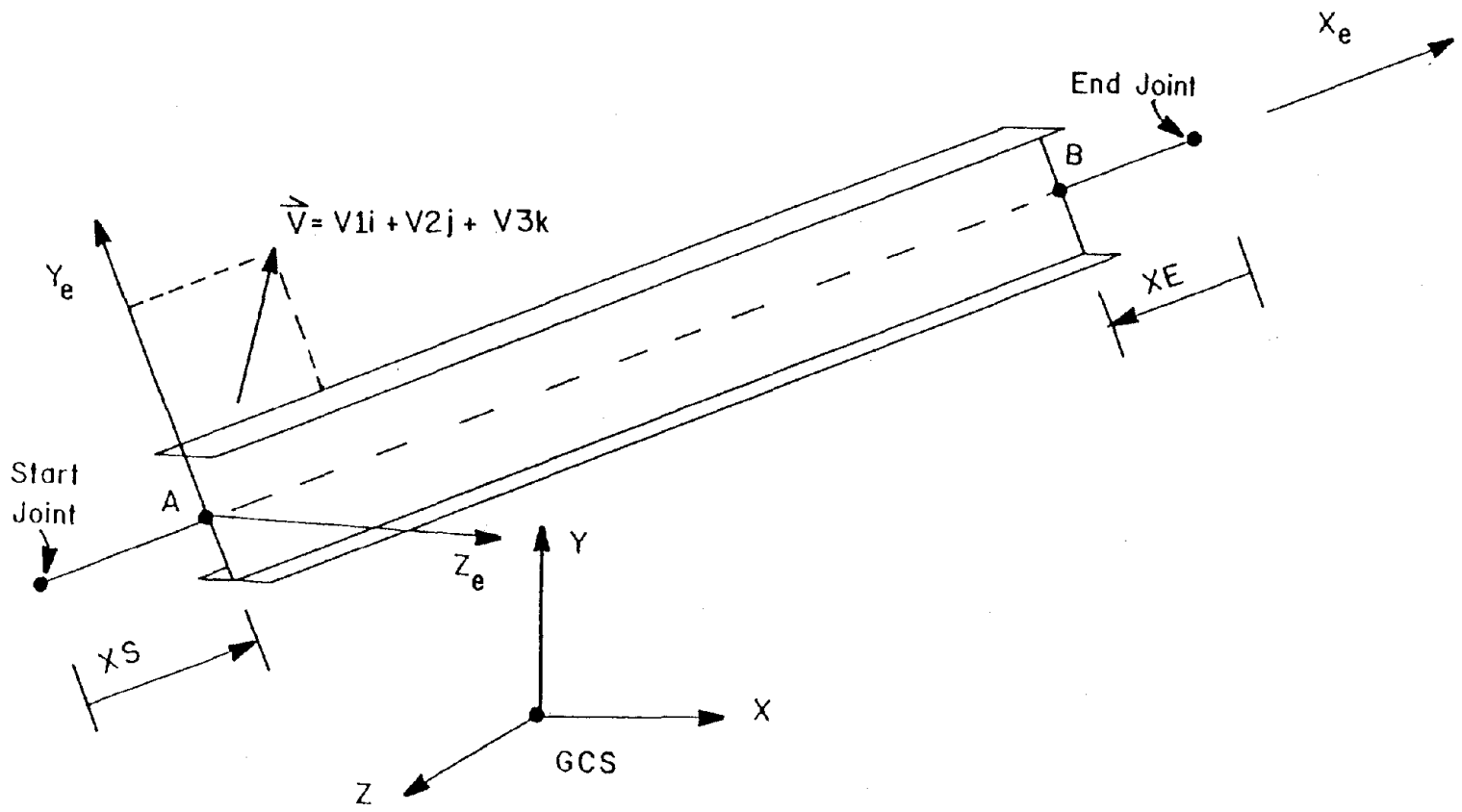


Figure 9. 3D Prismatic Beam Element

The beam element considers axial deformation, torsional deformation, and bending deformations about the Y_e and Z_e axes. Warping torsion and shear deformation are not considered. Two formulations of the geometric stiffness are also available. The 'lumped parameter' formulation only considers the second order shears at the end of the beam, while the 'consistent parameter' formulation considers the second order moments and shears at the end of the beam. The 3D-beam material is used with the beam element.

b. Spring Element. The spring element consist of an isolated spring that connects the start and end joints. At the start end of the spring, a rigid body of length XS is used to model the joint depth. A similar rigid body, of length XE is used at the end joint. The spring element coordinate system (ECS) X_e axis goes from end 'A' towards end 'B'. The orientation of the ECS Y_e axis is defined by the user. The ECS Z_e axis is perpendicular to the X_e and Y_e axes, right hand rule. When the distance between the start and end joints is zero, the orientation of the ECS is identical to the start joint's JCS. The distance between the start and end joints less the length of the rigid bodies, is the length of the spring element. Optionally, the user may define the length of the spring element. The spring element may behave elastically or nonlinearly depending on the material properties used and the magnitude of forces acting on the spring. Second order P-Δ forces are not calculated for the spring element.

The spring may be orientated in one of six positions as shown in Figure 10. Examples of axial, shear and rotational spring applications are shown in Figures 11, 12 and 13, respectively. The axial spring is parallel to the elements X_e axis. The rigid bodies at the ends of the spring reduce the length of the axial spring. The springs axial force, F_X, at end 'A' is positive in the X_e direction.

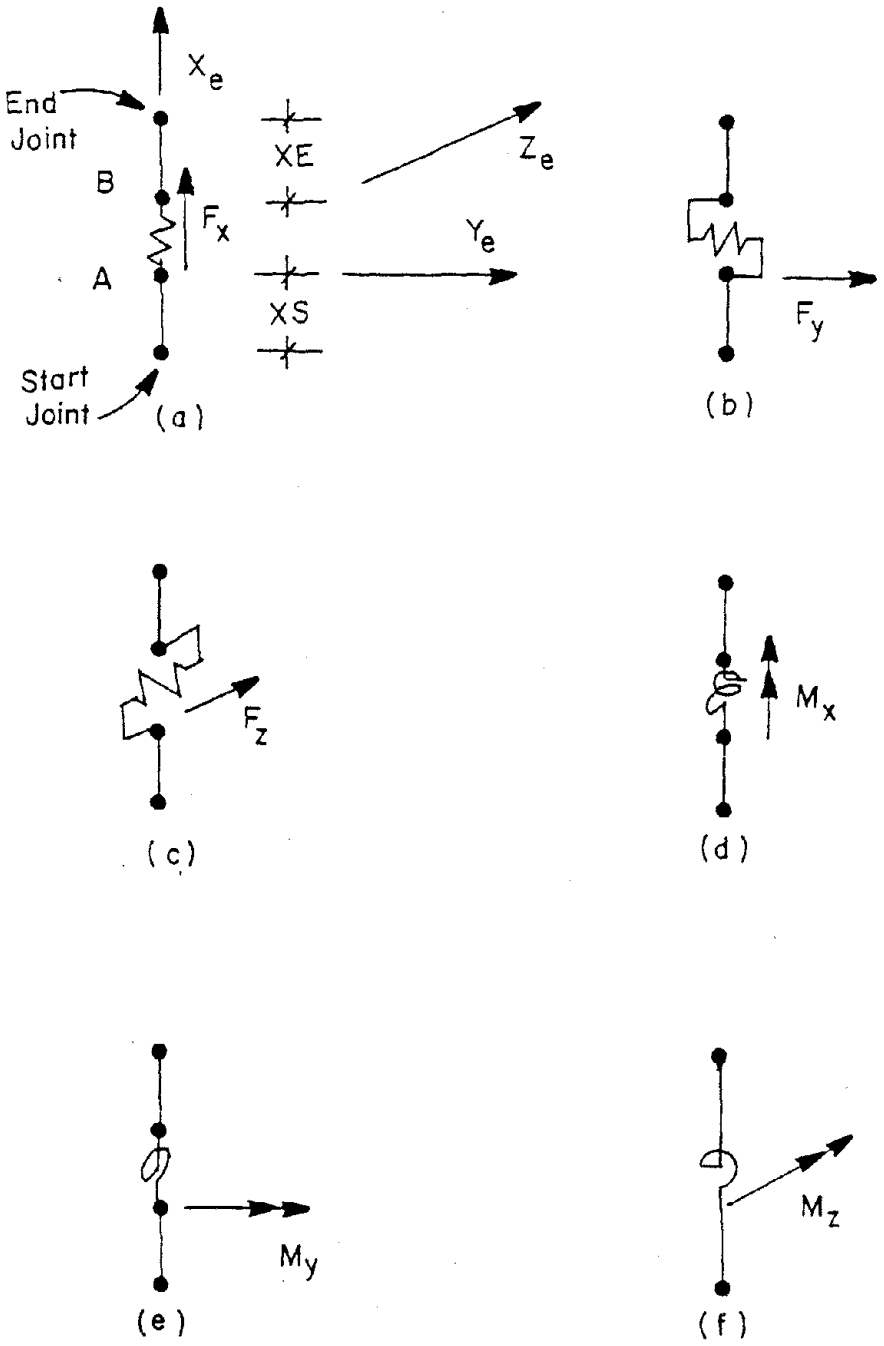


Figure 10. Spring Element: (a) Axial Spring and ECS, (b) Y-axis Shear Spring, (c) Z-axis Shear Spring, (d) Torsional Spring, (e) Y-axis Rotational Spring, (f) Z-axis Rotational Spring

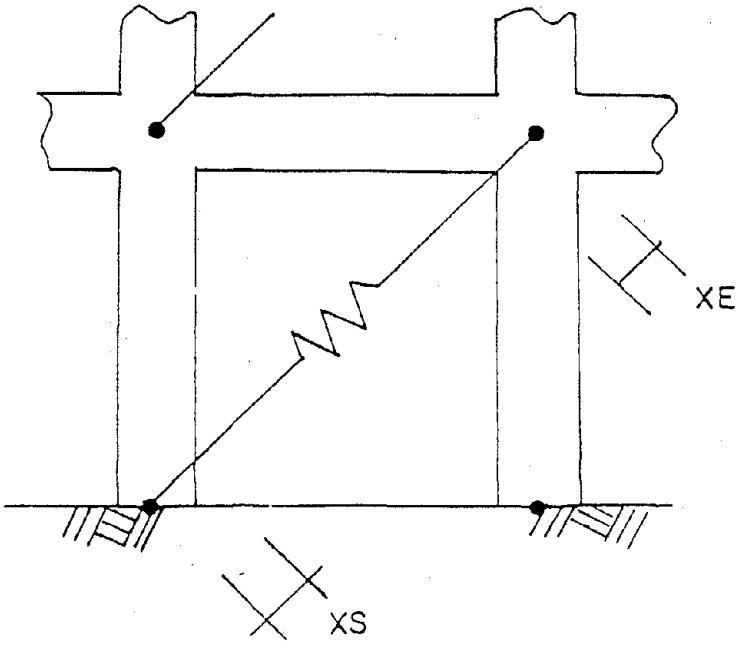


Figure 11. Diagonal Bracing Member Modeled With an Axial Spring

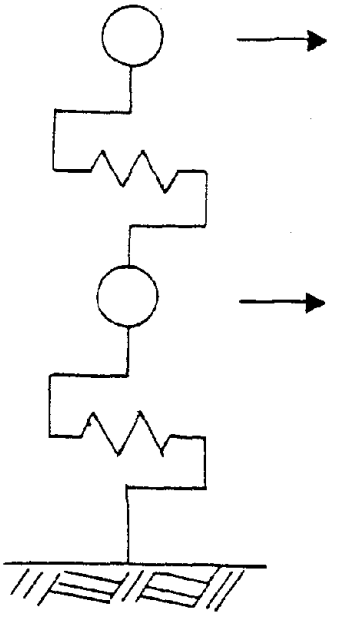


Figure 12. Two DOF Model With Shear Springs

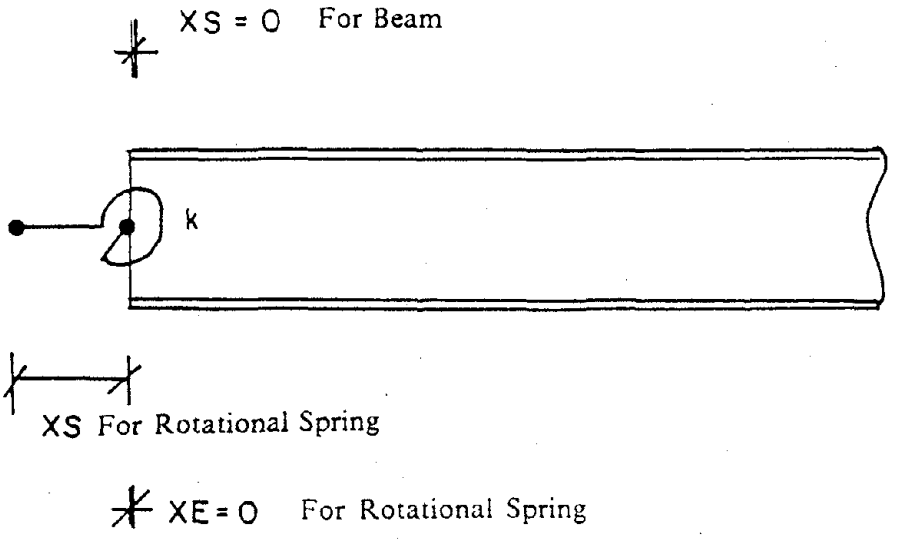


Figure 13. Member End Rotation of an Elastic 3D Prismatic Beam Modeled with a Rotational Spring

The Y-axis shear spring and Z-axis shear spring are orientated parallel to the elements Y_e and Z_e axes, respectively. The rigid bodies at the ends of the spring reduce the length of the shear spring and induce moments at the joints. The springs internal shears, F_Y and F_Z, at end 'A' are positive in the Y_e and Z_e directions.

The torsional spring is parallel to the elements X_e axis. The rigid bodies reduce the length of the torsional spring. The spring's internal torsion, M_X, at end 'A' is positive in the X_e direction.

The Y-axis rotational spring and the Z-axis rotational spring are rotational springs about the Y_e and Z_e axes, respectively. The rigid bodies at the ends of the spring reduce the length of the rotational spring. The spring's internal moments, M_Y and M_Z at end 'A' are positive in the Y_e and Z_e directions.

c. Reinforced Concrete Shear Wall Element. The reinforced concrete shear wall element consists of a panel linking four joints as shown in Figure 14. Bending and shear deformations in the plane of the wall are considered, along with axial deformation. The bending, shear and axial deformations are lumped into three springs. A rigid body, of length α, connects the joints at the top of the wall with the springs, while a second rigid body, of length β, connects the joints at the bottom of the wall with the springs. Bending and shear stiffness perpendicular to the plane of the wall are neglected. A 'lumped parameter' formulation of the geometric stiffness considers both in-plane and out of plane P-Δ effects.

The wall has 10 dof at the corner joints as shown in Figure 14. Fixing the bottom of the shear wall and applying a positive load to degree of freedom 1, the sign convention yields a positive moment and shear at end 'A' of the springs. Fixing the bottom of the shear wall and applying a positive load to degrees of freedom 2 and 4, yield a positive axial load at end 'A' of the spring.

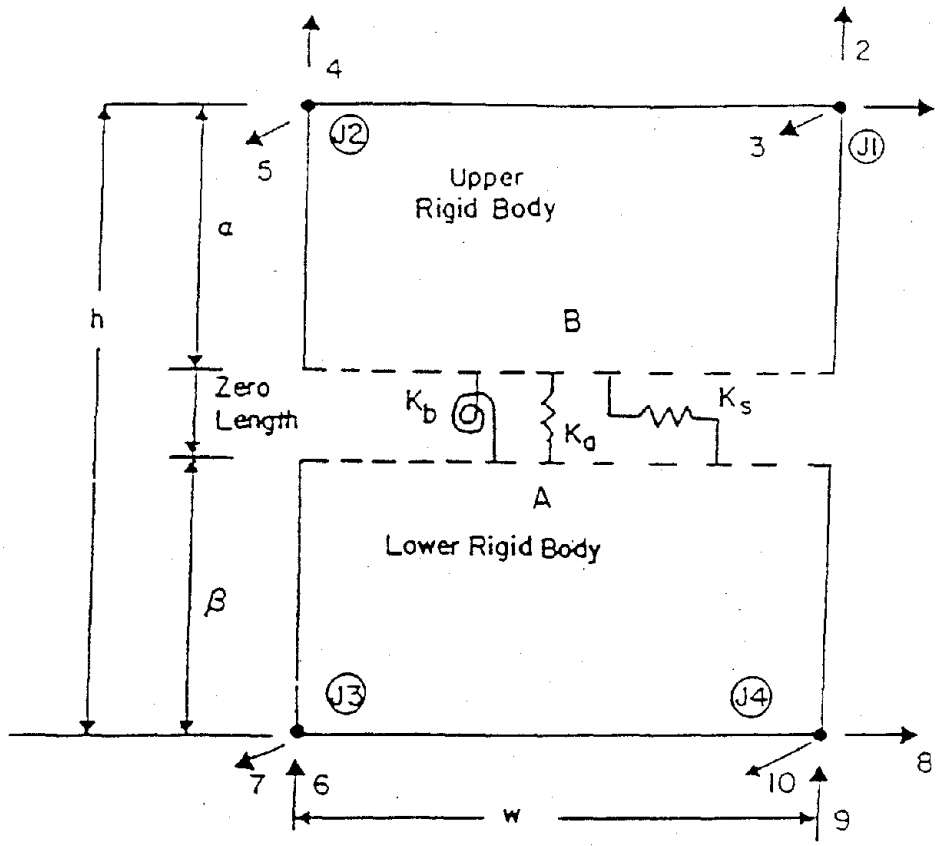


Figure 14. Shear Wall Element

Different materials are used to describe the stiffness of the bending, shear and axial springs. Typically the bending stiffness is defined by the B1 bending hysteresis model, the shear stiffness is defined by the S1 shear hysteresis model and the axial stiffness is defined by the axial hysteresis model.

B. SOL01 - ELASTIC STATIC ANALYSIS WITH MULTIPLE LOAD CASES

This block performs the elastic static analysis with multiple load cases. The flow chart for SOL01 is shown in Figure 15.

Step 1. --- Input Joint and Element Loadings. The joint loads and imposed displacements are input for each load case. Uniform and concentrated element loadings are input for each load case on the 3D-beam element.

Step 2. --- Form the Structural Stiffness and Load Matrices. The structural stiffness matrix is formed. Optionally, the geometric stiffness matrix, based on the user's input axial loads, is subtracted from the structural stiffness matrix. Joint loadings are determined for the imposed displacements (support settlements) and combined with the input joint loadings and element loadings for each load case.

Step 3. --- Calculate Displacements. The displacements for each load case are calculated by Gauss elimination.

Step 4. --- Calculate Reactions. The reactions at restrained degrees of freedom and the summation of reactions are calculated for each load case.

Step 5. --- Calculate Element Forces. The element forces are calculated for each load case.

C. SOL02 - ELASTIC / NONLINEAR SEISMIC TIME HISTORY RESPONSE

This block performs the elastic or nonlinear analysis of a structure subject to multiple ground accelerations. The flow chart for SOL02 is shown in Figure 16.

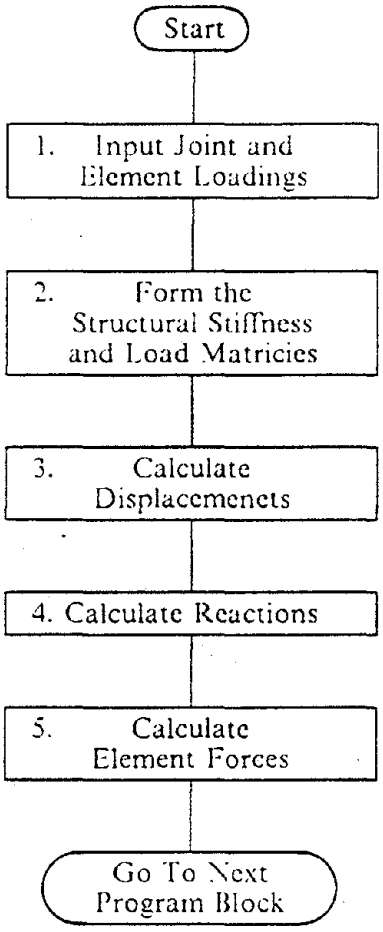


Figure 15. Block SOL01 - Static Analysis

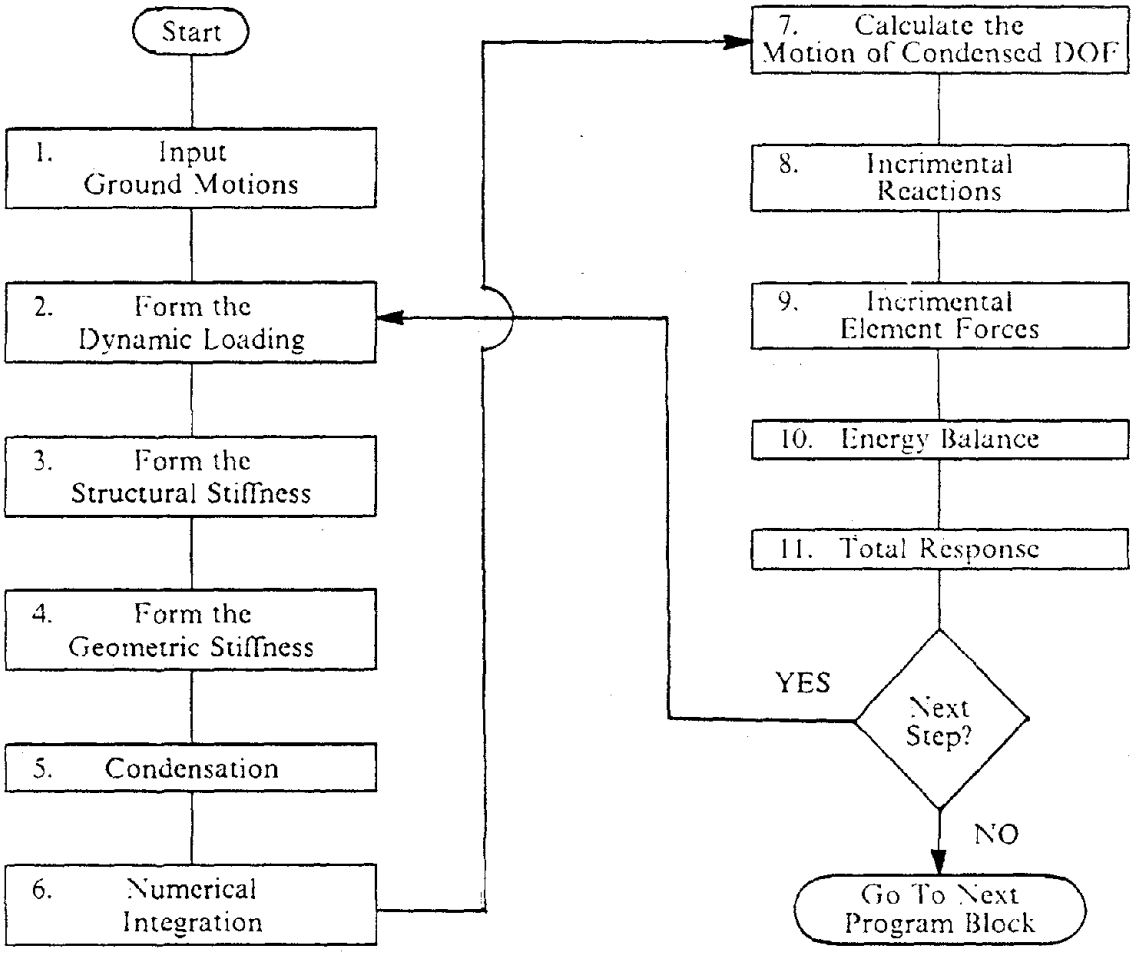


Figure 16. Block SOL02 - Elastic / Nonlinear Dynamic Analysis

Step 1. --- Input Ground Motions. The ground motions are input and stored. Transformation matrices to rotate the ground accelerations from the input coordinate system to the individual JCS are formed. The equation of motion for a dynamic loadings is

$$M\ddot{X} + C\dot{X} + KX = F(t) = - M'\ddot{X}_g \tag{1}$$

For seismic analyses, $M = M'$. For nonseismic analyses the special mass matrix, M', and a psudo-ground acceleration are input and stored. Thus the nonseismic forcing function function is equal to

$$F(t) = - M'\ddot{X}_g \tag{2}$$

Step 2. --- Form the Dynamic Loading. The dynamic loading matrix is formed and added to the unbalanced force matrix.

Step 3. --- Form the Structural Stiffness. The structural stiffness is formed 1) for the first time step, 2) for every time step that an element's stiffness is modified, and 3) for every time step that the geometric stiffness is modified. The element's stiffness is only modified during a nonlinear analysis.

Step 4. --- Form the Geometric Stiffness. The geometric stiffness is formed 1) for the first time step, and 2) for every time step if the actual element axial loads are used to calculate the geometric stiffness.

Step 5. --- Condensation. If condensed degrees of freedom exist: 1) the structural stiffness is condensed each time step that is formed, 2) the geometric stiffness is condensed each time step it is formed, if condensation of the geometric stiffness is desired, 3) the mass matrix is condensed the first time step, if needed, and 4) the dynamic loading matrix is condensed each time step.

Step 6. --- Numerical Integration. The incremental displacements, velocities and accelerations of free dof are calculated by the linear acceleration method or the average acceleration method.

Step 7. --- Calculate the Motion of Condensed DOF. The incremental displacements, velocities and accelerations of condensed dof are calculated.

Step 8. --- Incremental Reactions. The incremental reactions are calculated.

Step 9. --- Incremental Element Forces. The hysteresis models in the material library are called to calculate the incremental element forces, given the incremental displacements and previous loading history. For nonlinear analysis, if the element stiffness changes during the incremental displacement, 1) the element's unbalanced forces are calculated, and 2) a flag to reform the structural stiffness in step 3 is set for the next time step. The elastic and plastic strain energies for each element are also calculated.

Step 10. --- Energy Balance. The total input energy, kinetic energy, elastic strain energy, plastic strain energy and energy dissipated due to damping are calculated.

Step 11. --- Total Response. The total displacements, velocities, accelerations, reactions, and element forces are calculated. The unbalanced force vector for nonlinear analysis is assembled from the element's unbalanced forces. If desired, selected results may be written to output files.

Go to step 2 for the next time step.

D. SOL03 - ELASTIC NATURAL FREQUENCY / ELASTIC BUCKLING LOAD

This block calculates either the natural frequencies and mode shape of an elastic structure, or the buckling load and mode shape of an elastic structure. The flow chart for SOL03 is shown in Figure 17.

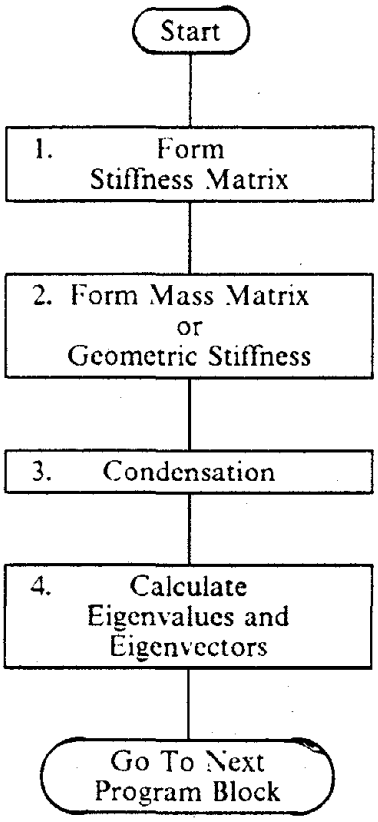


Figure 17. Block SOL03 - Natural Frequency / Elastic Buckling Load

Step 1. --- Form the Structural Stiffness Matrix. The structural stiffness matrix is formed.

Step 2. --- Form Mass Matrix or Geometric Stiffness Matrix. For the buckling solution, the geometric stiffness matrix is formed. For the natural frequency solution, the mass matrix that was formed in block STRUCT is used.

Step 3. --- Condensation. If condensed degrees of freedom exist: 1) the structural stiffness is condensed, 2) the geometric stiffness is condensed for buckling problems, if it is desired, 3) the mass matrix is condensed for natural frequency problems, if it is needed.

Step 4. --- Calculate Eigenvalue and Eigenvector. The eigenvalues and eigenvectors are calculated. Natural frequencies or buckling loads are extracted from the eigenvalues. The eigenvectors are normalized. The natural frequencies are printed along with the mode shape of each mode, or the buckling loads are printed along with the corresponding mode shape.

E. SOL04 - NONLINEAR STATIC CYCLIC RESPONSE

This block calculates the nonlinear static cyclic response for a given loading pattern. A loading pattern consisting of joint loads, imposed displacements and element loads is defined and stored in the vector {Q}. The loading pattern, {Q}, is multiplied by positive and negative load factors to generate cyclic loading cycles. Define Fj as the loading factor for the current cycle, and Fi as the loading factor for the previous cycle. The total loads on the structure for cycles i and j are Fi{Q} and Fj{Q}, respectively. The loading from Fi{Q} to Fj{Q} is carried out in a series of steps. A variable number of steps, with each step having incremental loads and displacements less than limiting values, may be used to load from Fi{Q} to Fj{Q} (loading option A). Alternately, a fixed number of equal load steps may be chosen to load from Fi{Q} to Fj{Q} (loading option B). The flow chart for SOL04 is shown in Figure 18.

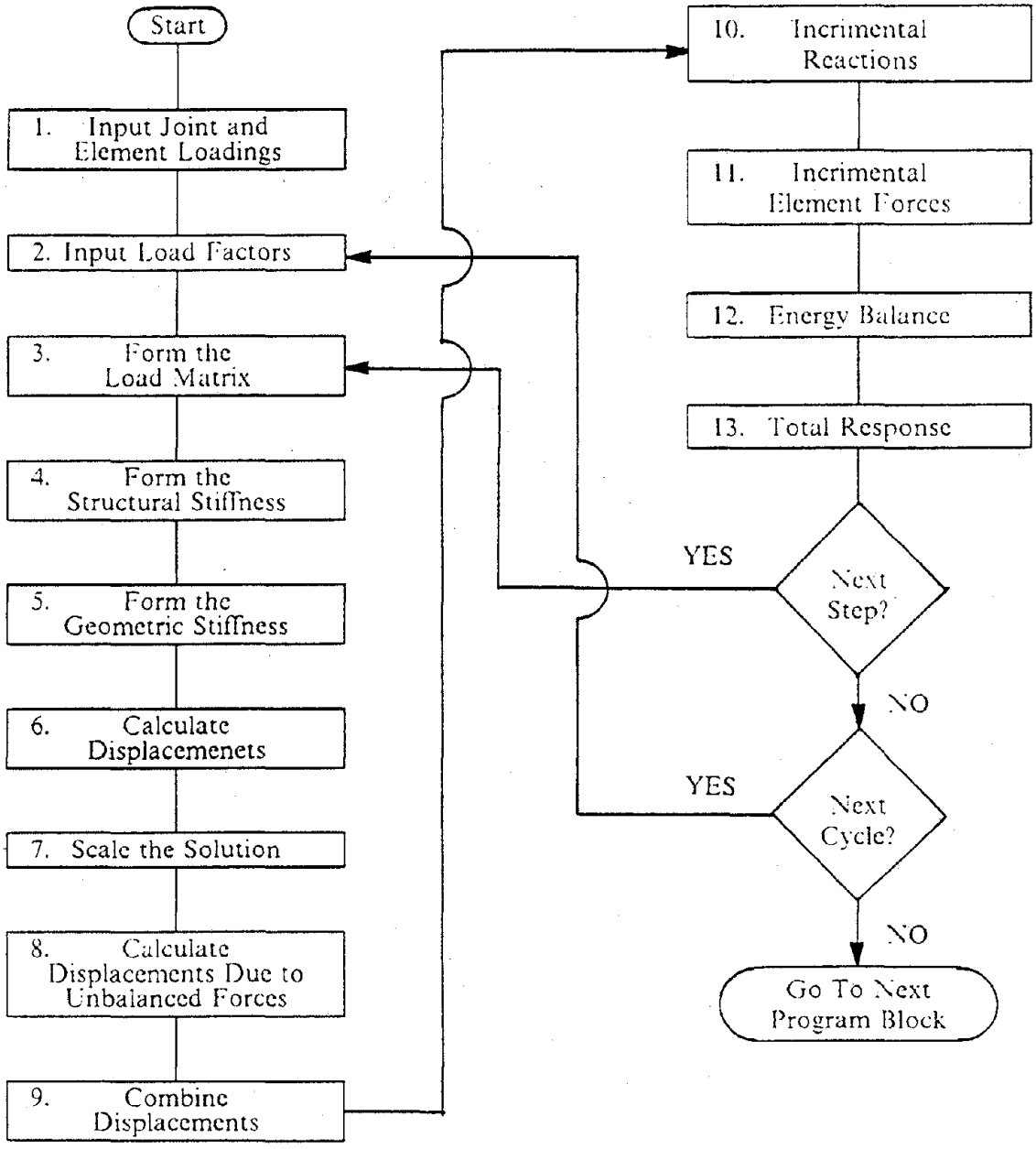


Figure 18. Block SOL04 - Nonlinear Static Cyclic Response

Step 1. --- Input Joint and Element Loadings. The joint loads and imposed displacements of the load pattern are input. Element loadings for the load pattern are input on the 3D-beam element.

Step 2. --- Input Load Factors. For each loading cycle, a load factor, and limits on the step size, or the number of load steps are input.

Step 3. --- Form the Load Matrix. The loading matrix for option A is the difference in the load at the end of the cycle and the current load. Alternately, the loading matrix for option B is the incremental load. $\frac{Fj - Fi}{N} \times \{Q\}$, where N is the number of load steps.

Step 4. --- Form the Structural Stiffness. The structural stiffness is formed 1) for the first load step, 2) for every load step that an element's stiffness is modified, and 3) for every load step that the geometric stiffness is modified.

Step 5. --- Form the Geometric Stiffness. The geometric stiffness is formed 1) for the first load step, and 2) for every load step if the actual element axial loads are used to calculate the geometric stiffness.

Step 6. --- Calculate Displacements. The incremental displacements due to the applied loadings are calculated by Gauss elimination.

Step 7. --- Scale the Solution. For loading option A, the response is scaled down such that the incremental loads and displacements for each step are less than the limiting values. This step is omitted for loading option B.

Step 8. --- Calculate Displacements due to Unbalanced Forces. The incremental displacements due to the unbalanced forces from the previous load step are calculated by Gauss elimination.

Step 9. --- Combine Displacements. The scaled down displacements due the applied loading and the displacements due to the unbalanced loadings are added together.

Step 10. --- Incremental Reactions. The incremental reactions are calculated.

Step 11. --- Incremental Element Forces. The hysteresis models in the material library are called to calculate the incremental element forces, given the incremental displacements and previous loading history. For nonlinear analysis, if the element's stiffness changes during the incremental displacement, 1) the element's unbalanced forces are calculated, and 2) a flag to reform the structural stiffness in step 4, is set for the next load step. The elastic and plastic strain energy for each element is also calculated.

Step 12. --- Energy Balance. The total input energy, elastic strain energy and plastic strain energy are calculated.

Step 13. --- Total Response. The total displacements, reactions, and element forces are calculated. The unbalanced force vector for nonlinear analysis is also calculated. If desired, selected results may be written to output files.

Go to step 3 for additional loading steps. Go to step 2 for the next loading cycle.

III. DESCRIPTION OF PROGRAM

The computer program INRESB-3D-SUP is described in this chapter. A flow chart of the program is given in Figures 19 and 20.

A. MAIN PROGRAM AND COMMON BLOCKS

A brief description of the main program and the three common blocks is presented in this section.

1. MAIN - Main Program. The main program 1) reads and echoes input data, 2) initializes data, 3) calls STRUCT to define the structural model, 4) calls SOLN to perform individual solutions, 5) calls COMDMP to dump the contents of the memory, 6) sets bug options to print stiffness matrices, etc, 7) releases memory from previous solutions, and 8) stops execution of the program.

2. Common Block DATA. This common block consists of a linear array that contains the program's dynamically allocated memory. The linear array consists of a real array, Z, and an integer array, NZ. The integer array and the real array share the same storage location in memory[1]. Common block DATA is used by routines MAIN, CKSTOR, STRUCT, MATLIB, ELELIB, DMPDAT FORM, SOLN, SOL01, SOL02, SOL03 and SOL04.

3. Common Block ZDATA. This common block contains flags which control the execution of the routines, the number of Gdof, joints, elements, etc. and the addresses of data in the Z array that is shared between different routines. ZDATA is used by routines MAIN, CKSTOR, STRUCT, MATLIB, ELELIB, DMPDAT, FORM, SOLN, SOL01, SOL02, SOL03, SOL04, COMPT and COMDMP.

---

[1] This is accomplished by the Fortran EQUIVALENCE statement.

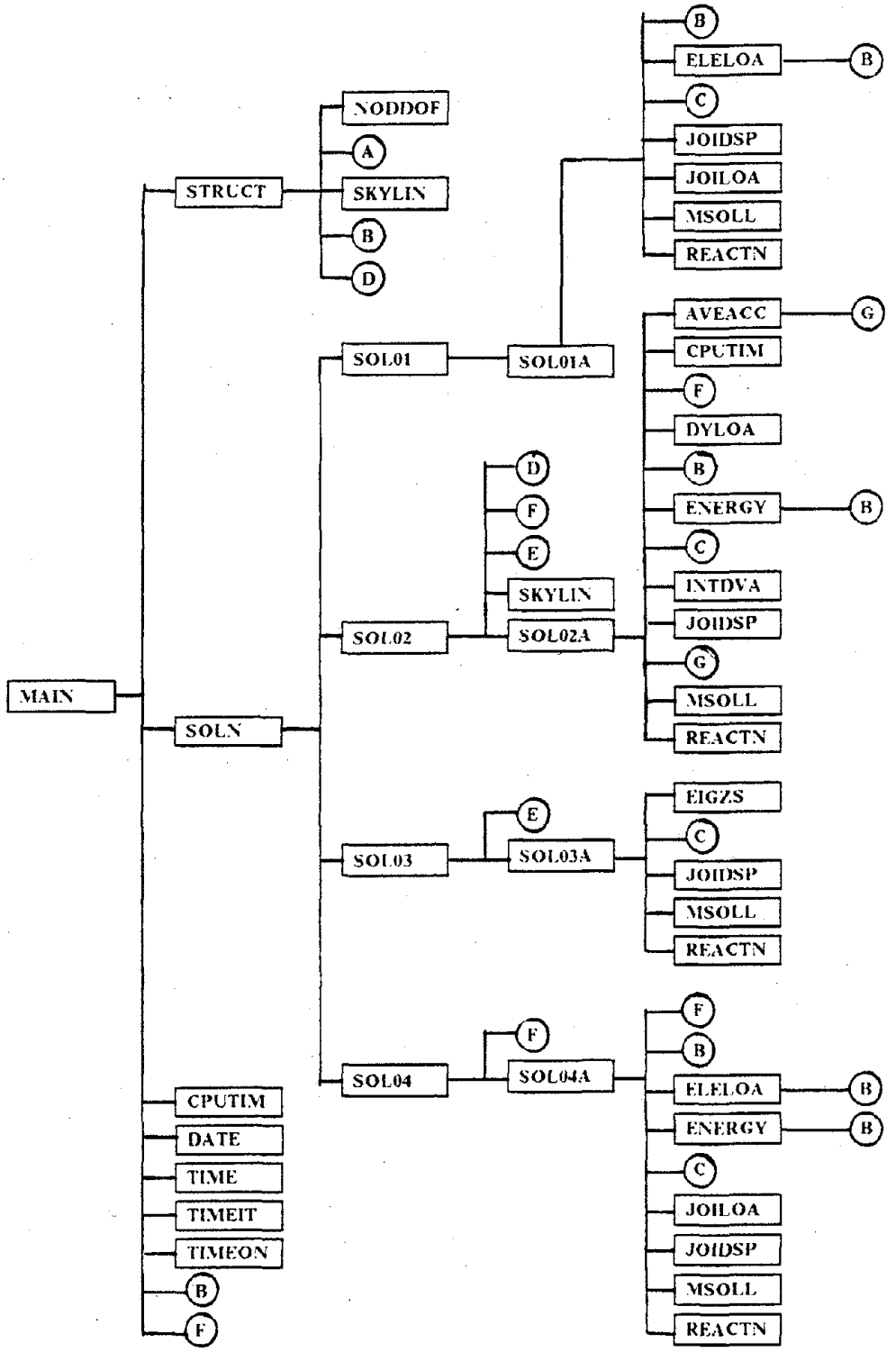


Figure 19. Flow Chart for Program INRESB-3D-SUP: Part A

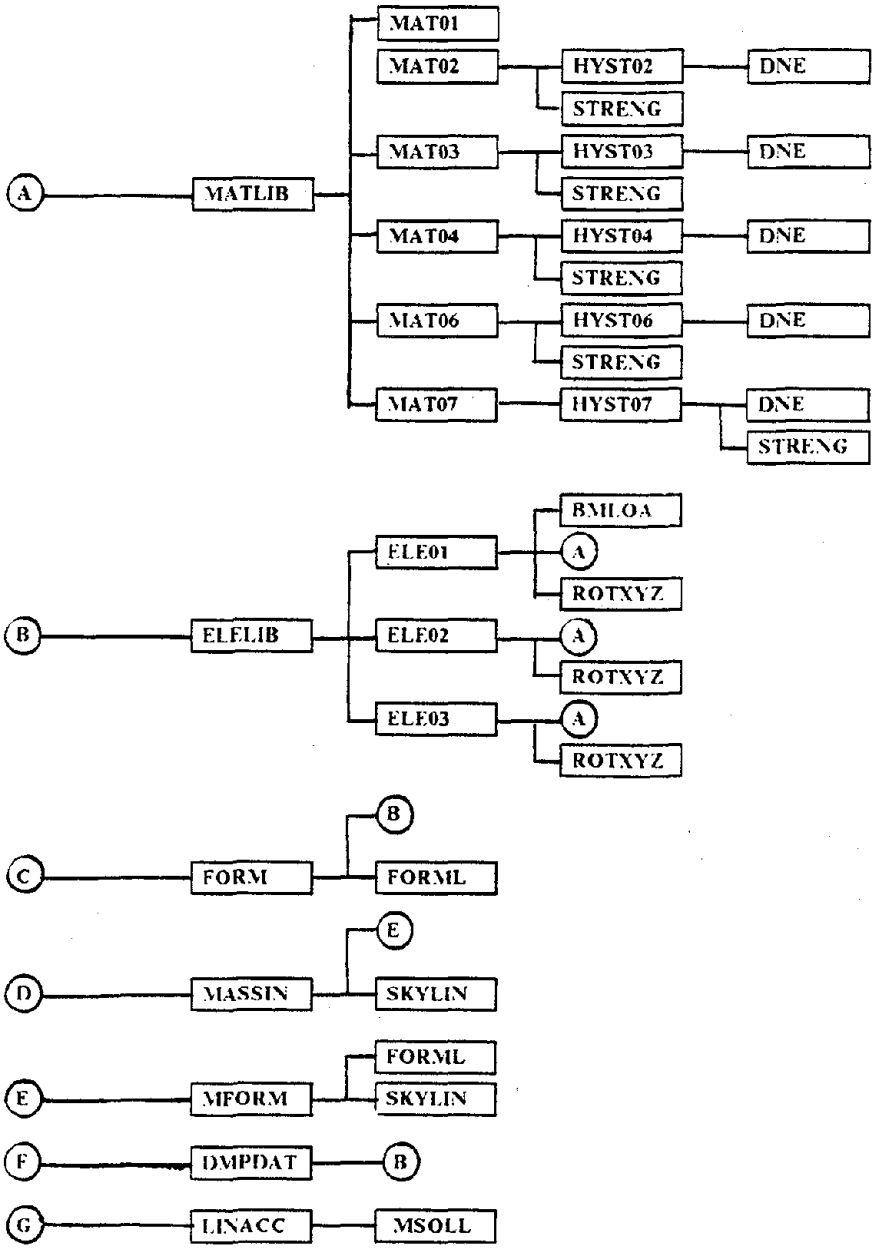


Figure 20. Flow Chart for Program INRESB-3D-SUP: Part B

4. Common Block ZDATAP. This common block contains the title, the bug options, and several other miscellaneous variables. ZDATAP is used by routines MAIN, CKSTOR, STRUCT, MATLIB, ELELIB, DMPDAT, FORM, SOLN, SOL01, SOL02, SOL03, SOL04, COMPT and COMDMP.

B. DESCRIPTION OF ROUTINES

The program INRESB-3D-SUP includes 73 routines. These are presented in alphabetical order with a brief description.

1. AVEACC. AVEACC and LINACC calculate the dynamic response by the average acceleration method. Called by SOL02A.

2. BMLOA. Calculate the fixed end moments and shears of a prismatic beam element for uniform and concentrated element loadings. Called by ELE01.

3. CKRNG. A utility routine called by ELELOA that checks the location of element loadings.

4. CKSTOR. A utility routine that aborts the program if the problem requires more storage than is available. Called by SOL02, DMPDAT, SOL03 and SOL04.

5. COMDMP. A utility routine that prints the contents of the dynamic memory. Called by MAIN.

6. COMPT. A utility routine that initializes the addresses of the dynamic memory. Called by MAIN.

7. CROSS. Vector cross product. Called by DYLOA and NODDOF.

8. DIRECT. A utility routine called by HYST03 and HYST04 to determine the current direction of loading.

9. DMPDAT. 1) Writes the results to user defined output files. 2) Reads the results from output files and prints data. Called by SOL02, SOL02A, SOL04, SOL04A, and MAIN.

10. DNE. Calculates ductilities and excursion ratios for HYST02, HYST03, HYST04, HYST06 and HYST07.

11. DYLOA. Generates the dynamic loadings due to base acceleration for multicomponent earthquakes. Called by SOL02A.

12. ELELIB - Element Library. This subroutine is a controller that calls the members of the element library. All calls to the elements must pass through this subroutine. Called by MAIN, FORM, STRUCT, SOL01A, SOL02A, SOL04A, ELELOA, ENERGY and DMPDAT.

13. ELELOA. Calculate the element loads for the elastic 3D prismatic beam element. Called by SOL01A and SOL04A.

14. ELE01 - Elastic 3D Prismatic Beam Element. This routine calculates the stiffness, geometric stiffness, element loads, displacements, forces, etc. for the elastic 3D prismatic beam element. Called by ELELIB.

15. ELE02 - Spring Element. This routine calculates the stiffness, geometric stiffness, displacements, forces, unbalanced forces, etc. for the spring element. Called by ELELIB.

16. ELE03 - Shear Wall Stiffness Element. This routine calculates the stiffness, geometric stiffness, displacements, forces, unbalanced forces, etc. for the shear wall stiffness element. Called by ELELIB.

17. ENERGY. Calculates the energy balance. Called by SOL02A and SOL04A.

18. ETMPLY. Performs the multiplication, $\{\dot{x}\}^t[M]\{\dot{x}\}$, to calculate kinetic energy, for an upper triangular mass matrix. Called by ENERGY.

19. FEMREL. Modifies the fixed end forces to account for end releases. Called by BMLOA.

20. FIRST. A utility routine that performs character manipulations. Called by GETCHR.

21. FORM. FORM and FORML are used to assemble the structural and geometric stiffness matrices. Called by SOL01A, SOL02A, SOL03A and SOL04A.

22. FORML. 1) Adds the element stiffness to the global structural stiffness. 2) Adds the element geometric stiffness to the global geometric stiffness. 3) Adds the joint mass to the global mass matrix. Called by FORM and MFORM.

23. GETCHR. A Utility routine that extracts a character string from input. Called by MAIN.

24. GETINT. A utility routine that extracts an integer from character input. Called by MAIN, LMATRX, SOLN, WMATRX and WMTRX2.

25. HYST02 - Axial Hysteresis Model. Calculates the axial stiffness of R/C elements based on the current load and the past loading history by the axial hysteresis model. Called by MAT02.

26. HYST03 - B1 Bending Hysteresis Model. Calculates the bending stiffness of R/C shear walls based on the current load and the past loading history by the B1 bending hysteresis model. Called by MAT03.

27. HYST04 - S1 Shear Hysteresis Model. Calculates the shear stiffness of R/C shear walls based on the current load and the past loading history by the S1 shear hysteresis model. Called by MAT04.

28. HYST06 - Takeda Hysteresis Model. Calculates the bending stiffness of R/C elements based on the current load and the past loading history by the Takeda hysteresis model. Called by MAT06.

29. HYST07 - Bilinear Hysteresis Model. Calculates the stiffness based on the current load and the past loading history by the bilinear hysteresis model. This model is also capable of producing elasto-plastic behavior. Called by MAT06.

30. IDRECT. A utility routine that determines the current direction for HYST02 and HYST06.

31. INTDVA. Reads and stores initial displacement, velocity and accelerations for dynamic solutions. Called by SOL02A.

32. INTSCT. A utility routine that determines the intersection of two lines in point slope form. Called by HYST02, HYST03 and HYST04.

33. IQUICK. A utility routine that finds the program's internal joint number, given the joint's ID number. Called by NODDOF, ELE01, ELE02, ELE03, JOILOA, INTDVA, DMPDAT and MFORM.

34. ISORT. A utility routine that sorts the joint input. Called by NODDOF.

35. JOIDSP. Calculates and prints the displacement, velocity, acceleration or eigenvector at each joint in the JCS and GCS. Called by SOL01A, SOL02A, SOL03A and SOL04A.

36. JOILOA. Reads and calculates the joint loads. Called by SOL01A and SOL04A.

37. KMIN. A utility routine used by the hysteresis models HYST03 and HYST04 to ensure the stiffness is between zero and the initial stiffness, SI.

38. LINACC. Calculates the dynamic response by the linear acceleration method. Called by SOL02A and AVEACC.

39. LMATRX. A utility routine that prints upper triangular mass, structural stiffness, element stiffness, geometric stiffness, etc. matrices. This routine is activated by the user input bug options, which are described later. Called by FORM, MFORM, ELE01, ELE02, ELE03, SOL01A, SOL02A, SOL03A, SOL04A, MSOLL, TMPLY, ETMPLY, UTMPLY and MASSIN.

40. MASSIN. Initializes the storage for the mass matrix. Called by STRUCT and SOL02.

41. MATLIB - Material Library. This subroutine is a controller that calls the members of the material library. All calls to the materials and hysteresis models must pass through this subroutine. Called by STRUCT, ELE01, ELE02 and ELE03.

42. MAT01 - Elastic 3D Prismatic Beam Stiffness. This routine reads and stores the stiffness for the elastic 3D prismatic beam element. Called by MATLIB.

43. MAT02 - Axial Stiffness for R/C Elements. 1) Reads and stores axial input data. 2) Calculates the axial stiffness with calls to the axial hysteresis model, HYST02. 3) Calculates the strain energy with calls to STRENG. Called by MATLIB.

44. MAT03 - Bending Stiffness for R/S Shear Walls. 1) Reads and stores bending input data. 2) Calculates the bending stiffness with calls to the B1 bending

hysteresis model, HYST03. 3) Calculates the strain energy with calls to STRENG. Called by MATLIB.

45. MAT04 - Shear Stiffness for R/C Shear Walls. 1) Reads and stores shear input data. 2) Calculates the shear stiffness with calls to the S1 shear hysteresis model, HYST04. 3) Calculates the strain energy with calls to STRENG. Called by MATLIB.

46. MAT06 - Bending Stiffness for R/C Elements. 1) Reads and stores bending input data. 2) Calculates the bending stiffness with calls to the Takeda hysteresis model, HYST06. 3) Calculates the strain energy with calls to STRENG. Called by MATLIB.

47. MAT07 - Bilinear Material. 1) Reads and stores input data. 2) Calculates the stiffness with calls to the bilinear hysteresis model, HYST07. 3) Calculates the strain energy with calls to STRENG. Called by MATLIB.

48. MFORM. Reads the joint masses and forms the mass matrix with calls to FORML. Called by MASSIN, SOL02 and SOL03.

49. MSOLL. 1) Performs Gauss elimination of the stiffness matrix. 2) Condenses the mass matrix while performing Gauss elimination. 3) Condenses the geometric stiffness while performing Gauss elimination. 4) Reduces the load matrix. 5) Performs back substitution. 6) Operations 1 through 5 may be performed on partitioned matrices. 7) Stiffness, geometric stiffness and mass matrices are upper triangular matrices with skyline storage. Called by SOL01A, SOL02A, SOL03A, SOL04A and LINACC.

50. NODDOF. Reads joints, constraints, restraints and condensation information, calculates the JCS cosine matrices, and determines the global degrees of freedom. Called by STRUCT.

51. PBIT. A utility routine that prints the joint restraint and constraint code for each joint. Called by NODDOF.

52. REACTN. 1) Modifies the loading to include restraint displacements. 2) Solves for the reactions. 3) Prints the reactions in the JCS and GCS coordinate systems. 4) Calculates the summation of reactions. Called by SOL01A, SOL02A, SOL03A and SOL04A.

53. ROTXYZ. 1) Calculates the local element cosine matrix. 2) Calculates the transformation from ECS to JCS. 3) Calculates the constraint transformation. 4) Calculates the transformation for a rigid body at the end of the beam element. Called by ELE01, ELE02 and ELE03.

54. SKYLIN. Determines the skyline of the structural stiffness, geometric stiffness and mass matrices. Called by STRUCT, MFORM and MASSIN.

55. SKYLN2. Determines the skyline of the dynamic stiffness matrix. Called by SOL02.

56. SMAX. A utility routine that determines and prints the the maximum and minimum input accelerations. Called by DYLOA.

57. SOLN - Solution Library. This subroutine is a controller that calls the solution requested by the user. All calls to the solutions must pass through this subroutine. Called by MAIN.

58. SOL01 - Elastic Static Analysis with Multiple Load Cases. This routine initializes the storage for the elastic static analysis. Called by SOLN.

59. SOL01A. This routine calculates the elastic static response, as described in Section B of Chapter II. Called by SOL01.

60. SOL02 - Dynamic Time History Response. This routine initializes the storage for the dynamic time history response. Called by SOLN.

61. SOL02A. This routine calculates the dynamic time history response for both elastic and nonlinear structures, as described in Section C of Chapter II. Called by SOL02.

62. SOL03 - Elastic Natural Frequency / Buckling Load. This routine initializes the storage for the elastic natural frequency or the buckling load. Called by SOLN.

63. SOL03A. This routine calls EIGZS to calculate the natural frequency or buckling load for elastic structures, as described in Section D of Chapter II. Called by SOL03.

64. SOL04 - Nonlinear Static Cyclic Response. This routine initializes the storage for the static nonlinear analysis. Called by SOLN.

65. SOL04A. This routine calculates the static nonlinear response, as described in Section E of Chapter II. Called by SOL04.

66. STRENG. Calculates the elastic and plastic strain energy. Called by MAT02, MAT03, MAT04, MAT06, HYST07.

67. STRUCT - Structural Definition. 1) Sets up storage for joints and calls NODDOF to input joint information and to generate degrees of freedom. 2) Sets up storage for materials and calls MATLIB to input materials and to initialize the hysteresis models. 3) Sets up storage for elements and calls ELELIB to input elements and to calculate initial element stiffnesses. 4) Calls SKYLIN to set up the storage for the structural stiffness, geometric stiffness and mass matrices. 5) Reads and initializes the mass matrix. These functions are described in greater detail in Section A of Chapter II. Called by MAIN.

68. TEST. A utility routine that examines character input, and sets a logical flag. Called by MAIN, STRUCT, SOL02, SOL03A, SOL04A, INTDVA and DMPDAT.

69. TMPLY. A utility routine that performs multiplication for triangular matrices Called by SOL02A, SOL04A and ETMPLY.

70. VCOS. A utility subroutine that calculates the angle between two vectors. Called by ROTXYZ and ELE03.

71. VCROSS. A utility subroutine that calculates the vector cross product. Called by ELE03.

72. WMATRX. A utility subroutine that prints a matrix. This routine is activated by the user input bug options which are described later. Called by ELE01, ELE02, ELE03, SOL01A, SOL04A, REACTN, MSOLL and DYLOA.

73. WMTRX2. A utility subroutine that prints a portion of a matrix. This routine is activated by the user input bug options which are described later. Called by ELE02 and ELE03.

C. DESCRIPTION OF MISCELLANEOUS ROUTINES

Several other miscellaneous routines are used by program INRESB-3D-SUP, which are listed below.

1. EIGZS. An IMSL eigenvalue routine that is called by SOL03A to calculate the natural frequency and buckling load. This routine is only required for SOL03. Omission of this routine will not affect blocks STRUCT, SOL01, SOL02 or SOL04.

2. SDUMP. An IBM routine that prints the contents of memory with bug option K. Called by COMDMP. Omission of this routine will only effect bug option K.

3. CPUTIM, TIMEON and TIMEIT. These are three routines on the local UMR system that are used to print the results of the analysis and abort SOL02 if the CPU time limit is close to being exceeded. Equivalent routines may be substituted on other systems, or these routines may be commented out. Called by MAIN and SOL02A.

4. TIME and DATE. These are two routines on the local UMR system that are used to print the time and date of the analysis on the header. Equivalent routines may be substituted on other systems, or these routines may be commented out. Called by MAIN.

D. DYNAMIC MEMORY MANAGEMENT

1. Dynamic Memory Allocation. The computer program INRESB-3D-SUP stores all of the joint, material, element, mass, stiffness, response, etc. data in a large linear array, Z, that resides in the common block DATA. The Z array consists of real numbers. An integer array NZ is set equal to the Z array, by a Fortran EQUIVALENCE statement, to provide storage for integer variables. Index variables beginning with IZ, such as IZxxxx, are used to store the address of the information in the Z array. The variable IZ is used to track the next available space in the Z array. The exact amount of space required to solve a specific problem is reserved in the Z array during the execution of the program. This dynamic allocation of memory allows the program to use the computers memory efficiently.

For example, assume that the Z array is empty and IZ = 1. The joint ID numbers are integer variables that are stored in the NZ array. Thus the index for the joint ID numbers is IZID = IZ = 1 and NUMJOI joint numbers are stored in the NZ array. Once the storage for the joint numbers is reserved, the next available storage location

is IZ = IZ + NUMJOI. Next the joint coordinates are stored in the Z array, beginning at IZCORD = IZ. The joint coordinates require 3*NUMJOI storage locations. Thus once the storage for the joint coordinates is reserved, the next available storage location is IZ = IZ + 3*NUMJOI. The Z array with joint numbers and coordinates is shown in Figure 21.

2. Compact Matrix Storage. The mass, stiffness, geometric stiffness and dynamic stiffness matrices are sparse symmetric matrices. Because the matrices are symmetric, only half (the upper triangular matrix) of each matrix needs to be stored. Additionally, many of the terms in the upper triangular matrix are zero. The nonzero terms, in a given column, are typically found near the main diagonal. The level of the upper nonzero term in each column is the skyline. Thus only the terms below the skyline of the upper triangular matrix are stored in the Z array. Additionally the linear array addresses of the main diagonal elements are stored in the array MD. For example, let the matrix B be the 5x5 full symmetric matrix below

$$B = \begin{bmatrix} 1 & 3 & 0 & 0 & 12 \\ 3 & 2 & 5 & 0 & 11 \\ 0 & 5 & 4 & 7 & 10 \\ 0 & 0 & 7 & 6 & 9 \\ 12 & 11 & 10 & 9 & 8 \end{bmatrix} \tag{3}$$

where $B_{1,3}$, $B_{1,4}$ and $B_{2,4}$ are zero. Thus the terms in the upper triangular matrix, below the skyline are

$$B_{\text{upper triangular}} = \begin{bmatrix} 1 & 3 & & 12 \\ & 2 & 5 & 11 \\ & & 4 & 7 & 10 \\ & & & 6 & 9 \\ & & & & 8 \end{bmatrix} \tag{4}$$

These elements are stored in the linear array by columns, beginning with the element on the main diagonal. Thus the B matrix in linear array form becomes

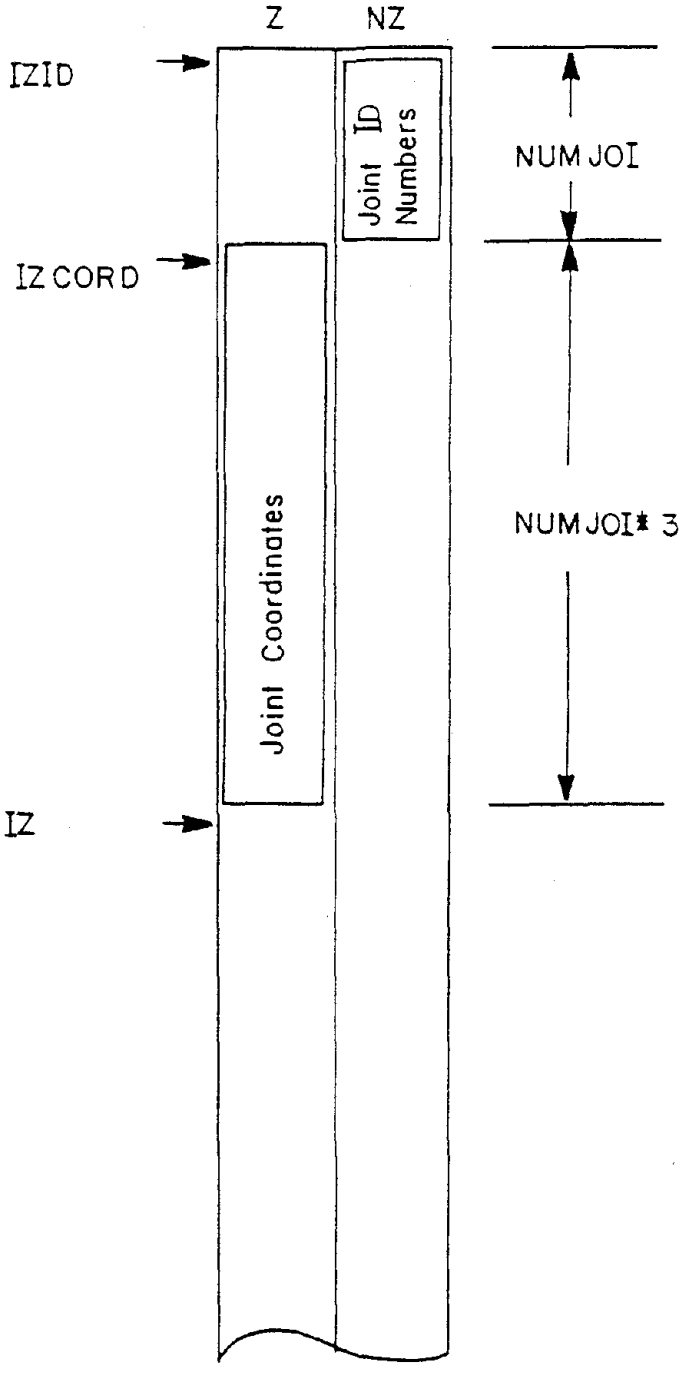


Figure 21. Dynamic Memory Example

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{bmatrix} \tag{5}$$

and the array, MD, containing the addresses of the main diagonal elements is

$$MD = \begin{bmatrix} 1 \\ 2 \\ 4 \\ 6 \\ 8 \\ 13 \end{bmatrix} \tag{6}$$

The last row of the array is a dummy element with the value $N+1$, where N is the number of elements in the linear array. The i,j element of the linear array is addressed with the algorithm

$$\begin{aligned} B(I,J) &= A(MD(J) + J - I) \quad \text{If } J > I \text{ and } (MD(J) + J - I) < MD(J+1) \\ &= 0 \qquad\qquad\qquad\quad \text{If } J > I \text{ and } (MD(J) + J - I) \geq MD(J+1) \end{aligned} \tag{7}$$

This storage method was also presented by Bath and Wilson (2).

3. Capacity of the Dynamic Memory.

The program's capacity is a function of the amount of memory available in the computer. The program instructions compiled in double precision on an IBM 4381 computer require about 0.8 megabytes of memory. Assuming 8 megabytes of total memory is available, the Z array may occupy $8 - 0.8 = 7.2$ megabytes of memory. For

double precision, on an IBM computer, each variable is 8 bytes long. Thus for 8 megabytes of memory, the Z array may contain up to $\frac{7.2 \times 10^6}{8} = 900{,}000$ variables.

The number of variables in the Z array required for a specific problem is a function of 1) the number and orientation of the joints, 2) the number of restraints, 3) the number of constraints, 4) the number and types of materials, 5) the number and types of elements and the materials each element uses, 6) the number of joint masses, 7) the amount of storage required by the structural stiffness, geometric stiffness, and mass matrices, which is a function of the bandwidth, 8) the type of solution, 9) the number of degrees of freedom, 10) the number load cases for SOL01, 11) the number of element loadings for SOL01 and SOL04, 12) the number of free dof for SOL02 and SOL03, and 13) the length of the ground acceleration record for SOL02. The actual number of variables used, and the percentage of total memory used are printed out at the end of each block. For the example in Section C of Chapter V, the following message is printed in the output.

```
*-------------------------------------*
*--- MEMORY UTILIZATION ..... ----*
*--- IZ=  21965,    MEM= 4.393%   ----*
*-------------------------------------*
```

This example required 21,965 variables. Four megabytes of memory are available for the Z array and the program is compiled in double precision on an IBM computer. Thus the Z array has a capacity of $\frac{4 \times 10^6}{8} = 500{,}000$ variables, and $\frac{21965}{500{,}000} \times 100 = 4.393\%$ of the available memory was used.

The capacity of the program is modified by changing the value of MAXZ in the PARAMETER statement

```
PARAMETER (MAXZ=500000, MAXDZ=MAXZ/2)
```

of file ZCOMN2 COPY. The macro library GEM3 is then regenerated and the program is recompiled.

E. ADDITION OF MATERIALS, ELEMENTS, AND SOLUTIONS

The modular form of the INRESB-3D-SUP program allows for the relatively easy addition of materials, elements and solutions. The changes required to add materials, elements and solutions are discussed below.

1. Addition of a Material to the Material Library. A subroutine with the name MATxx is written that contains the material information. The subroutine must have the following options to be consistent with the existing subroutines:

IOPT = 0 Determines the number of storage spaces required for each element utilizing this material. This storage is typically used by a hysteresis model that is called by routine MATxx.

IOPT = 1 Inputs and stores data. Returns the amount of storage required for the material.

IOPT = 2 Calculates the initial stiffness and initializes the hysteresis model.

IOPT = 3 Calculates the current stiffness and load for an incremental displacement, given the previous loading history. Calculates the strain energies, ductilities, and excursion ratios.

IOPT = 4 Calculates the strain energies, ductilities, and excursion ratios.

IOPT = 5 Calculates the damage index.

For nonlinear materials, the subroutine MATxx calls a companion hysteresis model, subroutine HYSTxx.

The call to MATxx is added in routine MATLIB. All communication with the subroutine MATxx is through the routine MATLIB. The name of the material and the material number (MATTYP = xx) are added to routine STRUCT.

2. Addition of an Element to the Element Library. A subroutine with the name ELExx is written that contains the element information. The subroutine must have the following options to be consistent with the existing subroutines:

IOPT = 1 Interprets the input data, calculates transformation matrices, initial structural stiffness and a unit load geometric stiffness. Determines the actual amount of storage required.

IOPT = 2 Determines the axial load used for the geometric stiffness.

IOPT = 3 Recalculates the structural stiffness, if one of the components of stiffness has changed.

IOPT = 4 Calculates the incremental forces, unbalanced forces and new components of stiffness.

IOPT = 5 Calculates the total element forces and energies.

IOPT = 6 Calculates the fixed end forces due to element loads, if applicable.

IOPT = 7 Writes element data to an output file.

IOPT = 8 Reads element data from an output file and print.

IOPT = 9 Determines the amount of storage required for the element's materials.

IOPT = 10 Determines the strain energy.

IOPT = 11 Determines the ductilities and excursion ratios.

IOPT = 12 Prints out the maximum forces and/or displacements that the element was subjected to.

IOPT = 13 Sets the internal forces and displacement equal to zero. Reinitializes the hysteresis models.

IOPT = 14 Calculates and prints the damage index.

The call to ELExx is added in routine ELELIB. All communication with the subroutine ELExx is through the routine ELELIB. The name of the element, the element number (NELTYP=xx), the number of variables input for the element, and the maximum amount of storage required for the element are added to routine STRUCT.

3. Addition of a Solution. A subroutine with the name SOLxx is written that reserves and initializes the storage required by the solution. SOLxx also calls CKSTOR to ensure that the program has enough storage to execute the solution. The call to SOLxx is added to routine SOLN. All communication with the subroutine SOLxx is through the routine SOLN.

Subroutine SOLxx calls subroutine SOLxxA which performs the actual solution. Subroutine SOLxxA may call other subroutine such as FORM to form the stiffness matrix, JOILOA to generate joint loadings, MSOLL to solve for displacements or ELELIB to calculate element forces. The functions of these subroutines and others are described in the previous section.

IV. INPUT DATA FOR PROGRAM INRESB-3D-SUP

The input for program INRESB-3D-SUP is divided into several blocks. Each block is briefly described below. The blocks may be executed in any order, except as noted. Multiple solutions of the same structure are possible by using multiple solution blocks. Multiple structures may be analyzed by redefining the structure with the STRUCT block.

BLOCK	DESCRIPTION
STRUCT	Defines the structure to be analyzed. Joints, materials, elements, mass and damping are defined.
SOL01	Elastic static solution of the structure with multiple load cases. SOL01 must be proceeded by the block STRUCT.
SOL02	Dynamic solution of linear and nonlinear structures subject to three dimensional ground motion by numerical integration. SOL02 must be proceeded by the block STRUCT.
SOL03	Eigenvalue solution for the natural frequency or buckling load of an elastic structure. SOL03 must be proceeded by the block STRUCT.
SOL04	Incremental static solution of a nonlinear structure. SOL04 must be proceeded by the block STRUCT.

Secondary Blocks

BUG	Sets flags to print out detailed information.
READ	Reads results written to an output file during SOL02 or SOL04, and prints the results.
NOECHO	Inhibits the input echo.
DUMP	Prints out the contents of the memory.
RELEASE	Releases the memory used for the previous solution.
STOP	Terminates execution of the program.

NOTES ON INPUT

1) Input is free format, unless otherwise noted.

2) Input variables beginning with I-N are integers, and should not contain a decimal point.

3) Input variables beginning with A-H, and O-Z are real and may contain a decimal point.

4) Logical variables are identified in the input description, and have the value .TRUE. or .FALSE..

5) Character variables are identified in the input description, and are enclosed in 'single quotes', except as noted.

6) Character and logical variables must be input in upper case.

7) The input data is read from unit 05, except as noted.

8) The output is printed on unit 06, except as noted.

9) Input cards are identified by a box. If that card is repeated, the entire box is repeated.

input card image.....

The data for one card may be input on one or more lines in the data file, provided that all of the character variables are on the first line.

10) Consistent units are used throughout the program. Thus, input in inches, kips, seconds yields output in inches, inch-kips etc. Mixing units will yield unpredictable results. Units are indicated parenthetically where appropriate.

Example:	Note
'BUG = K'	(1)
'STRUCT'	(2)
. Structure input is omitted.	
'SOL04'	(3)
. Solution input is omitted.	
'READ UNIT = 21'	(4)
'STOP'	(5)

(1) The bug option is set, this prints out the hysteresis model data for each time step.

(2) The structure is defined.

(3) An cyclic static solution of the structure is performed.

(4) Data printed on unit 21 during SOL04 is read from the disk file and printed in a report.

(5) The program is terminated.

A. STRUCTURE - DEFINE THE STRUCTURAL MODEL

These cards define the structural model to be analyzed. The following cards are each input once.

'STRUCT'
TITLE

STRUCT Signifies that the structural model is to be input. Character variable, enclose in single quotes.

TITLE User input title, 80 characters maximum. Character variable, enclose in single quotes.

1. Joints and Degrees of Freedom. These cards are used to define the coordinates of the joints, joint restraints, constraints and the degree of freedom to be condensed out. The degree of freedom numbers are assigned by the program, and printed in the output. The following card is input once.

NJOINT NCOS NSUPT NCOND NCONST SCALE

NJOINT The number of joints defined by the joint coordinate cards.

NCOS The number of joint direction cosine cards input.

NSUPT The number of joint restraint cards input.

NCOND The number of joint condensation cards input.

NCONST The number of joint constraint cards input.

SCALE Scale factor that the joint coordinates are to be multiplied by. If SCALE = 12, the user inputs the joint coordinates in feet, and the structure is defined in inches.

a. Joint Coordinates. These cards are used to define the coordinates of the joints, in the global coordinate system, GCS, and identify the direction cosine of the joint. The total number of joints defined in this section is less than or equal to NJOINT. The second card is only used when the preceding card has a value of IGEN that is greater than zero. These cards are repeated untill 1) NJOINT joints have been defined, or 2) An input or generated joint ID number is less than or equal to zero.

ID X Y Z ICOS IGEN
ΔID ΔX ΔY ΔZ

ID The joint IDentification number. ID numbers can be input in any convenient order and need not be consecutive. However, the band width of

the structural stiffness matrix is dependent on the joint ID numbers. An ID ≤ 0 terminates the input of the joint coordinates.

X The GCS X coordinate of the joint. (length)

Y The GCS Y coordinate of the joint. (length)

Z The GCS Z coordinate of the joint.(length)

ICOS The joint's direction cosine number.

IGEN The number of additional joints to be generated from this joint.

ΔID The increment between the generated ID number and the previous joint's ID number. A generated ID ≤ 0 terminates the input of the joint coordinates.

ΔX The increment between the generated joint's GCS X coordinate and the previous joint's X coordinate. (length)

ΔY The increment between the generated joint's GCS Y coordinate and the previous joint's Y coordinate. (length)

ΔZ The increment between the generated joint's GCS Z coordinate and the previous joint's Z coordinate. (length)

Example: NJOINT = 7

	Note
10 0. 0. 0. 1 2	(1,2)
10 0. 0. 3.	(2)
1 4. 3. -1. 2 0	(3)
-1 0 0 0 0 0	(4)

(1) Joint 10 has the coordinates (0,0,0) and uses direction cosine #1.

(2) Two joints are generated from joint 10: Joint 20 (0,0,3) and Joint 30 (0,0,6).

(3) Joint 1 has the coordinates (4,3,-1) and uses direction cosine #2.

(4) Input of the joint coordinates is terminated.

b. Joint Direction Cosines. These cards are used to input the joint direction cosines, which in turn define the joint coordinate system, JCS. The joint direction cosines are numbered from 1 to NCOS. This card is repeated NCOS times.

Vxi	Vxj	Vxk	Vyi	Vyj	Vyk

Vxi The projection on the GCS X axis of a unit vector parallel to the JCS X axis.

Vxj The projection on the GCS Y axis of a unit vector parallel to the JCS X axis.

Vxk The projection on the GCS Z axis of a unit vector parallel to the JCS X axis.

Vyi The projection on the GCS X axis of a unit vector parallel to the JCS Y axis.

Vyj The projection on the GCS Y axis of a unit vector parallel to the JCS Y axis.

Vyk The projection on the GCS Z axis of a unit vector parallel to the JCS Y axis.

c. Joint Restraints. These cards are used to define the joint restraints. This card is repeated NSUPT times.

ID	ITX	ITY	ITZ	IRX	IRY	IRZ	IGEN	ΔID

ID The joint IDentification number. An ID of zero indicates that all the joints are restrained by this card.

ITX Restraint flag for translation in the JCS X direction.

ITY Restraint flag for translation in the JCS Y direction.

ITZ Restraint flag for translation in the JCS Z direction.

IRX Restraint flag for rotation about the JCS X axis.

IRY Restraint flag for rotation about the JCS Y axis.

IRZ Restraint flag for rotation about the JCS Z axis.

IGEN The number of additional joints, with the same restraints, to be generated from this joint.

ΔID The increment between the generated ID number and the last ID number.

Valid joint restraint flags are:

0 Free or unrestrained degree of freedom.

1 Restrained degree of freedom.

2 Restrained degree of freedom. A restraint flag of 2 forces the program to assign the dof a higher number. This option can be used to reduce the bandwidth of the stiffness matrix.

Example: NSUPT = 2	Note
0 0 0 0 0 0 1 0 0	(1)
1 1 1 1 1 1 1 0 0	(2)

(1) The rotation about the JCS Z axis of all joints is restrained.

(2) Joint 1 has all six dofs restrained.

d. Joint Condensation. These cards are used to identify which dof are condensed out. This card is repeated NCOND times and is omitted if NCOND equals zero.

ID ITX ITY ITZ IRX IRY IRZ IGEN ΔID

ID The joint IDentification number. An ID of zero indicates that all of the joints are affected by this card.

ITX Condensation flag for translation in the JCS X direction.

ITY Condensation flag for translation in the JCS Y direction.

ITZ Condensation flag for translation in the JCS Z direction.

IRX Condensation flag for rotation about the JCS X axis.

IRY Condensation flag for rotation about the JCS Y axis.

IRZ Condensation flag for rotation about the JCS Z axis.

IGEN The number of additional joints, with the same condensation, to be generated from this joint.

ΔID The increment between the generated ID number and the last ID number.

Valid condensation flags are:

0 Degree of freedom is not condensed out.

1 Degree of freedom is condensed out. Condensation of a restrained dof is ignored.

Example: NCOND = 1

30 0 0 0 1 1 1 0 0

The rotations of joint 30 are condensed out. If the Z axis rotation had been previously restrained, only the X and Y axis rotations are condensed out.

e. Joint Constraints. These cards are used to identify which dof are constrained. This card is repeated NCONST times, and omitted if NCONST equals zero.

ITYPE MASTER ISLAVE IGEN ΔID

ITYPE The type of constraint.

0 Rigid body constraint. A rigid body constraint transfers all six joint dof from the slave to the master joint.

1 XY-planar constraint. An XY-planar constraint transfers the joint's JCS X and Y axes translational dof and the joint's JCS Z axis rotational dof from the slave to the master joint.

MASTER The joint IDentification number of the master joint.

ISLAVE The joint IDentification number of the slave joint.

IGEN The number of additional slave joints, constrained to the same master joint to be generated.

ΔID The increment between the generated ID number and the previous ID number.

Note: Both the slave and master joint must have the same joint direction cosine number (ICOS).

Example: NCONST = 1

1 10 20 0 0

Joint 20 is constrained in the JCS XY-plane to Joint 10.

2. Materials and Hysteresis Model Information. These cards are used to input the material and hysteresis model information. The first card is input once. The second card is repeated NMAT times.

NMAT
TYPE VALUE1 VALUE2 ...

NMAT The number of materials input.

TYPE The material type. Valid types are discussed below. Character variable, enclose in single quotes.

VALUEi The input required by a given material type. The values for each TYPE are discussed below. (real or integer)

Notes on material - element compatibility:

1) Individual elements may not use all of the information provided by a given material.

2) A given material may not be compatible with all of the elements. For example the Takeda hysteresis model may not be used with the elastic prismatic beam element. The compatible materials for each element are specified under the element input.

a. TYPE='3D-BEAM'. Material data for an elastic 3D-beam.

'3D-BEAM' E G AX AY AZ J IY IZ

E Young's modulus. (force/length^2)

G Shear modulus. (force/length^2)

AX Cross sectional area. (length^2)

AY Y axis shear area. (length^2)

AZ Z axis shear area. (length^2)

J Torsional moment of inertia. (length^4)

IY Moment of inertia, about the Y axis. (length^4)

IZ Moment of inertia, about the Z axis. (length^4)

b. TYPE='AXLMOD'. AXLMOD axial hysteresis model for reinforced concrete as shown in Figures 2 and 3.

'AXLMOD' Kc Kt Kt2 Fy α β μf βDI

Ks Compression stiffness of a unit length member. (force/length)

Kt Pre-yielding tensile stiffness of a unit length member. (force/length)

Kt2 Post-yielding tensile stiffness of a unit length member. (force/length)

Fy Yield force. (force)

α Unloading coefficient, usually 0.90.

β Pinching coefficient, usually 0.20.

μf Failure Ductility.

βDI Parameter for Ang's damage index discussed in Appendix B (1).

The stiffness terms Kc, Kt, and Kt2 are real variables and may contain a decimal point.

c. TYPE='BEND1'. Material data for the B1 hysteresis model. The B1 hysteresis model was developed to model the bending behavior of reinforced concrete shear walls and is shown in Figure 4. The input for the backbone curve is shown in Figure 5.

'BEND1' NSEG NI DY βDI
P(1) P(2) ... P(NSEG)
D(1) D(2) ... D(NSEG)

NSEG Number of points on the backbone curve.

NI Number of small amplitude loop reversal points that can be stored by the program at one time, usually 10.

DY Rotation of a unit length member, corresponding the yield point. Used to define the ductility ratio. (radian/length)

βDI Parameter for Ang's damage index discussed in Appendix B (1).

P(i) The moment of a point on the backbone curve. (force*length)

D(i) The rotation of a point on the backbone curve. Where the rotation is based on a unit length member. (radian/length)

d. TYPE='SHEAR1'. Material data for the S1 hysteresis model. The S1 hysteresis model was developed to model the shear deformation of reinforced concrete shear walls and is shown in Figure 6. The input for the backbone curve is shown in Figure 5.

'SHEAR1' NSEG NI DY βDI
P(1) P(2) ... P(NSEG)
D(1) D(2) ... D(NSEG)

NSEG Number of points on the backbone curve.

NI Number of small amplitude loop reversal points that can be stored by the program at one time, usually 10.

DY Shear deformation of a unit length member, corresponding the yield point. Used to define the ductility ratio. (length/length)

βDI Parameter for Ang's damage index discussed in Appendix B (1).

P(i) The shear of a point on the backbone curve. (force)

D(i) The shear deformation of a point on the backbone curve, where the shear deformation is based on a unit length member. (length/length)

e. TYPE='TAKEDA'. Material data for the TAKEDA hysteresis model. The TAKEDA hysteresis model was developed to model the bending deformations in reinforced concrete members and is shown in Figure 7.

'TAKEDA' PC DC PY DY PU DU βDI

PC Cracking moment. (force*length)

DC Cracking rotation, for a unit length member. (radian/length)

PY Yield moment. (force*length)

DY Yield rotation, for a unit length member. (radian/length)

PU Ultimate moment. (force*length)

DU Ultimate rotation, for a unit length member. (radian/length)

βDI Parameter for Ang's damage index discussed in Appendix B (1).

f. TYPE='BILINEAR'. The bilinear hysteresis model sketched in Figure 8.

'ELSPLS' E PY SIE μf βDI

E Elastic stiffness. (force/length)

PY Yield load. (force)

SIE Inelastic stiffness. (force/length)

μf Failure Ductility.

βDI Parameter for Ang's damage index discussed in Appendix B (1).

3. Geometric Stiffness Data. This card is used to determine the type of geometric stiffness used in the analysis. This card is input once.

KGLOAD KGTYPE KGFORM KGCOND

KGLOAD The type of axial force used to calculate the geometric stiffness.

- 0 The geometric stiffness is omitted.
- 1 The axial force is equal to the input force, magnified by the ground acceleration in the global Z direction, if applicable.
- 2 The internal element force of the previous load step is use to generate the geometric stiffness.

KGTYPE The type of geometric stiffness formulation.

- 0 The geometric stiffness is omitted.
- 1 A 'lumped parameter' formulation is used for the element geometric stiffness.

2 A 'consistent parameter' formulation is used for the element geometric stiffness. If a consistent parameter formulation for an individual element is not available, the lumped parameter formulation is used. Refer to individual element specifications for applicability.

KGFORM Form of the geometric stiffness used.

0 The geometric stiffness is omitted.

1 The geometric stiffness is subtracted from the structural stiffness.

2 Separate structural stiffness and geometric stiffness matrices are formed.

Refer to individual solution for applicability.

KGCOND A logical flag. If KGCOND=.TRUE., the geometric stiffness matrix is condensed when applicable. Logical variable.

4. Element Data. These cards are used to define the elements. The elements are numbered by the program in the order they are input from 1 to NELMT. The element numbers are used to identify the elements in the output. The first card is input once. The second and third card (if used) are repeated untill 1) NELMT elements are input, or 2) a TYPE='END' is encountered. The third card follows each second card with a value of IGEN > 1.

NELMT	
TYPE NAME VALUE1 VALUE2 ...	IGEN
ΔVALUE1 ΔVALUE2 ...	

NELMT The number of elements input.

TYPE The element type. Valid types are discussed below. Character variable, enclose in single quotes.

NAME A user defined name. Character variable, enclose in single quotes.

VALUEi The input required by a given element type. The values for each TYPE are discussed below. (real or integer)

IGEN The number of additional elements to be generated from this element.

ΔVALUEi The incremental value used to generate subsequent elements.

$$VALUEi_{generated} = VALUEi_{previous} + \Delta VALUEi$$

a. TYPE = '3D-BEAM'. This card is used to define the element data for the elastic prismatic 3D-beam shown in Figure 9.

'3D-BEAM' NAME MAT JOINTI JOINTJ V1 V2 V3 XS XE PKG IRELT IGEN

NAME A user defined name. Character variable, enclose in single quotes.

MAT Material number. The material number must correspond to the material type '3D-BEAM'. The terms EAX, GJ, EIY and EIZ from material '3D-BEAM' are used to generate the element stiffness.

JOINTI Start joint ID number.

JOINTJ End Joint ID number.

V1 The projection on the GCS X axis of a vector in the element's local XY-plane. This vector defines the orientation of the element's local Y-axis.

V2 The projection on the GCS Y axis of a vector in the element's local XY-plane.

V3 The projection on the GCS Z axis of a vector in the element's local XY-plane.

XS The off set distance from the start joint to the beginning of the element. Positive in the direction of the element's local X-axis. (length)

XE The off set distance from the end joint to the end of the element. Negative in the direction of the element's local X-axis. (length)

PKG Axial load used to calculate the geometric stiffness, if KGLOAD = 1. (force) Positive is compression.

IRELT A six digit release code that is used to release the rotational dof at the ends of the element. A nonzero value of the i'th digit signifies a released dof.

DIGIT

1 Releases the moment about the element's X axis at the start joint.

2 Releases the moment about the element's Y axis at the start joint.

3 Releases the moment about the element's Z axis at the start joint.

4 Releases the moment about the element's X axis at the end joint.

5 Releases the moment about the element's Y axis at the end joint.

6 Releases the moment about the element's Z axis at the end joint.

IGEN Element generation parameter. See discussion under 'Element Data'.

b. TYPE = 'SPRING'. This card is used to define the element data for the one dimensional spring shown in Figure 10.

'SPRING' NAME MAT JOINTI JOINTJ KTYPE XLEN V1 V2 V3 XS XE IGEN

NAME A user defined name. Character variable, enclose in single quotes.

MAT Material number.

JOINTI Start joint ID number.

JOINTJ End joint ID number.

KTYPE Type of spring.

1 Axial spring.

2 Shear spring in the element's local Y axis.

3 Shear spring in the element's local Z axis.

4 Torsional spring.

5 Rotational spring about the element's local Y axis.

6 Rotational spring about the element's local Z axis.

XLEN Length of the spring used to calculate the stiffness. If XLEN is zero the length between the start and end joints less XS and XE is used to formulate the stiffness. (length)

V1 The projection on the GCS X axis of a vector in the springs local XY-plane. This vector defines the orientation of the element's local Y-axis.

V2 The projection on the GCS Y axis of a vector in the springs local XY-plane.

V3 The projection on the GCS Z axis of a vector in the springs local XY-plane.

XS The off set distance from the start joint to the beginning of the spring. Positive in the direction of the element's local X axis. (length)

XE The off set distance from the end joint to the end of the spring. Negative in the direction of the element's local X axis. (length)

IGEN Element generation parameter. See discussion under 'Element Data'.

Notes:

1) The material specified for the spring element may consist of any of the following: 3D-BEAM, AXLMOD, BEND1, SHEAR1, TAKEDA, and ELSPLS.

2) The spring uses the axial stiffness from the 3D-BEAM material.

3) The BEND1 and TAKEDA models are usually based on moment-rotation, (KTYPE=4 to KTYPE=6) but they can be used as translational springs (KTYPE=1 to KTYPE=3) by changing the material input from moment-rotation to force-translation.

4) The AXLMOD and SHEAR1 models are usually based on force-translation, and are valid models for KTYPE = 1 to KTYPE = 3.

5) The program does not check the user's choice of model.

6) If the distance between the start and end joints is zero, the spring is orientated such that the ECS is parallel to the start joints JCS.

c. TYPE = 'SHEAR WALL'. This card is used to define the element data for the reinforced concrete shear wall shown in Figure 14.

'SHEAR WALL' NAME MATB MATS MATA J1 J2 J3 J4 ALPHA PKG IGEN

NAME A user defined name. Character variable, enclose in single quotes.

MATB Bending hysteresis model material number.

MATS Shear hysteresis model material number.

MATA Axial hysteresis model material number.

J1 Joint at the upper right hand corner of the element.

J2 Joint at the upper left hand corner of the element.

J3 Joint at the lower left hand corner of the element.

J4 Joint at the lower right hand corner of the element.

ALPHA Fraction of the length from the top of the element to the internal springs.

PKG Axial load used to calculate the geometric stiffness, if KGLOAD = 1. Positive is compression. (force)

IGEN Element generation parameter. See discussion under 'Element Data'.

Notes:

1) The material specified for the shear wall element may consist of any of the following: 3D-BEAM, AXLMOD, BEND1, SHEAR1, TAKEDA, and ELSPLS.

2) The BEND1 model is usually specified for the bending material.

3) The SHEAR1 model is usually specified for the shear material.

4) The AXLMOD model is usually specified for the axial material.

5) The 3D-BEAM material's axial stiffness is used for the spring's stiffness if the 3D-BEAM material is specified.

Example:	Note
5	(1)
'3D-BEAM' 'A1' 1 10 20 0. 0. 1. 0. 0. 0. 100000 2	(2, 3)
0 10 10 0. 0. 0. 0. 0. 0. 000000	(3)
'SPRING' 's1' 2 10 11 1 0. 0. 0. 1. 0. 0. 0	(4)
'SHEAR WALL' 'W1' 3 4 5 20 10 1 2 1. 35. 0	(5)

(1) Five elements are to be input.

(2) A 3D-beam is input between joints 10 and 20.

(3) Two more 3D-beams are generated from the first beam.

(4) An axial spring is input between joints 10 and 11.

(5) A shear wall element is input between joints 20, 10, 1 and 2.

5. Mass. These cards are used to input lumped masses at the joints. The first card is input once. The second card is repeated INMASS times, or until a joint ID ≤ 0 is encountered. If INMASS is less than one, or FMASS is zero, omit the second card.

INMASS FMASS MCOND
ID PX PY PX RXX RYY RZZ RXY RXZ RYZ IGEN ΔID

INMASS The number of mass cards to be read.

FMASS Mass flag.

0 The mass matrix is omitted.

1 The mass matrix due to concentrated joint masses is formed.

MCOND A logical flag. If MCOND = .TRUE., the mass matrix is condensed when applicable. Logical variable.

ID IDentification number of the joint.

PX Translational mass in the joint's JCS X direction. (mass)

PY Translational mass in the joint's JCS Y direction. (mass)

PZ Translational mass in the joint's JCS Z direction. (mass)

RXX Rotational mass moment of inertia about the joint's JCS X axis. (mass*length^2)

RYY Rotational mass moment of inertia about the joint's JCS Y axis. (mass*length^2)

RZZ Rotational mass moment of inertia about the joint's JCS Z axis. (mass*length^2)

RXY Rotational mass product of inertia about the joint's JCS XY axis. (mass*length^2)

RXZ Rotational mass product of inertia about the joint's JCS XZ axis. (mass*length^2)

RYZ Rotational mass product of inertia about the joint's JCS YZ axis. (mass*length^2)

IGEN Number of joints with identical mass, to be generated.

ΔID Increment of joint ID number for generated values.

Example: INMASS = 1, FMASS = 1,

10 8. 8. 8. 0 0 0 0 0 0 0 0

Joint 10 has a translational mass of 8.0 in the X, Y and Z directions.

6. Damping. This card is used to input proportional damping data.

ALPHA BETA

ALPHA Proportional damping coefficient for mass.

BETA Proportional damping coefficient for stiffness.

B. SOL01 - ELASTIC STATIC SOLUTION

The following solution is an elastic solution with multiple load cases. The following cards are input once.

'SOL01'
TITLE
NLOAD MAXELD

SOL01 Signifies solution #1. Character variable, enclose in single quotes.

TITLE User input title, 80 characters maximum. Character variable, enclose in single quotes.

NLOAD Number of load cases.

MAXELD Maximum number of element loads.

Notes:

1) Condensation increases the band width of the stiffness matrix for SOL01 without any other benefits. Condensation is not recommended for SOL01.
2) A separate geometric stiffness (KGFORM = 2) is not used by SOL01. The geometric stiffness may be included by using KGFORM = 1.
3) If the geometric stiffness is included, the element axial loads must be input, KGLOAD = 1.

1. Joint Loads. These cards are used to apply loads to joints. Loads applied to restrained joints are considered as displacements, yielding, support settlement or displacement control solutions. Loads applied to constrained joints are transferred to their 'master' joints. If the 'master' joint is restrained, loads transferred to restrained dof are considered as displacements. Joint loads are additive; applying two loads to the same joint results in the sum of the joint loads being considered. The following card is repeated untill the value of DIR is 'END'.

LOAD ID IGEN ΔID DIR VALUE

LOAD Number of the load case that the load is applied to.

ID Joint ID number that the load is applied to.

IGEN Number of identical joint loads to be generated.

ΔID Increment of joint ID number for generated values.

DIR Direction of load, in the joint coordinate system. Valid directions and the units of VALUE are given below. Character variable, enclose in single quotes.

- 'FX' Applied force in the joint's JCS X direction. (force)
- 'FY' Applied force in the joint's JCS Y direction. (force)
- 'FZ' Applied force in the joint's JCS Z direction. (force)
- 'MX' Applied moment about the joint's JCS X axis. (force*length)
- 'MY' Applied moment about the joint's JCS Y axis. (force*length)
- 'MZ' Applied moment about the joint's JCS Z axis. (force*length)
- 'END' Terminate the input of element loads.

VALUE Magnitude of the applied load.

Example:	Note
1 2 2 1 'FZ' -3.00	(1)
2 7 0 0 'MX' 33.5	(2)
0 0 0 0 'END' 0	(3)

(1) Joints 2, 3, and 4 have an applied force of -3.00 in the Z direction, load case 1.

(2) Joints 7 has an applied moment of 33.50 in the X direction, load case 2.

(3) Joint loading input is terminated.

2. Element Loads. These cards are used to apply element loads to the '3D-BEAM' element. The loads are applied to the portion of the beam between points A and B of Figure 9, in the ECS. The loads are transferred by the program to the start and end joints. The following card is only included if MAXELD > 0. The card is repeated MAXELD times, or until the value of TYPE is 'END'.

LOAD IELE IGEN ΔIELE TYPE DIR VALUE1, VALUE2, ...

LOAD Load case number.

IELE Element number.

IGEN Number of similar element loads to be generated.

ΔIELE Increment of element number for generated values.

TYPE Type of load. Valid types are described in detail below. Character variable, enclose in single quotes.

'CONC' Concentrated load applied in direction DIR. VALUE1 is the magnitude of the load (force or force*length). VALUE2 is the ratio of the distance to the load, divided by the flexible length of the member. The distance is measured from the beginning of the

flexible length at the member's start end. VALUE2 is between 0 and 1. Only two values are input.

'UNIF' Uniform load applied in direction DIR. VALUE1 is the magnitude of the load. Only one value is input. (force/length)

'FEM' Input the fixed end forces on the end of the member. DIR is not used and may be set to any value. VALUE1 to VALUE12 are required.

VALUE1 Fixed end axial force, at point A of Figure 9, in the ECS X direction. (force)

VALUE2 Fixed end shear, at point A of Figure 9, in the ECS Y direction. (force)

VALUE3 Fixed end shear, at point A of Figure 9, in the ECS Z direction. (force)

VALUE4 Fixed end torsion, at point A of Figure 9, about the ECS X axis. (force*length)

VALUE5 Fixed end moment, at point A of Figure 9, about the ECS Y axis. (force*length)

VALUE6 Fixed end moment, at point A of Figure 9, about the ECS Z axis. (force*length)

VALUE7 Fixed end axial force, at point B of Figure 9, in the ECS X direction. (force)

VALUE8 Fixed end shear, at point B of Figure 9, in the ECS Y direction. (force)

VALUE9 Fixed end shear, at point B of Figure 9, in the ECS Z direction. (force)

VALUE10 Fixed end torsion, at point B of Figure 9, about the ECS X axis. (force*length)

VALUE11 Fixed end moment, at point B of Figure 9, about the ECS Y axis. (force*length)

VALUE12 Fixed end moment, at point B of Figure 9, about the ECS Z axis. (force*length)

'END' Terminate the input of element loads.

DIR Direction of load, in element coordinate system. Valid directions given below. Character variable, enclose in single quotes.

'FX' Axial load is applied.

'FY' Force is applied in the local element's Y direction.

'FZ' Force is applied in the local element's Z direction.

'MX' Torque is applied.

'MY' Moment is applied about the local element's Y axis.

'MZ' Moment is applied about the local element's Z axis.

VALUEi Values used to calculate the element loads.

Notes:

1) Multiple loads may be put on a single element.

2) 'UNIF' 'MY' and 'UNIF' 'MZ' are not available.

3) Element load TYPE = 'FEM' is not modified to reflect member end releases.

Example: MAXELD = 1,

1 13 0 0 'UNIF' 'FZ' -3.00

Element 13 has a uniform load of -3.00 applied in the element's local Z direction, for load case 1.

C. SOL02 - DYNAMIC SOLUTION BY NUMERICAL INTEGRATION

This is a dynamic solution of the structure subject to ground accelerations. Both elastic and nonlinear behavior can be modeled. The following cards are input once.

'SOL02'
TITLE
INTEG THETA ELASTIC UNBAL
IPRINT IWRITE MAXACC SLMASS
TO ΔT TF GRAV

SOL02 Signifies solution #2. Character variable, enclose in single quotes.

TITLE User input title, 80 characters maximum. Character variable, enclose in single quotes.

INTEG Type of numerical integration used. Valid values of INTEG are given below. Character variable, enclose in single quotes.

'LINEAR' The linear acceleration method is used.

'AVERAGE' The average acceleration method is used.

THETA Not used, set THETA = 0.

ELASTIC A logical flag. If ELASTIC = .TRUE., the structure is assumed to behave elastically. Logical variable.

UNBAL A logical flag. If UNBAL = .TRUE., the unbalanced loads from the preceding step are added to the current dynamic loads for nonlinear analysis. Logical variable.

IPRINT Step increment for printed output.

IWRITE Step increment for data written to output files.

MAXACC The maximum number of points in each ground acceleration record.

SLMASS A logical flag. If SLMASS = .TRUE., a special mass matrix is input. This matrix is used, with the ground acceleration to generate loads. Logical variable.

TO Initial time at the beginning of the solution. (time)

ΔT Time step. (time) .

TF Final time at the end of the solution. (time)

GRAV Gravitational acceleration constant. (length/sec^2)

1. Special Loading Mass. The special loading mass matrix discussed in step 1 of Section C in Chapter II is input by these cards. If the special loading mass is input the dynamic joint loads are the ground acceleration times the special loading mass. If the special loading mass is not input, the dynamic joint loads are the ground acceleration times the structural mass. The first card is input once. The second card is repeated INMASS times, or until an ID less than one is encountered. If INMASS is less than one, or FMASS is zero, omit the second card. Omit both cards if SLMASS = .FALSE..

INMASS FMASS
ID PX PY PX RXX RYY RZZ RXY RXZ RYZ IGEN ΔID

These cards are identical to the 'Mass' cards in the block STRUCT. Refer to STRUCT for a detailed description.

2. Output Data to Disk Files. These cards control the data that is written to separate disk files. This data is used to print reports and plot data. The card is repeated until the value of TYPE is 'END'.

TYPE NUMB IUNIT IGEN ΔNUMB ΔIUNIT

TYPE Type of data to be written to output file. Valid types are given below. Character variable, enclose in single quotes.

- 'DOF' Data is printed for a degree of freedom. The degree of freedom ID number, as assigned by the program, is used.
- 'JOINT FX' Data is printed for the degree of freedom corresponding to the joint's JCS X translation.
- 'JOINT FY' Data is printed for the degree of freedom corresponding to the joint's JCS Y translation.
- 'JOINT FZ' Data is printed for the degree of freedom corresponding to the joint's JCS Z translation.
- 'JOINT MX' Data is printed for the degree of freedom corresponding to the joint's JCS X rotation.
- 'JOINT MY' Data is printed for the degree of freedom corresponding to the joint's JCS Y rotation.
- 'JOINT MZ' Data is printed for the degree of freedom corresponding to the joint's JCS Z rotation.
- 'ELE' Element data is printed.
- 'ENERGY' The structures energy balance is printed.
- 'SUMRCT' The structures summation of reactions is printed.
- 'GROUND' The ground response is printed.
- 'END' Terminate the input of element loads.

NUMB Element, dof or joint number. Set NUMB = 0 if TYPE = 'ENERGY'.

IUNIT Output file unit number.

IGEN Number of similar data groups to be printed.

ΔNUMB Incremental element dof or joint number for generation.

ΔIUNIT Incremental output file unit number for generation.

Example:	Note
'DOF' 5 10 0 0 0	(1)
'ELE' 32 11 1 1 1	(2)
'END' 0 0 0 0 0	(3)

(1) Degree of freedom #5 data is written to file 10.

(2) Element 32's data is written to file 11, and element 33's data is written to file 12.

(3) Output requests are terminated.

3. Initial Displacements, Velocities, Accelerations and Loads. This card is used to define initial conditions for dynamic analysis. This card is repeated until OPTION has a value of 'END'.

ID IGEN ΔID OPTION V1 V2 V3 V4 V5 V6

ID Joint ID number.

IGEN Number of joints with identical initial conditions.

ΔID Increment of the joint ID number for generation.

OPTION Type of initial condition. Valid options are listed below. Character variable, enclose in single quotes.

'DISPL' Initial displacements are given. V1, V2 and V3 are displacements (length), while V4, V5 and V6 are rotations (radians).

'VEL' Initial velocities are given. V1, V2 and V3 are translational velocities (length/time), while V4, V5 and V6 are rotational velocities (radians/time).

'ACC' Initial accelerations are given. V1, V2 and V3 are translational accelerations (length/time^2) , while V4, V5 and V6 are rotational accelerations (radians/time^2).

'LOA' Initial loads are given. V1, V2 and V3 are loads (force), while V4, V5 and V6 are moments (force*length).

V1 Initial value of translation or force in the joint's JCS X direction.

V2 Initial value of translation or force in the joint's JCS Y direction.

V3 Initial value of translation or force in the joint's JCS Z direction.

V4 Initial value of rotation or moment about the joint's JCS X axis.

V5 Initial value of rotation or moment about the joint's JCS Y axis.

V6 Initial value of rotation or moment about the joint's JCS Z axis.

Note:

1) If the acceleration at time TO is nonzero, initial conditions must be specified to ensure equilibrium.

2) The initial acceleration may be set equal to zero by one of the following methods:

   a) The initial time at the beginning of the solution is equal to the initial time of the acceleration record, (TO = TOG), set A(1) = 0.

   a) The initial time at the beginning of the solution is less than the initial time of the acceleration record, (TO<TOG), set A(1) ≠ 0.

3) If the dynamic solution, SOL02, follows a static solution, SOL01, then the resulting displacements and element forces are automatically included as the initial conditions for the dynamic solution. The results from the static solution may be excluded from the dynamic solution by releasing the memory between solutions.

Example:	Note
2 2 1 'ACC' 3. 0 0 0 0 0	(1)
0 0 0 'END' 0 0 0 0 0 0	(2)

(1) Joints 2, 3 and 4 have an initial acceleration of 3.0 in the X direction.

(2) Initial conditions are terminated.

4. Ground Acceleration Record. These cards are used to input the ground acceleration record, and its orientation. The following cards are input once.

NA ASCALE TOG ΔTG PRINT
Vxi Vzj Vxk Vyi Vyj Vyk

NA Number of components of ground acceleration input. $1 \leq NA \leq 3$

ASCALE Ground acceleration amplitude scale factor.

TOG Time at the first point of the ground acceleration. (time)

ΔTG Time increment for the input ground acceleration. (time)

PRINT A logical flag. If PRINT=.TRUE. then the 3D ground accelerations are printed.

Vxi The GCS X axis projection of a unit vector defining the X′ axis of the input ground acceleration.

Vxj The GCS Y axis projection of a unit vector defining the X′ axis of the input ground acceleration.

Vxk The GCS Z axis projection of a unit vector defining the X′ axis of the input ground acceleration.

Vyi The GCS X axis projection of a unit vector defining the Y′ axis of the input ground acceleration.

Vyj The GCS Y axis projection of a unit vector defining the Y′ axis of the input ground acceleration.

Vyk The GCS Z axis projection of a unit vector defining the Y′ axis of the input ground acceleration.

The following set of cards are repeated NA times. The two FORMAT cards are only used if the value of FMT=.TRUE.. The two ATITLE cards and A() are input on unit=IN.

IN NPTS IDIR FMT ECHO REWIND
FORMAT1
FORMAT2
ATITLE2
ATITLE2
A(1) A(2) A(3) ... ... A(NPTS)

IN Input unit number for acceleration record.

NPTS Number of acceleration points to be read. NPTS $\leq$ MAXACC

IDIR Direction of acceleration being read.

1 Ground acceleration is in the X′ direction.

2 Ground acceleration is in the Y′ direction.

3 Ground acceleration is in the Z′ direction.

FMT A logical flag. If FMT=.TRUE. then the input acceleration is formatted and the corresponding Fortran format codes, FORMAT1 and FORMAT2, must be input. Logical variable.

ECHO A logical Flag. If ECHO=.TRUE. then the input acceleration is printed. Logical variable.

REWIND A logical Flag. If REWIND=.TRUE. then the acceleration input file is rewound before it is read. This allows the same acceleration input file to be read more than once. IF IN=5 then set REWIND=.FALSE.. Logical variable.

FORMAT1 Fortran format code capable of reading the two 80-character titles. (Character variable, do not enclose in single quotes)

FORMAT2 Fortran format code capable of reading the input ground accelerations. (Character variable, do not enclose in single quotes)

ATITLE1 Ground acceleration title. Read from unit IN. (Character variable, do not enclose in single quotes)

ATITLE2 Ground acceleration title. Read from unit IN. (Character variable, do not enclose in single quotes)

A() Input ground acceleration. If FMT=.FALSE., A() must be separated by blank spaces or commas. Otherwise, A( ) is input by the fixed format code FORMAT2. Read from unit IN. (g's)

Example:	Note
2 1.0 0.0 0.01 .FALSE.	(1)
1 0 0 0 1 0	(2)
5 5 1 .TRUE. .FALSE.	(3)
(A/A)	(4)
(8F9.6)	(5)
ELCENTRO- 1940 EQ DATA FORMAT: (8F9.6)	(6)
DT=0.01 SEC + + + E-W DIRECTION + + +	(6)
-0.011848-0.008267-0.004687-0.002477-0.004856	(7)
5 5 2 .FALSE. .FALSE.	(8)
ELCENTRO- 1940 EQ DATA	(9)
DT=0.01 SEC + + + N-S DIRECTION + + +	(9)
0.010966 0.008610	(10)
0.006255	(10)
0.003900 .001545	(10)

(1) Two ground motions are to be input.

(2) Vectors defining the X' and Y' axes are input.

(3) Input the first ground motion in fixed format from unit 5.

(4) Format code for titles.

(5) Format code for acceleration data.

(6) Titles.

(7) Acceleration data.

(8) Input the second ground motion in free format from unit 5.

(9) Titles.

(10) Acceleration data.

D. SOL03 - EIGENVALUE SOLUTION

The following solution is used to calculate the elastic natural frequency and mode shape or the buckling load and mode shape. The following cards are input once.

'SOL03'
TITLE
OPTION IPRINT IPJT

SOL03 Signifies solution #3. Character variable, enclose in single quotes.

TITLE User input title, 80 characters maximum. Character variable, enclose in single quotes.

OPTION Choice of natural frequency or buckling load. Character variable, enclose in single quotes.

'FREQ' Solves for the natural frequency and mode shapes of the structure.

'BUCK' Solves for the buckling load and mode shapes of the structure.

IPRINT Number of modes to be printed.

IPJT Number of modes to be printed with detailed mode shape.

E. SOL04 - INCREMENTAL STATIC SOLUTION

The following solution is used to calculate the static cyclic response of nonlinear structures. The following cards are input once.

'SOL04'
TITLE
MAXELD IPRINT IWRITE UNBAL

SOL04 Signifies solution #4. Character variable, enclose in single quotes.

TITLE User input title, 80 characters maximum. Character variable, enclose in single quotes.

MAXELD Maximum number of element loads.

IPRINT Step increment for printed output.

IWRITE Step increment for data written to output files.

UNBAL A logical flag. If UNBAL=.TRUE., the unbalanced loads from the preceding step is added to the current load. Logical variable.

Note: The load case for this solution is always input as one.

1. Output Data to Disk Files. These cards control the data that is written to separate disk files. This data is used to print reports, or plot data. These cards are identical to the 'Output Data to Disk File' cards in block SOL02. Refer to SOL02 for a detailed description. The cards are repeated until the value of TYPE is 'END'.

TYPE NUMB IUNIT IGEN ΔNUMB ΔIUNIT

2. Joint Loads. These cards are used to apply loads to joints. These cards are identical to the 'Joint Load' cards in block SOL01. Refer to SOL01 for a detailed description. The following card is repeated untill the value of DIR is 'END'.

LOAD ID IGEN ΔID DIR VALUE

3. Element Loads. These cards are used to apply element loads to the '3D-BEAM' element. These cards are identical to the 'Element Load' cards in block SOL01. Refer to SOL01 for a detailed description. The following card is only included if MAXELD > 0. The card is repeated MAXELD times, or until the value of TYPE is 'END'.

LOAD IELE IGEN ΔIELE TYPE DIR VALUE1, VALUE2, ...

4. Load Factors. These cards contain the load factors that are used to generate incremental static loads. Two options are available to limit the size of the load step; 1) the results of each step are scaled such that joint loads and displacements on free dof are less than PMAX and DMAX, and the step is repeated until the value of FACTOR is achieved (loading option A), or 2) each load step is subdivided into N steps (loading option B). The card is repeated until the value of STEP is 'END'.

STEP FACTOR PMAX DMAX N

STEP User input step name, 80 characters maximum. Character variable, enclose in single quotes. If STEP is 'END', the current solution is terminated.

FACTOR Load factor. The applied load at the end of the step is FACTOR * (applied joint and element loads).

PMAX Maximum value for a free dof joint load, per load step. (force or force*length)

DMAX Maximum value for a free dof joint displacement, per load step. (length or radians)

N Number of load steps between the previous and current factor. N is only used if both PMAX and DMAX are equal to zero.

Example:	Note
'LOAD' 10.0 25. .25 0	(1)
'UNLOAD' 0.0 0. .0 20	(2)
'END' 0 0. .0 0	(3)

(1) The structure is loaded to 10 times the joint and element loads, in steps that do not have load increments greater than 25, or displacement increments greater than 0.25.

(2) The structure is then unloaded with 20 equal size load steps.

(3) SOLN04 is terminated.

F. BUG - SET BUG OPTIONS

This card is used to set the bug options, which print out the intermediate results listed below. The entire statement is a character variable, and is enclosed in single quotes.

'BUG = options'

Option	Description
A	Print element displacements. Print loads applied to degrees of freedom
B	Not used.

C Print plot data for program SEE to unit 07. SEE is a program that plots the structure on the UMR CALCOMP plotter. A listing of program SEE is given in Appendix C.

D Print joint, element, and dynamic loading data.

E Print numerical integration data for linear and average acceleration methods.

F Print the element's structural and geometric stiffness.

Print the global mass, structural stiffness, geometric stiffness, loads and displacements.

G Print the condensed global mass, structural stiffness and geometric stiffness matrices.

H Print the element transformations, structural and geometric stiffness, etc.

I Print contents of memory for elements.

J Print a skyline map for matrices.

K Print the material data for each load step.

L Print the energy balance.

M Print the skyline data for matrices.

N Print the contents of memory when DUMP is called.

Notes:

1) Any number of options may be specified at one time.

2) Options specified with the last bug statement are the only options active.

G. READ - READ OUTPUT FILES

This card is used to read and print data written to output files during SOL02 and SOL04. The entire statement is a character variable, and is enclosed in single quotes.

'READ INC=I UNIT=NO'

Where NO is the unit number of the file that contains the data, and I is the increment of the steps printed out. Multiple UNIT=NO statements may exist on each read card.

H. NOECHO - INHIBIT INPUT ECHO

This card is used to inhibit the input echo. Character variable, enclose in single quotes.

'NOECHO'

I. DUMP - PRINT MEMORY

This card is used to print the addresses of the data in memory. If 'BUG=N' was previously specified, 'DUMP' also prints the nonzero values in the linear array. Character variable, enclose in single quotes.

'DUMP'

J. RELEASE - RELEASE MEMORY

This card is used to release or 'free up' memory used for previous solutions. Global displacements, velocities etc. are reset to zero. The entire statement is a character variable, and is enclosed in single quotes.

'RELEASE OPTION'

If OPTION='ELEMENT', the element forces, displacements and hysteresis models are also reset to their initial values.

Example:	Note
'STRUCT'	(1)
. Structure input is omitted.	
'SOL01'	(2)
. Solution input is omitted.	
'RELEASE'	(3)
'SOL03'	(4)
. Solution input is omitted.	
'STOP'	(5)

(1) The structure is defined.

(2) A static solution is performed.

(3) The static solution is released. The memory required for load, displacement, stiffness, etc. is released. The element forces are not released because the ELEMENT statement was omitted from the RELEASE card.

(4) Natural frequencies or buckling loads are determined. For a buckling load solution, the geometric stiffness may be based on the axial load in the elements from SOL01. Releasing the memory after the static solution allows the same memory to be used for SOL03. If the memory had not been released, then the total memory required would be the sum of the memory required for SOL01 and SOL03.

(5) Terminate the program.

K. STOP - TERMINATE EXECUTION

This card is used to terminate execution of the program. The statement is a character variable, and is enclosed in single quotes.

'STOP'

V. RUNNING PROGRAM INRESB-3D-SUP

This chapter contains the instructions for running program INRESB-3D-SUP in the VM/SP CMS environment on an IBM 4381 computer. The instructions consist of several steps. First a macro library is generated that contains the common blocks. Second, the main program and of the each subroutines are compiled. The text files are stored on a disk in the users account. Third, the program is executed. The first and second steps are only done once to install the program.

A. GENERATING THE COMMON BLOCK MACRO LIBRARY

The common block macro library, GEM, is generated with the following CMS statements:

```
MACLIB GEN ZCOMN  GEM
MACLIB ADD ZCOMN1 GEM
MACLIB ADD ZCOMN2 GEM
MACLIB ADD ZCOMN3 GEM
```

where ZCOMN COPY, ZCOMN1 COPY, ZCOMN2 COPY and ZCOMN3 COPY are four CMS files that contain the common blocks. These files are listed with the program.

B. COMPILING THE PROGRAM

The program and subroutines are compiled with the following CMS commands:

```
GLOBAL MACLIB GEM
FORTVS2  INRESB (OPT(2)  AUTODBL(DBLPAD4)
```

The GLOBAL MACLIB GEM statement lets the compiler know where to locate the common blocks. The common blocks are put into the program by the Fortran INCLUDE statement, which is in the source code.

The compiler option OPT(2) specifies code optimization to reduce execution time. Vectorization may also be used to reduce execution time if the hardware and software are available.

The accuracy of computers differs according to their word length. Hence the program is written in single precision, and compiler options are used to convert the program to double precision when required. The IBM compiler option AUTODBL(DBLPAD4) automatically converts the program to double precision. The program has integer and real variables that share the same storage location. Thus the compiler option that is used to convert the program to double precision must pad the length of the integer variables, such that integer variables and real variables have the same length. Failure to use the DBLPAD4 compiler option an equivalent compiler option on other systems will lead to unpredictable results.

C. RUNNING THE PROGRAM

The program is run in the local CMS environment by the following commands:

```
GLOBAL TXTLIB VSF2LINK VSF2FORT CMSLIB IMSLDOUB UFORTLIB
FILEDEF  5 DISK   FN   FT05 A ( PERM LRECL  80  RECFM F
FILEDEF  6 DISK   FN   FT06 A ( PERM LRECL 132  RECFM F
FILEDEF 10 DISK   FN   FT10 A ( PERM LRECL  80  RECFM F
FILEDEF 11 DISK   FN   FT11 A ( PERM LRECL  80  RECFM F
    .
    .
    .
FILEDEF 92  DISK ELCENTRO EW    A
FILEDEF 93  DISK ELCENTRO NS    A
LOAD  INRESB ( CLEAR RESET  MAIN START  NOMAP
```

The GLOBAL TXTLIB command loads the Fortran libraries needed to run the program. The IMSL library is required to link IMSL subroutine EIGZS. The UFORTLIB library contains the local routines CPUTIM, TIMEON, TIMEIT, TIME and DATE. The input is contained in a files FN FT05 A, ELCENTRO EW A and

ELCENTRO NS A. The output is written to file FN FT06 A and the data files FN FT10 A, FN FT11 A, etc.

VI. EXAMPLE PROBLEMS

A. CANTILEVER BEAM

1. Description of Input Information. Thirteen joints and twelve beam segments are used to define the cantilever beam as shown in Figure 22(a). Each joint has a lumped mass of 1.0 kip – sec^2/in. Joint loads of -3 kips and -1 kip are applied at the end of the cantilever in the global Z and X directions, respectively. A uniform load of .08333 k/in is applied to the cantilever in the global Y direction as shown in Figure 22(b). The internal forces are shown for element 1 in figure 22(c).

The natural frequency is determined for the second solution. Note that the masses are defined at all 13 joints, and only joints 4, 7, 10 and 13 have lateral dof. The dof at joints 2, 3, 5, 6, 8, 9, 11 and 12 are condensed out and the dof at joint 1 are restrained. Thus the masses are condensed with the structural stiffness.

The elastic buckling load is determined for the third solution. The axial element forces from the first solution are used to generate the geometric stiffness. Similar to the second solution, the geometric stiffness is condensed out with the structural stiffness.

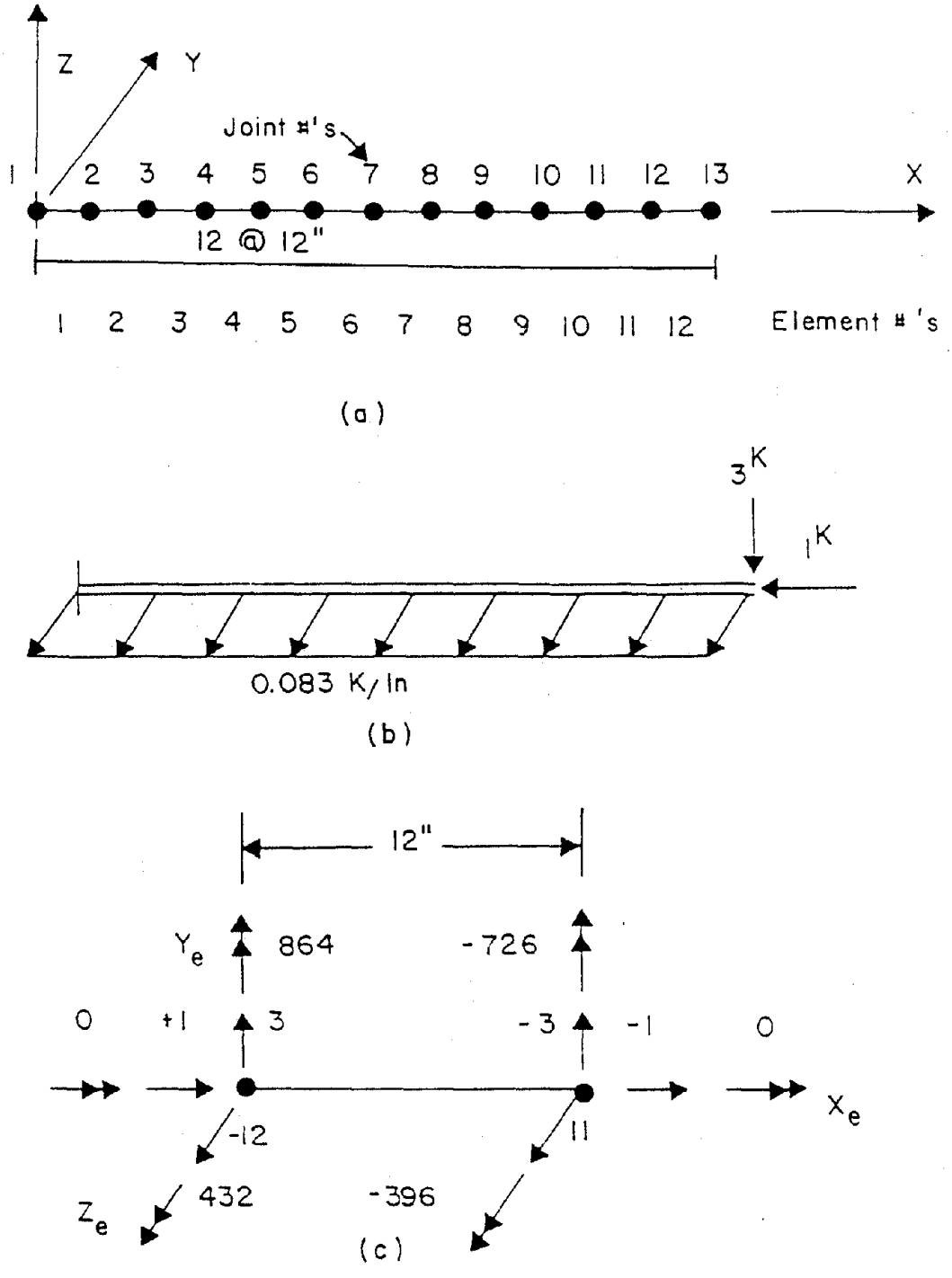


Figure 22. Cantilever Beam: (a) Joints, Elements, and Dof, (b) Applied loadings, (c) Member Forces for Element 1

2. Input Data.

```
ECHO OF INPUT DATA

LINE  ....|...10....|...20....|...30....|...40....|...50....|...60....|...70....|...80
   1:  'STRUCTURE DEFINATION'
   2:  'EXAMPLE #1. 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD,  4 DOF...'
   3:  13 1  1 5 0  12.00
   4:      1   0.00   0.00   0.00 1 12
   5:      1   1.00   0.00   0.00
   6:      1 0 0    0 1 0
   7:      1  1 1 1  1 1 1     0 0
   8:      0  1 0 0  1 1 1     0 0
   9:      2  1 1 1  0 0 0     1 1
  10:      5  1 1 1  0 0 0     1 1
  11:      8  1 1 1  0 0 0     1 1
  12:     11  1 1 1  0 0 0     1 1
  13:    1
  14: '3D-BEAM W12X40 ' 29000 11600 11.8 0 0 0.95 44.1 310.
  15:    2 2 2 .TRUE.
  16:   12
  17: '3D-BEAM' 'MEMBER  1' 1  1  2  0 0 1  0 0 0  000000          11
  18:                       0  1  1  0 0 0  0 0 0  000000           0
  19:  4 1  .TRUE.
  20:  1  .5 .5 .5  0.  0.  0. 0 0 0    0 0
  21:  2  1. 1. 1.  0.  0.  0. 0 0 0   10 1
  22: 13  .5 .5 .5  0.  0.  0. 0 0 0    0 0
  25:  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  24: 0 0
  25: 'SOL01  SOLUTION'
  26: 'STATIC LOADINGS'
  27:   1 1
  28: 1 13 0 0  'FX' -1.0
  29: 1 13 0 0  'FZ' -5.0
  30:  0 0 0 0  'END'  0
  31: 1 1 11 1 'UNIF' 'FZ'  0.08333333
  32: 'SOL03 - SOLUTION'
  33: 'FREQUENCY ANALYSIS'
  34: 'FREQ'          4  0
  35: 'SOL05 - SOLUTION'
  36: 'STABILITY ANALYSIS'
  37: 'BUCK'          4  0
  38: 'STOP'
```

3. Output.

```
STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD,  4 DOF...          TIME: 14:59:52. DATE: 06/07/89
SOLUTION.....:

*** PROGRAM FEM ***          DOUBLE PRECISION VERSION


   NODE COORDINATES AND DEGREES OF FREEDOM

  TOTAL NUMBER OF DEGREES OF FREEDOM.............    78
  NUMBER OF DEGREES OF FREEDOM CONDENSED OUT.....    64
  NUMBER OF FREE DEGREES OF FREEDOM..............     8
  NUMBER OF RESTRAINED DEGREES OF FREEDOM........     6

    NODE COS#    X-COORD        Y-COORD        Z-COORD        FX     FY     FZ     MX     MY     MZ
       1    1    0.00000E+00    0.00000E+00    0.00000E+00   75-R   76-R   77-R   78-R   79-R   80-R
       2    1     12.000        0.00000E+00    0.00000E+00    1      2      3      4      5      6
       3    1     24.000        0.00000E+00    0.00000E+00    7      8      9     10     11     12
       4    1     36.000        0.00000E+00    0.00000E+00   13     65     66     14     15     16
       5    1     48.000        0.00000E+00    0.00000E+00   17     18     19     20     21     22
       6    1     60.000        0.00000E+00    0.00000E+00   23     24     25     26     27     28
       7    1     72.000        0.00000E+00    0.00000E+00   29     67     68     30     31     32
       8    1     84.000        0.00000E+00    0.00000E+00   33     34     35     36     37     38
       9    1     96.000        0.00000E+00    0.00000E+00   39     40     41     42     43     44
      10    1     108.00        0.00000E+00    0.00000E+00   45     69     70     46     47     48
      11    1     120.00        0.00000E+00    0.00000E+00   49     50     51     52     53     54
      12    1     132.00        0.00000E+00    0.00000E+00   55     56     57     58     59     60
      13    1     144.00        0.00000E+00    0.00000E+00   61     71     72     62     63     64
        NOTE: R - RESTRAINED DEGREE OF FREEDOM
              C - CONSTRAINED DEGREE OF FREEDOM


   DIRECTION COSINES ...

COSX  1)  VX:  1.00000 I +0.00000 J +0.00000 K   VY:  0.00000 I +1.00000 J +0.00000 K   VZ:  0.00000 I +0.00000 J +1.00000 K

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD,  4 DOF...          TIME: 14:59:52. DATE: 06/07/89
SOLUTION.....:


   3-D ELASTIC BEAM ELEMENT
   ========================

  MATL.    E        GAMMA        AX         AY         AZ         IX         IY         IZ

     1 2.900E+04 1.160E+04  11.8      0.000E+00 0.000E+00 0.950      44.1       310.


STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD,  4 DOF...          TIME: 14:59:52. DATE: 06/07/89
SOLUTION.....:

   GEOMETRIC STIFFNESS DATA
   ========================

   KGLOAD   = 2. PREVIOUS LOAD STEP'S AXIAL LOAD IS USED
   KGTYPE   = 2. CONSISTENT FORMULATION
   KGFORM   = 2. SEPERATE GLOBAL KG IS FORMED

   - THE GEOMETRIC STIFFNESS MATRIX IS CONDENSED WITH THE STRUCTURAL STIFFNESS MATRIX.
```

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:39:52, DATE: 06/07/89
SOLUTION.....:

ELEMENT 08, 3D BEAM ELEMENT

	#	MATL	START	END	REL CD	LENGTH	V-AXIS	START DIST	END DIST	PKG
MEMBER 1	1	1	1	2		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	2	1	2	3		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	3	1	3	4		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	4	1	4	5		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	5	1	5	6		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	6	1	6	7		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	7	1	7	8		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	8	1	8	9		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	9	1	9	10		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	10	1	10	11		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	11	1	11	12		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00
GENERATED	12	1	12	13		12.00	0.00000 I +0.00000 J +1.00000 K	0.0000E+00	0.0000E+00	0.0000E+00

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:39:52, DATE: 06/07/89
SOLUTION.....:

LUMPED NODE MASSES

NODE	MX	MY	MZ	RXX	RYY	RZZ	RXY	RXZ	RYZ	IGEN	INC
1	0.5000	0.5000	0.5000	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
2	1.000	1.000	1.000	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	10	1
13	0.5000	0.5000	0.5000	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0

- THE MASS MATRIX IS CONDENSED WITH THE STRUCTURAL STIFFNESS MATRIX.

PROPORTIONAL DAMPING COIEFICIENTS

ALPHA= 0.00000E+00 BETA= 0.00000E+00

--- MEMORY UTILIZATION ..... ---
--- IZ= 7463, MEM= 1.495% ---
--- ELAPSED CPU TIME 0.41 SEC ---
--- TOTAL CPU TIME 0.41 SEC ---

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:39:52, DATE: 06/07/89
SOLUTION.....: STATIC LOADINGS TIME: 14:39:52, DATE: 06/07/89

SOLUTION #1, STATIC - ELASTIC ANALYSIS

NUMBER OF LOAD CASES 1
GEOMETRIC STIFFNESS IS NOT INCLUDED, USE KGFORM=1 TO INCLUDE GEOMETRIC STIFFNESS.

APPLIED JOINT LOADS

LOAD CASE: 1 JOINT: 13 DIRECTION: FX DOF(S) 61 MAGNITUDE: -1.00000
LOAD CASE: 1 JOINT: 13 DIRECTION: FZ DOF(S) 72 MAGNITUDE: -5.00000

ELEMENT LOADS

LOAD	IELE	IGEN	OIELE	TYPE	DIR	VALUES....
1	1	11	1	UNIF	FZ	8.333333E-02

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:39:52, DATE: 06/07/89
SOLUTION.....: STATIC LOADINGS TIME: 14:39:52, DATE: 06/07/89

GCS DISPLACEMENTS, LOADING # 1

NODE	DX	DY	DZ	RX	RY	RZ
1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
2	-3.506721E-05	-4.599578E-02	-5.363757E-05	0.000000E+00	5.526140E-04	-7.450152E-03
3	-7.013442E-05	-0.173849	-1.507052E-02	0.000000E+00	1.057175E-03	-1.566174E-02
4	-1.052016E-04	-0.369574	-2.854372E-02	0.000000E+00	1.515682E-03	-1.874756E-02
5	-1.402688E-04	-0.619753	-4.920667E-02	0.000000E+00	1.922156E-03	-2.281961E-02
6	-1.753361E-04	-0.913441	-7.448276E-02	0.000000E+00	2.282536E-03	-2.599109E-02
7	-2.104053E-04	-1.24057	-0.105795	0.000000E+00	2.594883E-03	-2.857458E-02
8	-2.454705E-04	-1.59172	-0.136568	0.000000E+00	2.859177E-03	-3.008210E-02
9	-2.805577E-04	-1.96008	-0.172225	0.000000E+00	3.075417E-03	-3.122685E-02
10	-3.156049E-04	-2.33957	-0.210186	0.000000E+00	3.245604E-03	-3.192118E-02
11	-3.506721E-04	-2.72484	-0.249878	0.000000E+00	3.363757E-03	-3.227774E-02
12	-3.857595E-04	-3.11313	-0.290723	0.000000E+00	3.435818E-03	-3.240910E-02
13	-4.208065E-04	-3.50221	-0.332145	0.000000E+00	3.459844E-03	-3.242787E-02

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:39:52, DATE: 06/07/89
SOLUTION.....: STATIC LOADINGS TIME: 14:39:52, DATE: 06/07/89

GCS RESTRAINT REACTIONS, LOADING # 1

NODE	FX	FY	FZ	MX	MY	MZ
1	1.000000	12.00000	5.000000	0.0000000E+00	-432.0000	864.0000
SUMMATION	1.000000	12.00000	5.000000	0.0000000E+00	-432.0000	864.0000

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:39:52, DATE: 06/07/89
SOLUTION.....: STATIC LOADINGS TIME: 14:39:52, DATE: 06/07/89

3D BEAM FORCES...

ELEMENT	LOAD		NODE	AXIAL	FV	FZ	TORSION	MY	MZ
1	1	FORCE	1	1.00000	5.00000	-12.0000	0.000000E+00	864.000	+32.000
		FORCE	2	-1.00000	-5.00000	11.0000	0.000000E+00	-726.000	-396.000
2	1	FORCE	2	1.00000	5.00000	-11.0000	0.000000E+00	726.000	396.000
		FORCE	3	-1.00000	-5.00000	10.0000	0.000000E+00	-600.000	-360.000
3	1	FORCE	3	1.00000	5.00000	-10.0000	0.000000E+00	600.000	360.000
		FORCE	4	-1.00000	-5.00000	9.00000	0.000000E+00	-486.000	-324.000
4	1	FORCE	4	1.00000	5.00000	-9.00000	0.000000E+00	486.000	324.000
		FORCE	5	-1.00000	-5.00000	8.00000	0.000000E+00	-384.000	-288.000
5	1	FORCE	5	1.00000	5.00000	-8.00000	0.000000E+00	384.000	288.000
		FORCE	6	-1.00000	-5.00000	7.00000	0.000000E+00	-294.000	-252.000
6	1	FORCE	6	1.00000	5.00000	-7.00000	0.000000E+00	294.000	252.000
		FORCE	7	-1.00000	-5.00000	6.00000	0.000000E+00	-216.000	-216.000
7	1	FORCE	7	1.00000	5.00000	-6.00000	0.000000E+00	216.000	216.000
		FORCE	8	-1.00000	-5.00000	5.00000	0.000000E+00	-150.000	-180.000
8	1	FORCE	8	1.00000	5.00000	-5.00000	0.000000E+00	150.000	180.000
		FORCE	9	-1.00000	-5.00000	4.00000	0.000000E+00	-96.0000	-144.000
9	1	FORCE	9	1.00000	5.00000	-4.00000	0.000000E+00	96.0000	144.000
		FORCE	10	-1.00000	-5.00000	3.00000	0.000000E+00	-54.0000	-108.000
10	1	FORCE	10	1.00000	5.00000	-3.00000	0.000000E+00	54.0000	108.000
		FORCE	11	-1.00000	-5.00000	2.00000	0.000000E+00	-24.0000	-72.0000
11	1	FORCE	11	1.00000	5.00000	-2.00000	0.000000E+00	24.0000	72.0000
		FORCE	12	-1.00000	-5.00000	1.00000	0.000000E+00	-6.00000	-36.0000
12	1	FORCE	12	1.00000	5.00000	-1.00000	0.000000E+00	6.00000	36.0000
		FORCE	13	-1.00000	-5.00000	1.746298E-12	0.000000E+00	1.952244E-11	1.09159+E-11

--- MEMORY UTILIZATION ..... ----
--- IZ= 7870, MEM= 1.574% ----
--- ELAPSED CPU TIME 0.15 SEC ----
--- TOTAL CPU TIME 0.53 SEC ----

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:59:52, DATE: 06/07/89
SOLUTION.....: FREQUENCY ANALYSIS TIME: 14:59:55, DATE: 06/07/89

SOLUTION #5, DETERMINE EIGENVALUES AND EIGENVECTORS

OPTION:.....NATURAL FREQUENCIES
NUMBER OF EIGENVALUES:..... 4
PREFORMANCE INDEX: 0.2906784684880794E3

MODE:	1	2	3	4
FREQ (RAD/SEC):	0.66219	1.7557	4.1268	10.941
FREQUENCY (HZ):	0.10539	0.27942	0.65680	1.7414
PERIOD (SEC):	9.4885	3.5788	1.5225	0.57426
EIGENVECTORS:				
DOF(65)	9.71210E-02	2.50500E-15	-0.41885	1.76607E-15
DOF(66)	-1.49621E-15	9.71210E-02	5.1553+E-16	-0.41885
DOF(67)	0.53911	7.26854E-15	-0.72162	2.68976E-16
DOF(68)	-5.50069E-15	0.53911	7.41779E-16	-0.72162
DOF(69)	0.65752	1.09734E-14	-0.1+744	-1.67126E-15
DOF(70)	-1.05996E-14	0.65752	1.78485E-15	-0.1+744
DOF(71)	1.0000	1.23237E-14	1.0000	1.9+012E-15
DOF(72)	-1.53597E-14	1.0000	5.16971E-15	1.0000

--- MEMORY UTILIZATION ..... ----
--- IZ= 8092, MEM= 1.618% ----
--- ELAPSED CPU TIME 0.16 SEC ----
--- TOTAL CPU TIME 0.69 SEC ----

STRUCTURE....: EXAMPLE #1, 12 ELEMENT BEAM VIBRATION AND BUCKLING LOAD, 4 DOF... TIME: 14:59:52, DATE: 06/07/89
SOLUTION.....: STABILITY ANALYSIS TIME: 14:59:55, DATE: 06/07/89

SOLUTION #5, DETERMINE EIGENVALUES AND EIGENVECTORS

OPTION:.....ELASTIC BUCKLING LOAD
NUMBER OF EIGENVALUES:..... 4
PREFORMANCE INDEX: 0.2690082949720555E0

MODE:	1	2	3	4
EIGENVALUE:	152.18	1069.8	1374.4	5950.6
EIGENVECTORS:				
DOF(65)	7.61205E-02	-3.5209+E-16	0.32087	0.80996
DOF(66)	-8.51674E-17	7.61205E-02	2.60076E-16	-5.95682E-17
DOF(67)	0.29289	-8.64958E-16	0.88753	1.0000
DOF(68)	-1.86999E-17	0.29289	9.45507E-16	-9.47499E-17
DOF(69)	0.61752	-8.54068E-16	1.0000	4.45905E-02
DOF(70)	-1.49916E-16	0.61752	1.80181E-15	-1.19719E-16
DOF(71)	1.0000	-1.56099E-16	0.51978	0.58579
DOF(72)	-5.96887E-16	1.0000	2.87797E-15	-9.17196E-17

--- MEMORY UTILIZATION ..... ----
--- IZ= 8514, MEM= 1.665% ----
--- ELAPSED CPU TIME 0.17 SEC ----
--- TOTAL CPU TIME 0.86 SEC ----

Notes:

(1) Vector in the beam's local XY-plane, $\vec{V} = V1i + V2j + V3k$

(2) IMSL performance index for routine EIGZS. When PI<1, the performance of the eigensolution is excellent, and when PI>100, the performance of the eigensolution is poor.

B. ONE STORY R/C BOX TYPE BUILDING

1. Description of Input Information.

The one story R/C box type building consists of four shear walls and a rigid roof diaphragm as shown in Figure 23. Nine joints are used to define the three dimensional structural model. At the base, joints 1 through 4 define the corners of the box. Joints 11 through 14 define the corners of the box at the roof. Joint 10 is at the mass center of the roof. The JCS for all of the joints is parallel to the GCS. The three dimensional structural model consists of four shear wall elements, numbered 1 through 4. The wall's center line dimensions are used for the structural model, thus each of the shear wall elements is 75 cm wide, and 80 cm tall. P-Δ effects are neglected.

Each joint has 6 degrees of freedom. At the base of the box, joints 1 through 4, all six degrees of freedom are restrained. The slab at the top of the box is assumed to be a rigid diaphragm. A planar constraint is used to transfer the translations in the X and Y axes and the rotation about the Z axis from joints 11 through 14, at the corners of the box, to joint 10, at the mass center. The planar constraint at the roof reduces the number of degrees of freedom at joints 11 through 14, leaving rotational dof about the X and Y axes which are restrained and translational dof in the Z axis which is free to displace. None of the elements are connected to joint 10. Thus joint 10 does not have any stiffness untill joints 11 through 14 are constrained to it. As a 'master' joint,

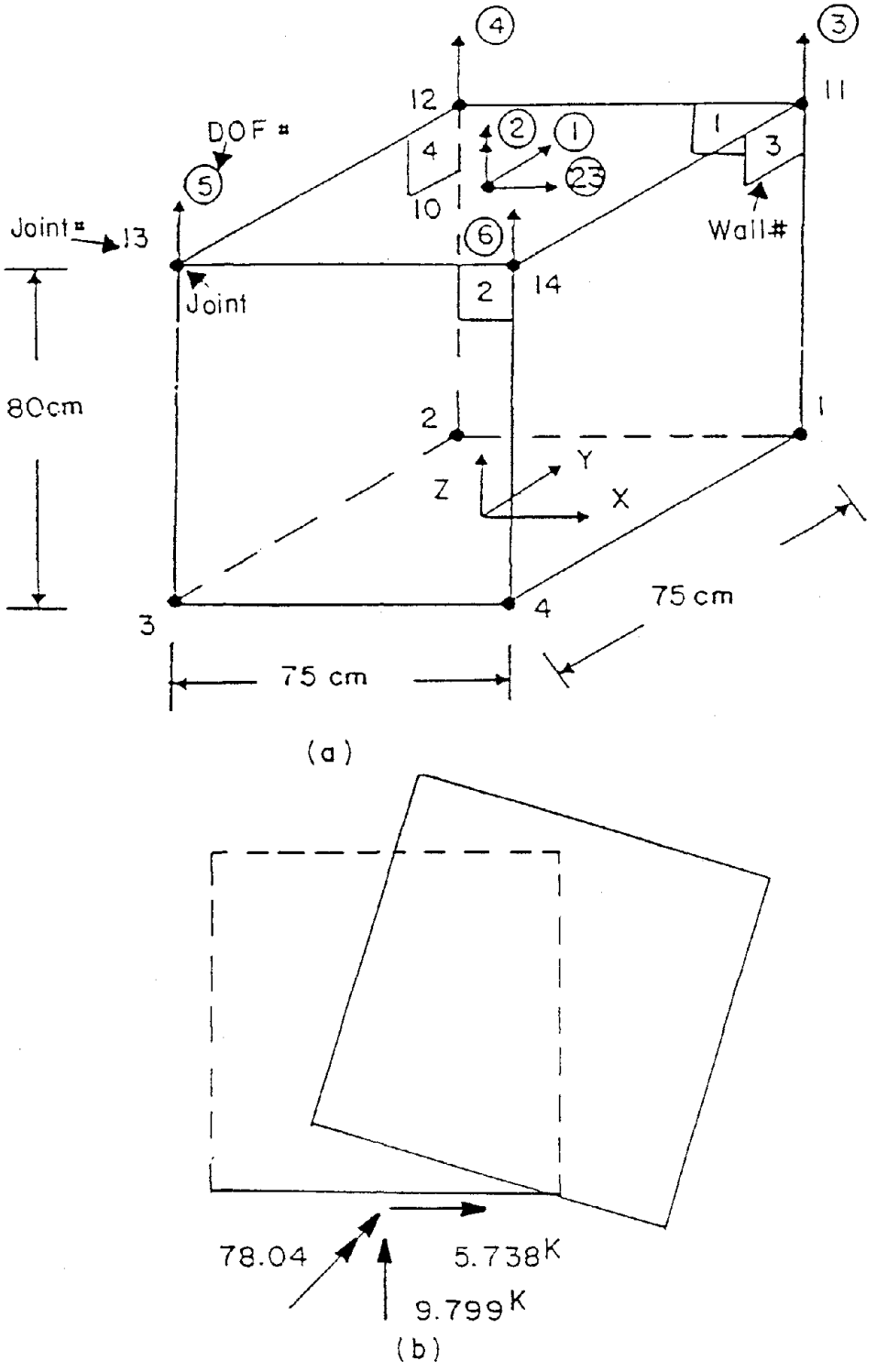


Figure 23. One Story R/C Box Type Building: (a) Building, (b) Forces and Deformation of Wall #1 at Load Step 50

joint 10 has translational stiffness in the X and Y axes, and rotational stiffness about the Z axis. Loading consist of the imposed displacement of joint 10 in the X direction. Thus the X translational dof of joint 10 is restrained. The Z translational dof and the rotations about the X and Y axes are also restrained, because these dof do not have any stiffness. The Y translational dof and the Z rotational dof of joint 10, are free to displace. The dof are numbered in accordance with the global degree of freedom (Gdof) numbering scheme (17), as shown in Table I. All of the rotations about the X and Y axes are restrained with a restraint flag of 2.

2. Input Data.

```
ECHO OF INPUT DATA

LINE  ....|...10....|...20....|...30....|...40....|...50....|...60....|...70....|...80
   1: 'STRUCTURE DEFINATION'
   2: 'UNIV. OF TOKYO BOX 86  3-D MODEL,  4/6/89    '
   3:   9 1   5 0 1    1.
   4:      1  57.5  57.5   0.    1 0
   5:      2 -57.5  57.5   0.    1 0
   6:      3 -57.5 -57.5   0.    1 0
   7:      4  57.5 -57.5   0.    1 0
   8:     10   0      0    80.    1 0
   9:     11  57.5  57.5  80.    1 0
  10:     12 -57.5  57.5  80.    1 0
  11:     13 -57.5 -57.5  80.    1 0
  12:     14  57.5 -57.5  80.    1 0
  13:        1 0 0    0 1 0
  14:      0  0 0 0 0 2 2 0    0  0
  15:      1  1 1 1 1 1 1 1    3  1
  16:     10  1 0 1 2 2 0      0  0
  17:  1 10 11 3 1
  18:     3
  19: 'AXLMOD ' 172800  155520  172.8   14.06  .40    .2   15 .2
  20: 'BEND1 BOT' 9 10  3.073819E-5              .2
  21:     80.0000       100.000       160.000       280.000       360.000
  22:     391.000       420.000       440.000       460.000
  23:   1.409580E-06  2.297770E-06  5.720770E-06  1.592260E-05  2.401340E-05
  24:   5.073819E-05  5.967050E-05  4.950700E-05  5.073818E-04
  25: 'SHEAR1 BOT' 10 10 2.72587E-3              .2
  26:     4.00000       5.00000       8.00000       12.0000       16.0000
  27:     19.0000       19.5500       21.0000       22.0000       22.1000
  28:   1.245750E-04  1.958900E-04  4.987020E-04  1.020270E-03  1.705120E-03
  29:   2.552620E-03  2.725875E-03  3.440410E-03  4.277610E-03  2.725875E-02
  30:    0 0 0 .FALSE.
  31:    4
  32: 'SHEAR WALL' 'NORTH' 2  3 1   11  12   2   1  1.00    0.0 0 0
  33: 'SHEAR WALL' 'SOUTH' 2  3 1   14  15   3   4  1.00    0.0 0 0
  34: 'SHEAR WALL' 'EAST ' 2  3 1   11  14   4   1  1.00    0.0 0 0
  35: 'SHEAR WALL' 'WEST ' 2  3 1   12  15   3   2  1.00    0.0 0 0
  36:   0  0 .FALSE.
  37:   0 0
  38: 'SOL04 '
  39: 'MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION '
  40:    0 99999  1 .TRUE.
  41:    'ELE' 1 21    3 1 1
  42:    'JOINT FX' 10  31 0   0 0
  43:    'JOINT FY' 10  32 0   0 0
  44:    'JOINT FZ' 11  41 0   0 0
  45:    'JOINT FZ' 12  42 0   0 0
  46:    'JOINT FZ' 13  43 0   0 0
  47:    'JOINT FZ' 14  44 0   0 0
  48:    'ENERGY '   00 37 0   0 0
  49:    'SUMRCT '   00 38 0   0 0
  50:    'END' 0  0        0  0 0
  51:   1 10 0 0 'FX'     0.1
  52:   0  0 0 0 'END'  0.
```

Table I. DEGREES OF FREEDOM FOR UT BOX B6

Joint	Condensed DOF						Free DOF						Restrained DOF *						Restrained DOF **					
#	TX	TY	TZ	RX	RY	RZ	TX	TY	TZ	RX	RY	RZ	TX	TY	TZ	RX	RY	RZ	TX	TY	TZ	RX	RY	RZ
1													7	8	9			10						25 26
2													11	12	13			14						27 28
3													15	16	17			18						29 30
4													19	20	21			22						31 32
10							1					2	23					24						33 34
11							1c		3			2c	23c											35 36
12							1c		4			2c	23c											37 38
13							1c		5			2c	23c											39 40
14							1c		6			2c	23c											41 42

Notes:

1) TX, TY and TZ signify translation in the joints X, Y and Z axes respectively.
2) RX, RY and RZ signify rotation about the joints X, Y and Z axes respectively.
3) The constrained dof of 'slave' joints are represented by #c, where # is the dof number of the 'master' joint.

* Restraint flag = 1 on the joint restraint input card.

** Restraint flag = 2 on the joint restraint input card.

```
 ECHO OF INPUT DATA

LINE  ....|...10....|...20....|...30....|...40....|...50....|...60....|...70....|...80
  53:   'LOAD- 1'   0.01       0  .0       1
  54:   'LOAD- 1'   0.46       0  .0       4 6
  55:   'LOAD- 1'   0.4599     0  .0       2
  56:   'LOAD- 2'  -0.46       0  .0       9 2
  57:   'LOAD- 2'  -0.4599     0  .0       2
  58:   'LOAD- 3'   0.7000     0  .0      10
  59:   'LOAD- 3'   1.70       0  .0      10 6
  60:   'LOAD- 3'   1.6999     0  .0       2
  61:   'LOAD- 4'  -0.8200     0  .0      35
  62:   'LOAD- 4'  -1.82       0  .0      10 2
  63:   'LOAD- 4'  -1.8199     0  .0       2
  64:   'LOAD- 5'   0.7000     0  .0      35
  65:   'LOAD- 5'   1.70       0  .0      10 2
  66:   'LOAD- 5'   1.6999     0  .0       2
  67:   'LOAD- 6'  -0.5600     0  .0      32
  68:   'LOAD- 6'  -1.56       0  .0      10 6
  69:   'LOAD- 6'  -1.5599     0  .0       2
  70:   'LOAD- 7'   2.8000     0  .0      53
  71:   'LOAD- 7'   3.80       0  .0      10 6
  72:   'LOAD- 7'   3.7999     0  .0       2
  73:   'LOAD- 8'  -5.0000     0  .0      78
  74:   'LOAD- 8'  -4.00       0  .0      10 0
  75:   'LOAD- 8'  -3.9999     0  .0       2
  76:   'LOAD- 9'   2.9000     0  .0      79
  77:   'LOAD- 9'   3.90       0  .0      10 0
  78:   'LOAD- 9'   3.8999     0  .0       2
  79:   'LOAD-10'  -2.9000     0  .0      78
  80:   'LOAD-10'  -3.90       0  .0      10 0
  81:   'LOAD-10'  -3.8999     0  .0       2
  82:   'LOAD-11'   3.2200     0  .0      81
  83:   'LOAD-11'   4.22       0  .0      10 2
  84:   'LOAD-11'   4.2199     0  .0       2
  85:   'LOAD-12'  -2.9000     0  .0      81
  86:   'LOAD-12'  -3.90       0  .0      10 2
  87:   'LOAD-12'  -3.8999     0  .0       2
  88:   'LOAD-13'    .0000     0  .0     119
  89:   'LOAD-13'   7.0600     0  .0     119
  90:   'LOAD-13'   8.06       0  .0      10 6
  91:   'LOAD-13'   8.0599     0  .0       2
  92:   'LOAD-14'  -7.1800     0  .0     162
  93:   'LOAD-14'  -8.18       0  .0      10 4
  94:   'LOAD-14'  -8.1799     0  .0       2
  95:   'LOAD-15'   7.0000     0  .0     161
  96:   'LOAD-15'   8.00       0  .0      10 8
  97:   'LOAD-15'   7.9999     0  .0       2
  98:   'LOAD-16'  -7.1800     0  .0     161
  99:   'LOAD-16'  -8.18       0  .0      10 8
 100:   'LOAD-16'  -8.1799     0  .0       2
 101:   'LOAD-17'  14.6000     0  .0     557
 102:   'LOAD-17'  15.60       0  .0      10 8
 103:   'LOAD-17'  15.5999     0  .0       2
 104:   'LOAD-18'   9.7000     0  .0      50
 105:   'LOAD-18'  10.70       0  .0      10 0
 106:   'LOAD-18'  10.69999    0  .0       2
 107:   'LOAD-19'  19.4000     0  .0      97
 108:   'LOAD-19'  20.40       0  .0      10 0
 109:     'END'     0    0     0    0
 110:  'READ INC=50  UNIT=21 UNIT=57 '
 111:  'STOP'
```

3. Output.

```
STRUCTURE.....: UNIV. OF TOKYO BOX 86  3-D MODEL,  4/6/89                    TIME: 16:59:04, DATE: 05/31/89
SOLUTION......:

*** PROGRAM FEM ***          DOUBLE PRECISION VERSION


    NODE CONSTRAINTS

  XY-PLANE        CONSTRAINT, MASTER:    10 SLAVE:    11
  XY-PLANE        CONSTRAINT, MASTER:    10 SLAVE:    12
  XY-PLANE        CONSTRAINT, MASTER:    10 SLAVE:    15
  XY-PLANE        CONSTRAINT, MASTER:    10 SLAVE:    14


    NODE COORDINATES AND DEGREES OF FREEDOM

   TOTAL NUMBER OF DEGREES OF FREEDOM.............    42
   NUMBER OF DEGREES OF FREEDOM CONDENSED OUT.....     0
   NUMBER OF FREE DEGREES OF FREEDOM..............     6
   NUMBER OF RESTRAINED DEGREES OF FREEDOM........    36

     NODE COS#     X-COORD        Y-COORD        Z-COORD        FX     FY     FZ     MX     MY     MZ
        1    1      57.500         57.500      0.00000E+00     7-R    8-R    9-R   25-R   26-R   10-R
        2    1     -57.500         57.500      0.00000E+00    11-R   12-R   15-R   27-R   28-R   14-R
        5    1     -57.500        -57.500      0.00000E+00    15-R   16-R   17-R   29-R   30-R   18-R
        4    1      57.500        -57.500      0.00000E+00    19-R   20-R   21-R   31-R   32-R   22-R
       10    1    0.00000E+00    0.00000E+00     80.000      23-R    1     24-R   33-R   34-R    2
       11    1      57.500         57.500        80.000      23-C   1-C     5     35-R   36-R   2-C
       12    1     -57.500         57.500        80.000      23-C   1-C     4     37-R   38-R   2-C
       13    1     -57.500        -57.500        80.000      23-C   1-C     5     39-R   40-R   2-C
       14    1      57.500        -57.500        80.000      23-C   1-C     6     41-R   42-R   2-C
         NOTE: R - RESTRAINED DEGREE OF FREEDOM
               C - CONSTRAINED DEGREE OF FREEDOM


    DIRECTION COSINES ...

COSC  1)   VK:  1.00000 I +0.00000 J +0.00000 K   VV:  0.00000 I +1.00000 J +0.00000 K   VZ:  0.00000 I +0.00000 J +1.00000 K
```

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL, 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....:

AXLMOD HYSTERESIS MODEL- FOR UNIT LENGTH MEMBER

MAT.	SC	ST1	ST2	PY	ALPHA	BETA	MAX DUCT.	BETA-DI
1	172800.	155520.	172.800	14.0600	0.400000	0.200000	15.0000	0.200000

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL, 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....:

BEND1 HYSTERESIS MODEL DATA - UNIT LENGTH MEMBER

BACKBONE CURVE POINTS - MATL	NI	DY	BETA	MOMENT	ROTATION
2	10	0.307582E-04	0.200000	80.0000	0.140958E-05
				100.000	0.229777E-05
				160.000	0.572077E-05
				280.000	0.139226E-04
				360.000	0.240134E-04
				391.000	0.307382E-04
				420.000	0.396705E-04
				440.000	0.495070E-04
				460.000	0.307382E-03

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL, 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....:

SHEAR1 HYSTERESIS MODEL DATA - UNIT LENGTH MEMBER

BACKBONE CURVE POINTS - MAT.	NI	DY	BETA	STRESS	STRAIN
5	10	0.272587E-02	0.200000	4.00000	0.124575E-03
				5.00000	0.195890E-03
				8.00000	0.498702E-03
				12.0000	0.102027E-02
				16.0000	0.170512E-02
				19.0000	0.253262E-02
				19.5500	0.272587E-02
				21.0000	0.3[illegible]1E-02
				22.0000	0.[illegible]27761E-02
				22.1000	0.272587E-01

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL, 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....:

ELEMENT 05, R/C SHEAR WALL ELEMENT

	ELEM #	BEND MATL	SHEAR MATL	AXIAL MATL	JOINT #1	JOINT #2	JOINT #3	JOINT #4	LENGTH	WIDTH	ALPHA	PKG
NORTH	1	2	5	1	11	12	2	1	80.00	75.00	1.000	0.0000E+00
SOUTH	2	2	5	1	14	15	5	4	80.00	75.00	1.000	0.0000E+00
EAST	3	2	5	1	11	14	4	1	80.00	75.00	1.000	0.0000E+00
WEST	4	2	5	1	12	15	5	2	80.00	75.00	1.000	0.0000E+00

ZERO MASS MATRIX

PROPORTIONAL DAMPING COIEFICIENTS

ALPHA= 0.00000E+00 BETA= 0.00000E+00

--- MEMORY UTILIZATION ..... ---
--- IZ= 2625, MEM= 0.525% ---
--- ELAPSED CPU TIME 0.20 SEC ---
--- TOTAL CPU TIME 0.20 SEC ---

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL, 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:39:05, DATE: 05/31/89
SOLUTION ##, STATIC NONLINEAR SOLUTION

INTERVAL FOR PRINTING DATA.....99999
INTERVAL FOR WRITING DATA TO FILE..... 1

(1)

UNBALANCED JOINT FORCES ARE ADDED TO THE NEXT CYCLE

DATA WRITTEN TO FILES

ELEMENT # 1 IS WRITTEN TO UNIT # 21
ELEMENT # 2 IS WRITTEN TO UNIT # 22
ELEMENT # 3 IS WRITTEN TO UNIT # 23
ELEMENT # 4 IS WRITTEN TO UNIT # 24
DEGREE OF FREEDOM # 23 IS WRITTEN TO UNIT # 31 JOINT: 10 DIRECTION: FX
DEGREE OF FREEDOM # 1 IS WRITTEN TO UNIT # 32 JOINT: 10 DIRECTION: FY
DEGREE OF FREEDOM # 3 IS WRITTEN TO UNIT # 41 JOINT: 11 DIRECTION: FZ
DEGREE OF FREEDOM # 4 IS WRITTEN TO UNIT # 42 JOINT: 12 DIRECTION: FZ
DEGREE OF FREEDOM # 5 IS WRITTEN TO UNIT # 43 JOINT: 13 DIRECTION: FZ
DEGREE OF FREEDOM # 6 IS WRITTEN TO UNIT # 44 JOINT: 14 DIRECTION: FZ
ENERGY BALANCE IS WRITTEN TO UNIT # 37
SUMMATION OF REACTIONS IS WRITTEN TO UNIT # 38

APPLIED JOINT LOADS

LOAD CASE: 1 JOINT: 10 DIRECTION: FX DOF(S) 23 MAGNITUDE 0.100000 ...JOINT DISPLACEMENT...

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL, 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:39:05, DATE: 05/31/89

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:59:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:59:05, DATE: 05/31/89

, LOADING # 0 MAXIMUM DISPLACEMENTS

GCS DISPLACEMENTS
==========

NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

NODE	DX	DY	DZ	RX	RY	RZ
1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
2	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
3	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
4	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
10	2.04000	2.224492E-16	0.000000E+00	0.000000E+00	0.000000E+00	-8.109588E-18
11	2.04000	-8.166050E-17	0.542589	0.000000E+00	0.000000E+00	-8.109588E-18
12	2.04000	5.265588E-16	1.60591	0.000000E+00	0.000000E+00	-8.109588E-18
13	2.04000	5.265588E-16	1.60591	0.000000E+00	0.000000E+00	-8.109588E-18
14	2.04000	-8.166050E-17	0.542589	0.000000E+00	0.000000E+00	-8.109588E-18

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:59:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:59:05, DATE: 05/31/89

, LOADING # 0 MAXIMUM REACTIONS

MAXIMUM GCS RESTRAINT REACTIONS NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY
==========

NODE	FX	FY	FZ	MX	MY	MZ
1	-21.34217	1.7242261E-14	22.76498	0.0000000E+00	0.0000000E+00	0.0000000E+00
2	0.0000000E+00	1.5276904E-14	-22.76498	0.0000000E+00	0.0000000E+00	0.0000000E+00
3	0.0000000E+00	0.0000000E+00	-22.76498	0.0000000E+00	0.0000000E+00	0.0000000E+00
4	-21.34217	0.0000000E+00	22.76498	0.0000000E+00	0.0000000E+00	0.0000000E+00
10	42.68434	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
11	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
12	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
13	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
14	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
MAX OF ALL						
GCS SUMM.	0.5872458E-12	0.1496954E-15	0.1550051E-12	0.1875853E-11	-0.5029754E-10	0.3069545E-11

MAXIMUM RESULTANT OF REACTIONS, FORCE= 5.8734E-13 MOMENT= 5.0359E-11

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:59:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:59:05, DATE: 05/31/89

, LOADING # 0 PEAK ELEMENT FORCES

R/C SHEAR WALL FORCES... MAXIMUM LOADS AND DISPL AT MAXIMUM LOADS NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

ELEMENT	LOAD	JNT-1	JNT-2	JNT-3	JNT-4	MOMENT	ROTATION	SHEAR	SHEAR DISPL.	AXIAL	AXIAL DISPL.
1	2	11	12	2	1	457.261	2.177222E-02	21.3422	0.298222	15.7879	0.787452
2	2	14	13	3	4	457.261	2.177222E-02	21.3422	0.298222	15.7879	0.787452
3	2	11	14	4	1	2.102525E-12	2.965674E-18	-4.502961E-14	-1.072085E-16	-49.1242	-2.900605E-02
4	2	12	13	3	2	4.276439E-12	6.028264E-18	-6.765052E-14	-1.685513E-16	-49.0505	-2.067145E-02

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:59:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:59:05, DATE: 05/31/89

, LOADING # 0 PEAK DUCTILITIES AND EXCURSION RATIO'S

R/C SHEAR WALL DUCTILITY AND EXCURSION RATIOS

ELEM#	BENDING COMPONENT DEFN 1 DUCT.	BENDING COMPONENT DEFN 1 EXCR.	BENDING COMPONENT DEFN 2 DUCT.	BENDING COMPONENT DEFN 2 EXCR.	BENDING COMPONENT DEFN 3 DUCT.	BENDING COMPONENT DEFN 3 EXCR.	SHEAR COMPONENT DEFN 1 DUCT.	SHEAR COMPONENT DEFN 1 EXCR.	SHEAR COMPONENT DEFN 2 DUCT.	SHEAR COMPONENT DEFN 2 EXCR.	SHEAR COMPONENT DEFN 3 DUCT.	SHEAR COMPONENT DEFN 3 EXCR.	AXIAL COMPONENT DEFN 1 DUCT.	AXIAL COMPONENT DEFN 1 EXCR.	AXIAL COMPONENT DEFN 2 DUCT.	AXIAL COMPONENT DEFN 2 EXCR.	AXIAL COMPONENT DEFN 3 DUCT.	AXIAL COMPONENT DEFN 3 EXCR.
1	6.60	15.95	8.09	25.06	9.55	22.28	1.25	0.22	2.24	8.42	1.54	2.57	108.88	125.22	18.54	38.11	109.95	178.52
2	6.60	15.95	8.09	25.06	9.55	22.28	1.25	0.22	2.24	8.42	1.54	2.57	108.88	125.22	18.54	38.11	109.95	178.52
3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	75.02	207.45	21.90	55.65	124.04	261.10
4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	221.76	381.64	29.14	88.48	274.62	548.05

(2)

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:59:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:59:05, DATE: 05/31/89

, LOADING # 0 DAMAGE INDEX

ELEM#	BENDING COMPONENT DAMAGE	BENDING COMPONENT ESE	BENDING COMPONENT PSE	SHEAR COMPONENT DAMAGE	SHEAR COMPONENT ESE	SHEAR COMPONENT PSE	AXIAL COMPONENT DAMAGE	AXIAL COMPONENT ESE	AXIAL COMPONENT PSE
1	1.24058	1.04057E-02	0.21545	0.17396	2.34124E-02	9.91175E-02	8.94944	4.69800E-05	0.16121
2	1.24058	1.04057E-02	0.21545	0.17396	2.34124E-02	9.91175E-02	8.94944	4.69800E-05	0.16121
3	0.00000	1.71267E-32	1.27541E-57	0.00000	7.85050E-33	5.40540E-47	8.55245	1.75789E-02	0.51756
4	0.00000	1.10166E-51	5.95951E-58	0.00000	7.12661E-52	2.18955E-46	21.16058	7.71512E-05	0.60788

(3)

STRUCTURE DAMAGE INDEX= 9.8144

---------- MEMORY UTILIZATION ..... ----
--- IZ= 537%, MEM= 0.67% ----
---------- ELAPSED CPU TIME 68.79 SEC ----
--- TOTAL CPU TIME 68.99 SEC ----

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:39:05, DATE: 05/31/89

ELEMENT # 1 IS READ FROM UNIT # 21

R/C SHEAR WALL FORCES...
NODE1: 11 NODE2: 12 NODE3: 2 NODE4: 1

STEP	TIME	MOMENT	ROTATION	SHEAR	SHEAR DISPL.	AXIAL	AXIAL DISPL.	ESE	PSE
0	0.0000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.0000E+00	0.0000E+00
50	0.0000E+00	78.040	3.84400E-04	5.7380	6.72400E-02	9.7990	5.18000E-05	0.1248	0.6599
100	0.0000E+00	-90.850	-5.21000E-04	-6.4310	-6.27100E-02	8.7640	4.72600E-05	8.9450E-02	0.9725
150	0.0000E+00	37.380	1.84400E-04	1.8420	4.22400E-02	7.1710	3.98800E-05	2.4250E-02	1.180
200	0.0000E+00	-6.5970	-4.27100E-05	2.0950	2.15700E-02	5.1970	5.07400E-05	1.9500E-02	1.518
250	0.0000E+00	309.20	2.14400E-05	11.770	0.14740	11.140	6.74400E-02	0.7804	5.250
300	0.0000E+00	-130.90	-5.20700E-04	-7.3910	-9.12700E-02	-1.8860	4.26600E-02	0.2520	5.255
350	0.0000E+00	-244.40	-1.85500E-03	-5.5830	-0.10510	6.2370	6.69400E-02	0.2977	4.171
400	0.0000E+00	180.30	1.10600E-03	7.3290	9.66900E-02	2.1770	6.21100E-02	0.2812	4.243
450	0.0000E+00	164.50	1.23500E-03	1.4810	6.45400E-02	1.9250	6.18100E-02	8.4890E-02	4.590
500	0.0000E+00	-268.40	-1.76700E-03	-11.050	-0.15120	5.7010	6.65000E-02	0.6470	4.819
550	0.0000E+00	-85.990	-7.35700E-04	0.54090	-1.46900E-02	1.7670	6.16200E-02	2.3570E-02	5.026
600	0.0000E+00	369.00	2.39200E-03	16.390	0.18070	11.220	7.28600E-02	1.594	5.467
650	0.0000E+00	56.250	6.80400E-04	-1.2200	-1.04100E-02	1.4710	7.06500E-02	1.5660E-02	5.740
700	0.0000E+00	-390.80	-2.60800E-03	-16.470	-0.18140	10.280	8.16500E-02	1.457	6.199
750	0.0000E+00	-225.50	-1.76100E-03	-5.9880	-8.52900E-02	3.5970	7.33000E-02	0.2101	6.194
800	0.0000E+00	-81.560	-6.91200E-04	0.65440	-6.96900E-03	1.3500	7.04900E-02	2.1670E-02	6.158
850	0.0000E+00	159.00	1.18700E-03	6.4910	8.89500E-02	1.5450	7.07400E-02	0.2220	6.262
900	0.0000E+00	407.80	5.51700E-03	18.640	0.19920	14.290	0.11190	1.874	7.184
950	0.0000E+00	443.30	7.35800E-03	19.520	0.21740	14.600	0.25540	2.281	10.88
1000	0.0000E+00	44.490	4.18000E-03	-0.87610	1.20200E-03	-0.42040	0.21290	9.8120E-03	10.90
1050	0.0000E+00	-257.20	-2.32800E-04	-8.4760	-0.11610	-2.8910	0.17520	0.4656	11.38
1100	0.0000E+00	-441.30	-5.28400E-03	-15.390	-0.18240	6.1790	0.19060	1.577	12.97
1150	0.0000E+00	-202.80	-5.94500E-03	-5.8110	-9.72100E-02	2.8940	0.25820	0.2057	14.81
1200	0.0000E+00	65.920	-2.00400E-03	2.8610	5.89000E-02	0.45670	0.25160	4.4850E-02	14.87
1250	0.0000E+00	266.40	5.21900E-03	8.1040	0.11250	0.41200	0.25080	0.4866	15.58
1300	0.0000E+00	587.30	7.16400E-03	15.670	0.19860	11.620	0.25260	1.551	17.09
1350	0.0000E+00	8.3050	3.76100E-03	-0.91350	-5.83500E-04	0.40900	0.25040	3.4630E-03	16.90
1400	0.0000E+00	-177.80	-1.01400E-03	-5.7600	-9.00200E-02	0.46670	0.25060	0.2312	17.32
1450	0.0000E+00	-568.50	-6.01600E-03	-15.190	-0.16130	5.3880	0.24010	1.157	18.48
1500	0.0000E+00	-234.90	-6.25100E-03	-5.1920	-0.11510	4.5940	0.25990	0.3111	19.27
1550	0.0000E+00	6.1280	-3.61500E-03	1.2440	1.18100E-02	0.47590	0.25160	6.0750E-03	19.20
1600	0.0000E+00	133.50	-2.77500E-04	4.5980	8.29400E-02	0.44560	0.25150	0.1401	19.48
1650	0.0000E+00	277.50	3.49000E-03	8.5850	0.11950	0.41130	0.25070	0.5265	20.10
1700	0.0000E+00	407.70	6.90000E-03	15.130	0.18470	7.0480	0.24360	1.470	21.14
1750	0.0000E+00	446.40	1.05100E-02	19.940	0.25550	14.890	0.57320	2.466	24.46
1800	0.0000E+00	450.20	1.44900E-02	20.440	0.25520	15.210	0.52180	2.685	28.66
1850	0.0000E+00	105.20	1.55700E-02	2.1420	0.10860	0.82440	0.55180	6.1450E-02	30.56
1900	0.0000E+00	168.90	1.34700E-02	10.400	0.16260	3.6890	0.50580	0.5529	30.67
1950	0.0000E+00	445.50	1.76900E-02	20.730	0.27390	15.460	0.65490	2.813	33.42

STRUCTURE....: UNIV. OF TOKYO BOX 86 3-D MODEL. 4/6/89 TIME: 16:39:04, DATE: 05/31/89
SOLUTION.....: MONOTONIC PSUDO-STATIC EQ LOADING GLOBAL X DIRECTION TIME: 16:39:05, DATE: 05/31/89

ENERGY DATA IS READ FROM UNIT # 57

STEP	TIME	INPUT	ELASTIC STRAIN	KINETIC	PLASTIC STRAIN	DAMPINED	RELATIVE ERROR %
0	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.0000
50	0.00000E+00	1.9944	0.34182	0.00000E+00	1.6400	0.00000E+00	0.6302
100	0.00000E+00	2.9921	0.35454	0.00000E+00	2.6277	0.00000E+00	0.5326
150	0.00000E+00	3.4619	9.72605E-02	0.00000E+00	3.3537	0.00000E+00	0.5155
200	0.00000E+00	5.9599	7.92542E-02	0.00000E+00	5.8610	0.00000E+00	0.4979
250	0.00000E+00	11.516	1.9006	0.00000E+00	9.5918	0.00000E+00	0.2067
300	0.00000E+00	11.242	0.65243	0.00000E+00	10.565	0.00000E+00	0.2242
350	0.00000E+00	15.518	0.70458	0.00000E+00	14.594	0.00000E+00	0.1510
400	0.00000E+00	16.425	0.76168	0.00000E+00	15.638	0.00000E+00	0.1455
450	0.00000E+00	17.765	0.17827	0.00000E+00	17.555	0.00000E+00	0.1645
500	0.00000E+00	20.963	1.7867	0.00000E+00	19.149	0.00000E+00	0.1314
550	0.00000E+00	20.602	7.14255E-02	0.00000E+00	20.507	0.00000E+00	0.1126
600	0.00000E+00	26.616	5.7755	0.00000E+00	22.821	0.00000E+00	0.0751
650	0.00000E+00	29.471	6.57818E-02	0.00000E+00	24.581	0.00000E+00	0.1055
700	0.00000E+00	50.957	5.8690	0.00000E+00	27.045	0.00000E+00	0.0805
750	0.00000E+00	27.891	0.48509	0.00000E+00	27.413	0.00000E+00	0.0915
800	0.00000E+00	27.550	6.77752E-02	0.00000E+00	27.456	0.00000E+00	0.0958
850	0.00000E+00	29.105	0.60596	0.00000E+00	28.475	0.00000E+00	0.0380
900	0.00000E+00	56.886	4.8586	0.00000E+00	52.002	0.00000E+00	0.0682
950	0.00000E+00	49.408	5.7962	0.00000E+00	45.587	0.00000E+00	0.0509
1000	0.00000E+00	44.206	5.86909E-02	0.00000E+00	44.151	0.00000E+00	0.0821
1050	0.00000E+00	48.383	1.2071	0.00000E+00	47.141	0.00000E+00	0.0734
1100	0.00000E+00	59.472	4.3640	0.00000E+00	55.072	0.00000E+00	0.0610
1150	0.00000E+00	62.208	0.44825	0.00000E+00	61.715	0.00000E+00	0.0715
1200	0.00000E+00	62.587	0.15829	0.00000E+00	62.402	0.00000E+00	0.0755
1250	0.00000E+00	67.731	1.2427	0.00000E+00	66.441	0.00000E+00	0.0698
1300	0.00000E+00	77.577	4.2955	0.00000E+00	73.040	0.00000E+00	0.0559
1350	0.00000E+00	75.376	1.95520E-02	0.00000E+00	73.306	0.00000E+00	0.0689
1400	0.00000E+00	76.419	0.61070	0.00000E+00	75.758	0.00000E+00	0.0663
1450	0.00000E+00	84.845	3.2168	0.00000E+00	81.579	0.00000E+00	0.0579
1500	0.00000E+00	85.918	0.75645	0.00000E+00	85.157	0.00000E+00	0.0521
1550	0.00000E+00	85.272	2.88542E-02	0.00000E+00	85.196	0.00000E+00	0.0544
1600	0.00000E+00	87.184	0.58190	0.00000E+00	86.756	0.00000E+00	0.0552
1650	0.00000E+00	91.572	1.3588	0.00000E+00	90.187	0.00000E+00	0.0506
1700	0.00000E+00	98.491	4.5948	0.00000E+00	94.856	0.00000E+00	0.0413
1750	0.00000E+00	112.44	6.6295	0.00000E+00	105.77	0.00000E+00	0.0394
1800	0.00000E+00	126.09	7.1768	0.00000E+00	118.87	0.00000E+00	0.0351
1850	0.00000E+00	125.42	0.13484	0.00000E+00	125.25	0.00000E+00	0.0429
1900	0.00000E+00	127.89	2.0508	0.00000E+00	125.79	0.00000E+00	0.0536
1950	0.00000E+00	145.30	7.5282	0.00000E+00	135.69	0.00000E+00	0.0552
	MAXIMUN	158.10	8.1861	0.00000E+00	149.84	0.00000E+00	

GROSS RE: 0.0646%

--- MEMORY UTILIZATION ..... ---
--- IZ= 3574, MEM= 0.675% ---
--- ELAPSED CPU TIME 3.42 SEC ---
--- TOTAL CPU TIME 72.41 SEC ---

Notes:

(1) The results for an individual step are written to the output file every 99999th step. The choice of a large increment (99999) suppresses the output for individual steps.

(2) The ductility and excursion ratio's are defined in Appendix I.

(3) The damage index is defined in Appendix II.

C. TWO STORY UNSYMMETRIC BUILDING WITH ISOLATED SHEAR WALLS

1. Description of Input Information. The two story unsymmetric building with isolated shear walls is shown in Figures 24 and 25. This 'L' shaped, two-story building has four 30' wide shear walls. All of the lateral loads are resisted by the shear walls. Some of the gravity loads in the building are carried by a separate vertical load resisting system, with the shear walls carrying the remainder of the gravity loads.

The model used for the unsymmetric building has 8 shear wall elements, 16 vertical degrees of freedom, 2 translational degrees of freedom at each level and one rotational degree of freedom at each level as shown in Figure 24. All 16 of the vertical degrees of freedom are condensed out, leaving the 2 translational and and one rotational degrees of freedom at each level, Gdof 17 through 22, for the dynamic analysis. The separate vertical load resisting system is not included in the model. The geometric stiffness of the unsymmetric building reflects the gravity loads on the structure.

For the first solution the natural frequency and mode shapes of the structure are calculated. For the second solution, the nonlinear response to the first 10 seconds of the two component 1940 El Centro earthquake is calculated. The north south

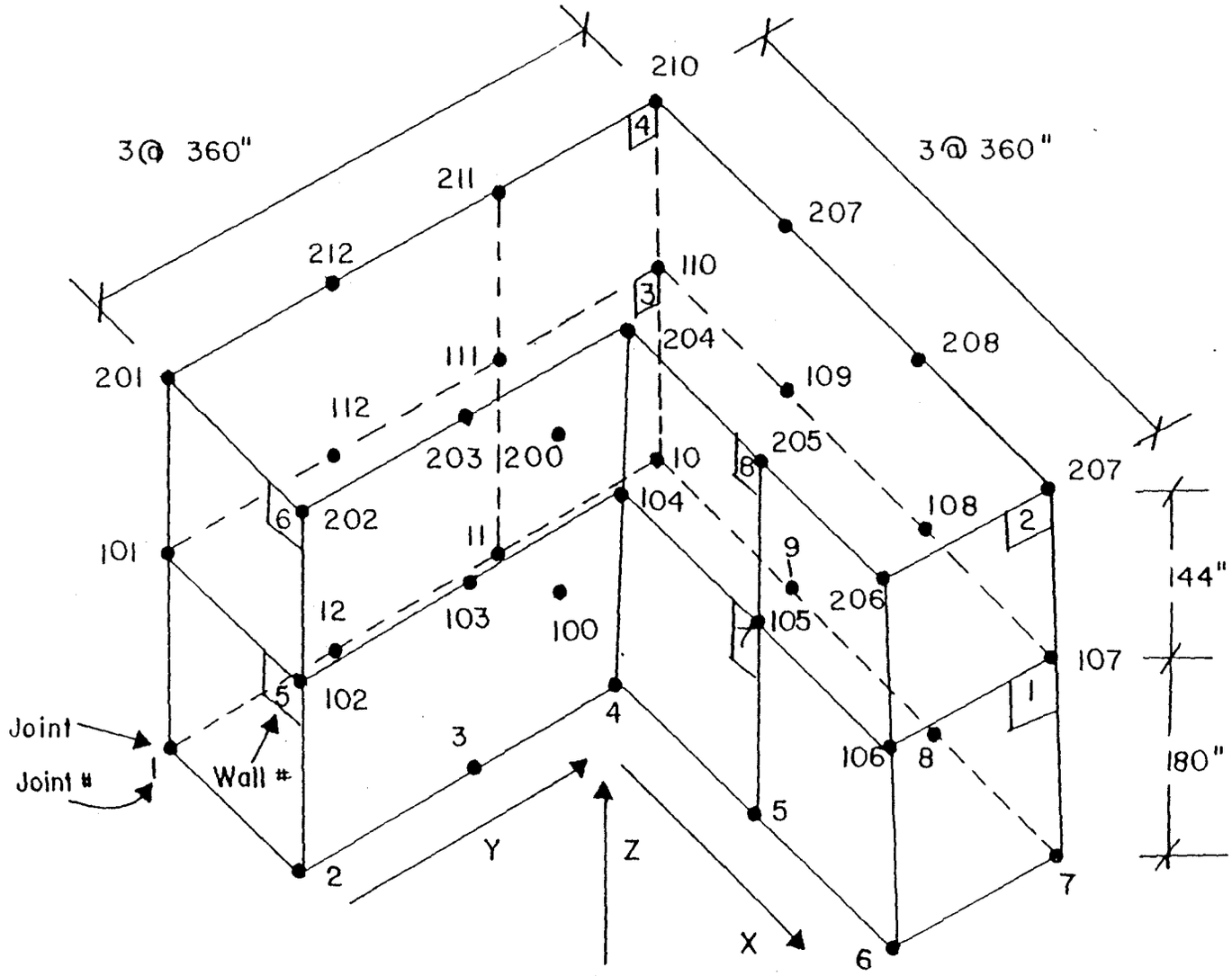


Figure 24. Two-story Unsymmetric Building: Dimensions, Joints and Walls

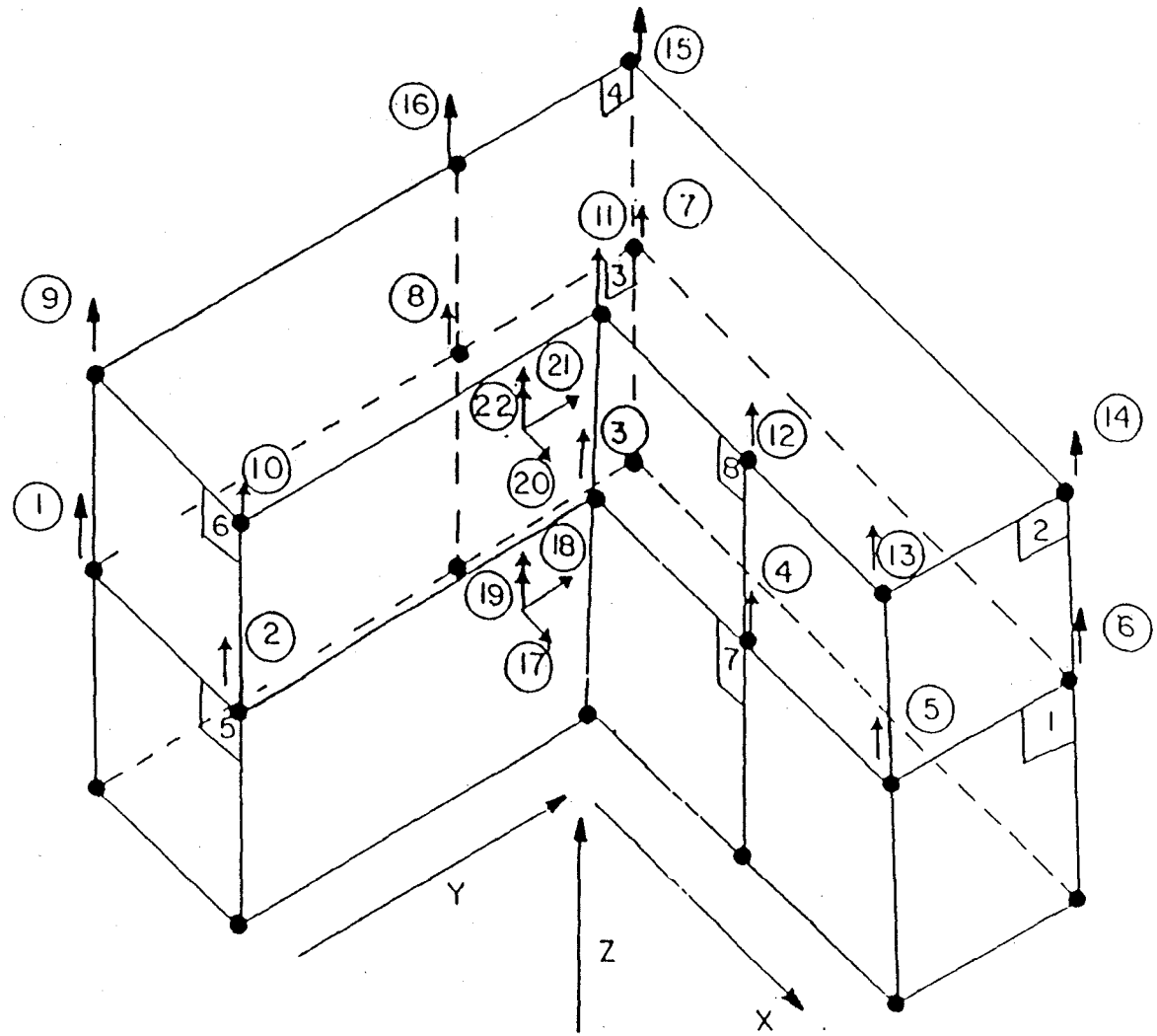


Figure 25. Two-story Unsymmetric building: Degrees of Freedom

component of the El Centro earthquake is stored in the file accessed by unit 92, while the east west component is stored in the file accessed by unit 93. For the third solution the natural frequency and mode shapes of the damaged structure are calculated.

2. Input Data.

```
 ECHO OF INPUT DATA

LINE  ....|...10....|...20....|...30....|...40....|...50....|...60....|...70....|...80
   1: 'STRUCTURE DEFINATION'
   2: 'BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG'
   3:  38 1 12 1 2   12.
   4:     1   0    0     0    1 0
   5:     2  50    0     0    1 0
   6:     5  50   50     0    1 0
   7:     4  50   60     0    1 0
   8:     5  60   60     0    1 0
   9:     6  90   60     0    1 0
  10:     7  90   90     0    1 0
  11:     8  60   90     0    1 0
  12:     9  50   90     0    1 0
  13:    10   0   90     0    1 0
  14:    11   0   60     0    1 0
  15:    12   0   30     0    1 0
  16:   100  54.7598   55.2602     15   1 0
  17:   101   0    0    15    1 0
  18:   102  50    0    15    1 0
  19:   103  50   30    15    1 0
  20:   104  50   60    15    1 0
  21:   105  60   60    15    1 0
  22:   106  90   60    15    1 0
  23:   107  90   90    15    1 0
  24:   108  60   90    15    1 0
  25:   109  50   90    15    1 0
  26:   110   0   90    15    1 0
  27:   111   0   60    15    1 0
  28:   112   0   30    15    1 0
  29:   200  54.4811   55.5191    27   1 0
  30:   201   0    0    27    1 0
  31:   202  30    0    27    1 0
  32:   203  30   30    27    1 0
  33:   204  30   60    27    1 0
  34:   205  60   60    27    1 0
  35:   206  90   60    27    1 0
  36:   207  90   90    27    1 0
  37:   208  60   90    27    1 0
  38:   209  30   90    27    1 0
  39:   210   0   90    27    1 0
  40:   211   0   60    27    1 0
  41:   212   0   30    27    1 0
  42:       1 0 0     0 1 0
  43:     1  1 1 1 1 1 1   11  1
  44:     0  0 0 0 0 2 2 0   0  0
  45:   100  0 0 2 0 0 0 0   0  0
  46:   105  0 0 2 0 0 0 0   0  0
  47:   108  0 0 2 0 0 0 0   0  0
  48:   109  0 0 2 0 0 0 0   0  0
  49:   112  0 0 2 0 0 0 0   0  0
  50:   200  0 0 2 0 0 0 0   0  0
  51:   205  0 0 2 0 0 0 0   0  0
  52:   208  0 0 2 0 0 0 0   0  0
  53:   209  0 0 2 0 0 0 0   0  0
  54:   212  0 0 2 0 0 0 0   0  0
  55:     0  0 0 1 0 0 0 0   0  0
  56:     1 100 101 11 1
```

```
ECHO OF INPUT DATA

LINE  ....|...10....|...20....|...30....|...40....|...50....|...60....|...70....|...80
  57:     1 200 201 11 1
  58:     5
  59: 'AXLMOD ' 7.68E6 6.91E6  7.68E3  257.6  .9 .2  15 .2
  60: 'BEND1 BOT' 9 10  5.5588SE-6      .2
  61:     44100.0       46550.0       56644.0       66150.0       68600.0
  62:     71050.0       75500.0       75960.0       77179.3
  63:   9.111390E-07  1.250030E-06  3.558854E-06  7.359490E-06  1.065460E-05
  64:   1.641140E-05  2.357860E-05  3.125210E-05  3.558855E-05
  65: 'BEND1 TOP' 10 10 3.32151E-6      .2
  66:     37440.0       38880.0       51840.0       52450.4       60480.0
  67:     61920.0       64800.0       67680.0       70560.0       73506.5
  68:   6.782970E-07  8.315580E-07  3.210960E-06  5.521510E-06  5.859730E-06
  69:   6.857840E-06  1.025430E-05  1.602140E-05  2.589400E-05  5.521510E-05
  70: 'SHEAR1 BOT' 10 10 2.90442E-4     .2
  71:     140.000       160.000       210.000       251.200       270.000
  72:     280.000       290.000       300.000       310.000       360.5
  73:   5.650959E-05  7.255400E-05  2.177690E-04  2.904420E-04  4.957469E-04
  74:   6.590021E-04  8.475210E-04  1.096060E-03  1.334810E-03  2.904420E-03
  75: 'SHEAR1 TOP' 10 10 4.851258E-4     .2
  76:     250.000       260.000       540.000       564.100       450.000
  77:     440.000       460.000       480.000       500.000       539.2
  78:   1.146500E-04  1.411250E-04  3.816141E-04  4.851258E-04  8.824540E-04
  79:   1.005020E-03  1.372420E-03  1.883950E-03  2.587450E-03  4.851256E-03
  80:     1 1 2 .FALSE.
  81:     8
  82: 'SHEAR WALL' 'NS F' 2  4 1 107 106   6   7  1.00  250.5 0 0
  83: 'SHEAR WALL' 'NS S' 3  5 1 207 206 106 107  1.00   71.1 0 0
  84: 'SHEAR WALL' 'NS F' 2  4 1 110 111  11  10  1.00  250.5 0 0
  85: 'SHEAR WALL' 'NS S' 3  5 1 210 211 111 110  1.00   71.1 0 0
  86: 'SHEAR WALL' 'EW F' 2  4 1 102 101   1   2  1.00  250.5 0 0
  87: 'SHEAR WALL' 'EW S' 3  5 1 202 201 101 102  1.00   71.1 0 0
  88: 'SHEAR WALL' 'EW F' 2  4 1 105 104   4   5  1.00  250.5 0 0
  89: 'SHEAR WALL' 'EW S' 3  5 1 205 204 104 105  1.00   71.1 0 0
  90: 24  1 .FALSE.
  91:   101  .067547 .067547 .067547   0 0 0   0 0 0   0 0
  92:   102  .067547 .067547 .067547   0 0 0   0 0 0   0 0
  93:   106  .067547 .067547 .067547   0 0 0   0 0 0   0 0
  94:   107  .067547 .067547 .067547   0 0 0   0 0 0   0 0
  95:   110  .074017 .074017 .074017   0 0 0   0 0 0   0 0
  96:   201  .148486 .148486 .148486   0 0 0   0 0 0   0 0
  97:   202  .148486 .148486 .148486   0 0 0   0 0 0   0 0
  98:   206  .148486 .148486 .148486   0 0 0   0 0 0   0 0
  99:   207  .148486 .148486 .148486   0 0 0   0 0 0   0 0
 100:   210  .148486 .148486 .148486   0 0 0   0 0 0   0 0
 101:   105  .051242 .051242 .0        0 0 0   0 0 0   0 0
 102:   108  .051242 .051242 .0        0 0 0   0 0 0   0 0
 103:   109  .057712 .057712 .0        0 0 0   0 0 0   0 0
 104:   112  .051242 .051242 .0        0 0 0   0 0 0   0 0
 105:   205  .129270 .129270 .0        0 0 0   0 0 0   0 0
 106:   208  .129270 .129270 .0        0 0 0   0 0 0   0 0
 107:   209  .129270 .129270 .0        0 0 0   0 0 0   0 0
 108:   212  .129270 .129270 .0        0 0 0   0 0 0   0 0
 109:   105  .079175 .079175 .079157   0 0 0   0 0 0   0 0
 110:   111  .085665 .085665 .085665   0 0 0   0 0 0   0 0
 111:   205  .192158 .192158 .192158   0 0 0   0 0 0   0 0
 112:   211  .192158 .192158 .192158   0 0 0   0 0 0   0 0

ECHO OF INPUT DATA

LINE  ....|...10....|...20....|...30....|...40....|...50....|...60....|...70....|...80
 113:   104  .097508 .097508 .097508   0 0 0   0 0.0   0 0
 114:   204  .255851 .255851 .255851   0 0 0   0 0 0   0 0
 115:   1.6845  .000635589
 116:       'SOL05 - SOLUTION'
 117:       'NATURAL FREQUENCIES AND MODE SHAPES'
 118:       'FREQUENCY'  6   0
 119: 'SOL02 - SOLUTION'
 120: 'NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75'
 121: 'LINEAR' 0 .FALSE. .TRUE.
 122: 99999 40  2400 .FALSE.
 123:   0.00 .00025 10.    386.4
 124:     'ELE' 1 21     7 1 1
 125:     'JOINT FX' 100 S1 1 100 S
 126:     'JOINT FY' 100 S2 1 100 S
 127:     'JOINT MZ' 100 S3 1 100 S
 128:     'ENERGY  '  00 S7 0   0 0
 129:     'SUMRCT  '  00 S8 0   0 0
 130:     'END' 0  0       0  0 0
 131:   0 0 0 'END'              0 0 0   0 0 0
 132:   2   1.75 .00  .01 .FALSE.
 133:   1 0 0   0 1 0     V1, V2
 134:  92 2400 2 .TRUE. .FALSE.   .TRUE.
 135: (A/A)
 136: (F9.6/(8F9.6))
 137:  93 2400 1 .TRUE. .FALSE.   .TRUE.
 138: (A/A)
 139: (F9.6/(8F9.6))
 140: 'READ INC=1000 UNIT=57 '
 141:       'SOL05 - SOLUTION'
 142:       'NATURAL FREQUENCIES AND MODE SHAPES'
 143:       'FREQUENCY'  6   0
 144: 'STOP'
```

3. Output.

```
STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG                    TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....:

*** PROGRAM FEM ***          DOUBLE PRECISION VERSION


   NODE CONSTRAINTS

 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   101
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   102
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   105
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   104
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   105
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   106
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   107
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   108
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   109
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   110
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   111
 XY-PLANE       CONSTRAINT, MASTER:   100 SLAVE:   112
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   201
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   202
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   205
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   204
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   205
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   206
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   207
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   208
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   209
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   210
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   211
 XY-PLANE       CONSTRAINT, MASTER:   200 SLAVE:   212

   NODE COORDINATES AND DEGREES OF FREEDOM

 TOTAL NUMBER OF DEGREES OF FREEDOM.............   156
 NUMBER OF DEGREES OF FREEDOM CONDENSED OUT.....    16
 NUMBER OF FREE DEGREES OF FREEDOM..............     6
 NUMBER OF RESTRAINED DEGREES OF FREEDOM........   154

  NODE COS#    X-COORD        Y-COORD        Z-COORD        FX     FY     FZ     MX     MY     MZ
    1    1   0.00000E+00    0.00000E+00    0.00000E+00   25-R   2+-R   25-R   71-R   72-R   26-R
    2    1     560.00       0.00000E+00    0.00000E+00   27-R   28-R   29-R   73-R   74-R   30-R
    3    1     560.00         560.00       0.00000E+00   31-R   32-R   33-R   75-R   76-R   34-R
    4    1     560.00         720.00       0.00000E+00   35-R   36-R   37-R   77-R   78-R   38-R
    5    1     720.00         720.00       0.00000E+00   39-R   40-R   41-R   79-R   80-R   42-R
    6    1    1080.0          720.00       0.00000E+00   43-R   44-R   45-R   81-R   82-R   46-R
    7    1    1080.0         1080.0        0.00000E+00   47-R   48-R   49-R   83-R   84-R   50-R
    8    1     720.00        1080.0        0.00000E+00   51-R   52-R   53-R   85-R   86-R   54-R
    9    1     560.00        1080.0        0.00000E+00   55-R   56-R   57-R   87-R   88-R   58-R
   10    1   0.00000E+00     1080.0        0.00000E+00   59-R   60-R   61-R   89-R   90-R   62-R
   11    1   0.00000E+00      720.00       0.00000E+00   63-R   64-R   65-R   91-R   92-R   66-R
   12    1   0.00000E+00      560.00       0.00000E+00   67-R   68-R   69-R   95-R   94-R   70-R
  100    1     416.88         663.12         180.00      17     18     95-R   96-R   97-R    19
  101    1   0.00000E+00    0.00000E+00      180.00      17-C   18-C    1     98-R   99-R   19-C
  102    1     360.00       0.00000E+00      180.00      17-C   18-C    2     100-R  101-R  19-C
  103    1     360.00         360.00         180.00      17-C   18-C   102-R  103-R  104-R  19-C
  104    1     360.00         720.00         180.00      17-C   18-C    3     105-R  106-R  19-C
  105    1     720.00         720.00         180.00      17-C   18-C    4     107-R  108-R  19-C
  106    1    1080.0          720.00         180.00      17-C   18-C    5     109-R  110-R  19-C
  107    1    1080.0         1080.0          180.00      17-C   18-C    6     111-R  112-R  19-C
  108    1     720.00        1080.0          180.00      17-C   18-C   113-R  114-R  115-R  19-C
  109    1     360.00        1080.0          180.00      17-C   18-C   116-R  117-R  118-R  19-C
  110    1   0.00000E+00     1080.0          180.00      17-C   18-C    7     119-R  120-R  19-C
  111    1   0.00000E+00      720.00         180.00      17-C   18-C    8     121-R  122-R  19-C
  112    1   0.00000E+00      560.00         180.00      17-C   18-C   123-R  124-R  125-R  19-C
  200    1     415.77         666.25         524.00      20     21     126-R  127-R  128-R   22
  201    1   0.00000E+00    0.00000E+00      524.00      20-C   21-C    9     129-R  130-R  22-C
  202    1     360.00       0.00000E+00      524.00      20-C   21-C   10     131-R  132-R  22-C
  203    1     360.00         360.00         524.00      20-C   21-C   133-R  134-R  135-R  22-C
  204    1     360.00         720.00         524.00      20-C   21-C   11     136-R  137-R  22-C
  205    1     720.00         720.00         524.00      20-C   21-C   12     138-R  139-R  22-C
  206    1    1080.0          720.00         524.00      20-C   21-C   13     140-R  141-R  22-C
  207    1    1080.0         1080.0          524.00      20-C   21-C   14     142-R  143-R  22-C
  208    1     720.00        1080.0          524.00      20-C   21-C   144-R  145-R  146-R  22-C
  209    1     360.00        1080.0          524.00      20-C   21-C   147-R  148-R  149-R  22-C
  210    1   0.00000E+00     1080.0          524.00      20-C   21-C   15     150-R  151-R  22-C
  211    1   0.00000E+00      720.00         524.00      20-C   21-C   16     152-R  153-R  22-C
  212    1   0.00000E+00      560.00         524.00      20-C   21-C   154-R  155-R  156-R  22-C
     NOTE: R - RESTRAINED DEGREE OF FREEDOM
           C - CONSTRAINED DEGREE OF FREEDOM

   DIRECTION COSINES ...

COSC  1>  VX:  1.00000 I +0.00000 J +0.00000 K   VY:  0.00000 I +1.00000 J +0.00000 K   VZ:  0.00000 I +0.00000 J +1.00000 K

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG                    TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....:


   AXLMOD HYSTERESIS MODEL- FOR UNIT LENGTH MEMBER
   ===============================================

MAT.        SC            ST1           ST2          PY          ALPHA        BETA       MAX DUCT.     BETA-DI
  1   0.768000E+07   0.691000E+07     7680.00       257.600     0.900000     0.200000      15.0000      0.200000
```

(1)

BEND1 HYSTERESIS MODEL DATA - UNIT LENGTH MEMBER

BACKBONE CURVE POINTS - MAT.	NI	DV	BETA	MOMENT	ROTATION	
2	10	0.555885E-05	0.200000	44100.0	0.911159E-06	
				46550.0	0.125005E-05	
				56644.0	0.555885E-05	
				66150.0	0.755949E-05	
				68600.0	0.106546E-04	(2)
				71050.0	0.164114E-04	
				73500.0	0.235786E-04	
				75950.0	0.512521E-04	
				77179.3	0.555885E-04	
3	10	0.332151E-05	0.200000	57440.0	0.678297E-06	
				58880.0	0.851558E-06	
				51840.0	0.321096E-05	
				52430.4	0.332151E-05	
				60480.0	0.585973E-05	
				61920.0	0.685784E-05	(3)
				64800.0	0.102545E-04	
				67680.0	0.160214E-04	
				70560.0	0.258940E-04	
				73506.3	0.332151E-04	

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....:

SHEAR1 HYSTERESIS MODEL DATA - UNIT LENGTH MEMBER

BACKBONE CURVE POINTS - MAT.	NI	DV	BETA	STRESS	STRAIN	
4	10	0.290442E-05	0.200000	140.000	0.565094E-04	
				160.000	0.725508E-04	
				210.000	0.217769E-05	
				231.200	0.290442E-05	
				270.000	0.495747E-05	
				280.000	0.639002E-05	(2)
				290.000	0.847521E-05	
				300.000	0.109606E-02	
				310.000	0.133481E-02	
				360.300	0.290442E-02	
5	10	0.485124E-05	0.200000	250.000	0.114650E-05	
				260.000	0.141125E-05	
				340.000	0.581614E-05	
				369.100	0.485124E-05	(3)
				430.000	0.882454E-05	
				440.000	0.100502E-02	
				460.000	0.157242E-02	
				480.000	0.188595E-02	
				500.000	0.258745E-02	
				539.200	0.485124E-02	

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....:

GEOMETRIC STIFFNESS DATA

KGLOAD = 1, LOAD= F(INPUT)*(1.00+ACCEL)
F(INPUT) IS POSITIVE IN COMPRESSION
ACCEL IS POSITIVE GLOBAL Z-AXIS ACCELERATION
KGTYPE = 1, LUMPED FORMULATION
KGFORM = 2, SEPERATE GLOBAL KG IS FORMED

- THE GEOMETRIC STIFFNESS MATRIX IS NOT CONDENSED WITH THE STRUCTURAL STIFFNESS MATRIX.

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....:

ELEMENT 05, R/C SHEAR WALL ELEMENT

	ELEM #	BEND MATL	SHEAR MATL	AXIAL MATL	JOINT #1	JOINT #2	JOINT #3	JOINT #4	LENGTH	WIDTH	ALPHA	PKG
NS F	1	2	4	1	107	106	6	7	180.0	560.0	1.000	250.5
NS S	2	5	5	1	207	206	106	107	144.0	560.0	1.000	71.10
NS F	3	2	4	1	110	111	11	10	180.0	560.0	1.000	250.5
NS S	4	5	5	1	210	211	111	110	144.0	560.0	1.000	71.10
EW F	5	2	4	1	102	101	1	2	180.0	560.0	1.000	250.5
EW S	6	5	5	1	202	201	101	102	144.0	560.0	1.000	71.10
EW F	7	2	4	1	105	104	4	5	180.0	560.0	1.000	250.5
EW S	8	5	5	1	205	204	104	105	144.0	560.0	1.000	71.10

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....:

LUMPED NODE MASSES

NODE	MX	MY	MZ	RXX	RYY	RZZ	RXY	RXZ	RYZ	IGEN	INC
101	6.7547E-02	6.7547E-02	6.7547E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
102	6.7547E-02	6.7547E-02	6.7547E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
106	6.7547E-02	6.7547E-02	6.7547E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
107	6.7547E-02	6.7547E-02	6.7547E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
110	7.4017E-02	7.4017E-02	7.4017E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
201	0.1485	0.1485	0.1485	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
202	0.1485	0.1485	0.1485	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
206	0.1485	0.1485	0.1485	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
207	0.1485	0.1485	0.1485	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
210	0.1485	0.1485	0.1485	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
105	5.1242E-02	5.1242E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
108	5.1242E-02	5.1242E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
109	5.7712E-02	5.7712E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
112	5.1242E-02	5.1242E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
203	0.1295	0.1295	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
208	0.1295	0.1295	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
209	0.1295	0.1295	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
212	0.1295	0.1295	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
105	7.9173E-02	7.9173E-02	7.9157E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
111	8.5663E-02	8.5663E-02	8.5663E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
205	0.1922	0.1922	0.1922	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
211	0.1922	0.1922	0.1922	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
104	9.7508E-02	9.7508E-02	9.7508E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0
204	0.2558	0.2558	0.2558	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0

- THE MASS MATRIX IS NOT CONDENSED WITH THE STRUCTURAL STIFFNESS MATRIX.

PROPORTIONAL DAMPING COIEFICIENTS

ALPHA= 1.6845 BETA= 6.55389E-04

--- MEMORY UTILIZATION ..... ---
--- IZ= 7727. MEM= 1.54% ---
--- ELAPSED CPU TIME 0.46 SEC ---
--- TOTAL CPU TIME 0.46 SEC ---

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NATURAL FREQUENCIES AND MODE SHAPES TIME: 17:01:23, DATE: 05/31/89

SOLUTION #5, DETERMINE EIGENVALUES AND EIGENVECTORS

OPTION:.....NATURAL FREQUENCIES
NUMBER OF EIGENVALUES:..... 6
PREFORMANCE INDEX: 0.6027446578153274E-02

MODE:	1	2	3	4	5	6
FREQ (RAD/SEC):	33.480	40.526	60.262	192.82	234.55	346.54
FREQUENCY (HZ):	5.3286	6.4498	9.5910	30.689	37.326	55.154
PERIOD (SEC):	0.18767	0.15504	0.10426	3.2585E-02	2.6790E-02	1.8131E-02
EIGENVECTORS:						
DOF(17)	0.51200	-0.20888	0.51856	1.0000	-0.41158	1.0000
DOF(18)	0.20820	0.51553	0.21111	0.40965	1.0000	0.41860
DOF(19)	-5.3603E-04	-6.2085E-03	2.0747E-05	-1.0575E-05	7.0967E-07	5.8117E-03
DOF(20)	1.0000	-0.40681	1.0000	-0.22480	9.1964E-02	-0.21311
DOF(21)	0.40853	1.0000	0.40021	-9.2995E-02	-0.22541	-8.8160E-02
DOF(22)	-1.0451E-03	1.4042E-03	4.0547E-05	2.4746E-04	1.5991E-06	-8.7525E-04

--- MEMORY UTILIZATION ..... ---
--- IZ= 7967. MEM= 1.59% ---
--- ELAPSED CPU TIME 0.09 SEC ---
--- TOTAL CPU TIME 0.55 SEC ---

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 TIME: 17:01:25, DATE: 05/31/89

SOLUTION #2, LINEAR ACCELERATION METHOD OF NUMERICAL INTEGRATION

INITIAL TIME ..... 0.000000E+00
TIME STEP 2.500000E-04
FINAL TIME 10.0000
ACCELERATION DUE TO GRAVITY 386.400
STEP INTERVAL FOR PRINTING 99999

DATA WRITTEN TO FILES

ELEMENT # 1 IS WRITTEN TO UNIT # 21
ELEMENT # 2 IS WRITTEN TO UNIT # 22
ELEMENT # 3 IS WRITTEN TO UNIT # 23
ELEMENT # 4 IS WRITTEN TO UNIT # 24
ELEMENT # 5 IS WRITTEN TO UNIT # 25
ELEMENT # 6 IS WRITTEN TO UNIT # 26
ELEMENT # 7 IS WRITTEN TO UNIT # 27
ELEMENT # 8 IS WRITTEN TO UNIT # 28
DEGREE OF FREEDOM # 17 IS WRITTEN TO UNIT # 51 JOINT: 100 DIRECTION: FX
DEGREE OF FREEDOM # 20 IS WRITTEN TO UNIT # 54 JOINT: 200 DIRECTION: FX
DEGREE OF FREEDOM # 18 IS WRITTEN TO UNIT # 52 JOINT: 100 DIRECTION: FY
DEGREE OF FREEDOM # 21 IS WRITTEN TO UNIT # 55 JOINT: 200 DIRECTION: FY
DEGREE OF FREEDOM # 19 IS WRITTEN TO UNIT # 53 JOINT: 100 DIRECTION: MZ
DEGREE OF FREEDOM # 22 IS WRITTEN TO UNIT # 56 JOINT: 200 DIRECTION: MZ
ENERGY BALANCE IS WRITTEN TO UNIT # 57
SUMMATION OF REACTIONS IS WRITTEN TO UNIT # 58

GROUND ACCELERATION RECORD

INPUTING TRANSLATIONAL ACCELERATION RECORD # 1, FROM UNIT: 92
ELCENTRO- 1940 EQ DATA FORMAT: (8F9.6)
DT=0.01 SEC <<< E-W DIRECTION >>> 2400 DATA
THE RECORD CONTAINS 2400 POINTS, BEGINING AT TIME: 0.000000E+00 WITH A TIME INCREMENT OF: 1.000000E-02
DIRECTION OF ACCELERATION: 0.00000 I +1.00000 J +0.00000 K

	MAXIMUM	AT	TIME	MINIMUM	AT	TIME	AVERAGE	STANDARD DEV.	RMS
INPUT ACCELERATION	0.21490		1.8700	-0.16138		11.050	-4.2494E-05	5.4490E-02	5.4376E-02

GROUND ACCELERATION RECORD

INPUTING TRANSLATIONAL ACCELERATION RECORD # 2, FROM UNIT: 95
ELCENTRO- 1940 EQ DATA
DT=0.01 SEC <<< N-S DIRECTION >>> 2400 DATA
THE RECORD CONTAINS 2400 POINTS, BEGINING AT TIME: 0.000000E+00 WITH A TIME INCREMENT OF: 1.000000E-02
DIRECTION OF ACCELERATION: 1.00000 I +0.00000 J +0.00000 K

	MAXIMUM	AT	TIME	MINIMUM	AT	TIME	AVERAGE	STANDARD DEV.	RMS
INPUT ACCELERATION	0.29242		2.2200	-0.31450		2.0000	1.05550E-04	6.8568ZE-02	6.855+0E-02

	MAXIMUM	AT	TIME	MINIMUM	AT	TIME	AVERAGE	STANDARD DEV.	RMS
X-AXIS ACCELERATION	0.51174		2.2200	-0.55005		2.0000	1.84712E-04	0.11999	0.11997
Y-AXIS ACCELERATION	0.32607		1.8700	-0.28242		11.050	-7.456+8E-05	9.5e558E-02	9.5e159E-02
Z-AXIS ACCELERATION	0.00000E+00		0.00000E+00	0.00000E+00		0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 TIME: 17:01:25, DATE: 05/31/89

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

GCS MAXIMUM DISPLACEMENTS

NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

NODE	DX	DY	DZ	RX	RY	RZ
1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
2	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
3	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
4	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
5	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
6	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
7	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
8	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
9	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
10	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
11	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
12	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
100	0.904062	0.557587	0.000000E+00	0.000000E+00	0.000000E+00	-8.952485E-04
101	0.511729	0.929962	0.506021	0.000000E+00	0.000000E+00	-8.952485E-04
102	0.511729	0.608595	-0.506021	0.000000E+00	0.000000E+00	-8.952485E-04
103	0.655298	0.608595	0.000000E+00	0.000000E+00	0.000000E+00	-8.952485E-04
104	0.954868	0.608595	0.849100	0.000000E+00	0.000000E+00	-8.952485E-04
105	0.954868	0.286825	-0.849100	0.000000E+00	0.000000E+00	-8.952485E-04
106	0.954868	-5.474606E-02	0.464751	0.000000E+00	0.000000E+00	-8.952485E-04
107	1.27644	-5.474606E-02	-0.464751	0.000000E+00	0.000000E+00	-8.952485E-04
108	1.27644	0.286825	0.000000E+00	0.000000E+00	0.000000E+00	-8.952485E-04
109	1.27644	0.608595	0.000000E+00	0.000000E+00	0.000000E+00	-8.952485E-04
110	1.27644	0.929962	-0.858259	0.000000E+00	0.000000E+00	-8.952485E-04
111	0.954868	0.929962	0.858259	0.000000E+00	0.000000E+00	-8.952485E-04
112	0.655298	0.929962	0.000000E+00	0.000000E+00	0.000000E+00	-8.952485E-04
200	1.58562	0.979505	0.000000E+00	0.000000E+00	0.000000E+00	-1.557565E-05
201	0.545927	1.62378	0.520578	0.000000E+00	0.000000E+00	-1.557565E-05
202	0.545927	1.06306	-0.520578	0.000000E+00	0.000000E+00	-1.557565E-05
203	1.10665	1.06306	0.000000E+00	0.000000E+00	0.000000E+00	-1.557565E-05
204	1.66757	1.06306	0.865567	0.000000E+00	0.000000E+00	-1.557565E-05
205	1.66757	0.502536	-0.865567	0.000000E+00	0.000000E+00	-1.557565E-05
206	1.66757	-5.858670E-02	0.477672	0.000000E+00	0.000000E+00	-1.557565E-05
207	2.22809	-5.858670E-02	-0.477672	0.000000E+00	0.000000E+00	-1.557565E-05
208	2.22809	0.502536	0.000000E+00	0.000000E+00	0.000000E+00	-1.557565E-05
209	2.22809	1.06306	0.000000E+00	0.000000E+00	0.000000E+00	-1.557565E-05
210	2.22809	1.62378	-0.850990	0.000000E+00	0.000000E+00	-1.557565E-05
211	1.66757	1.62378	0.850990	0.000000E+00	0.000000E+00	-1.557565E-05
212	1.10665	1.62378	0.000000E+00	0.000000E+00	0.000000E+00	-1.557565E-05

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

GCS MAXIMUM VELOCITIES

NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

NODE	DX	DY	DZ	RX	RY	RZ
1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
2	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
3	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
4	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
5	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
6	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
7	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
8	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
9	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
10	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
11	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
12	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
100	-10.4173	5.58568	0.000000E+00	0.000000E+00	0.000000E+00	9.567675E-05
101	-4.07230	1.59715	-5.57271	0.000000E+00	0.000000E+00	9.567675E-05
102	-4.07280	5.04149	5.57271	0.000000E+00	0.000000E+00	9.567675E-05
103	-7.51716	5.04149	0.000000E+00	0.000000E+00	0.000000E+00	9.567675E-05
104	-10.9615	5.04149	-9.61586	0.000000E+00	0.000000E+00	9.567675E-05
105	-10.9615	8.48586	9.61586	0.000000E+00	0.000000E+00	9.567675E-05
106	-10.9615	11.9502	6.16408	0.000000E+00	0.000000E+00	9.567675E-05
107	-14.4059	11.9502	-6.16408	0.000000E+00	0.000000E+00	9.567675E-05
108	-14.4059	8.48586	0.000000E+00	0.000000E+00	0.000000E+00	9.567675E-05
109	-14.4059	5.04149	0.000000E+00	0.000000E+00	0.000000E+00	9.567675E-05
110	-14.4059	1.59715	-7.21577	0.000000E+00	0.000000E+00	9.567675E-05
111	-10.9615	1.59715	7.21577	0.000000E+00	0.000000E+00	9.567675E-05
112	-7.51716	1.59715	0.000000E+00	0.000000E+00	0.000000E+00	9.567675E-05
200	-17.7672	9.62214	0.000000E+00	0.000000E+00	0.000000E+00	1.659585E-02
201	-6.71189	2.75606	-5.45652	0.000000E+00	0.000000E+00	1.659585E-02
202	-6.71189	8.72985	5.45652	0.000000E+00	0.000000E+00	1.659585E-02
203	-12.6857	8.72985	0.000000E+00	0.000000E+00	0.000000E+00	1.659585E-02
204	-18.6594	8.72985	-9.84467	0.000000E+00	0.000000E+00	1.659585E-02
205	-18.6594	14.7056	9.84467	0.000000E+00	0.000000E+00	1.659585E-02
206	-18.6594	20.6774	6.45004	0.000000E+00	0.000000E+00	1.659585E-02
207	-24.6332	20.6774	-6.45004	0.000000E+00	0.000000E+00	1.659585E-02
208	-24.6332	14.7036	0.000000E+00	0.000000E+00	0.000000E+00	1.659585E-02
209	-24.6332	8.72983	0.000000E+00	0.000000E+00	0.000000E+00	1.659585E-02
210	-24.6332	2.75606	-7.59555	0.000000E+00	0.000000E+00	1.659585E-02
211	-18.6594	2.75606	7.59555	0.000000E+00	0.000000E+00	1.659585E-02
212	-12.6857	2.75606	0.000000E+00	0.000000E+00	0.000000E+00	1.659585E-02

STRUCTURE....: BUILDING B2, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

GCS MAXIMUM ACCELERATIONS

NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

NODE	DX	DY	DZ	RX	RY	RZ
1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
2	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
3	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
4	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
5	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
6	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
7	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
8	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
9	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
10	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
11	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
12	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
100	242.837	131.458	0.000000E+00	0.000000E+00	0.000000E+00	-0.216085
101	99.5459	221.539	-219.546	0.000000E+00	0.000000E+00	-0.216085
102	99.5459	143.749	219.546	0.000000E+00	0.000000E+00	-0.216085
103	177.336	143.749	0.000000E+00	0.000000E+00	0.000000E+00	-0.216085
104	255.127	143.749	277.762	0.000000E+00	0.000000E+00	-0.216085
105	255.127	65.9581	-277.762	0.000000E+00	0.000000E+00	-0.216085
106	255.127	-11.8325	-225.538	0.000000E+00	0.000000E+00	-0.216085
107	332.918	-11.8325	225.538	0.000000E+00	0.000000E+00	-0.216085
108	332.918	65.9581	0.000000E+00	0.000000E+00	0.000000E+00	-0.216085
109	332.918	143.749	0.000000E+00	0.000000E+00	0.000000E+00	-0.216085
110	332.918	221.539	-245.012	0.000000E+00	0.000000E+00	-0.216085
111	255.127	221.539	245.012	0.000000E+00	0.000000E+00	-0.216085
112	177.336	221.539	0.000000E+00	0.000000E+00	0.000000E+00	-0.216085
200	422.660	226.354	0.000000E+00	0.000000E+00	0.000000E+00	-0.520782
201	208.946	559.085	190.171	0.000000E+00	0.000000E+00	-0.520782
202	208.946	243.605	-190.171	0.000000E+00	0.000000E+00	-0.520782
203	524.427	243.605	0.000000E+00	0.000000E+00	0.000000E+00	-0.520782
204	439.909	243.605	254.263	0.000000E+00	0.000000E+00	-0.520782
205	439.909	128.122	-254.263	0.000000E+00	0.000000E+00	-0.520782
206	439.909	12.6400	212.052	0.000000E+00	0.000000E+00	-0.520782
207	555.390	12.6400	-212.052	0.000000E+00	0.000000E+00	-0.520782
208	555.390	128.122	0.000000E+00	0.000000E+00	0.000000E+00	-0.520782
209	555.390	243.605	0.000000E+00	0.000000E+00	0.000000E+00	-0.520782
210	555.390	359.085	-178.809	0.000000E+00	0.000000E+00	-0.520782
211	439.909	359.085	178.809	0.000000E+00	0.000000E+00	-0.520782
212	524.427	359.085	0.000000E+00	0.000000E+00	0.000000E+00	-0.520782

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

MAXIMUM GCS RESTRAINT REACTIONS NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

NODE	FX	FY	FZ	MX	MY	MZ
1	0.0000000E+00	0.0000000E+00	-197.0129	0.0000000E+00	0.0000000E+00	0.0000000E+00
2	-250.1184	0.0000000E+00	197.0129	0.0000000E+00	0.0000000E+00	0.0000000E+00
3	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
4	0.0000000E+00	0.0000000E+00	-206.2897	0.0000000E+00	0.0000000E+00	0.0000000E+00
5	-272.2428	0.0000000E+00	206.2897	0.0000000E+00	0.0000000E+00	0.0000000E+00
6	0.0000000E+00	0.0000000E+00	-194.5335	0.0000000E+00	0.0000000E+00	0.0000000E+00
7	0.0000000E+00	245.5584	194.5335	0.0000000E+00	0.0000000E+00	0.0000000E+00
8	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
9	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
10	0.0000000E+00	-262.5301	206.5734	0.0000000E+00	0.0000000E+00	0.0000000E+00
11	0.0000000E+00	0.0000000E+00	-206.5734	0.0000000E+00	0.0000000E+00	0.0000000E+00
12	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
100	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
101	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
102	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
103	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
104	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
105	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
106	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
107	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
108	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
109	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
110	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
111	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
112	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
200	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
201	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
202	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
203	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
204	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
205	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
206	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
207	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
208	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
209	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
210	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
211	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
212	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00	0.0000000E+00
MAX OF ALL						
GCS SUMM.	-505.9169	450.8115	0.9258572E-11	-125848.8	-144080.5	-418525.9

MAXIMUM RESULTANT OF REACTIONS, FORCE= 555.5 MOMENT= 4.4138E+05

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

R/C SHEAR WALL FORCES... MAXIMUM LOADS AND DISPL AT MAXIMUM LOADS NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY

ELEMENT	LOAD	JNT-1	JNT-2	JNT-3	JNT-4	MOMENT	ROTATION	SHEAR	SHEAR DISPL.	AXIAL	AXIAL DISPL.
1	2	107	106	6	7	70160.5	2.5819516-05	-245.440	-6.5710506-02	-1.8401456-12	-4.1584556-17
2	2	207	206	106	107	27851.6	7.2660506-05	195.414	1.277276E-02	2.506996E-12	5.179999E-17
3	2	110	111	11	10	74229.9	4.6569926-05	262.647	8.197792E-02	-7.5+7715E-12	-1.747299E-16
4	2	210	211	111	110	28995.7	7.5639696-05	201.345	1.329650E-02	2.992704E-12	6.192185E-17
5	2	102	101	1	2	70711.4	2.8112296-05	250.724	7.070848E-02	5.+09559E-12	8.826102E-17
6	2	202	201	101	102	50748.0	8.0216356-05	215.528	1.+101026-02	-2.++7125E-12	-+.555515E-17
7	2	105	104	+	5	74269.5	+.678649E-05	275.976	9.955061E-02	-+.970622E-12	-1.14884E-16
8	2	205	204	104	105	50819.5	8.0402866-05	214.024	1.+15380E-02	7.741659E-12	1.608874E-16

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

R/C SHEAR WALL DUCTILITY AND EXCURSION RATIOS

ELEM#	BENDING COMPONENT						SHEAR COMPONENT						AXIAL COMPONENT					
	DEFN 1		DEFN 2		DEFN 3		DEFN 1		DEFN 2		DEFN 3		DEFN 1		DEFN 2		DEFN 3	
	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.	DUCT.	EXCR.
1	+.05	29.34	5.85	25.75	4.08	25.65	1.26	1.25	5.14	27.57	1.81	8.44	0.00	0.00	1.09	0.19	1.00	0.00
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.11	0.11	1.00	0.00
3	7.27	+8.25	7.39	51.70	10.99	41.05	1.57	1.71	5.42	20.51	2.15	8.05	0.00	0.00	1.11	0.21	1.00	0.00
+	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.11	0.20	1.00	0.00
5	+.59	26.64	5.05	22.42	6.11	26.48	1.35	1.42	3.50	18.41	2.06	7.06	0.00	0.00	1.11	0.00	1.00	0.00
6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.11	0.11	1.00	0.00
7	7.56	+9.42	7.28	58.66	10.85	54.62	1.91	5.77	5.84	23.16	2.66	10.94	0.00	0.00	1.09	0.16	1.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.81	0.88	1.00	0.00

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUM VALUES FOR ALL STEPS

ELEM#	BENDING COMPONENT			SHEAR COMPONENT			AXIAL COMPONENT		
	DAMAGE	ESE	PSE	DAMAGE	ESE	PSE	DAMAGE	ESE	PSE
1	0.75305	2.54705E-05	5.5276	0.25252	6.62819E-04	0.55870	0.00000	+.2857+E-34	6.21581E-33
2	0.01519	1.59958E-07	8.27006E-16	0.01856	1.70829E-07	0.00000E+00	0.00000	2.57557E-51	4.52470E-32
3	1.51107	1.64954E-02	5.8875	0.25725	1.61609E-05	0.55717	0.00000	2.84255E-50	7.56860E-33
+	0.01581	+.64029E-04	8.25126E-16	0.01911	5.66462E-04	0.00000E+00	0.00000	2.++177E-51	6.+626+E-32
5	0.80954	1.42972E-02	5.7344	0.22598	1.77779E-05	0.29795	0.00000	1.90840E-51	8.59995E-32
6	0.01677	2.49975E-04	+.57555E-16	0.02026	5.05154E-04	0.00000E+00	0.00000	5.51427E-51	1.+5477E-32
7	1.54247	1.28096E-05	8.1249	0.53434	7.97790E-05	0.48074	0.00000	1.0+901E-50	+.69828E-33
8	0.01681	+.10854E-05	2.97874E-16	0.02051	5.01548E-05	0.00000E+00	0.00000	1.50597E-50	+.5++50E-31

STRUCTURE DAMAGE INDEX= 1.1570

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

MAXIMUN VALUES FOR ALL STEPS

PEAK ENERGY VALUES
MAX INPUT ENERGY..... 5738.1
MAX ELASTIC STRAIN ENERGY..... 123.55
MAX PLASTIC STRAIN ENERGY..... 4094.8
MAX KINETIC ENERGY..... 16.405
MAX DAMPING ENERGY..... 1589.5

--- MEMORY UTILIZATION ..... ---
--- IZ= 21725, MEM= 4.345% ---
--- ELAPSED CPU TIME 4857.27 SEC ---
--- TOTAL CPU TIME 4857.82 SEC ---

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NONLINEAR RESPONSE TO (1940 ELCENTRO NS & EW)*1.75 — TIME: 17:01:25, DATE: 05/31/89

ENERGY DATA IS READ FROM UNIT # 57

STEP	TIME	INPUT	ELASTIC STRAIN	KINETIC	PLASTIC STRAIN	DAMPINED	RELATIVE ERROR %
0	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.0000
1000	0.25000	4.5152	0.14857	4.1255	1.15270E-15	3.1585+E-02	0.1789
2000	0.50000	8.4657	0.89772	7.2519	2.16555E-14	0.39516	-0.9104
3000	0.75000	67.570	0.41480	64.964	5.82791E-14	2.2551	-0.0945
4000	1.0000	82.079	5.2810	72.921	0.28674	5.8929	-0.5695
5000	1.2500	55.664	9.1222	33.398	2.5+31	11.318	-0.9296
6000	1.5000	1650.4	2.9165	1590.5	28.722	17.081	-0.5292
7000	1.7500	622.59	52.765	415.01	136.46	57.684	-2.7817
8000	2.0000	2122.8	72.172	1589.0	395.99	88.145	-1.0614
9000	2.2500	3010.9	27.990	1679.8	1080.6	254.15	-1.0520
10000	2.5000	2715.1	42.772	857.35	1462.6	405.25	-1.2098
11000	2.7500	2497.6	65.692	440.18	1590.5	440.47	-1.5654
12000	3.0000	2465.7	1.9506	457.62	1587.1	470.19	-1.5466
13000	3.2500	2236.6	10.744	156.28	1609.3	485.90	-1.1325
14000	3.5000	2291.3	10.121	97.955	1680.1	524.86	-0.9449
15000	3.7500	2315.9	0.85257	505.17	1765.4	559.47	-0.5148
16000	4.0000	2496.7	21.962	63.126	1851.3	587.32	-0.2808
17000	4.2500	2505.3	18.620	24.772	1865.8	608.07	-0.3980
18000	4.5000	3772.0	17.808	940.84	2115.6	716.58	-0.4977
19000	4.7500	3458.3	40.459	61.955	2550.9	855.32	-0.8201
20000	5.0000	4484.6	48.382	661.52	2848.8	955.57	-0.6595
21000	5.2500	4702.7	99.820	402.15	3176.1	1066.6	-0.8923
22000	5.5000	4851.2	19.270	440.66	3299.4	1121.6	-0.6109
23000	5.7500	4822.7	52.391	223.58	3396.6	1178.8	-0.1812
24000	6.0000	4702.6	9.3645	36.558	3462.0	1202.5	-0.1665
25000	6.2500	4716.5	12.595	6.2502	3485.2	1218.4	-0.0850
26000	6.5000	4758.6	1.5705	41.416	3490.8	1229.4	-0.0968
27000	6.7500	4737.0	5.5428	3.7055	3499.1	1233.6	-0.1055
28000	7.0000	4790.7	3.7615	21.742	3513.5	1254.0	-0.0488
29000	7.2500	4959.9	21.108	58.455	3596.2	1289.2	-0.1028
30000	7.5000	4996.6	2.1952	39.318	3634.1	1324.5	-0.0699
31000	7.7500	5145.6	8.7271	74.698	3699.8	1364.8	-0.0969
32000	8.0000	5313.8	0.87374	254.78	3716.2	1372.5	-0.1986
33000	8.2500	5186.1	2.1959	93.194	3728.7	1381.7	-0.3798
34000	8.5000	5157.0	11.902	13.213	3763.0	1395.4	-0.5156
35000	8.7500	5287.3	6.9645	110.48	3779.1	1420.7	-0.5668
36000	9.0000	5447.9	8.2668	271.16	3911.0	1484.3	-0.4749
37000	9.2500	5585.5	21.891	64.100	4000.6	1525.0	-0.4669
38000	9.5000	5685.3	1.9463	109.43	4037.1	1552.9	-0.2859
39000	9.7500	5675.1	7.2102	56.192	4064.8	1575.3	-0.1477
MAXIMUN		5738.0	122.87	2260.7	4092.5	1589.2	

GROSS RE: 0.4489%

--- MEMORY UTILIZATION ..... ---
--- IZ= 21725, MEM= 4.345% ---
--- ELAPSED CPU TIME 0.81 SEC ---
--- TOTAL CPU TIME 4858.63 SEC ---

STRUCTURE....: BUILDING 82, L SHAPED TWO STORY BOX STRUCTURE, WITH KG — TIME: 17:01:22, DATE: 05/31/89
SOLUTION.....: NATURAL FREQUENCIES AND MODE SHAPES — TIME: 18:44:16, DATE: 05/31/89

SOLUTION #3, DETERMINE EIGENVALUES AND EIGENVECTORS

OPTION:.....NATURAL FREQUENCIES
NUMBER OF EIGENVALUES:..... 6
PREFORMANCE INDEX: 0.12794595161555466 4E-02

MODE:	1	2	3	4	5	6
FREQ (RAD/SEC):	12.324	20.547	24.332	146.06	180.46	252.41
FREQUENCY (HZ):	1.9614	3.2701	3.8726	23.246	28.722	40.172
PERIOD (SEC):	0.50984	0.30580	0.25825	4.30176E-02	3.48168E-02	2.48927E-02
EIGENVECTORS:						
DOF(17)	7.63947E-02	0.55409	0.56612	0.52999	1.0000	1.0000
DOF(18)	0.57560	-0.28112	0.25051	1.0000	-0.71089	0.57858
DOF(19)	-5.47059E-04	-7.32726E-04	1.27136E-05	-1.09182E-05	-5.84490E-04	5.47856E-05
DOF(20)	0.14275	1.0000	1.0000	-0.12689	-0.24581	-0.24554
DOF(21)	1.0000	-0.47729	0.43305	-0.25301	0.17776	-0.15995
DOF(22)	-9.49768E-04	-1.29555E-05	2.17057E-03	2.85772E-04	1.49215E-04	-8.98719E-04

--- MEMORY UTILIZATION ..... ---
--- IZ= 21965, MEM= 4.395% ---
--- ELAPSED CPU TIME 0.16 SEC ---
--- TOTAL CPU TIME 4858.79 SEC ---

Notes:

(1) The Gdof are printed for the six dof of each joint. The suffix -C denotes a constrained, 'slave' dof. The suffix -R denotes a restrained dof.

(2) First floor walls with a moment to shear ratio of 246".

(3) Second floor walls with a moment to shear ratio of 144".

BIBLIOGRAPHY

1. Ang, A. H-S., and Park, Y. J., "Reliability and Damage Analysis Under Extreme Natural Hazards," Proceedings of the CCNAA-AIT Joint Seminar on Research for Multiple Hazards Mitigation, NCKU, Taiwan, 1984, pp. 150-163.

2. Bath, K.J., Wilson, E.L., Numerical Methods on Finite Elements, Prentice-Hall, 1976.

3. Cheng, F. Y., "Dynamic Response of Nonlinear Space Frames by Finite Element Methods," Symposium on International Association for Shell Structures, Tokyo amd Kyoto, Japan, pp 817-826, 1972.

4. Cheng, F. Y., "Inelastic Analysis of 3-D Mixed Steel and Reinforced Concrete Seismic Building Structures," Journal of Computers and Structures, Vol 13, pp 189-196, 1981.

5. Cheng, F. Y. and Ger, J. F., "Maximum Response of Buildings to Multi-Seismic Input," Dynamics of Structures, ASCE, pp 397-410, 1987.

6. Cheng, F. Y. and Ger, J. F., "Translational-Torsional Spectral Method and Maximum Response for Six-Component Seismic Input," Proceedings of the Korea-Japan Joint Seminar on Engineering Technologies in Structural Engineering and Mechanics, pp 334-345, 1988.

7. Cheng, F. Y. and Juang, D. S., "Assessment of Various Code Provisions Based on Optimum Design of Steel Structures," International Journal of Earthquake Engineering and Structural Dynamics, Vol. 16, pp 45-61, 1988.

8. Cheng, F. Y., and Kitipitayangkul, P., "Multicomponent Earthquake Analysis of Mixed Structural Systems," U.S.A.-Japan Seminar on Composite Structures and Mixed Structural Systems; Developments in Composite and Mixed Construction (Ed. Ben Kayto and Le-Wu Lu) Gihodo Shuppan Co., Ltd., Tokyo, pp 337-347, 1980.

9. Cheng, F. Y., and Kitipitayangkul, P., "Investigation of the Effect of 3-D Parametric Earthquake Motions on the Stability of Elastic and Inelastic Building Systems," Technical Report prepared for the National Sceince Foundation. Available at the U.S. Department of Commerce, National Technical Information Service, Springfield, VA 22151, PB80-176936 (392 pages) 1977. Reinforced-Concrete Steel Buildings Subjected to 3-Dimensional

10. Cheng, F. Y., and Mertz, G. E., "Recent Studies of Nuclear Power Plant Auxiliary Buildings Subjected to Seismic Excitations," Proceedings of the CCNAA-AIT Joint Seminar on Research and Application for Multiple Hazard Mitigation, Taipei, Taiwan, April 1988.

11. Cheng, F. Y., and Mertz, G. E., "Hysteresis Models of Low-Rise Shear Walls with Coupled Bending and Shear Deformations," Proceedings of 10th International Conference on Structural Mechanics in Reactor Technology, August 1989.

12. Cheng, F. Y., and Oster, K. B., "Ultimate Instability of Earthquake Structures," Journal of the Structural Division, ASCE, Vol. 102, pp 961-972, 1976.

13. Cheng, F. Y., and Oster, K. B., "Effect of Coupling Earthquake Motions on Inelastic Structural Models," International Symposium on Earthquake Structural Engineering, Proceedings, Vol 1, pp 107-126, 1976.

14. Cheng, F. Y., and Oster, K. B., "Dynamic Instability and Ultimate Capacity of Inelastic Systems Parametrically Excitated by Earthquakes-- Part II," Technical Report prepared for the National Sceince Foundation. Available at the U.S. Department of Commerce, National Technical Information Service, Springfield, VA 22151, PB261097/AS (313 pages) 1976.

15. Cheng, F. Y. and Truman, K. Z., Optimun Design of Reinforced Concrete and Steel 3-D Static and Seismic Building Systems with Assessment of ATC-03,Final Report Series 85-20 for the National Sceince Foundation. Available at the U.S. Department of Commerce, National Technical Information Service, Springfield, VA 22151, PB87-168564/AS (414 pages) 1985. Civil Engineering Studies, Structural Research Series No. 82-9, Civil Engineering Studies, Structural Research Series No. 82-10,

16. Kabeyasawa, T., Shiohara, H., Otani, S., and Aoyama, H., "Analysis of the Full-Scale Seven-Story Reinforced Concrete Test Structure," Journal of the Faculity of Engineering, The University of Tokyo, Vol. XXXVII, No 2, 1983, pp. 431-478.

17. Mertz, G.E., Nonlinear Analysis of Low-Rise Reinforced Concrete Shear Wall Buildings Subjected to Multicomponent Seismic Input, Ph.D. Dissertation, University of Missouri-Rolla, 1989.

18. Park, Y., and Ang, A. H-S., "Mechanistic Seismic Damage Model for Reinforced Concrete," Journal of the Structural Division, ASCE, Vol. 111, No. 4, April, 1985, pp. 722-739.

19. Park, Y., and Ang, A. H-S., "Seismic Damage Analysis of Reinforced Concrete Buildings," Journal of the Structural Division, ASCE, Vol. 111, No. 4, April, 1985, pp. 740-757.

20. Sheu, M. S., "Behavior of Low Rise R.C. Shear Walls Subjected to Reversed Cyclic Loading," Technical Report to the National Science Council, Architectural Engineering Department, National Cheng Kung University, July, 1988 (in Chinese).

21. Takeda, T., Sozen, M. A., and Nielsen N. N., "Reinforced Concrete Response to Simulated Earthquakes," Journal of the Structural Division, ASCE, Vol. 96, No. ST12, Dec 1970, pp. 2557-2573.

APPENDIX A

DUCTILITY AND EXCURSION RATIO

Three definitions of ductility are considered in this study. The first and most common definition is the displacement definition as shown in Figure 26. Let the displacement ductility be defined as

$$\mu_1 = \frac{|\Delta_{max}|}{\Delta_y} \tag{8}$$

when Δ_{max} represents the maximum displacement, rotation or strain in the structure or element, and Δ_y is the yield displacement, rotation or strain. The ductility for the elastic 3D prismatic beam element is not calculated. For the reinforced concrete shear wall element the ductility is calculated for the bending, shear and axial components of deformation. The type of the spring element determines if its ductility is calculated for axial, shear, torsional or rotational deformation.

Cheng, et al, (14, 9) have proposed several energy based ductility definitions. Both the variable strain energy and the constant strain energy formulations are used in this study. The variable strain energy definition of the ductility is shown in Figure 27 and defined as

$$\mu_2 = 1 + \frac{PSE}{ESE} \tag{9}$$

where PSE corresponds to the plastic strain energy for the current half cycle. The constant strain energy definition of the ductility is shown in Figure 28 and defined as

$$\mu_3 = 1 + \frac{PSE}{CSE} \tag{10}$$

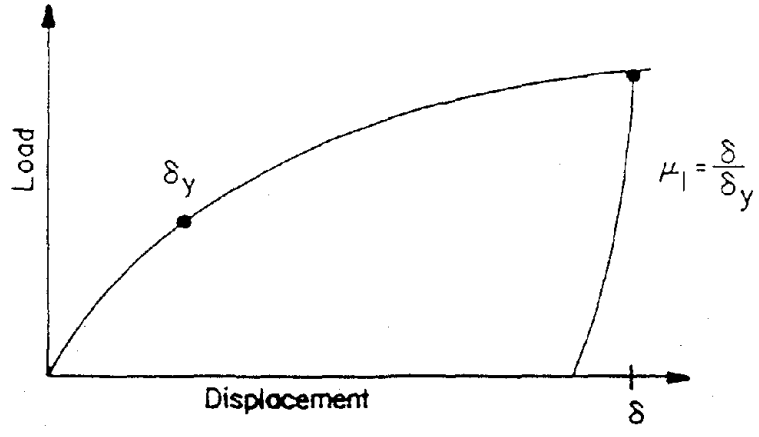


Figure 26. Displacement Definition of Ductility

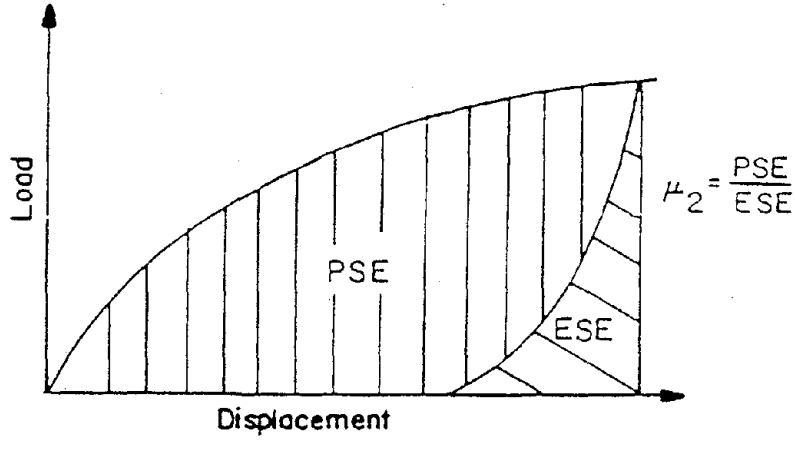


Figure 27. Variable Strain Energy Definition of Ductility

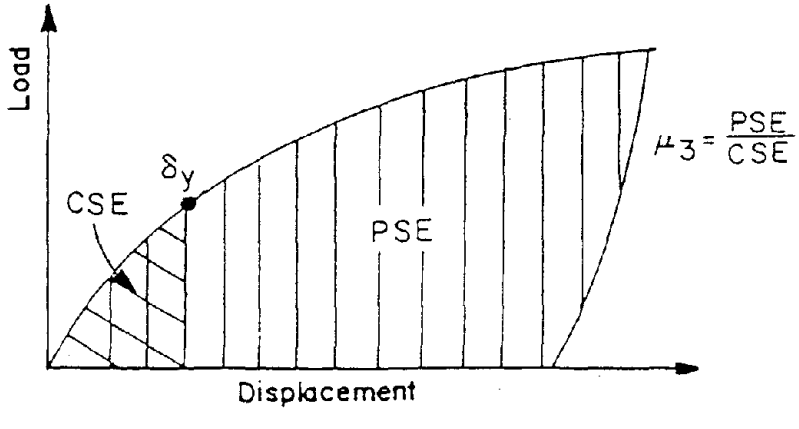


Figure 28. Constant Strain Energy Definition of Ductility

where CSE is the constant strain energy corresponding to displacement at yield.

For each ductility ratio, a corresponding excursion ratio exist, where the excursion ratio is given by

$$\varepsilon = \sum(\mu - 1) \tag{11}$$

and the summation is carried out for each half cycle.

APPENDIX B

DAMAGE INDEX

The damage index is a parameter developed by Ang, et al, to assess the damage in a structure (18, 19, 1). A damage index greater than 1.0 indicates total damage or collapse. The damage index is defined as

$$DI = \frac{\Delta_{max}}{\Delta_{ult}} + \frac{\beta}{Fy\,\Delta_{ult}} \int_0^t d(PSE) \tag{12}$$

where Δ_{max} is the maximum displacement, Δ_{ult} is the failure displacement under monotonic loading, Fy is the yield force, and β is a hysteretic energy coefficient. For R/C shear walls, Sheu (20) determined $\beta = 0.20$, based on NCKU walls SW1a through SW6.

The damage index for the elastic 3D prismatic beam element is not calculated. For the reinforced concrete shear wall element the damage index is calculated independently for the bending, shear and axial components of deformation. The type of the spring element determines if its damage index is calculated for axial, shear, torsional or rotational deformation. The damage index for each of the shear wall element's components and the spring element are printed in the output. The damage index for the entire structure is then calculated by taking a weighted average of each individual component's damage index, where the total strain energy, SE_i, for each component is used as the weighting factor. Thus

$$DI = \frac{\sum SE_i\, DI_i}{\sum SE_i} \tag{13}$$

where the summation is carried out for all the members. A sample calculation of the damage index is included in Section B.1 of Chapter VI.

APPENDIX C

LISTING OF PROGRAM INRESB-3D-SUP

A. COMMON BLOCKS

FILE: ZCOMN COPY

```
C---------- BEGIN COMMON ---------- ZC000010
      INCLUDE (ZCOMN1)              ZC000020
      INCLUDE (ZCOMN2)              ZC000030
C---------- END COMMON ------------ ZC000040
```

FILE: ZCOMN1 COPY

```
C---------- BEGIN BLOCK - ZCOMN1 ---------- ZC000010
      CHARACTER*112 TITLE(2)                ZC000020
      CHARACTER*60  STEPID                  ZC000030
      LOGICAL BUG,MCOND,KGCOND,DOUBLE,NEMK,NEMKG,ELSTIC,ABORT  ZC000040
      INTEGER FMASS,FDAMP,CPUREM,CPUELA     ZC000050
      DOUBLE PRECISION EIE,ESE,PSE,EKE,EDD,EDUMMY  ZC000060
      COMMON /ZDATAP/ TITLE ,BUG    ,IBUG  ,MAXEOF,MD(25),STEPID,GRAV  ZC000070
      COMMON /ZDATA/  IZ    ,IZREL ,CPUREM,CPUELA,ITLCPU,INCCPU,       ZC000080
     &  NEMK  ,KGAME ,NEMKG ,MCOND ,KGCOND,DOUBLE,ELSTIC,ABORT ,       ZC000090
     &  NNODE ,NDOF  ,NCOND ,NFREE ,NREST ,MAXNOD,NCOS  ,               ZC000100
     &  NMAT  ,NELMT ,NMASS ,                                           ZC000110
     &  NLOAD ,NSOLN ,                                                  ZC000120
     &  MAXELD,NELD  ,                                                  ZC000130
C                                                                       ZC000140
     &  IZID  ,IZIDOF,IZCORD,IZCOS ,IZCNST,IZJFLG,IZJCOS,               ZC000150
     &  IZMAT ,IZELE ,                                                  ZC000160
     &  IZIELD,IZELD ,                                                  ZC000170
     &  IZKE  ,IZLM  ,IZSTIF,IZMD  ,IZKEKG,IZLMKG,                      ZC000180
     &  IZMASS,IZMDMS,IZMDAT,IZDAMP,IZKGDT,IZKG  ,IZMDKG,               ZC000190
     &  IZLOAD,IZDISP,IZVEL ,IZACC ,                                    ZC000200
     &  IZDLOA,IZDDSP,IZDVEL,IZDACC,                                    ZC000210
     &  IZAGTK,IZAGTV,IZAGTZ,                                           ZC000220
     &  IZDUMP,IZFUB ,IZAN  ,IZAR  ,                                    ZC000230
C                                                                       ZC000240
     &  FMASS ,FDAMP,IUNIT,                                             ZC000250
     &  EIE,ESE,PSE,EKE,EDD,EDUMMY,                                     ZC000260
     &  SUMRCT(6),GROUND(9),INC,                                        ZC000270
     &  IDUMMY(500)                                                     ZC000280
C                                                                       ZC000290
C---------- END   BLOCK - ZCOMN1 ---------- ZC000300
```

FILE: ZCOMN2 COPY

```
C---------- BEGIN BLOCK - ZCOMN2 ----------C ZC000010
C                                          C ZC000020
      PARAMETER (MAXZ=500000, MAXDZ=MAXZ/2)  ZC000050
      COMMON /DATA/ Z(MAXZ)                  ZC000070
      DIMENSION NZ(MAXZ),DZ(MAXDZ)           ZC000080
      DOUBLE PRECISION DZ                    ZC000090
      EQUIVALENCE  (Z(1),NZ(1),DZ(1))        ZC000100
C                                          C ZC000110
C---------- END   BLOCK - ZCOMN2 ----------C ZC000120
```

FILE: ZCOMN3 COPY

```
C---------- BEGIN BLOCK - ZCOMN3 ---------- ZC000010
      DATA LZDATA/500/                      ZC000020
C------- VARIABLES IN COMMON                ZC000030
C TITLE  = USER INPUT TITLES DESCRIBING STRUCTURE AND SOLUTION  ZC000040
C IZ     = NEXT AVAILIABLE LOCATION IN LINEAR ARRAY Z           ZC000050
C IZREL  = VALUE OF IZ IF STORAGE FOR A SOLUTION IS RELEASED    ZC000060
C BUG    = FLAG, IF TRUE, DETAILED OUTPUT IS PRINTED FOR DEBUGING  ZC000070
C MCOND  = FLAG, IF TRUE, MASS HAS 'NOT' BEEN CONDENSED OUT     ZC000080
C NNODE  = NUMBER OF NODES                                      ZC000090
C                                                               ZC000100
C NDOF   = TOTAL NUMBER OF DEGREES OF FREEDOM                   ZC000110
C NCOND  = NUMBER OF DOF TO BE CONDENSED OUT                    ZC000120
C NFREE  = NUMBER OF FREE DOF                                   ZC000130
C NREST  = NUMBER OF RESTRAINED DOF                             ZC000140
C MAXNOD = MAXIMUN NUMBER OF NODES INPUT                        ZC000150
C NMAT   = NUMBER OF MATERIAL PROPERTIES                        ZC000160
C NELMT  = NUMBER OF ELEMENTS                                   ZC000170
C                                                               ZC000180
C NLOAD  = NUMBER OF LOADING CASES                              ZC000190
C NSOLN  = SOLUTION NUMBER                                      ZC000200
C MAXEOF = MAXIMUN NUMBER OF ELEMENT DOF                        ZC000210
C MD     = MAIN DIAGONAL ADDRESSES FOR AN UPPER TRIANGULAR STIFFNESS  ZC000220
C NCOS   = NUMBER OF JOINT DIRECTION COSINES                    ZC000230
C MAXELD = MAXIMUN NUMBER OF ELEMENT LOADS                      ZC000240
C NELD   = NUMBER OF ELEMENT LOADS                              ZC000250
C                                                               ZC000260
C..... ALL VARIABLES BEGINING WITH IZ____ ARE THE ABSOLUTE ADDRESS  ZC000270
C..... IN THE LINEAR ARRAY FOR THE ____ DATA.                   ZC000280
C IZID   = EXTERNAL JOINT NUMBERS                               ZC000290
C IZIDOF = DEGREE OF FREEDOM MATRIX FOR JOINT DOF'S             ZC000300
C IZCORD = MATRIX OF JOINT COORD'S                              ZC000310
C IZCOS  = MATRIX OF JOINT DIRECTION COSINES                    ZC000320
C IZCNST = MATIRX OF JOINT CONSTRAINT COIEFF                    ZC000330
C IZMAT  = MATERIAL DATA                                        ZC000340
C IZELE  = ELEMENT DATA                                         ZC000350
C                                                               ZC000360
C IZJFLG = MATRIX OF JOINT DOF FLAGS (RELEASE CONSTRAIN CONDENSE ELIM.)  ZC000370
C IZJCOS = MATRIX OF JOINT COSINE ID NUMBERS                    ZC000380
C IZKE   = ADDRESS OF ELEMENT STIFFNESS                         ZC000390
C IZLM   = ADDRESS OF ELEMENT GLOBAL TO LOCAL MAPPING MATRIX    ZC000400
C IZKEKG = ADDRESS OF ELEMENT GEOMETRIC STIFFNESS               ZC000410
C IZLMKG = ADDRESS OF ELEMENT KG GLOBAL TO LOCAL MAPPING MATRIX ZC000420
C IZSTIF = GLOBAL STIFFNESS                                     ZC000430
C IZMD   = ADDRESS OF MAIN DIAGONAL STIFFNESS TERMS             ZC000440
C IZLOAD = LOAD MATRIX                                          ZC000450
C IZDISP = DISPLACMENT                                          ZC000460
C IZIELD = INTEGER ELEMENT LOAD DATA                            ZC000470
C                                                               ZC000480
C IZIELD = REAL ELEMENT LOAD DATA                               ZC000490
C                                                               ZC000500
C IZLOAD = TOTAL LOAD                                           ZC000510
C IZDISP = TOTAL DISPLACEMENT                                   ZC000520
C IZVEL  = TOTAL VELOCTIV                                       ZC000530
C IZACC  = TOTAL ACCELERATION                                   ZC000540
C                                                               ZC000550
C IZDLOA = INCRIMENTAL LOAD                                     ZC000560
C IZDDSP = INCRIMENTAL DISPLACEMENT                             ZC000570
C IZDVEL = INCRIMENTAL VELOCTIV                                 ZC000580
C IZDACC = INCRIMENTAL ACCELERATION                             ZC000590
C                                                               ZC000600
C IZAGTK = TRANSLATIONAL GROUND ACCELERATION - X DIRECTION      ZC000610
C IZAGTV = TRANSLATIONAL GROUND ACCELERATION - Y DIRECTION      ZC000620
C IZAGTZ = TRANSLATIONAL GROUND ACCELERATION - Z DIRECTION      ZC000630
C                                                               ZC000640
C IZFUB  = UNBALANCED FORCE VECTOR                              ZC000650
C IZQ    = TEMPORARY VECTOR                                     ZC000660
C IZR    = TEMPORARY VECTOR                                     ZC000670
C                                                               ZC000680
C FDAMP  = DAMPING TYPE                                         ZC000690
C FMASS  = MASS TYPE                                            ZC000700
C IUNIT  = UNIT # FOR PRINTING DATA TO AN OUTPUT FILE           ZC000710
C                                                               ZC000720
C---------- END   COMMON BLOCK - ZCOMN ---------- ZC000730
```

B. PROGRAM INRESB-3D-SUP

FILE: INRESB FORTRAN

```
C=============================================================C CFEM00010
C                                                             C CFEM00020
C   PROGRAM INRESB                                            C CFEM00030
C                                                             C CFEM00040
C=============================================================C CFEM00050
C                                                             C CFEM00060
C   NOTE: USE IBM FORTVS2 COMPILER OPTION AUTODBL(DBLPAD*)    C CFEM00070
C         TO COMPILE THIS PROGRAM IN DOUBLE PRECISION         C CFEM00080
C                                                             C CFEM00090
C   NOTE: USE IBM FORTVS2 COMPILER OPTION OPT(2)              C CFEM00100
C         TO OPTIMIZE THE CODE FOR FASTER EXECUTION           C CFEM00110
C                                                             C CFEM00120
C=============================================================C CFEM00130
      CHARACTER*80 OPTION,INPUT,BLANK                          FEM00140
      CHARACTER*8  TIME1,DATE1                                 FEM00150
      CHARACTER*1  CHR(0:25), NAME                             FEM00160
      LOGICAL TEST,BTEST,HEAD ,AXIAL                           FEM00170
      INCLUDE (ZCOMN)                                          FEM00180
      INCLUDE (ZCOMN3)                                         FEM00190
      DATA CHR/'A','B','C','D','E','F','G','H','I','J','K','L','M',  FEM00200
     &         'N','O','P','Q','R','S','T','U','V','W','Z','Y','Z'/  FEM00210
C                                                              FEM00220
C=======================================                       FEM00230
C== INITILIZE THE UMRBX TIMING SUBROUTINE ==                   FEM00240
C=======================================                       FEM00250
      CPUREM=0                                                 FEM00260
      CPUELA=0                                                 FEM00270
      CALL CPUTIM(CPUREM,CPUELA)                               FEM00280
      ITLCPU=CPUREM                                            FEM00290
                                                               FEM00300
C                                                              FEM00310
C============================                                  FEM00320
C== INITILIZE THE CPU CLOCK ===                                FEM00330
C============================                                  FEM00340
      CALL TIMEON                                              FEM00350
      TCPU=0                                                   FEM00360
C                                                              FEM00370
C============================                                  FEM00380
C== ECHO THE INPUT.....  ==                                    FEM00390
C============================                                  FEM00400
      I=55                                                     FEM00410
      L=0                                                      FEM00420
9995  READ (5,'(A)',END=9996) INPUT                            FEM00430
      IF (INDEX(INPUT,'NOECHO').NE.0) GO TO 9996               FEM00440
      I=I+1                                                    FEM00450
      L=L+1                                                    FEM00460
      IF (  I.GT.55 ) THEN                                     FEM00470
        WRITE (6,9994) (KK,KK=1,8)                             FEM00480
        I=0                                                    FEM00490
      ENDIF                                                    FEM00500
      WRITE (6,9997) L,INPUT                                   FEM00510
      IF (INDEX(INPUT,'STOP' ).EQ.0) GO TO 9995                FEM00520
9994  FORMAT ('1    ECHO OF INPUT DATA '//                     FEM00530
     &        1X,5X,'LINE  ',8('....I...',I1,'0'))             FEM00540
9997  FORMAT (1X,I7,': ',A)                                    FEM00550
9996  REWIND(5)                                                FEM00560
C                                                              FEM00570
C============================                                  FEM00580
C== INITIALIZE DATA.....  ==                                   FEM00590
C============================                                  FEM00600
      TITLE(1)=' '                                             FEM00610
      TITLE(2)=' '                                             FEM00620
      BLANK  =' '                                              FEM00630
      BUG =.FALSE.                                             FEM00640
```

```
      DOUBLE =.FALSE.                                                   FEM00650
      IBUG=0                                                            FEM00660
      MAXEDF=24                                                         FEM00670
      MDC(1)=1                                                          FEM00680
      DO 1 I=2,(MAXEDF+1)                                               FEM00690
    1   MDC(I)=MDC(I-1)+(I-1)                                           FEM00700
      IZ=0                                                              FEM00710
C                                                                       FEM00760
   20 READX(5,*,END=1000) OPTION                                        FEM00770
      RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                    FEM00780
C                                                                       FEM00790
C=========================                                            FEM00800
C== DEFINE THE STRUCTURE ===                                            FEM00810
C=========================                                            FEM00820
      IF (INDEX(OPTION,'STR').NE.0)  THEN                               FEM00830
          ABORT=.FALSE.                                                 FEM00840
          CALL  COMPT(1,LZDATA)                                         FEM00850
          GRAV=386.4                                                    FEM00860
          READ (5,*,END=1000) TITLE(1)                                  FEM00870
          CALL TIME(TIME1)                                              FEM00880
          CALL DATE(DATE1)                                              FEM00890
          WRITE (TITLE(1)(80:),45) TIME1,DATE1                          FEM00900
          RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                FEM00910
          WRITE (1,52)' BUILDING STRUCTURAL MODEL ----------------*',FEM00920
     &                  IZ,RATIZ                                        FEM00930
          WRITE (1,51) TITLE(1)(1:70)                                   FEM00940
          ABORT=.FALSE.                                                 FEM00950
          CALL STRUCT                                                   FEM00960
          IZREL=IZ                                                      FEM00970
          RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                FEM00980
          CALL TIMEIT(CPU)                                              FEM00990
          TCPU=CPU+TCPU                                                 FEM01000
          WRITE (6,55) IZ,RATIZ,CPU,TCPU                                FEM01010
C                                                                       FEM01020
C=================================                                    FEM01030
C== DEFINE THE LOADING AND SOLVE ==                                     FEM01040
C=================================                                    FEM01050
      ELSE IF (INDEX(OPTION,'SOL').NE.0)  THEN                          FEM01060
          READ (5,*,END=1000) TITLE(2)                                  FEM01070
          CALL TIME(TIME1)                                              FEM01080
          CALL DATE(DATE1)                                              FEM01090
          WRITE (TITLE(2)(80:),45) TIME1,DATE1                          FEM01100
          WRITE (1,52)' SOLVING STRUCTURAL MODEL -----------------*',FEM01120
     &                  IZ,RATIZ                                        FEM01130
          WRITE (1,51) TITLE(1)(1:70)                                   FEM01140
          WRITE (1,51) TITLE(2)(1:70)                                   FEM01150
          ABORT=.FALSE.                                                 FEM01160
          CALL SOLVE(OPTION)                                            FEM01170
          RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                FEM01180
          CALL TIMEIT(CPU)                                              FEM01190
          TCPU=CPU+TCPU                                                 FEM01200
          WRITE (6,55) IZ,RATIZ,CPU,TCPU                                FEM01210
C                                                                       FEM01220
C=================================                                    FEM01230
C== DUMP THE MEMORARY...        ==                                      FEM01240
C=================================                                    FEM01250
      ELSE IF (INDEX(OPTION,'DUMP').NE.0)  THEN                         FEM01260
          WRITE (1,52)' DUMPING MEMORY ---------------------------*',FEM01270
     &                  IZ,RATIZ                                        FEM01280
          WRITE (1,51) TITLE(1)(1:70)                                   FEM01290
          WRITE (1,51) TITLE(2)(1:70)                                   FEM01300
          CALL CONDMP(Z,NZ,DZ)                                          FEM01310
          RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                FEM01320
          CALL TIMEIT(CPU)                                              FEM01330
          TCPU=CPU+TCPU                                                 FEM01340
          WRITE (6,55) IZ,RATIZ,CPU,TCPU                                FEM01350
C                                                                       FEM01360
C=================================                                    FEM01370
C== STOP... END OF PROGRAM      ==                                      FEM01380
C=================================                                    FEM01390
      ELSE IF (INDEX(OPTION,'STOP').NE.0)  THEN                         FEM01400
          WRITE (1,52)' NORMAL STOP ------------------------------*',FEM01410
     &                  IZ,RATIZ                                        FEM01420
          STOP                                                          FEM01430
C                                                                       FEM01440
C=================================                                    FEM01450
C==  READ RESULTS FROM INPUT FILE ===                                   FEM01460
C=================================                                    FEM01470
      ELSE IF (INDEX(OPTION,'READ').NE.0)  THEN                         FEM01480
  140     CALL GETINT(OPTION,'UNIT=',IUNIT,0,.TRUE.,.TRUE. )            FEM01490
          CALL GETINT(OPTION, 'INC=',INC  ,1,.TRUE.,.FALSE.)            FEM01500
          IF (IUNIT.GT.0) THEN                                          FEM01510
            WRITE (1,52)                                                FEM01520
     &      ' READING DATA FROM FILE -------------------------*',IZ,RATIZ FEM01530
            IOPT=4                                                      FEM01540
            CALL OMPOAT(IOPT,ENRITE,TO,DT)                              FEM01550
            GO TO 140                                                   FEM01560
          ENDIF                                                         FEM01570
          RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                FEM01580
          CALL TIMEIT(CPU)                                              FEM01590
          TCPU=CPU+TCPU                                                 FEM01600
          WRITE (6,53) IZ,RATIZ,CPU,TCPU                                FEM01610
C                                                                       FEM01620
C=========================                                            FEM01630
C== SET BUG OPTION      ==                                              FEM01640
C=========================                                            FEM01650
      ELSE IF (TEST(OPTION,'BUG',,FALSE.))  THEN                        FEM01660
          CALL GETCHR(OPTION,'BUG=',INPUT,,TRUE.,,FALSE.)               FEM01670
          IBUG=0                                                        FEM01680
          WRITE (OPTION,160) INPUT(1:70)                                FEM01690
          DO 150 I=0,25                                                 FEM01700
            IF (TEST(INPUT,CHR(I),,FALSE.)) THEN                        FEM01710
                IBUG=IBSET(IBUG,I)                                      FEM01720
            ENDIF                                                       FEM01730
  150     CONTINUE                                                      FEM01740
  160     FORMAT ('BUG=',A)                                             FEM01750
          WRITE (1,52)' SET BUG OPTION: --------------------------*',FEM01760
     &                  IZ,RATIZ                                        FEM01770
          WRITE (1,51)  OPTION(1:70)                                    FEM01780
          WRITE (6,51)  'SET BUG OPTION: '                              FEM01790
          WRITE (6,51)  OPTION(1:70)                                    FEM01800
C                                                                       FEM01810
C=========================                                            FEM01820
C== SET BUG OPTION      ==                                              FEM01830
C=========================                                            FEM01840
      ELSE IF (INDEX(OPTION,'BUG').NE.0)  THEN                          FEM01850
          WRITE (1,51) ('SET BUG OPTION: '//OPTION(1:56) )              FEM01860
          BUG=.TRUE.                                                    FEM01870
          IF (INDEX(OPTION,'NOBUG').NE.0) BUG=.FALSE.                   FEM01880
C                                                                       FEM01890
C=========================================                            FEM01900
C== RELEASE MEMORY FROM LAST SOLUTION ==                                FEM01910
C=========================================                            FEM01920
      ELSE IF (INDEX(OPTION,'RELEASE').NE.0)  THEN                      FEM01950
          WRITE (6,52)' ------------------------------------------*',   FEM01940
          WRITE (1,52)' RELEASING MEMORY -------------------------*',   FEM01950
          WRITE (6,52)' RELEASING MEMORY -------------------------*',   FEM01960
          IF (INDEX(OPTION,'ELEMENT').NE.0)  THEN                       FEM01970
            WRITE (6,52)' RESETING ELEMENT FORCES ------------------*',FEM01980
            WRITE (6,52)' ------------------------------------------*',FEM01990
            DO 170 IELNO=1,NELMT                                        FEM02000
              LSTTYP=0                                                  FEM02010
              HEAD =.FALSE.                                             FEM02020
              CALL ELELIB                                               FEM02030
  170       &  (15   ,LSTTYP ,.FALSE.,IREL ,HEAD  ,INPUT,EESE,EPSE,DAMAGE,FEM02040
     &          IELNO,IELDOF,KGDOF,PGEOM,Z ,AXIAL,IZDOSP,IZDLOA,MSTOR)   FEM02050
          ELSE                                                          FEM02060
            WRITE (6,52)' ------------------------------------------*'  FEM02070
          ENDIF                                                         FEM02080
          IZ=IZREL                                                      FEM02090
          IZACC =0                                                      FEM02100
          IZDACC=0                                                      FEM02110
          IZDDSP=0                                                      FEM02120
          IZDISP=0                                                      FEM02130
          IZDLOA=0                                                      FEM02140
          IZDVEL=0                                                      FEM02150
          IZFUB =0                                                      FEM02160
          IZLOAD=0                                                      FEM02170
          IZVEL =0                                                      FEM02180
          RATIZ=100.*REAL(IZ)/REAL(MAXZ)                                FEM02190
          CALL TIMEIT(CPU)                                              FEM02200
          TCPU=CPU+TCPU                                                 FEM02210
          WRITE (6,55) IZ,RATIZ,CPU,TCPU                                FEM02220
C                                                                       FEM02840
C=====================================                                FEM02850
C== DUMMY STATEMENTS DURING ABORT    ==                                 FEM02860
C=====================================                                FEM02870
      ELSE IF (ABORT) THEN                                              FEM02880
          GO TO  20                                                     FEM02890
C                                                                       FEM02900
C=====================================                                FEM02910
C== INVALID OPTION                   ==                                 FEM02920
C=====================================                                FEM02930
      ELSE                                                              FEM02940
          IF (TEST(OPTION,'NOECHO',,FALSE.) ) GO TO 20                  FEM02950
          IF (OPTION.EQ.BLANK) GO TO 20                                 FEM02960
          WRITE (6,30) OPTION                                           FEM02970
          WRITE (1,51) ('INVALID OPTION: '//OPTION(1:54))               FEM02980
      ENDIF                                                             FEM02990
      GO TO  20                                                         FEM03000
C                                                                       FEM05010
C                                                                       FEM05020
   45 FORMAT(' TIME: ',A,'  DATE: ',A)                                  FEM05050
   50 FORMAT (A1,' STRUCTURE....: ',A,/                                 FEM05060
     &        1X,' SOLUTION.....: ',A,/)                                FEM05070
   51 FORMAT (1X,A)                                                     FEM05080
   52 FORMAT (1X,'*---',A45,'*',,' IZ=',I7,' MEM=',F6.3,'%')            FEM05090
   55 FORMAT  (//                                                       FEM05100
     &  1X,'*----',35('-'),'*'/                                         FEM05110
     &  1X,'*--- MEMORY UTILIZATION ......... ---*'/                   FEM05120
     &  1X,'*--- IZ=', I7,',    MEM=',F6.5,'%   ---*'/                 FEM05130
     &  1X,'*----',35('-'),'*'/ ,                                       FEM05140
     &  1X,'*--- ELAPSED CPU TIME ',F8.2,' SEC ---*'/                   FEM05150
     &  1X,'*--- TOTAL CPU TIME   ',F8.2,' SEC ---*'/                   FEM05160
     &  1X,'*----',35('-'),'*'/ )                                       FEM05170
   90 FORMAT (' INVALID OPTION= ',A)                                    FEM05180
  110 FORMAT (' ZMATRX(',I6,'):',I15,1P,G20.10,:,G20.10)                FEM05190
 1000 STOP                                                              FEM05200
      END                                                               FEM05210
C=====================================================================AVE00010
      SUBROUTINE AVEACC(IOPT,BUG,DT,TO,TF,T,THETA,NEMKDV,               AVE00040
     &    L1,L2,L3,L4,L5,L6,IMDMAS,IMD,IMDS,INDKG,NDOF,NFREE,           AVE00050
     &    KGLOAD,KGFORM,                                                AVE00060
     &    A    ,V    ,D    ,P    ,                                      AVE00070
     &    DA   ,DV   ,DD   ,DP   ,WORK,                                 AVE00080
     &    MASS ,MDMASS,DAMP ,STIFF ,MD   ,                              AVE00090
     &    KG   ,MDKG ,ACC  ,GRAV ,                                      AVE00100
     &    S    ,MDS  ,Q    ,R    ,PBAR ,                               AVE00110
     &    FDAMP,FMASS,C0,C1,C2,C3,C4,C5,C6,ALPHA,BETA,C10,C11,ABORT)    AVE00120
      INTEGER FDAMP,FMASS                                               AVE00130
      REAL MASS,KG                                                      AVE00140
      LOGICAL BUG,NEMKDV,ABORT                                          AVE00150
      LOGICAL BTEST                                                     AVE00160
      DIMENSION A(NDOF),V(NDOF),D(NDOF),DA(NDOF),DV(NDOF),DD(NDOF)      AVE00170
      DIMENSION P(NDOF),DP(NDOF),WORK(2*NDOF)                           AVE00180
      DIMENSION PBAR(L5:L4),Q(L5:L4),R(L5:L4)                           AVE00190
      DIMENSION MDMASS(NDOF+1),MASS(IMDMAS)                             AVE00200
      DIMENSION MDKG(NDOF+1)  ,KG(INDKG)                                AVE00210
      DIMENSION MDC(NDOF+1)   ,STIFF(IMD)                               AVE00220
      DIMENSION MDS(NDOF+1)   ,S(IMDS)    ,DAMP(5)                      AVE00230
C                                                                       AVE00240
C---------------------------------------------------------------------AVE00250
C     DETERMINE THE RESPONSE AT TIME DT*2                               AVE00260
C                                                                       AVE00270
      TWODT=DT*2.0                                                      AVE00280
      THETA=2.0                                                         AVE00290
      CALL LINACC(IOPT,BUG,TWODT,TO,TF,T,THETA,NEMKDV,                  AVE00300
     &    L1,L2,L3,L4,L5,L6,IMDMAS,IMD,IMDS,INDKG,NDOF,NFREE,           AVE00310
     &    KGLOAD,KGFORM,                                                AVE00320
     &    A    ,V    ,D    ,P    ,                                      AVE00330
     &    DA   ,DV   ,DD   ,DP   ,WORK,                                 AVE00340
     &    MASS ,MDMASS,DAMP ,STIFF ,MD   ,                              AVE00350
     &    KG   ,MDKG ,ACC  ,GRAV ,                                      AVE00360
     &    S    ,MDS  ,Q    ,R    ,PBAR ,                               AVE00370
     &    FDAMP,FMASS,C0,C1,C2,C3,C4,C5,C6,ALPHA,BETA,C10,C11,ABORT)    AVE00380
C                                                                       AVE00390
      IF (IOPT.NE.2) RETURN                                             AVE00400
C                                                                       AVE00410
C-------- DETERMINE ACCELERATION, VELOCITY AND DISPLACEMENT AT TIME DT   AVE00420
C       DA = INCRIMENTAL ACCELERATION BETWEEN TIMES T AND T+DT          AVE00430
C       DV = INCRIMENTAL VELOCITY BETWEEN TIMES T AND T+DT              AVE00440
C       DD = INCRIMENTAL DISPLACMENT BETWEEN TIMES T AND T+DT           AVE00450
C                                                                       AVE00460
      DO 270 I=3,L4                                                     AVE00470
        DSP2DT=DD(I)                                                    AVE00480
        VEL2DT=DV(I)                                                    AVE00490
        ACC2DT=DA(I)                                                    AVE00500
        IF (BUG) WRITE (6,271) I,DSP2DT,VEL2DT,ACC2DT                   AVE00510
        DA(I)= ACC2DT/2.00                                              AVE00520
        DV(I)= (4*A(I) + ACC2DT)*DT/4.                                  AVE00530
  270   DD(I)=V(I)*DT + (6.00*A(I) + ACC2DT)*DT*DT/12.                  AVE00540
  271   FORMAT (' DOF:',I6,' AT T+2DT,  DSP:',1P,G14.6,'  VEL:',G14.6,  AVE00550
     &          '  ACC:',G14.6)                                        AVE00560
C                                                                       AVE00570
      RETURN                                                            AVE00580
C                                                                       AVE00590
      END                                                               AVE00600
C=====================================================================BML00020
```

```
      SUBROUTINE BMLOAC(IELNO,FLAG,L,REL,MAXELD,NELD,MLOAD,
     &     FEM,IELD,ELD)
C     INTEGER RY,RZ
      REAL    L,MA,MB
      INTEGER REL
      LOGICAL B0,B1,B2,B3,B4,B5,BTEST
      LOGICAL TEST,FLAG
      DIMENSION IELD(5,MAXELD),ELD(12,MAXELD)
      DIMENSION FEM(12,MLOAD),FORCE(12)
      EQUIVALENCE (FORCE( 1),F1 ),(FORCE( 2),F2 ),(FORCE( 3),F3 )
      EQUIVALENCE (FORCE( 4),F4 ),(FORCE( 5),F5 ),(FORCE( 6),F6 )
      EQUIVALENCE (FORCE( 7),F7 ),(FORCE( 8),F8 ),(FORCE( 9),F9 )
      EQUIVALENCE (FORCE(10),F10),(FORCE(11),F11),(FORCE(12),F12)
      FLAG=.FALSE.
      L=L
      REL=REL
      B0=BTEST(REL,0)
      B1=BTEST(REL,1)
      B2=BTEST(REL,2)
      B3=BTEST(REL,3)
      B4=BTEST(REL,4)
      B5=BTEST(REL,5)
C
      IF (MAXELD.LE.0) RETURN
C
C-------- LOOP FOR EACH LOAD INPUT
      DO 1000 N=1,NELD
C
C..... DETERMINE IF MEMBER IS AFFECTED BY THIS LOAD
         TEST=.FALSE.
         IEL=IELD(2,N)
         IGEN=IELD(3,N)
         INC=IELD(4,N)
         IF (IEL.EQ.IELNO) TEST=.TRUE.
         IF ((.NOT.TEST) .AND. IGEN.GT.0 .AND. INC.NE.0) THEN
            IRMD=MOD(IELNO-IEL,INC)
            JGEN=(IELNO-IEL)/INC
            IF (IRMD.EQ.0 .AND. JGEN.LE.IGEN .AND. JGEN.GE.0)
     &           TEST=.TRUE.
         ENDIF
         IF (.NOT. TEST) GO TO 1000
C
C..................... ZERO FEM
         IF (.NOT. FLAG) THEN
            DO 10 I=1,12
            DO 10 J=1,NLOAD
   10          FEM(I,J)=0
               FLAG=.TRUE.
         ENDIF
C
C..................... ZERO FORCE
         DO 15 I=1,12
   15       FORCE(I)=0
C
C..... DETERMINE LOAD CODES...
         KODE=IELD(5,N)
         KDOF=MOD(KODE,100)
         KGRP=KODE-KDOF
         LOAD=IELD(1,N)
C
C
C..... CONCENTRATED LOADING
       IF (KGRP.EQ.100) THEN
         P=ELD(1,N)
         R=ELD(2,N)
         IF (KDOF.EQ.1) THEN
           F1=-(1-R)*P
           F7=-R*P
         ELSE IF (KDOF.EQ.2) THEN
           F6 =-P*R*L*(1-R)**2
           F12= P*(1-R)*L*(R)**2
           CALL FEMREL(F6 ,F12,B5 ,B0 ,0,IELNO)
           F8=(-F6-F12-P*R*L )/L
           F2=-P-F8
         ELSE IF (KDOF.EQ.3) THEN
           F5 = P*R*L*(1-R)**2
           F11=-P*(1-R)*L*(R)**2
           CALL FEMREL(F5 ,F11,B4 ,B1 ,0,IELNO)
           F9 =(F5+F11-P*R*L)/L
           F3 =-P-F9
         ELSE IF (KDOF.EQ.4) THEN
           F4 =-P*(1-R)
           F10=-P*R
           CALL FEMREL(F4 ,F10,B5 ,B2 ,1,IELNO)
         ELSE IF (KDOF.EQ.5) THEN
           F5 =-P*(1 - 4*R + 3*R**2)
           F11= P*( 2*R - 3*R**2)
           CALL FEMREL(F5 ,F11,B4 ,B1 ,0,IELNO)
           F9=( F5+F11+P)/L
           F5=-F9
         ELSE IF (KDOF.EQ.6) THEN
           F6 =-P*(1 - 4*R + 3*R**2)
           F12= P*( 2*R - 3*R**2)
           CALL FEMREL(F6 ,F12,B5 ,B0 ,0,IELNO)
           F8 =-(F6+F12+P)/L
           F2 = -F8
         ENDIF
C
C..... UNIFORM LOADING
       ELSE IF (KGRP.EQ.200) THEN
         W=ELD(1,N)
         IF (KDOF.EQ.1) THEN
           F1=-W*L/2
           F7=-W*L/2
         ELSE IF (KDOF.EQ.2) THEN
           F6 =-W*L*L/12
           F12= W*L*L/12
           CALL FEMREL(F6 ,F12,B5 ,B0 ,0,IELNO)
           F8=(-F6-F12)/L  -W*L/2
           F2=-W*L-F8
         ELSE IF (KDOF.EQ.5) THEN
           F5 = W*L*L/12
           F11=-W*L*L/12
           CALL FEMREL(F5 ,F11,B4 ,B1 ,0,IELNO)
           F9 =(F5+F11)/L  - W*L/2
           F3 =-W*L-F9
         ELSE IF (KDOF.EQ.4) THEN
           F4 =-W*L/2
           F10=-W*L/2
           CALL FEMREL(F4 ,F10,B5 ,B2 ,1,IELNO)
         ELSE
           WRITE (6,900) IELNO,' UNIFORM WY OR UNIFORM WZ'
         ENDIF
C       USER INPUT FIXED END FORCE
        ELSE IF (KGRP.EQ.400) THEN
          DO 400 I=1,12
  400     FORCE(I)= ELD(I,N)
C
C..... INVALID LOAD GROUP
        ELSE
          WRITE (6,900) IELNO,'INVALID LOAD GROUP'
        ENDIF
C
C..... TRANSFER BEAM LOADS TO FIXED END FORCES.....
          DO 800 I=1,12
  800     FEM(I,LOAD)=FEM(I,LOAD)-FORCE(I)
C
C
 1000 CONTINUE
C
  900 FORMAT(' ER',' ROR CALCULATING FIXED END FORCES FOR ELEMENT #',
     &  I6,' ',A,' IS NOT AVAILABLE')
      RETURN
      END
C=======================================================================
      SUBROUTINE CKRNGC(V,A,B,NAME)
      CHARACTER*(*) NAME
      IF (A.LT.V .AND. V.LT.B) RETURN
      WRITE (6,10) NAME,A,V,B
   10 FORMAT (' ====>',A,' IS OUT OF RANGE, SET ',1P,G13.5,' LE INPUT=',
     &  G13.5,' LE ',G13.5)
      V=MIN(V,B)
      V=MAX(V,A)
      RETURN
      END
C=======================================================================
      SUBROUTINE CKSTOR(ITEMP,IH)
      INCLUDE (ZCOMN)
C
C================================
C= CHECK STORAGE REQUIREMENTS ===
C================================
      N=MAX(IZ,IH)+ITEMP
      IF (N.LE.MAXZ) RETURN
C
      WRITE (6,10) N,MAXZ
   10 FORMAT (///1X,49('*') /
     &  ' *** AVAILABLE STORAGE IS EXCEEDED'          ,T48,'***'/
     &  ' *** REQUIRED STORAGE =',I10                 ,T48,'***'/
     &  ' *** AVAILABLE STORAGE =',I10                ,T48,'***'/
     &  ' *** EXECUTION IS TERMINATED.'               ,T48,'***'/
     &  ' *** RECOMPILE PROGRAM WITH INCREASED MEMORY',T48,'***'/
     &  1X,49('*') )
C
C------ INITIATE IBM ER_ROR TRACEBACK ROUTINE
      CALL ERRTRA
C
C------ ABORT RUN
      STOP 'REQUIRED MEMORY > AVAILABLE MEMORY '
      END
C=======================================================================
      SUBROUTINE COMDMP(Z,NZ,DZ)
      DIMENSION Z(1),NZ(1),DZ(1)
      DOUBLE PRECISION DZ
      LOGICAL BTEST
      INCLUDE (ZCOMN1)
C
      WRITE (6,50) TITLE(1),TITLE(2)
      WRITE (6,51) 'DUMPING MEMORY'
      CALL SDUMP
      IF (.NOT. BTEST(IBUG,15) ) RETURN
C
            I=0
            K=0
            DO 100 J=1,IZ,2
               I=I+1
               K=K+1
               IF (NZ(I).NE.0) WRITE(6,110) I,NZ(I),Z(I),DZ(K)
               I=I+1
  100          IF (NZ(I).NE.0) WRITE(6,110) I,NZ(I),Z(I)
C
   50 FORMAT ('1 STRUCTURE....: ',A,/
     &        ' SOLUTION.....: ',A,/)
   51 FORMAT (1X,A)
  110 FORMAT (' ZMATRX(',I6,'):',I15,1P,G20.10,:,G20.10)
      RETURN
      END
C=======================================================================
      SUBROUTINE COMPT(IOPT,LZDATA)
      CHARACTER*112 TITLE(2)
      COMMON /ZDATAP/ TITLE
      COMMON /ZDATA/  IZ  ,IDATA(1)
C
      IF (IOPT.EQ.1)  THEN
         DO 10 I=1,LZDATA-1
   10    IDATA(I)=0
         IZ=1
         TITLE(1)=' '
         TITLE(2)=' '
      ENDIF
C
      END
C=======================================================================
      SUBROUTINE CROSS(VZ,VX,VY,NORM)
      DIMENSION VX(3),VY(3),VZ(3)
      LOGICAL BTEST
C
      CROSS(A1,B1,C1,A2,B2,C2)=B1*C2-B2*C1
      CROSS(K,A1,B1,C1,A2,B2,C2)=C1*A2-C2*A1
      CROSS(A1,B1,C1,A2,B2,C2)=A1*B2-A2*B1
      VALUE (A1,A2,A3)=SQRT(A1**2+A2**2+A3**2)
C
C ------- CALCULATE VZ, IF VY HAS CORRECTLY INPUT
      VZ(1)=CROSS(VX(1),VX(2),VX(3),VY(1),VY(2),VY(3))
      VZ(2)=CROSS(K,VX(1),VX(2),VX(3),VY(1),VY(2),VY(3))
      VZ(3)=CROSS(K,VX(1),VX(2),VX(3),VY(1),VY(2),VY(3))
C
C -------- NORMALIZE VX,VY,VZ
      IF (NORM.NE.0) THEN
           AVX=VALUE(VX(1),VX(2),VX(3))
           AVY=VALUE(VY(1),VY(2),VY(3))
           AVZ=VALUE(VZ(1),VZ(2),VZ(3))
           DO 10 I=1,3
              VX(I)=VX(I)/AVX
              VY(I)=VY(I)/AVY
```

```
   10         VZ(I)=VZ(I)/AVZ                                          CR000260
C                                                                      CR000270
      ENDIF                                                            CR000280
      RETURN                                                           CR000290
      END                                                              CR000300
C=====================================================================
C  - DIRECTION OF LOADING ..........................................DIR00010
C     P       = CURRENT LOAD                                         DIR00040
C     PL      = LAST LOAD                                            DIR00050
C     D       = CURRENT DISPLACEMENT                                 DIR00060
C     DIR     = CURRENT DIRECTION                                    DIR00070
C               DIR=1, POSITIVE LOADING                              DIR00080
C               DIR=2, POSITIVE UNLOADING                            DIR00090
C               DIR=3, NEGATIVE LOADING                              DIR00100
C               DIR=4, NEGATIVE UNLOADING                            DIR00110
C     DIRL    = LAST DIRECTION                                       DIR00120
C     VEL     = PRODUCT OF LAST AND CURRENT VELOCITIES               DIR00130
                                                                     DIR00140
      SUBROUTINE DIRECT(DIR,DIRL,P,PL,VEL)                           DIR00150
C                                                                    DIR00160
      IMPLICIT REAL(A-H,O-Z)                                         DIR00170
      IMPLICIT INTEGER(I-N)                                          DIR00180
      INTEGER DIR,DIRL                                               DIR00190
      LOGICAL BTEST                                                  DIR00200
                                                                     DIR00210
      IF (P.GT.0) THEN                                               DIR00220
         IF (P.GT.PL .AND. PL.GT.0) THEN                             DIR00230
            DIRL=1                                                   DIR00240
         ELSE IF (P.GT.PL .AND. PL.LT.0) THEN                        DIR00250
            DIRL=4                                                   DIR00260
         ELSE IF (P.LT.PL) THEN                                      DIR00270
            DIRL=2                                                   DIR00280
         ELSE                                                        DIR00290
            DIRL=DIR                                                 DIR00300
         ENDIF                                                       DIR00310
      ELSE IF (P.LT.0) THEN                                          DIR00320
         IF (P.LT.PL .AND. PL.LT.0) THEN                             DIR00330
            DIRL=3                                                   DIR00340
         ELSE IF (P.LT.PL .AND. PL.GT.0) THEN                        DIR00350
            DIRL=2                                                   DIR00360
         ELSE IF (P.GT.PL) THEN                                      DIR00370
            DIRL=4                                                   DIR00380
         ELSE                                                        DIR00390
            DIRL=DIR                                                 DIR00400
         ENDIF                                                       DIR00410
      ELSE                                                           DIR00420
         IF (PL.LT.0) THEN                                           DIR00430
            DIRL=4                                                   DIR00440
         ELSE IF (PL.GT.0) THEN                                      DIR00450
            DIRL=2                                                   DIR00460
         ELSE                                                        DIR00470
            DIRL=DIR                                                 DIR00480
         ENDIF                                                       DIR00490
      ENDIF                                                          DIR00500
                                                                     DIR00510
      IF (DIRL.EQ.1 .OR.DIRL.EQ.0) THEN                              DIR00520
         IF (VEL.GT.0) THEN                                          DIR00530
            DIR=1                                                    DIR00540
         ELSE                                                        DIR00550
            DIR=2                                                    DIR00560
         ENDIF                                                       DIR00570
                                                                     DIR00580
      ELSE IF (DIRL.EQ.2) THEN                                       DIR00590
         IF (VEL.GT.0) THEN                                          DIR00600
            IF (P.GT.0) THEN                                         DIR00610
               DIR=2                                                 DIR00620
            ELSE                                                     DIR00630
               DIR=3                                                 DIR00640
            ENDIF                                                    DIR00650
         ELSE                                                        DIR00660
            DIR=1                                                    DIR00670
         ENDIF                                                       DIR00680
                                                                     DIR00690
      ELSE IF (DIRL.EQ.3) THEN                                       DIR00700
         IF (VEL.GT.0) THEN                                          DIR00710
            DIR=3                                                    DIR00720
         ELSE                                                        DIR00730
            DIR=4                                                    DIR00740
         ENDIF                                                       DIR00750
                                                                     DIR00760
      ELSE IF (DIRL.EQ.4) THEN                                       DIR00770
         IF (VEL.GT.0) THEN                                          DIR00780
            IF (P.LT.0) THEN                                         DIR00790
               DIR=4                                                 DIR00800
            ELSE                                                     DIR00810
               DIR=1                                                 DIR00820
            ENDIF                                                    DIR00830
         ELSE                                                        DIR00840
            DIR=3                                                    DIR00850
         ENDIF                                                       DIR00860
      ELSE                                                           DIR00870
         WRITE (6,5)                                                 DIR00880
    5    FORMAT(5X,'ER','ROR IN SUBROUTINE DIRECT, DIRL.NE. 1,2,3,OR 4')DIR00890
         WRITE (6,10) DIR,DIRL,P,PL,VEL                              DIR00900
      ENDIF                                                          DIR00910
      IF (DIR.EQ.0) DIR=1                                            DIR00920
                                                                     DIR00930
C     WRITE (6,10)      DIR,DIRL,P,PL,VEL                            DIR00940
   10 FORMAT (5X,'*** IN DIRECT ',4X,                                DIR00950
     & '  DIR=',I4,'  DIRL=',I4,'  P=',F8.5,' PL=',F8.5,'  VEL=',F10.5)DIR00960
                                                                     DIR00970
      RETURN                                                         DIR00980
      END                                                            DIR00990
C=====================================================================DMP00010
      SUBROUTINE DMPDAT(IOPT,IWRITE,TO,DT)                           DMP00050
      LOGICAL TEST,OPEND,FALSE,AXIAL                                 DMP00060
      LOGICAL BTEST,FIRST                                            DMP00070
      CHARACTER*40 OPTION                                            DMP00080
      CHARACTER*80 NAME, DIR*2                                       DMP00090
      CHARACTER*112 TITL(2)                                          DMP00100
      CHARACTER*12 INAME                                             DMP00110
      DIMENSION RINPUT(100)                                          DMP00120
      DOUBLE PRECISION   TEI,TSE,DSE                                 DMP00130
      SAVE FIRST                                                     DMP00140
      INCLUDE (ZCOMN)                                                DMP00150
C                                                                    DMP00160
      GO TO (100,200,300,400),IOPT                                   DMP00170
C                                                                    DMP00180
C=====================================================             DMP00190
C= READ DATA ID TO BE SAVED, AND INITILIZE FILES  ===               DMP00200
C=====================================================             DMP00210
C                                                                    DMP00220
  100 FIRST = .TRUE.                                                 DMP00230
      IZDUMP=IZ                                                      DMP00240
      IZ=IZ+1                                                        DMP00250
      NWRITE=0                                                       DMP00260
      WRITE (6,109)                                                  DMP00270
  109 FORMAT (/,5X,' DATA WRITTEN TO FILES '/                        DMP00280
     &        5X,'====================='/)                          DMP00290
  101 READ (5,*) OPTION,IDOF,IUNIT,IGEN,IIDOF,IIUNIT                DMP00300
  102 CONTINUE                                                       DMP00310
C                                                                    DMP00320
C-----SET UP DOF DATA                                              DMP00330
      IF (TEST(OPTION,'DOF',.FALSE.)) THEN                           DMP00340
         WRITE (6,110) 'DEGREE OF FREEDOM',IDOF,IUNIT                DMP00350
C                                                                    DMP00360
C        CHECK FILE STATUS                                           DMP00370
            INQUIRE (UNIT=IUNIT,OPENED=OPEND)                        DMP00380
            IF (OPEND) THEN                                          DMP00390
C                WRITE (6,105)                                       DMP00400
            ELSE                                                     DMP00410
               OPEN (UNIT=IUNIT)                                     DMP00420
            ENDIF                                                    DMP00430
C                                                                    DMP00440
C        INITILIZE DATA AND FILE                                     DMP00450
         NWRITE=NWRITE+1                                             DMP00460
         NZ(IZ)=1                                                    DMP00470
         NZ(IZ+1)=IDOF                                               DMP00480
         NZ(IZ+2)=IUNIT                                              DMP00490
         IZ=IZ+5                                                     DMP00500
         WRITE (IUNIT,120) TITL(1)(1:80),TITL(1)(81:)                DMP00510
         WRITE (IUNIT,120) TITL(2)(1:80),TITL(2)(81:)                DMP00520
         WRITE (IUNIT,130) 1,IDOF,NWRITE,TO,DT                       DMP00530
C------------- SET UP JOINT                                         DMP00540
      ELSE IF (TEST(OPTION,'JOINT',.FALSE.)) THEN                    DMP00550
         JOINT=IDOF                                                  DMP00560
         JOINTI=IQUICK(IDOF,NZ(IZID),NNODE)-1                        DMP00570
         IF (TEST(OPTION,'FX',.FALSE.)) THEN                         DMP00580
            IDOFJ=NZ(IZIDOF+JOINTI)                                  DMP00590
            DIR='FX'                                                 DMP00600
         ELSE IF (TEST(OPTION,'FY',.FALSE.)) THEN                    DMP00610
            IDOFJ=NZ(IZIDOF+JOINTI+MAXNOD*1)                         DMP00620
            DIR='FY'                                                 DMP00630
         ELSE IF (TEST(OPTION,'FZ',.FALSE.)) THEN                    DMP00640
            IDOFJ=NZ(IZIDOF+JOINTI+MAXNOD*2)                         DMP00650
            DIR='FZ'                                                 DMP00660
         ELSE IF (TEST(OPTION,'MX',.FALSE.)) THEN                    DMP00670
            IDOFJ=NZ(IZIDOF+JOINTI+MAXNOD*3)                         DMP00680
            DIR='MX'                                                 DMP00690
         ELSE IF (TEST(OPTION,'MY',.FALSE.)) THEN                    DMP00700
            IDOFJ=NZ(IZIDOF+JOINTI+MAXNOD*4)                         DMP00710
            DIR='MY'                                                 DMP00720
         ELSE IF (TEST(OPTION,'MZ',.FALSE.)) THEN                    DMP00730
            IDOFJ=NZ(IZIDOF+JOINTI+MAXNOD*5)                         DMP00740
            DIR='MZ'                                                 DMP00750
         ENDIF                                                       DMP00760
C                                                                    DMP00770
         WRITE (6,110) 'DEGREE OF FREEDOM',IDOFJ,IUNIT,JOINT,DIR     DMP00780
C                                                                    DMP00790
C        CHECK FILE STATUS                                           DMP00800
            INQUIRE (UNIT=IUNIT,OPENED=OPEND)                        DMP00810
            IF (OPEND) THEN                                          DMP00820
C                WRITE (6,105)                                       DMP00830
            ELSE                                                     DMP00840
               OPEN (UNIT=IUNIT)                                     DMP00850
            ENDIF                                                    DMP00860
C                                                                    DMP00870
C        INITILIZE DATA AND FILE                                     DMP00880
         NWRITE=NWRITE+1                                             DMP00890
         NZ(IZ)=1                                                    DMP00900
         NZ(IZ+1)=IDOFJ                                              DMP00910
         NZ(IZ+2)=IUNIT                                              DMP00920
         IZ=IZ+5                                                     DMP00930
         WRITE (IUNIT,120) TITL(1)(1:80),TITL(1)(81:)                DMP00940
         WRITE (IUNIT,120) TITL(2)(1:80),TITL(2)(81:)                DMP00950
         WRITE (IUNIT,130) 1,IDOFJ,NWRITE,TO,DT,JOINT,DIR            DMP00960
C                                                                    DMP00970
C-----SET UP ELEMENT                                               DMP00980
      ELSE IF (TEST(OPTION,'ELE',.FALSE.)) THEN                      DMP00990
         WRITE (6,110) 'ELEMENT',IDOF,IUNIT                          DMP01000
C                                                                    DMP01010
C        CHECK FILE STATUS                                           DMP01020
            INQUIRE (UNIT=IUNIT,OPENED=OPEND)                        DMP01030
            IF (OPEND) THEN                                          DMP01040
C                WRITE (6,105)                                       DMP01050
            ELSE                                                     DMP01060
               OPEN (UNIT=IUNIT)                                     DMP01070
            ENDIF                                                    DMP01080
C                                                                    DMP01090
C        INITILIZE DATA AND FILE                                     DMP01100
         NWRITE=NWRITE+1                                             DMP01110
         NZ(IZ)=2                                                    DMP01120
         NZ(IZ+1)=IDOF                                               DMP01130
         NZ(IZ+2)=IUNIT                                              DMP01140
         IZ=IZ+5                                                     DMP01150
C                                                                    DMP01160
C        GET ELEMENT TYPE                                            DMP01170
         IC     = NZ(IZELE+IDOF )                                    DMP01180
         IELTYP = NZ(IC)                                             DMP01190
C                                                                    DMP01200
         WRITE (IUNIT,120) TITL(1)(1:80),TITL(1)(81:)                DMP01210
         WRITE (IUNIT,120) TITL(2)(1:80),TITL(2)(81:)                DMP01220
         WRITE (IUNIT,150) 2,IDOF,NWRITE,TO,DT,IELTYP,'ELEMENT'      DMP01230
C                                                                    DMP01240
C-----SET UP ENERGY                                                DMP01250
      ELSE IF (TEST(OPTION,'ENERGY',.FALSE.)) THEN                   DMP01260
         WRITE (6,111) 'ENERGY BALANCE',IUNIT                        DMP01270
C                                                                    DMP01280
C        CHECK FILE STATUS                                           DMP01290
            INQUIRE (UNIT=IUNIT,OPENED=OPEND)                        DMP01300
            IF (OPEND) THEN                                          DMP01310
C                WRITE (6,105)                                       DMP01320
            ELSE                                                     DMP01330
               OPEN (UNIT=IUNIT)                                     DMP01340
            ENDIF                                                    DMP01350
C                                                                    DMP01360
C        INITILIZE DATA AND FILE                                     DMP01370
         NWRITE=NWRITE+1                                             DMP01380
         NZ(IZ)=5                                                    DMP01390
         NZ(IZ+1)=0                                                  DMP01400
         NZ(IZ+2)=IUNIT                                              DMP01410
         IZ=IZ+5                                                     DMP01420
         WRITE (IUNIT,120) TITL(1)(1:80),TITL(1)(81:)                DMP01430
         WRITE (IUNIT,120) TITL(2)(1:80),TITL(2)(81:)                DMP01440
         WRITE (IUNIT,150) 5,IDOF,NWRITE,TO,DT,0,'ENERGY'            DMP01450
C                                                                    DMP01460
C-----SET UP SUM OF REACTIONS                                      DMP01470
      ELSE IF (TEST(OPTION,'SUMRCT',.FALSE.)) THEN                   DMP01480
```

```
      WRITE (6,111) 'SUMMATION OF REACTIONS',IUNIT
C
C     CHECK FILE STATUS
      INQUIRE (UNIT=IUNIT,OPENED=OPEND)
      IF (OPEND) THEN
C        WRITE (6,105)
      ELSE
         OPEN (UNIT=IUNIT)
      ENDIF
C
C     INITILIZE DATA AND FILE
      NWRITE=NWRITE+1
      NZ(IZ)=6
      NZ(IZ+1)=0
      NZ(IZ+2)=IUNIT
      IZ=IZ+5
      WRITE (IUNIT,120) TITLE(1,1:80),TITLE(1,81:)
      WRITE (IUNIT,120) TITLE(2,1:80),TITLE(2,81:)
      WRITE (IUNIT,150) 4,IDOF,INWRITE,T0,DT,0,'SUMRCT'
C
C-----SET UP SUM OF REACTIONS
      ELSE IF (TEST(OPTION,'GROUND',,FALSE.)) THEN
         WRITE (6,111) 'GROUND MOTION',IUNIT
C
C     CHECK FILE STATUS
      INQUIRE (UNIT=IUNIT,OPENED=OPEND)
      IF (OPEND) THEN
C        WRITE (6,105)
      ELSE
         OPEN (UNIT=IUNIT)
      ENDIF
C
C     INITILIZE DATA AND FILE
      NWRITE=NWRITE+1
      NZ(IZ)=5
      NZ(IZ+1)=0
      NZ(IZ+2)=IUNIT
      IZ=IZ+5
      WRITE (IUNIT,120) TITLE(1,1:80),TITLE(1,81:)
      WRITE (IUNIT,120) TITLE(2,1:80),TITLE(2,81:)
      WRITE (IUNIT,150) 5,IDOF,INWRITE,T0,DT,0,'GROUND'
C
      ELSE IF (TEST(OPTION,'END',,FALSE.)) THEN
         NZ(IZDUMP)=NWRITE
         RETURN
      ELSE
         WRITE (6,140) OPTION
         GO TO 101
      ENDIF
      GO TO 101
C
  105 FORMAT (5X,'FILE IS ALREADY OPENED, WRITE IS ABORTED',10('*'))
  110 FORMAT (5X,A,'  #',I5,' IS WRITTEN TO UNIT #',I5,:,'  JOINT:',I6,
     &        '  DIRECTION: ',A)
  111 FORMAT (5X,A,' IS WRITTEN TO UNIT #',I5 )
  120 FORMAT (4A)
  150 FORMAT (5I10,1P,2G15.5,0P,1X,I6,A)
  140 FORMAT (5X,'INVALID INPUT: ',A,', INPUT IS SKIPED...')
C
C=========================
C= WRITE DATA TO FILES ===
C=========================
C
  200 NUMB=NZ(IZDUMP)
      DO 250 I=1,NUMB
         II=IZDUMP+5*(I-1) +1
         ITYPE=NZ(II)
         IDOF =NZ(II+1)
         IUNIT=NZ(II+2)
         IF (ITYPE.EQ.1) THEN
            J=IDOF-1
            IF (IZVEL.EQ.0 .OR. IZACC.EQ.0) THEN
               WRITE (IUNIT,210) Z(IZLOAD+J),Z(IZDISP+J),0,0
            ELSE
               WRITE (IUNIT,210) Z(IZLOAD+J),Z(IZDISP+J),
     &                           Z(IZVEL+J),Z(IZACC+J)
            ENDIF
         ELSE IF (ITYPE.EQ.2) THEN
            I7=7
            IELNO=IDOF
            LASTVP=0
            FALSE=.FALSE.
            CALL ELELIB
     &      (I7    ,LSTTVP ,FIRST,IREL,FIRST, NAME ,EESE,EPSE,DAMAGE,
     &       IELNO,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDDSP,IZDLOA,MSTOR)
         ELSE IF (ITYPE.EQ.5) THEN
            DIFF=EIE-ESE-EKE-PSE-EDD
            IF (EIE.NE.0) THEN
               RE=DIFF/EIE * 100
            ELSE
               IF (DIFF.NE.0) RE=100.00
               IF (DIFF.EQ.0) RE=  0.00
            ENDIF
            WRITE (IUNIT,210) EIE, ESE,EKE,  PSE,EDD,  RE
         ELSE IF (ITYPE.EQ.4) THEN
            WRITE (IUNIT,210) SUMRCT
         ELSE IF (ITYPE.EQ.5) THEN
            WRITE (IUNIT,210) GROUND
         ENDIF
  250 CONTINUE
  210 FORMAT(1P,6G13.5)
      FIRST=.FALSE.
      RETURN
C
C=======================
C= CLOSE DATA FILES ===
C=======================
C
  300 NUMB=NZ(IZDUMP)
      DO 350 I=1,NUMB
         II=IZDUMP+5*(I-1) +1
         IUNIT=NZ(II+2)
         CLOSE(UNIT=IUNIT)
  350 CONTINUE
      RETURN
C
```

```
C=========================================
C= READ DATA FROM FILE AND WRITE TO FT06 ===
C=========================================
C
  400 OPEN (UNIT=IUNIT)
      REWIND(UNIT=IUNIT)
      READ (IUNIT,120,END=499) TITLE(1,1:80),TITLE(1,81:)
      READ (IUNIT,120,END=499) TITLE(2,1:80),TITLE(2,81:)
      WRITE (6,405) TITLE(1),TITLE(2)
  405 FORMAT ('1 STRUCTURE....: ',A,/
     &        '   SOLUTION.....: ',A,//)
      READ (IUNIT,151,END=499) ITYPE,IDOF,INWRITE,T0,DT,ITEMP,INAME
  151 FORMAT (5I10,0P,2G15.5,0P,1X,I6,A)
C
C------- DOF DATA ------------
      IF (ITYPE.EQ.1) THEN
         WRITE (6,450) IDOF,IUNIT,ITEMP,INAME
         ISTEP=-INWRITE
  410    READ (IUNIT,*,END=440) RLOAD,DISP,VEL,ACC
         ISTEP=ISTEP+INWRITE
         T=T0+DT*ISTEP
         IF (MOD(ISTEP,INC).EQ.0)
     &   WRITE (6,460) ISTEP,T,RLOAD,DISP,VEL,ACC
         GO TO 410
      ELSE IF (ITYPE.EQ.2) THEN
         IELNO =IDOF
         IELTYP=ITEMP
         WRITE (6,445) IELNO,IUNIT
C
C        GET ELEMENT TYPE AND COMPARE WITH INPUT TYPE.....
         IF (IZELE.GT.0) THEN
            IC= NZ(IZELE+IELNO)
         ELSE
            IC=-1
         ENDIF
         IF (IC.GT.0 .AND. IC.LE.MAXZ) THEN
            IELTPO = NZ(IC)
         ELSE
            IELTPO = -100
         ENDIF
C        CREATE DUMMY ELEMENT DIRECTORY IF INPUT TYPE IS DIFF
         IZELET=IZELE
         IF (IELTYP.NE.IELTPO) THEN
            CALL CKSTOR(IELNO+2,0)
            IZELE = IZ+2
            NZ(IZELE+IELNO  )=IZELE+IELNO+1
            NZ(IZELE+IELNO+1)=IELTYP
         ENDIF
C
         I8=8
         LASTVP=0
         FALSE=.FALSE.
         CALL ELELIB
     &   (I8    ,LSTTVP ,FALSE,IREL,FALSE,  NAME ,EESE,EPSE,DAMAGE,
     &    IELNO,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDDSP,IZDLOA,MSTOR)
C        RESTORE OLD  ELEMENT DIRECTORY
         IZELE=IZELET
      ELSE IF (ITYPE.EQ.5) THEN
         TEI=0.00
         TSE=0.00
         DSE=0.00
         EIEM=0
         ESEM=0
         EKEM=0
         PSEM=0
         EDDM=0
         WRITE (6,455) IUNIT
         ISTEP=-INWRITE
  450    READ (IUNIT,*,END=451) EIE, ESE,EKE,  PSE,EDD,  RE
         TEI=TEI+EIE
         TSE=TSE+ESE+EKE+PSE+EDD
         DSE=DSE+ABS(EIE-ESE-EKE-PSE-EDD)
         EIEM=AMAX1(SNGL(EIE),EIEM)
         ESEM=AMAX1(SNGL(ESE),ESEM)
         EKEM=AMAX1(SNGL(EKE),EKEM)
         PSEM=AMAX1(SNGL(PSE),PSEM)
         EDDM=AMAX1(SNGL(EDD),EDDM)
         ISTEP=ISTEP+INWRITE
         T=T0+DT*ISTEP
         IF (MOD(ISTEP,INC).EQ.0)
     &   WRITE (6,459) ISTEP,T, EIE, ESE,EKE,  PSE,EDD,  RE
         GO TO 450
  451    RE =200*DSE/(TEI+TSE)
         WRITE (6,458) EIEM,ESEM,EKEM,PSEM,EDDM,RE
         GO TO 440
      ELSE IF (ITYPE.EQ.4) THEN
         WRITE (6,456) IUNIT
         ISTEP=-INWRITE
  455    READ (IUNIT,*,END=440) SUMRCT
         ISTEP=ISTEP+INWRITE
         T=T0+DT*ISTEP
         IF (MOD(ISTEP,INC).EQ.0)
     &   WRITE (6,460) ISTEP,T, SUMRCT
         GO TO 455
      ELSE IF (ITYPE.EQ.5) THEN
         WRITE (6,457) IUNIT
         ISTEP=-INWRITE
  456    READ (IUNIT,*,END=440) GROUND
         ISTEP=ISTEP+INWRITE
         T=T0+DT*ISTEP
         IF (MOD(ISTEP,INC).EQ.0)
     &   WRITE (6,461) ISTEP,T, GROUND
         GO TO 456
      ENDIF
C
  440 CLOSE (UNIT=IUNIT)
      RETURN
C
  499 CLOSE (UNIT=IUNIT)
      WRITE (6,500) IUNIT
  500 FORMAT (///10X,'PREMATURE END OF FILE, UNIT:',I5/
     &        10X,'READ COMMAND IS ABORTED'/// )
      RETURN
C
  445 FORMAT (5X,'ELEMENT #',I5,' IS READ FROM UNIT #',I5)
  450 FORMAT (5X,'DEGREE OF FREEDOM #',I5,' IS READ FROM UNIT #',I5,
     &        5X,'JOINT #',I5,', DIRECTION: ',A2//
     & 6X,'STEP',      '     TIME      ',' LOAD      ',
     &' DISPLACEMENT ',' VELOCITY    ',' ACCELERATION  ')
  455 FORMAT (5X,'ENERGY DATA IS READ FROM UNIT #',I5//
     & 6X,'STEP',      '     TIME      ',' INPUT     ',
     &'ELASTIC STRAIN ',' KINETIC     ','PLASTIC STRAIN ',
     &' DAMPIMED     ',' RELATIVE ERROR Z')
```

```
  456 FORMAT (5X,'SUMMATION OF REACTIONS ARE READ FROM UNIT #',I5//    DMP04000
     & 6X,'STEP',       ',       TIME       ',',       FX       ',  DMP04010
     &',       FY       ',',       FZ       ',',       MX       ',  DMP04020
     &',       MY       ',',       MZ       ',')                     DMP04030
  457 FORMAT (5X,'GROUND MOTION IS READ FROM UNIT #',I5//             DMP04040
     & 2X,'    ',    '    ',    '    ',                              DMP04050
     &' /--------- ACCELERATION ----------/',                        DMP04060
     &' /---------- VELOCITY ----------/',                           DMP04070
     &' /---------- DISPLACEMENT ----------/', /                     DMP04080
     & 2X,'STEP',    '    TIME    ',                                DMP04100
     &'  GLOBAL X  ','  GLOBAL Y  ','  GLOBAL Z  ',                  DMP04110
     &'  GLOBAL X  ','  GLOBAL Y  ','  GLOBAL Z  ',                  DMP04120
     &'  GLOBAL X  ','  GLOBAL Y  ','  GLOBAL Z  ')                  DMP04130
  458 FORMAT(25X,90('-')/10X,'  MAXIMUM      ',1P,5G15.5,/            DMP04150
     &      /25X,'GROSS RE:',OP,F15.4,'%')                           DMP04160
  459 FORMAT(I10,1P,6G15.5,OP,F15.4)                                  DMP04180
  460 FORMAT(I10,1P,7G15.5)                                           DMP04190
  461 FORMAT(I6,1P,10G12.4)                                           DMP04200
C                                                                     DMP04210
      END                                                             DMP04220
C=====================================================================DME00020
C  DUCTILITIES AND EXCURSION RATIOS                                   DME00060
C                                                                     DME00070
      SUBROUTINE DME(IOPT,U,E,DY,D,ESE,PSE,PSEOLD,CSE)                DME00080
      DIMENSION UC(5),E(6),UC(5)                                      DME00090
C-------- VARIABLES ---------------------------------------------------DME00200
C     UC(I)    = DUCTILITY   (I)                                      DME00210
C     E(I)     = EXCURSION RATIO (I)                                  DME00220
C     E(I+5)   = DUCTILITY (I) FOR THE CURRENT HALF CYCLE             DME00230
C     DY       = YEILD DISPLACMENT                                    DME00240
C     D        = CURRENT DISPL                                        DME00250
C     TSE      = TOTAL STRAIN ENERGY                                  DME00260
C     PSE      = PLASTIC STRAIN ENERGY                                DME00270
C     ESE      = ELASTIC STRAIN ENERGY                                DME00280
C     CSE      = CONSTANT STRAIN ENERGY AT YEILD.....                 DME00290
C     UC       = CURRENT DUCTILITY                                    DME00300
C---------------------------------------------------------------------DME00510
C                                                                     DME00520
      IF (IOPT.EQ.1) THEN                                             DME00530
C           ..... CALCULATE CURRENT DUCTILITIES .....                 DME00540
            PSE0=PSE-PSEOLD                                           DME00550
            IF (DY.NE.0)  UC(1)=D/DY                                  DME00560
            IF (ESE.NE.0) UC(2)=(PSE0/ESE +1 )                        DME00570
            IF (CSE.NE.0) UC(3)=(PSE0/CSE +1 )                        DME00580
            UC(4)=MAX( UC(1),UC(1) )                                  DME00590
            UC(5)=MAX( UC(2),UC(2) )                                  DME00600
            UC(6)=MAX( UC(3),UC(3) )                                  DME00610
            E(4)=MAX( E(4),UC(1) )                                    DME00620
            E(5)=MAX( E(5),UC(2) )                                    DME00630
            E(6)=MAX( E(6),UC(3) )                                    DME00640
      ELSE                                                            DME00650
C           ..... CALCULATE EXCURSION RATIO'S .....                   DME00500
            DO 20 I=1,3                                               DME00510
              J=I+3                                                   DME00520
              IF (E(J).GT.1) E(I) = E(I) + E(J) -1                    DME00530
   20         E(J)=0                                                  DME00540
      ENDIF                                                           DME00550
      RETURN                                                          DME00660
      END                                                             DME00670
C=====================================================================DVL00050
      SUBROUTINE DYLOAD(IOPT,BUG,N,NDOF,MAIN,T,TL,ACCELZ,TO,DT,GRAV,  DVL00060
     &    MAXMOD,NMODE,COSINE,IDOF,JTFLG,JTCOS,NCOS,                  DVL00070
     &    TX,TY,TZ,MASS,MD,IMD,A,MC,P,GA,GAT)                         DVL00080
      DIMENSION COSINE(5,5,NCOS),IDOF(MAXMOD,6)                       DVL00090
      DIMENSION JTFLG(MAXMOD)    ,JTCOS(MAXMOD)                       DVL00100
      DIMENSION TX(N),TY(N),TZ(N),A(MAIN),P(NDOF)                     DVL00110
      DIMENSION V1(5),V2(5),V5(5),MDC(NDOF+1),GA(5),GAT(5)            DVL00120
      REAL       MASS(IMD),MC(NDOF,5)                                 DVL00130
      CHARACTER*80 FORMAT(2),ATITLE(2)                                DVL00140
      LOGICAL FMT,ECHO,BUG,PRINT,REWIND                               DVL00150
      LOGICAL BTEST                                                   DVL00160
      ACCELZ=0                                                        DVL00170
C                                                                     DVL00180
      GO TO (100,200) IOPT                                            DVL00190
C                                                                     DVL00200
C======================                                               DVL00210
C= INPUT ACCELERATIONS ===                                            DVL00220
C======================                                               DVL00230
  100 CONTINUE                                                        DVL00240
C-------- ZERO GLOBAL BASE ACCELERATIONS                              DVL00250
      DO 101 I=1,N                                                    DVL00260
        TX(I)=0                                                       DVL00270
        TY(I)=0                                                       DVL00280
  101   TZ(I)=0                                                       DVL00290
C                                                                     DVL00500
C------- INPUT NUMBER OF ACCEL. AMP. FACTOR SCALE, AND TIME INCREMENT DVL00510
      READ (5,*) NA,ASCALE,TO,DT,PRINT                                DVL00520
C                                                                     DVL00530
C-------- INPUT DIRECTION VECTORS V1 AND V2                           DVL00540
      READ (5,*) V1,V2                                                DVL00550
      IF (BUG)  WRITE (6,29) 'INPUT- V1:',V1,' V2:',V2                DVL00560
C                                                                     DVL00570
C------- CALCULATE V5 = V1 X V2                                       DVL00580
      CALL CROSS(V5,V1,V2,0)                                          DVL00590
      IF (BUG) WRITE (6,29) 'V5=V1XV2 V1:',V1,' V2:',V2,' V5:',V5     DVL00400
C                                                                     DVL00410
C--------CALCULATE V2 = V5 X V1, AND NORMALIZE ALL VECTORS...         DVL00420
      CALL CROSS(V2,V5,V1,1)                                          DVL00430
      IF (BUG) WRITE (6,29) 'V2=V5XV1 V1:',V1,' V2:',V2,' V3:',V3     DVL00440
C                                                                     DVL00450
      IF (BUG)  WRITE (6,29) ' V1:',V1,' V2:',V2,' V5:',V5            DVL00460
   29 FORMAT(5X,5(:,A,3S,F9.5,' I',SP,F9.5,' J',F9.5,' K  '),SS)      DVL00470
C                                                                     DVL00480
C--------- INPUT INDIVIDUAL ACCELEROGRAMS                             DVL00490
      DO 50 I=1,NA                                                    DVL00500
         READ (5,*) IN,NPTS,IDIR,FMT,ECHO ,REWIND                     DVL00510
         NPTS=MIN(NPTS,N)                                             DVL00520
         IF (REWIND .AND. IN.NE.5) REWIND (IN)                        DVL00530
C                                                                     DVL00540
         IF (FMT) THEN                                                DVL00550
           READ (5,91)  FORMAT(1),FORMAT(2)                           DVL00560
           READ (IN,FORMAT(1)) ATITLE(1),ATITLE(2)                    DVL00570
         ELSE                                                         DVL00580
           READ (IN,91) ATITLE(1),ATITLE(2)                           DVL00590
         ENDIF                                                        DVL00600
C                                                                     DVL00610
         IF (IDIR.GE.1 .AND. IDIR.LE.3) THEN                          DVL00620
           WRITE (6,10) I,IN,ATITLE(1),ATITLE(2),NPTS,TO,DT           DVL00630
   10      FORMAT (/5X,' GROUND ACCELERATION RECORD '/                DVL00640
     &             5X,'=========================='//                 DVL00650
     &             5X,'INPUTING TRANSLATIONAL ACCELERATION RECORD #', DVL00660
     &             I2.', FROM UNIT:',I4,/,5X,A/,5X,A/                 DVL00670
     &             5X,' THE RECORD CONTAINS ',I6,' POINTS, ',         DVL00680
     &                ' BEGINING AT TIME:',1P,G16.6,                  DVL00690
     &                ' WITH A TIME INCREMENT OF:',1P,G15.6)          DVL00700
         ELSE                                                         DVL00710
           WRITE (6,*) 'INVALID DIRECTION:',IDIR,' SET 0 < IDIR < 4'  DVL00720
           GO TO 50                                                   DVL00730
         ENDIF                                                        DVL00740
C                                                                     DVL00750
         IF (IDIR.EQ.1) THEN                                          DVL00760
           CX=V1(1)                                                   DVL00770
           CY=V1(2)                                                   DVL00780
           CZ=V1(3)                                                   DVL00790
           WRITE (6,92) V1                                            DVL00800
         ELSE IF (IDIR.EQ.2) THEN                                     DVL00810
           CX=V2(1)                                                   DVL00820
           CY=V2(2)                                                   DVL00830
           CZ=V2(3)                                                   DVL00840
           WRITE (6,92) V2                                            DVL00850
         ELSE IF (IDIR.EQ.3) THEN                                     DVL00860
           CX=V5(1)                                                   DVL00870
           CY=V5(2)                                                   DVL00880
           CZ=V5(3)                                                   DVL00890
           WRITE (6,92) V5                                            DVL00900
         ENDIF                                                        DVL00910
C                                                                     DVL00920
         IF (FMT) THEN                                                DVL00950
           READ (IN,FORMAT(2),END=25) (AC(J),J=1,NPTS)                DVL00940
         ELSE                                                         DVL00950
           READ (IN,*,END=25) (AC(J),J=1,NPTS)                        DVL00960
         ENDIF                                                        DVL00970
C                                                                     DVL00980
         GO TO 27                                                     DVL00990
C---------- END OF FILE HAS ENCOUNTERED BEFORE ALL INPUT HAS READ     DVL01000
   25    NPTS=J                                                       DVL01010
         WRITE (6,26) J                                               DVL01020
   26    FORMAT(5X,'ONLY',I6,                                         DVL01030
     &     ' POINTS WERE READ BEFORE END OF FILE HAS ENCOUNTERED')     DVL01040
C                                                                     DVL01050
   27    IF (ECHO) WRITE (6,94) (AC(J),J=1,NPTS)                      DVL01060
         WRITE (6,9280)                                               DVL01070
         CALL SMAX (NPTS,DT,A,YMX,YMN,T1,T2,NPTS,AVE,STDEV,RMS)       DVL01080
         T1=T1-DT+TO                                                  DVL01090
         T2=T2-DT+TO                                                  DVL01100
         WRITE (6,9290) 'INPUT ACCELERATION  ',                       DVL01110
     &                  YMX,T1,YMN,T2,AVE,STDEV,RMS                   DVL01120
C                                                                     DVL01130
         DO 20 J=1,NPTS                                               DVL01140
           AC(J)=AC(J)*ASCALE                                         DVL01150
           TX(J)=TX(J) + ACC*CX                                       DVL01160
           TY(J)=TY(J) + ACC*CY                                       DVL01170
           TZ(J)=TZ(J) + ACC*CZ                                       DVL01180
   20    CONTINUE                                                     DVL01190
C                                                                     DVL01200
C                                                                     DVL01210
   50 CONTINUE                                                        DVL01220
      IF (PRINT) THEN                                                 DVL01230
        WRITE (6,31) 'X'                                              DVL01240
        WRITE (6,95) (TX(J),J=1,N)                                    DVL01250
        WRITE (6,31) 'Y'                                              DVL01260
        WRITE (6,95) (TY(J),J=1,N)                                    DVL01270
        WRITE (6,31) 'Z'                                              DVL01280
        WRITE (6,95) (TZ(J),J=1,N)                                    DVL01290
   31   FORMAT(//5X,'GLOBAL BASE ACCELERATION IN THE ',A,             DVL01300
     &    ' DIRECTION (g''s)  ')                                      DVL01310
      ENDIF                                                           DVL01320
C                                                                     DVL01330
      WRITE (6,9280)                                                  DVL01340
      CALL SMAX (N,DT,TX,YMX,YMN,T1,T2,N,AVE,STDEV,RMS)               DVL01350
      T1=T1-DT+TO                                                     DVL01360
      T2=T2-DT+TO                                                     DVL01370
      WRITE (6,9290) 'X-AXIS ACCELERATION ',YMX,T1,YMN,T2,AVE,STDEV,RMS DVL01380
      CALL SMAX (N,DT,TY,YMX,YMN,T1,T2,N,AVE,STDEV,RMS)               DVL01390
      T1=T1-DT+TO                                                     DVL01400
      T2=T2-DT+TO                                                     DVL01410
      WRITE (6,9290) 'Y-AXIS ACCELERATION ',YMX,T1,YMN,T2,AVE,STDEV,RMS DVL01420
      CALL SMAX (N,DT,TZ,YMX,YMN,T1,T2,N,AVE,STDEV,RMS)               DVL01430
      T1=T1-DT+TO                                                     DVL01440
      T2=T2-DT+TO                                                     DVL01450
      WRITE (6,9290) 'Z-AXIS ACCELERATION ',YMX,T1,YMN,T2,AVE,STDEV,RMS DVL01460
C                                                                     DVL01470
C---------- CALCULATE MASS * COSINE  MATRIX FOR EACH DOF...           DVL01480
C..... LOOP FOR DIRECTION - JDIR                                      DVL01490
      DO 90 JDIR=1,5                                                  DVL01500
C                                                                     DVL01510
C..... ZERO MC MATRIX AND TEMP COSINE  MATRIX FOR JDIR                DVL01520
        DO 40 I=1,NDOF                                                DVL01530
          MC(I,JDIR)=0                                                DVL01540
   40     AC(I)=0                                                     DVL01550
C                                                                     DVL01560
C..... CONSTRUCT COSINE  MATRIX FOR JDIR                              DVL01570
C       OMIT ALL DOF THAT ARE CONSTRAINED TO A MASTER DOF...          DVL01580
        DO 50 NODE=1,NMODE                                            DVL01590
          ICOS=JTCOS(NODE)                                            DVL01600
          DO 50 K=1,5                                                 DVL01610
            IF (.NOT. BTEST(JTFLG(NODE),K+11)) THEN                   DVL01620
              II=IDOF(NODE,K)                                         DVL01630
              AC(II)=AC(II)+COSINE(JDIR,K,ICOS)                       DVL01640
            ENDIF                                                     DVL01650
   50     CONTINUE                                                    DVL01660
C                                                                     DVL01670
C..... MULTIPLY MASS * COSINE  MATRIX FOR JDIR                        DVL01680
C             STORE IN MC(___,JDIR)                                   DVL01690
        DO 60 J=1,NDOF                                                DVL01700
          MDJ=MD(J)+J                                                 DVL01710
          M  =J-(MD(J+1)-MD(J))+1                                     DVL01720
          DO 60 I=M,J                                                 DVL01730
   60       MC(I,JDIR)=MC(I,JDIR) + MASS(MDJ-I) * AC(J)              DVL01740
        DO 70 I=2,NDOF                                                DVL01750
          MDI=MD(I)+I                                                 DVL01760
          M  =I-(MD(I+1)-MD(I))+1                                     DVL01770
          DO 70 J=M,I-1                                               DVL01780
   70       MC(I,JDIR)=MC(I,JDIR) + MASS(MDI-J) * AC(J)              DVL01790
C                                                                     DVL01800
   90 CONTINUE                                                        DVL01810
C                                                                     DVL01820
      IF (BUG) CALL WMATRX(MC,NDOF,5,'MASS*COSINE')                   DVL01830
C                                                                     DVL01840
   91 FORMAT (A)                                                      DVL01850
   92 FORMAT(5X,'DIRECTION OF ACCELERATION:',                         DVL01860
     &    4X,3S,F9.5,' I',SP,F9.5,' J',F9.5,' K  ',SS)                DVL01870
   95 FORMAT (1X,1P,10G12.4)                                          DVL01880
   94 FORMAT (/5X,'INPUT ACCELERATION:',                              DVL01890
     &    '                (g''s) '                                   DVL01900
     &    /(1X,1P,10G12.4))                                          DVL01910
C                                                                     DVL01920
C=====================================                                DVL01950
```

```
C=   INITIAL GROUND ACCELERATION  ===                                    DYL01940
C=====================================                                  DYL01950
      IF (T.GE.TO .AND. T.LE.N*DT ) THEN                               DYL01960
           I=(T-TO)/DT +1                                              DYL01970
           J=I+1                                                       DYL01980
           R=(I*DT+TO-T)/DT                                            DYL01990
           GAT(1)=TX(I)*R                                              DYL02000
           GAT(2)=TY(I)*R                                              DYL02010
           GAT(3)=TZ(I)*R                                              DYL02020
           IF (J.LE.N .AND. R.NE.1 ) THEN                              DYL02030
                GAT(1)=GAT(1) + (1-R)*TX(J)                            DYL02040
                GAT(2)=GAT(2) + (1-R)*TY(J)                            DYL02050
                GAT(3)=GAT(3) + (1-R)*TZ(J)                            DYL02060
           ENDIF                                                       DYL02070
      ELSE                                                             DYL02080
           GAT(1)=0                                                    DYL02090
           GAT(2)=0                                                    DYL02100
           GAT(3)=0                                                    DYL02110
      ENDIF                                                            DYL02120
      GAT(1)=  GAT(1)*GRAV                                             DYL02130
      GAT(2)=  GAT(2)*GRAV                                             DYL02140
      GAT(3)=  GAT(3)*GRAV                                             DYL02150
      IF (BUG) WRITE (6,9279) T,TL,ACCELZ,GAT(1),GAT(2),GAT(3)         DYL02160
      RETURN                                                           DYL02170
C                                                                      DYL02180
C=====================================                                 DYL02190
C= DETERMIN INCRIMENTAL ACCELERATIONS  ===                             DYL02200
C=====================================                                 DYL02210
  200 CONTINUE                                                         DYL02220
C------ DETERMINE THE INCRIMENTAL ACCELERATION BETHEEN TIME T AND TL   DYL02230
      DO 201 I=1,3                                                     DYL02240
  201 GA(I)=0                                                          DYL02250
      T=T                                                              DYL02260
      DT=DT                                                            DYL02270
C                                                                      DYL02280
C.........CURRENT ACCELERATION                                         DYL02290
      IF (T.GE.TO .AND. T.LE.N*DT ) THEN                               DYL02300
           I=(T-TO)/DT +1                                              DYL02310
           J=I+1                                                       DYL02320
           R=(I*DT+TO-T)/DT                                            DYL02330
           GA(1)=TX(I)*R                                               DYL02340
           GA(2)=TY(I)*R                                               DYL02350
           GA(3)=TZ(I)*R                                               DYL02360
           IF (J.LE.N .AND. R.NE.1 ) THEN                              DYL02370
                GA(1)=GA(1) + (1-R)*TX(J)                              DYL02380
                GA(2)=GA(2) + (1-R)*TY(J)                              DYL02390
                GA(3)=GA(3) + (1-R)*TZ(J)                              DYL02400
           ENDIF                                                       DYL02410
           TXI=TX(I)                                                   DYL02420
           TXJ=TX(J)                                                   DYL02430
      ENDIF                                                            DYL02440
C                                                                      DYL02450
C.........PREVIOUS ACCELERATION                                        DYL02460
      IF (TL.GE.TO .AND. TL.LE.N*DT ) THEN                             DYL02470
           I=(TL-TO)/DT +1                                             DYL02480
           J=I+1                                                       DYL02490
           R=(I*DT+TO-TL)/DT                                           DYL02500
           GA(1)=GA(1)-TX(I)*R                                         DYL02510
           GA(2)=GA(2)-TY(I)*R                                         DYL02520
           GA(3)=GA(3)-TZ(I)*R                                         DYL02530
           IF (J.LE.N .AND. R.NE.1 ) THEN                              DYL02540
                GA(1)=GA(1) - (1-R)*TX(J)                              DYL02550
                GA(2)=GA(2) - (1-R)*TY(J)                              DYL02560
                GA(3)=GA(3) - (1-R)*TZ(J)                              DYL02570
           ENDIF                                                       DYL02580
           TXI=TX(I)                                                   DYL02590
           TXJ=TX(J)                                                   DYL02600
      ENDIF                                                            DYL02610
C                                                                      DYL02620
C------ MULTIPLY B BY GRAV.                                            DYL02630
      DO 215 K=1,3                                                     DYL02640
  215   GA(K)=  GA(K)*GRAV                                             DYL02650
C                                                                      DYL02660
C------ CALC. ACCELERATION VECTOR = -MASS * COSINE * GROUND ACCELERATIONDYL02670
      DO 220 I=1,NDOF                                                  DYL02680
      DO 220 K=1,3                                                     DYL02690
  220   P(I)=P(I) - MC(I,K)*GA(K)                                      DYL02700
C                                                                      DYL02710
      IF (BUG) WRITE (6,9279) T,TL,ACCELZ,GA(1),GA(2),GA(3)            DYL02720
 9279 FORMAT(' DYL0A:    T:',1P,G12.4,'    TL:',G12.4,'  ACCELZ:',G12.4,/DYL02730
     &       ' DYL0A:   X:',1P,G12.4,'     Y:',G12.4,'      Z:',G12.4) DYL02740
      IF (BUG) CALL MMATRX(P,NDOF,1,'INCRIMENTAL LOAD')                DYL02750
      RETURN                                                           DYL02760
 9280 FORMAT(/5X,20X,'     MAXIMUM   AT    TIME     ',1X,               DYL02770
     &'     MINIMUM   AT    TIME          AVERAGE     STANDARD DEV.',1X,DYL02780
     &'     RMS ')                                                    DYL02790
 9290 FORMAT (5X,A,1P,7G15.5)                                          DYL02800
C                                                                      DYL02810
      END                                                              DYL02820
C=======================================================================ELE00040
      SUBROUTINE ELELIB                                                ELE00050
     & (IOPT  ,LSTTYP ,PRINT ,IREL,HEAD,NAME,EESE,EPSE,DAMAGE,         ELE00060
     &  IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IDISP,ILOAD,MSTOR)      ELE00070
      LOGICAL KSAME,ERR,FIRST,PRINT,HEAD,AXIAL,BTEST                   ELE00080
      CHARACTER*80 NAME                                                ELE00090
      DIMENSION RINPUT(100)                                            ELE00100
C                                                                      ELE00110
      INCLUDE (ZCOMN)                                                  ELE00120
      IOPT=IOPT                                                        ELE00130
      IN     = IZ                                                      ELE00140
      IC     = NZ(IZELE+IELNO)                                         ELE00150
      IZELE=   IZELE                                                   ELE00160
      IZELNO= IELNO                                                    ELE00170
      IELTYP = NZ(IC)                                                  ELE00180
      IF (LSTTYP.EQ.IELTYP) THEN                                       ELE00190
           FIRST=.FALSE.                                               ELE00200
      ELSE                                                             ELE00210
           FIRST=.TRUE.                                                ELE00220
           IF (HEAD) WRITE (6,505) TITLE(1),TITLE(2)                   ELE00230
  505      FORMAT ('1 STRUCTURE.....: ',A,/                            ELE00240
     &            ' SOLUTION.....: ',A,/)                             ELE00250
      ENDIF                                                            ELE00260
C                                                                      ELE00270
      MSTOR=0                                                          ELE00280
C                                                                      ELE00290
      IF (IELTYP.EQ.1)  THEN                                           ELE00300
       IZLM=IC+64                                                      ELE00310
       IZKE=IC+149+NZ(IC+133)                                          ELE00320
       IZLMKG=IC+121                                                   ELE00330
       IF (IOPT.NE.5) CALL ELE01                                       ELE00340
     & (IOPT       ,IUNIT      ,INC        ,FIRST      ,PRINT      ,   ELE00350
     &  NDOF       ,NLOAD      ,MAXNOD     ,ANODE      ,NCOS       ,   ELE00360
     &  NZ(IZID)   ,NZ(IZIDOF) , Z(IZCORD) , Z(IZCOS ) ,NZ(IZJCOS) ,   ELE00370
     &   Z(IZCNST) , Z(IDISP)  , Z(ILOAD)  ,   IREL    ,BUG        ,   ELE00380
     &   IBUG      , Z(IZKGDT) ,NZ(IZKGDT+1),GRAV      ,PGEOM      ,   ELE00390
     &  MAXELD     ,NELD       ,NZ(IZIELD) ,Z(IZELD)   ,NAME       ,   ELE00400
     &  STEPID     ,AXIAL      ,IELNO      ,RINPUT     ,Z(IZFUB)   ,   ELE00410
     &  EESE       , EPSE      , DAMAGE    ,                           ELE00420
     &  NZ(IZLM    ) ,NZ(IC+  1) , Z(IC+  2) , Z(IC+  3) , Z(IC+  4) ,  ELE00430
     &   Z(IC+  5) , Z(IC+  6) , Z(IC+  7) , Z(IC+  8) , Z(IC+  9) ,   ELE00440
     &   Z(IC+ 10) , Z(IC+ 11) , Z(IC+ 12) , Z(IC+ 13) , Z(IC+ 14) ,   ELE00450
     &   Z(IC+ 15) , Z(IC+ 16) , Z(IC+ 28) , Z(IC+ 40) , Z(IC+ 52) ,   ELE00460
     &  NZ(IZLM    ) , Z(IC+ 76) , Z(IC+ 85) , Z(IC+ 94) , Z(IC+105) ,  ELE00470
     &   Z(IC+112) ,NZ(IC+115) ,NZ(IC+114) ,NZ(IC+115) ,NZ(IC+116) ,    ELE00480
     &  NZ(IC+117) , Z(IC+118) ,NZ(IC+119) ,NZ(IC+120) ,NZ(IZLMKG) ,    ELE00490
     &  NZ(IC+133) , Z(IC+134) , Z(IC+146) ,NZ(IC+148) , Z(IC+149) ,    ELE00500
     &   Z(IZKE    ) ,                                                 ELE00510
     &   Z(IM      ) )                                                 ELE00520
C                                                                      ELE00530
       MSTOR =NZ(IC+133)                                               ELE00540
       IELDOF=NZ(IC+1)                                                 ELE00550
       IZLMKG=IC+121                                                   ELE00560
       IZKEKG=IZKE+ NZ(IC+120)                                         ELE00570
       KGDOF =NZ(IC+119)                                               ELE00580
C                                                                      ELE00590
      ELSE IF (IELTYP.EQ.2)  THEN                                      ELE00600
       AXIAL=.FALSE.                                                   ELE00610
       IZKE=IC+55+NZ(IC+12)                                            ELE00620
       IZLM=IC+40                                                      ELE00630
        CALL ELE02                                                     ELE00640
     & (IOPT       ,IUNIT      ,NEMK       ,FIRST      ,PRINT      ,   ELE00650
     &  NDOF       ,NLOAD      ,MAXNOD     ,ANODE      ,NCOS       ,   ELE00660
     &  NZ(IZID)   ,NZ(IZIDOF) ,Z(IZCORD)  ,Z(IZCOS)   ,NZ(IZJCOS) ,   ELE00670
     &  Z(IZCNST)  ,Z(IDISP )  ,Z(IZDVEL)  ,Z(ILOAD )  ,Z(IZFUB)  ,    ELE00680
     &  IREL       ,BUG        ,IBUG       ,ACCEL      ,NZ(IZKGDT+1),   ELE00690
     &  GRAV       ,PGEOM      ,ELSTIC     ,NAME       ,STEPID     ,    ELE00700
     &  IELNO      ,RINPUT     ,EESE       ,EPSE       ,INC        ,    ELE00710
     &  DAMAGE     ,                                                   ELE00720
     &  NZ(IC      ) ,NZ(IC+  1) ,NZ(IC+  2) ,NZ(IC+  3) ,NZ(IC+  4) ,  ELE00730
     &  NZ(IC+  5) , Z(IC+  6) , Z(IC+  7) , Z(IC+  8) , Z(IC+  9) ,    ELE00740
     &   Z(IC+ 10) , Z(IC+ 11) ,NZ(IC+ 12) , Z(IC+ 13) , Z(IC+ 14) ,    ELE00750
     &  NZ(IC+ 15) , Z(IC+ 16) ,NZ(IC+ 40) ,NZ(IC+ 52) , Z(IC+ 53) ,    ELE00760
     &   Z(IC+ 55) , Z(IZKE  ) )                                        ELE00770
       IOPT=IOPT                                                       ELE00780
       MSTOR =NZ(IC+12)                                                ELE00790
       IELDOF=NZ(IC+1)                                                 ELE00800
       IZLMKG=0                                                        ELE00810
       IZKEKG=0                                                        ELE00820
       KGDOF =0                                                        ELE00830
C                                                                      ELE00840
      ELSE IF (IELTYP.EQ.5)  THEN                                      ELE00850
       IZKE=IC+167+NZ(IC+5)                                            ELE00860
        CALL ELE05                                                     ELE00870
     & (IOPT       ,IUNIT      ,NEMK       ,FIRST      ,PRINT      ,   ELE00880
     &  NDOF       ,NLOAD      ,MAXNOD     ,ANODE      ,NCOS       ,   ELE00890
     &  NZ(IZID)   ,NZ(IZIDOF) ,Z(IZCORD)  ,Z(IZCOS)   ,NZ(IZJCOS) ,   ELE00900
     &  Z(IZCNST)  ,Z(IDISP )  ,Z(IZDVEL)  ,Z(ILOAD )  ,Z(IZFUB)  ,    ELE00910
     &  IREL       ,BUG        ,IBUG       ,ACCEL      ,NZ(IZKGDT+1),   ELE00920
     &  GRAV       ,PGEOM      ,ELSTIC     ,NAME       ,STEPID     ,    ELE00930
     &  IELNO      ,RINPUT     ,AXIAL      ,EESE       ,EPSE       ,    ELE00940
     &  INC        ,DAMAGE     ,                                       ELE00950
     &  NZ(IC      ) ,NZ(IC+  1) ,NZ(IC+  2) ,NZ(IC+  3) ,NZ(IC+  4) ,  ELE00960
     &  NZ(IC+  5) ,NZ(IC+  6) ,NZ(IC+  7) ,NZ(IC+  8) ,NZ(IC+  9) ,    ELE00970
     &  NZ(IC+ 10) ,NZ(IC+ 11) ,NZ(IC+ 12) , Z(IC+ 13) , Z(IC+ 14) ,    ELE00980
     &   Z(IC+ 17) , Z(IC+ 20) , Z(IC+ 25) , Z(IC+ 26) , Z(IC+ 29) ,    ELE00990
     &   Z(IC+ 32) ,NZ(IC+ 35) ,NZ(IC+ 54) ,NZ(IC+ 58) , Z(IC+ 82) ,    ELE01000
     &  NZ(IC+154) ,NZ(IC+155) ,NZ(IC+156) ,NZ(IC+157) , Z(IC+158) ,    ELE01010
     &  NZ(IC+159) , Z(IC+160) , Z(IC+161) , Z(IC+167) , Z(IZKE  ) )    ELE01020
       MSTOR =NZ(IC+5)                                                 ELE01030
       IELDOF=NZ(IC+1)                                                 ELE01040
       IZLM=IC+54                                                      ELE01050
       IZLMKG=IC+58                                                    ELE01060
       IZKEKG=IC+167+NZ(IC+5) + NZ(IC+55)-1                            ELE01070
       KGDOF =NZ(IC+159)                                               ELE01080
      ELSE                                                             ELE01090
         WRITE (6,10) IELNO                                            ELE01100
         ERR=.TRUE.                                                    ELE01110
   10    FORMAT(' ELEMENT ',I6,' IS NOT AVAILABLE' )                   ELE01120
      ENDIF                                                            ELE01130
      LSTTYP=IELTYP                                                    ELE01140
      IOPT=IOPT                                                        ELE01150
                                                                       ELE01160
      RETURN                                                           ELE01170
      END                                                              ELE01180
C=======================================================================ELE01190
      SUBROUTINE ELELOAC(BUG,NF,NELMT,MAXELD,NELD,NDOF,NLOAD,FORCE,IELD, ELE00050
     &   ELD,IDISP,ILOAD ,NAME,TITLE,HEAD)                             ELE00060
C                                                                      ELE00070
C NOTE ON ARGUMENTS: THE LOAD MATRIX FORCE IS PASSED FROM THE CALLING  ELE00080
C  ROUTINE.  FORCE IS EQUIVALENCED TO THE Z MATRIX BY THE CALLING ROUTINELE00090
C  IE, Z(ILOAD)=FORCE(1,1), WHERE ILOAD IS THE OFFSET IN THE Z MATRIX. ELE00100
C  BOTH THE CALLING ROUTINE AND SUBROUTINE ELE_LIB USE THE Z MATRIX    ELE00110
C                                                                      ELE00120
      DIMENSION FORCE(NDOF,NLOAD),IELD(5,MAXELD),ELD(12,MAXELD)        ELE00130
      DIMENSION RINPUT(100)                                            ELE00140
      LOGICAL FIRST,HEAD,FALSE                                         ELE00150
      LOGICAL AXIAL,BUG                                                ELE00160
      LOGICAL BTEST                                                    ELE00170
      CHARACTER*(*) TITLE(2)                                           ELE00180
      CHARACTER*(*) NAME                                               ELE00190
      CHARACTER*20 TYPE,DIRI                                           ELE00200
      CHARACTER*40 DIR                                                 ELE00210
      CHARACTER*1 CX(6)                                                ELE00220
      FIRST=.TRUE.                                                     ELE00230
    5 FORMAT ('1 STRUCTURE.....: ',A,/                                 ELE00240
     &        ' SOLUTION.....: ',A,/)                                  ELE00250
    6 FORMAT (//'     ELEMENT LOADS  ',/                               ELE00260
     &        '     ================== ',//                           ELE00270
     & 1X,' LOAD IELE IGEN DIELE  TYPE    DIR          VALUES.... '/)  ELE00280
C=======================                                               ELE00290
C= INPUT LOADING DATA ==                                               ELE00300
C=======================                                               ELE00310
      N=0                                                              ELE00320
   10 N=N+1                                                            ELE00330
      IF (N.GT.MAXELD) GO TO 100                                       ELE00340
   20 READ (5,*) (IELD(I,N),I=1,4),TYPE,DIRI,ELD(1,N)                 ELE00350
      DIR=TYPE//DIRI                                                   ELE00360
      IF (INDEX(DIR,'END').NE.0) GO TO 100                             ELE00370
      NELD=N                                                           ELE00380
      IF (FIRST)  THEN                                                 ELE00390
          IF (HEAD) WRITE (6,5) TITLE(1),TITLE(2)                      ELE00400
          WRITE (6,6)                                                  ELE00410
          FIRST=.FALSE.                                                ELE00420
      ENDIF                                                            ELE00430
C                                                                      ELE00440
C------- PROCESS LOAD GROUPS AND READ ADDITIONAL INFO...               ELE00450
      IF (INDEX(DIR,'CONC').NE.0) THEN                                 ELE00460
         KODE=100                                                      ELE00470
```

```
            J=2                                                                 ELE00490
            BACKSPACE(5)                                                        ELE00500
            READ (5,*) (IELDK(I,N),I=1,4),TYPE,DIRI,(ELDK(I,N),I=1,J)           ELE00510
            DIR=TYPE//DIRI                                                      ELE00520
            CALL CKRNG(ELDK(2,N),0.,1.,'DISTANCE')                              ELE00530
          ELSE IF (INDEX(DIR,'UNIF').NE.0) THEN                                 ELE00540
            J=1                                                                 ELE00550
            KODE=200                                                            ELE00560
          ELSE IF (INDEX(DIR,'VARR').NE.0) THEN                                 ELE00570
            KODE=300                                                            ELE00580
            J=4                                                                 ELE00590
            BACKSPACE(5)                                                        ELE00600
            READ (5,*) (IELDK(I,N),I=1,4),TYPE,DIRI,(ELDK(I,N),I=1,J)           ELE00610
            DIR=TYPE//DIRI                                                      ELE00620
            CALL CKRNG(ELDK(3,N),0.      ,ELDK(4,N),'DISTANCE')                 ELE00630
            CALL CKRNG(ELDK(4,N),ELDK(3,N),1.      ,'DISTANCE')                 ELE00640
          ELSE IF (INDEX(DIR,'FEM').NE.0) THEN                                  ELE00650
            KODE=400                                                            ELE00660
            J=12                                                                ELE00670
            BACKSPACE(5)                                                        ELE00680
            READ (5,*) (IELDK(I,N),I=1,4),TYPE,DIRI,(ELDK(I,N),I=1,J)           ELE00690
            DIR=TYPE//DIRI                                                      ELE00700
          ELSE                                                                  ELE00710
            WRITE(6,30) 'INVALID LOAD TYPE',                                    ELE00720
     &                  (IELDK(I,N),I=1,4),DIR,ELDK(1,N)                        ELE00730
            GO TO 20                                                            ELE00740
          ENDIF                                                                 ELE00750
          IF (INDEX(DIR,'FX').NE.0) THEN                                        ELE00760
            KODE=KODE+1                                                         ELE00770
          ELSE IF (INDEX(DIR,'FY').NE.0) THEN                                   ELE00780
            KODE=KODE+2                                                         ELE00790
          ELSE IF (INDEX(DIR,'FZ').NE.0) THEN                                   ELE00800
            KODE=KODE+3                                                         ELE00810
          ELSE IF (INDEX(DIR,'MX').NE.0) THEN                                   ELE00820
            KODE=KODE+4                                                         ELE00830
          ELSE IF (INDEX(DIR,'MY').NE.0) THEN                                   ELE00840
            KODE=KODE+5                                                         ELE00850
          ELSE IF (INDEX(DIR,'MZ').NE.0) THEN                                   ELE00860
            KODE=KODE+6                                                         ELE00870
          ELSE IF (KODE.NE.400) THEN                                            ELE00880
            WRITE(6,30) 'INVALID LOAD DIRECTION',                               ELE00890
     &                  (IELDK(I,N),I=1,4),DIR,(ELDK(1,N),I=1,J)                ELE00900
            GO TO 20                                                            ELE00910
          ENDIF                                                                 ELE00920
          IELDK(5,N)=KODE                                                       ELE00930
          WRITE(6,40) (IELDK(I,N),I=1,4),DIR,(ELDK(1,N),I=1,J)                  ELE00940
          GO TO 10                                                              ELE00950
C                                                                               ELE00960
   30 FORMAT (' ',A, ' --- LOAD IS SKIPPED'/                                    ELE00970
     &        1X,4I5,1X,A20,1P,6G14.6,:/(9X,6G14.6) )                          ELE00980
   40 FORMAT (1X,4I5,1X,A20,1P,6G14.6,:/(9X,6G14.6) )                           ELE00990
C                                                                               ELE01000
C------- RETURN IF NO ELEMENT LOADS WERE INPUT                                  ELE01010
  100 IF (FIRST) RETURN                                                         ELE01020
C                                                                               ELE01030
C------- LOOP TO GENERATE MEMBER LOADS FOR EACH ELEMENT                         ELE01040
      IOPT=0                                                                    ELE01050
      DO 101 IELNO=1,NELMT                                                      ELE01060
        FALSE=.FALSE.                                                           ELE01070
  101   CALL ELELIB                                                             ELE01080
     &  (IOPT  ,LSTTYP ,FALSE ,IREL,FALSE, NAME ,EESE,EPSE,DAMAGE,             ELE01090
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IDISP,ILOAD,MSTOR)             ELE01100
C                                                                               ELE01110
C------- PRINT LOADS AT EACH DOF                                                ELE01120
      IF (BUG) THEN                                                             ELE01130
        N5=NDOF+5                                                               ELE01140
        DO 120 L=1,NLOAD                                                        ELE01150
          IF (HEAD) WRITE (6,5) TITLE(1),TITLE(2)                               ELE01160
          WRITE (6,130) L                                                       ELE01170
          DO 120 I=1,N5,5                                                       ELE01180
          I5=MIN(I+4,NDOF)                                                      ELE01190
          DO 110 J=I,I5                                                         ELE01200
            K=J-I+1                                                             ELE01210
            CK(K)=' '                                                           ELE01220
  110       IF (J.GT.NF) CK(K)='*'                                            ELE01230
            WRITE (6,140) (J,FORCE(J,L),CK(J-I+1),J=I,I5)                       ELE01240
  120     CONTINUE                                                              ELE01250
      ENDIF                                                                     ELE01260
C                                                                               ELE01270
  130 FORMAT (' GLOBAL JOINT FORCES DUE TO ELEMENT LOADS ',5X,                  ELE01280
     &        '   LOADING #',I5,10X,                                           ELE01290
     &        ' NOTE: * DENOTES SUPPORT DISPLACEMENT'/                         ELE01300
     &        ' ==========================================='//                 ELE01310
     &        5X,5(' DOF',7X,'LOAD    ')/)                                    ELE01320
  140   FORMAT (5X,5(I5,1P,G14.6,A) )                                           ELE01330
C                                                                               ELE01340
      RETURN                                                                    ELE01350
      END                                                                       ELE01360
C=======================================================================        ELE00040
      SUBROUTINE ELE01                                                          ELE00050
     &  (IOPT  ,IUNIT  ,INC    ,FIRST  ,PRINT  ,                                ELE00060
     &   NDOF  ,NLOAD  ,MAXNOD ,NNODE  ,NCOS  ,                                ELE00070
     &   ID    ,IDOF   ,COORD  ,COSINE ,JTCOS  ,                               ELE00080
     &   CONST ,DISPL  ,FORCE  ,IREL   ,BUG   ,                                ELE00090
     &   IBUG  ,ACCEL  ,KGDATA ,GRAV   ,PGEOM ,                                ELE00100
     &   MAXELD,NELD   ,IELD   ,ELD    ,NAME  ,                                ELE00110
     &   STEPID,AXIAL  ,IELNO  ,RINPUT ,FUB   ,                                ELE00120
     &   EESE  , EPSE  ,DAMAGE ,       ,      ,                                ELE00130
     &   IELTYP,IELDOF ,PKG    ,H      ,Q     ,                                ELE00140
     &   AI    ,AJ     ,C      ,BI     ,BJ    ,                                ELE00150
     &   S22   ,S26    ,S212   ,S33    ,S35   ,                                ELE00160
     &   S311  ,R      ,RT     ,F      ,FT    ,                                ELE00170
     &   LM    ,CTS    ,CTE    ,BS     ,BE    ,                                ELE00180
     &   D     ,NODEI  ,NODEJ  ,JOINTI ,JOINTJ ,                               ELE00190
     &   REL   ,LENGTH ,KGDOF  ,LKG    ,LMKG  ,                                ELE00200
     &   ISE   ,ENGDAT ,FMAX   ,MAT    ,SE,                                    ELE00210
     &   STIFF ,                                                                ELE00220
     &   FEM   )                                                                ELE00230
C                                                                               ELE00240
      IMPLICIT REAL(A-H,O-Z)                                                    ELE00250
      LOGICAL ERR,FIRST,PRINT,BUG,FEMFLG,FALSE,AXIAL,BTEST                      ELE00260
      CHARACTER*80 NAME                                                         ELE00270
      CHARACTER*(*) STEPID                                                      ELE00280
      CHARACTER*6  IRELC                                                        ELE00290
      DIMENSION KGDATA(5)                                                       ELE00300
      DIMENSION  IDOF(MAXNOD,6),COORD(MAXNOD,6),COSINE(5,3,NCOS)               ELE00310
      DIMENSION CONST(MAXNOD,6),   ID(MAXNOD)  ,JTCOS(MAXNOD)                  ELE00320
      DIMENSION ASAT(12,12),SAT(12,12),STIFF(156)                               ELE00330
      DIMENSION A(12,12),  S(12,12),LMK(12),LM1(12),LMKG(12),FMAX(12)           ELE00340
      DIMENSION CTS( 3, 3),CTE( 3, 3),COSLOC(3,3),VX(3),VV(3),VZ(3)             ELE00350
      DIMENSION BS( 5, 5),BE( 3, 3),SE(15),ENGDAT(2)                            ELE00360
      DIMENSION R(12),DISPL(NDOF,NLOAD),F(12),RT(12),FT(12),FUB(NDOF)           ELE00370
      DIMENSION FORCE(NDOF,NLOAD),IELD(5,MAXELD),ELD(12,MAXELD)                 ELE00380
      DIMENSION ASAV(12,12),RINPUT(100),FEMK(12,NLOAD),GFEM(12)                 ELE00390
      DIMENSION DUCT(3),EXCR(6)                                                 ELE00400
      DIMENSION VC(12),VL(12)                                                   ELE00410
      REAL LENGTH                                                               ELE00420
      REAL IX,IV,IZ,KII2,KJJZ,KIJZ,KIIV,KJJV,KIJV,KTI,TVJ                      ELE00430
      INTEGER REL,RELT                                                          ELE00440
      DIMENSION MOK(15)                                                         ELE00450
      DATA MO/1,2,4,7,11,16,22,29,37,46,56,67,79/                               ELE00460
C                                                                               ELE00470
C-----------------------------------------------------------------------        ELE00480
C VARIABLES:                                                                    ELE00490
C-----------------------------------------------------------------------        ELE00500
C---------- GLOBAL VARIABLES                                                    ELE00510
C IOPT   = 1, INITIALIZE BEAM COLUMN ELEMENT                                    ELE00520
C        = 2, CALCULATE GEOMETRIC STIFFNESS                                     ELE00530
C        = 3, FORM STIFFNESS                                                    ELE00540
C        = 4, CALCULATE INCREMENTAL FORCES                                      ELE00550
C        = 5, CALCULATE TOTAL FORCES AND ENERGIES                               ELE00560
C IUNIT  = OUTPUT FILE UNIT # FOR PRINTING OUTPUT                               ELE00570
C ERR    = ERR OR FLAG. IF AN ERR OR HAS OCCURED, ERR=.TRUE.                    ELE00580
C FIRST  = FLAG. FIRST=.TRUE., PRINT HEADERS                                    ELE00590
C PRINT  = FLAG. PRINT=.TRUE., PRINT DATA                                       ELE00600
C NDOF   = # OF GLOBAL DOF                                                      ELE00610
C NLOAD  = # OF LOAD COMBINATIONS                                               ELE00620
C MAXNOD = # ROW DIMENSION OF NODE ARRAY                                        ELE00630
C NNODE  = # OF NODES                                                           ELE00640
C ID     = ARRAY OF EXTERNAL NODE NUMBERS                                       ELE00650
C IDOF   = ARRAY OF DEGREES OF FREEDOM                                          ELE00660
C COORD  = ARRAY OF COORDINATES                                                 ELE00670
C COSINE = ARRAY OF DIRECTION COSINES OF NODES                                  ELE00680
C CONST  = ARRAY OF CONSTRAINT TRANSFORMATIONS                                  ELE00690
C DISPL  = GLOBAL DISPLACEMENT MATRIX                                           ELE00700
C IZMAT  = LOCATION OF MATERIAL DATA IN GLOBAL STORAGE VECTOR                   ELE00710
C NAME   = ELEMENT NAME                                                         ELE00720
C IELNO  = ELEMENT NUMBER                                                       ELE00730
C RINPUT = INPUT DATA                                                           ELE00740
C STIFF  = OUTPUT ELEMENT STIFFNESS (UPPER TRIANGULAR FORM)                     ELE00750
C LM     = OUTPUT LOCAL TO GLOBAL STIFFNESS MAPPING MATRIX                      ELE00760
C---------- ELEMENT VARIABLES                                                   ELE00770
C H,Q,AI,AJ,C,BI,BJ,S22,S26,S212,S33,S35,S311,D                                 ELE00780
C        = INDIVIDUAL TERMS IN THE ELEMENT STIFFNESS (S) MATRIX                 ELE00790
C PKG    = INPUT LOAD FOR FORMING GEOMETRIC STIFFNESS MATRIX...                 ELE00800
C R      = INCRIMENTAL DISPLACEMENTS                                            ELE00810
C RT     = TOTAL DISPLACEMENTS                                                  ELE00820
C F      = INCRIMENTAL FORCES                                                   ELE00830
C FT     = TOTAL FORCES                                                         ELE00840
C LM     = LOCAL TO GLOBAL STIFFNESS MAPPING MATRIX                             ELE00850
C CTS    = LOCAL TO GLOBAL ROTATION MATRIX START                                ELE00860
C CTE    = LOCAL TO GLOBAL ROTATION MATRIX END                                  ELE00870
C BS     = LOCAL TO GLOBAL ROTATED ECCENTRICITY/CONSTRAINT  MATRIX START        ELE00880
C BE     = LOCAL TO GLOBAL ROTATED ECCENTRICITY/CONSTRAINT  MATRIX END          ELE00890
C STIFF  = ELEMENT STIFFNESS                                                    ELE00900
C IELTYP = ELEMENT TYPE = 10                                                    ELE00910
C NODEI  = EXTERNAL JOINT NUMBER, END I                                         ELE00920
C NODEJ  = EXTERNAL JOINT NUMBER, END J                                         ELE00930
C JOINTI = INTERNAL JOINT NUMBER, END I                                         ELE00940
C JOINTJ = INTERNAL JOINT NUMBER, END J                                         ELE00950
C---------- TEMPORARY VARIABLES                                                 ELE00960
C VX     = VECTOR DEFINING THE ELEMENT X AXIS                                   ELE00970
C VV     = VECTOR DEFINING THE ELEMENT Y AXIS                                   ELE00980
C COSLOC = LOCAL COSINE MATRIX                                                  ELE00990
C A      = GLOBAL TO LOCAL FORCE TRANSFORMATION MATRIX                          ELE01000
C S      = ELEMENT STIFFNESS AT LOCAL COORD, AT ENDS OF FLEXABLE PART           ELE01010
C SAT    = PRODUCT OF S*A                                                       ELE01020
C ASAT   = PRODUCT OF A*SAT                                                     ELE01030
C-----------------------------------------------------------------------        ELE01040
C                                                                               ELE01050
C------- CHOOSE OPTION                                                          ELE01060
        IF (IOPT.LT.1 .OR.IOPT.GT.14)                                           ELE01070
     &  WRITE (6,*) 'INVALID OPTION IN ELE-01, IOPT=',IOPT                      ELE01080
C                                                                               ELE01090
      GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,                ELE01100
     &                                           1300,1400),IOPT                ELE01110
C                                                                               ELE01120
  100 CONTINUE                                                                  ELE01130
C                                                                               ELE01140
C------ ZERO STRAIN ENERGY                                                     ELE01150
      ENGDAT(1) = 0                                                             ELE01160
      ENGDAT(2) = 0                                                             ELE01170
C                                                                               ELE01180
      IF ( BTEST(IBUG,7) ) THEN                                                 ELE01190
        WRITE (6,*) NAME                                                        ELE01200
        CALL HMATRX(RINPUT,10, 1,'RINPUT')                                      ELE01210
      ENDIF                                                                     ELE01220
C                                                                               ELE01230
C------ INTERPERTATE INPUT DATA                                                ELE01240
      IELTYP= 1                                                                 ELE01250
      MAT   = RINPUT( 1)                                                        ELE01260
      NODEI = RINPUT( 2)                                                        ELE01270
      NODEJ = RINPUT( 3)                                                        ELE01280
      VX(1) = RINPUT( 4)                                                        ELE01290
      VX(2) = RINPUT( 5)                                                        ELE01300
      VX(3) = RINPUT( 6)                                                        ELE01310
      XS    = RINPUT( 7)                                                        ELE01320
      XE    = RINPUT( 8)                                                        ELE01330
      PKG   = RINPUT( 9)                                                        ELE01340
      RELT  = RINPUT(10)                                                        ELE01350
C                                                                               ELE01360
C------ GET INTERNAL NODE # ASSOCIATED WITH EXTERNAL NODE #'S                  ELE01370
      JOINTI=IQUICK(NODEI,ID,NNODE)                                             ELE01380
      JOINTJ=IQUICK(NODEJ,ID,NNODE)                                             ELE01390
C                                                                               ELE01410
C------ PRINT DATA FOR PLOTTING                                                ELE01420
      IF (BTEST(IBUG,2)) THEN                                                   ELE01430
        WRITE(7,10) 0,(COORD(JOINTJ,I),I=1,3),(COORD(JOINTI,J),J=1,3)           ELE01440
   10   FORMAT (I5,1P,6G12.4)                                                   ELE01450
      ENDIF                                                                     ELE01460
C                                                                               ELE01470
C-----------------------------------------------------------------------        ELE01480
C  GET GLOBAL TO LOCAL TRANSFORMATION MATRIX                                    ELE01490
C-----------------------------------------------------------------------        ELE01500
C------ DEFINE VECTOR X, LENGTH                                                ELE01510
      VX(1)=COORD(JOINTJ,1)-COORD(JOINTI,1)                                     ELE01520
      VX(2)=COORD(JOINTJ,2)-COORD(JOINTI,2)                                     ELE01530
      VX(3)=COORD(JOINTJ,3)-COORD(JOINTI,3)                                     ELE01540
      LENGTH=SQRT(VX(1)**2+VX(2)**2+VX(3)**2)-XS-XE                             ELE01550
      IF (LENGTH.LE.0) THEN                                                     ELE01560
        WRITE(6,101) IELNO,JOINTI,JOINTJ,LENGTH                                 ELE01570
  101   FORMAT(1X,20('*'),'ER','ROR - LENGTH IS LE 0',/                         ELE01580
     &  5X,'IELNO=',I5,5X,'JOINTI=',I5,5X,'JOINTJ=',I5,                        ELE01590
     &  5X,'LENGTH',1P,G15.6/                                                   ELE01600
     &  5X,'REVISE INPUT.....................'/)                               ELE01610
        LENGTH=1.0                                                              ELE01620
      ENDIF                                                                     ELE01630
      IF ( BTEST(IBUG,7) ) THEN                                                 ELE01640
```

```
        WRITE (6,*) 'JOINTS',JOINTI,JOINTJ
        WRITE (6,*) 'VX',VX
        WRITE (6,*) 'VV',VV
        WRITE (6,*) 'LENGTH  ',LENGTH
      ENDIF
C----- GET LOCAL TO GLOBAL TRANSFORMATION MATRICIES CT AND B
      ICS=JTCOSC JOINTI)
      ICE=JTCOSC JOINTJ)
      CALL ROTXYZC VX,VV,COSINE(1,1,ICS),CTS, XS ,0.,0.,BS,
     &      CONST(JOINTI,1),MAXNOD)
      CALL ROTXYZC VX,VV,COSINE(1,1,ICE),CTE,-XE ,0.,0.,BE,
     &      CONST(JOINTJ,1),MAXNOD)
      IF ( BTEST(IBUG,7) ) THEN
          CALL MMATRXC CTS   , 3, 3,'CTS   ')
          CALL MMATRXC CTE   , 3, 3,'CTE   ')
      ENDIF
C
C----- GET GLOBAL CONSTRAINT MATRIX BS, BE
C
C
      IF ( BTEST(IBUG,7) ) THEN
         CALL MMATRXC BS, 3, 3,'BS   ')
         CALL MMATRXC BE, 3, 3,'BE   ')
      ENDIF
C
C----- ZERO MATRICIES
      DO 40 I=1,12
        FMAXC I)=0
        F (I)=0
        FT(I)=0
        R (I)=0
        RT(I)=0
        DO 40 J=1,12
          S(I,J)=0.
   40   A(I,J)=0.
C
C----- ASSEMBLE TRANSFORMATION MATRIX A
       DO 50 I=1,3
       DO 50 J=1,3
         A(I  ,J  )=CTS(I,J)
         A(I+3,J+3)=CTS(I,J)
         A(I+6,J+6)=CTE(I,J)
         A(I+9,J+9)=CTE(I,J)
         A(I+3,J  )=BS(I,J)
   50    A(I+9,J+6)=BE(I,J)
      IF ( BTEST(IBUG,7) ) CALL MMATRXC A    ,12,12,' A    ')
C
C----- GET MATERIAL PROPERTIES
       FALSE=.FALSE.
       LSTTYP=0
       CALL MATLIB
     & ( 2    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE,DUCT,EXCR,
     &   FT   ,F      ,RT    ,R     ,V     ,VL    ,LELEM ,LMAT  ,
     &   MAT  ,MATTYP,SE,IELNO,DAMAGE)
C
      IF (MATTYP.NE.1) THEN
         WRITE (6,190) IELNO,MATTYP,NODEI,NODEJ
  190    FORMAT (' ELEMENT: ',I6,'  INCOMPATIBLE MATERIAL PROP.#',I5,
     &    '  START JOINT:',I5,'  END JOINT:',I5,/
     &    ' THE ELEMENT IS REJECTED AND ER','ROR CHECKING CONTINUES')
         ERR=.TRUE.
         GO TO 191
      ENDIF
C
      EAX   =SE( 1)
      GAV   =SE( 2)
      GAZ   =SE( 3)
      GIX   =SE( 4)
      EIV   =SE( 5)
      EIZ   =SE( 6)
C
C-----------------------------------------------------------------
C  DETERMINE THE MEMBER RELEASE CODES
C-----------------------------------------------------------------
C     RELT  IS AN INPUT MEMBER RELEASE CODE
C     REL   IS THE INTERNAL MEMBER RELEASE CODE
C           BIT 6 AXIAL -          TRUE = AXIAL DEFORMATION CONSIDERED
C           BIT 5 MV AT END I  |
C           BIT 4 MV AT END I  |----> TRUE=RELEASE
C           BIT 3 MZ AT END I  |
C           BIT 2 MV AT END J  |
C           BIT 1 MV AT END J  |
C           BIT 0 MZ AT END J  |
C
      IF (BTEST(IBUG,7)) WRITE (6,*) 'RELT',RELT
      REL=0
      IRELC=' '
      IF (MOD(RELT,10).NE.0) THEN
          REL=IBSET(REL,0)
          IRELC(6:6)='Z'
      ENDIF
      RELT=RELT/10
      IF (MOD(RELT,10).NE.0) THEN
          REL=IBSET(REL,1)
          IRELC(5:5)='V'
      ENDIF
      RELT=RELT/10
      IF (MOD(RELT,10).NE.0) THEN
          REL=IBSET(REL,2)
          IRELC(4:4)='X'
      ENDIF
      RELT=RELT/10
      IF (MOD(RELT,10).NE.0) THEN
          REL=IBSET(REL,3)
          IRELC(3:3)='Z'
      ENDIF
      RELT=RELT/10
      IF (MOD(RELT,10).NE.0) THEN
          REL=IBSET(REL,4)
          IRELC(2:2)='V'
      ENDIF
      RELT=RELT/10
      IF (MOD(RELT,10).NE.0) THEN
          REL=IBSET(REL,5)
          IRELC(1:1)='X'
      ENDIF
C--------- SET STIFFNESS COIEFFICIENTS...
C--------- SET CONSISTENT MASS GEOMETRIC STIFFNESS COIEFFICIENTS...
      AZ=0
      BZ=0
      CZ=0
      DZ=0
      EZ=0
      FZ=1.0 / LENGTH
C..... BOTH ENDS RELEASED, Z-AXIS .....................
      IF (BTEST(REL,0) .AND. BTEST(REL,5)) THEN
        KIIZ=0
        KJJZ=0
        KIJZ=0
C..... START END RELEASED, Z-AXIS .....................
      ELSE IF (BTEST(REL,5)) THEN
        KIIZ=0
        KJJZ=5
        KIJZ=0
        BZ=  0.20 * LENGTH
        EZ=  0.20
        FZ=  1.20 / LENGTH
C..... END END RELEASED, Z-AXIS .......................
      ELSE IF (BTEST(REL,0)) THEN
        KIIZ=5
        KJJZ=0
        KIJZ=0
        AZ=  0.20 * LENGTH
        DZ=  0.20
        FZ=  1.20 / LENGTH
C..... BOTH ENDS RELEASED, Z-AXIS .....................
      ELSE
        KIIZ=4
        KJJZ=4
        KIJZ=2
        AZ=  2 * LENGTH / 15.
        BZ=  AZ
        CZ=  LENGTH / 30.
        DZ=  0.10
        EZ=  0.10
        FZ=  1.2 / LENGTH
      ENDIF
C
      AV=0
      BV=0
      CV=0
      DV=0
      EV=0
      FV=1.0 / LENGTH
C..... BOTH ENDS RELEASED, Y-AXIS .....................
      IF (BTEST(REL,1) .AND. BTEST(REL,4)) THEN
        KIIV=0
        KJJV=0
        KIJV=0
C..... START END RELEASED, Y-AXIS .....................
      ELSE IF (BTEST(REL,4)) THEN
        KIIV=0
        KJJV=5
        KIJV=0
        BV=  0.20 * LENGTH
        EV=  0.20
        FV=  1.20 / LENGTH
C..... END END RELEASED, Y-AXIS .......................
      ELSE IF (BTEST(REL,1)) THEN
        KIIV=5
        KJJV=0
        KIJV=0
        AV=  0.20 * LENGTH
        DV=  0.20
        FV=  1.20 / LENGTH
C..... BOTH ENDS RELEASED, Z-AXIS .....................
      ELSE
        KIIV=4
        KJJV=4
        KIJV=2
        AV=  2 * LENGTH / 15.
        BV=  AV
        CV=  LENGTH / 30.
        DV=  0.10
        EV=  0.10
        FV=  1.2 / LENGTH
      ENDIF
C
      IF (BTEST(REL,2) .AND. BTEST(REL,5)) THEN
        KTI  =0
        KTJ  =0
      ELSE IF (BTEST(REL,5)) THEN
        KTI  =0
        KTJ  =1
      ELSE IF (BTEST(REL,2)) THEN
        KTI  =1
        KTJ  =0
      ELSE
        KTI  =1
        KTJ  =1
      ENDIF
C
      IF ( BTEST(IBUG,7) ) WRITE (6,*) 'IRELC',IRELC,'REL',REL
C
C-----------------------------------------------------------------
C  ASSEMBLE LOCAL STIFFNESS MATRIX
C-----------------------------------------------------------------
      H=EAX/LENGTH
      Q=GIX/LENGTH
      AI=KIIZ*(EIZ/LENGTH)
      AJ=KJJZ*(EIZ/LENGTH)
      C =KIJZ*(EIZ/LENGTH)
      BI=KIIV*(EIV/LENGTH)
      BJ=KJJV*(EIV/LENGTH)
      D =KIJV*(EIV/LENGTH)
      S22 = (AI+2*C+AJ)/(LENGTH**2)
      S26 = (AI+  C   )/(LENGTH   )
      S212= (     C+AJ)/(LENGTH   )
      S33 = (BI+2*D+BJ)/(LENGTH**2)
      S35 = (BI+  D   )/(LENGTH   )
      S311= (     D+BJ)/(LENGTH   )
      SK( 1, 1) = H
      SK( 1, 7) =-H
      SK( 2, 2) = S22
      SK( 2, 6) = S26
      SK( 2, 8) =-S22
      SK( 2,12) = S212
      SK( 3, 3) = S33
      SK( 3, 5) = S35
      SK( 3, 9) =-S35
      SK( 5,11) = S311
      SK( 4, 4) = Q * KTI
      SK( 4,10) =-Q
      SK( 5, 5) = BI
      SK( 5, 9) = S35
      SK( 5,11) = D
      SK( 6, 6) = AI
```

```
      S( 6, 8) =-SZ6
      S( 6,12) = C
      S( 7, 7) = M
      S( 8, 8) = S22
      S( 8,12) =-S212
      S( 9, 9) = S55
      S( 9,11) = S311
      S(10,10) = Q * KTJ
      S(11,11) = BJ
      S(12,12) = AJ
      S( 5, 3) = S( 3, 5)
      S( 6, 2) = S( 2, 6)
      S( 7, 1) = S( 1, 7)
      S( 8, 2) = S( 2, 8)
      S( 8, 6) = S( 6, 8)
      S( 9, 3) = S( 3, 9)
      S( 9, 5) = S( 5, 9)
      S(10, 4) = S( 4,10)
      S(11, 3) = S( 3,11)
      S(11, 5) = S( 5,11)
      S(11, 9) = S( 9,11)
      S(12, 2) = S( 2,12)
      S(12, 6) = S( 6,12)
      S(12, 8) = S( 8,12)
C
C----- CALCULATE SAT
      DO 60 I=1,12
        DO 60 J=1,12
        SAT(I,J)=0
          DO 60 K=1,12
   60       SAT(I,J)=SAT(I,J)+S(I,K)*A(J,K)
C
C----- CALCULATE ASAT
      DO 70 I=1,12
        DO 70 J=1,12
        ASAT(I,J)=0
          DO 70 K=1,12
   70       ASAT(I,J)=ASAT(I,J)+A(I,K)*SAT(K,J)
      IF ( BTEST(IBUG,8) ) CALL MMATRX(ASAT  ,12,12,' ASAT ')
C
C----- DEGREES OF FREEDOM
C     REMOVE DOF'S THAT ARE CONSTRAINED TO THE SAME DOF...
      DO 75 I=1,6
        IF ( IDOF(JOINTI,I).NE.IDOF(JOINTJ,I) ) THEN
          LM(I  )=IDOF(JOINTI,I)
          LM(I+6)=IDOF(JOINTJ,I)
        ELSE
          LM(I  )= 0
          LM(I+6)= 0
        ENDIF
   75   CONTINUE
C
C----- CONVERT TO HALF STORAGE MODE  BY COLUMNS
C
C          1  3  6 10 15 21 28 36 45 55 66 78  NUMBER INDICATES
C             2  5  9 14 20 27 35 44 54 65 77  LOCATION OF DATA
C                4  8 13 19 26 34 43 53 64 76  IN ARRAY STIFF
C                   7 12 18 25 33 42 52 63 75
C                     11 17 24 32 41 51 62 74
C                        16 23 31 40 50 61 73
C                           22 30 39 49 60 72
C                              29 38 48 59 71
C                                 37 47 58 70
C                                    46 57 69
C                                       56 68
C                                          67
C
      L=0
      IELDOF=0
      DO 90 I=1,12
        IF (LM(I).EQ.0 .OR. ASAT(I,I).LE.0) GO TO 90
        IELDOF=IELDOF+1
        LM(IELDOF)=LM(I)
        DO 80 J=I,1,-1
          IF (LM(J).EQ.0 .OR. ASAT(J,J).LE.0 ) GO TO 80
          L=L+1
          STIFF(L)=ASAT(I,J)
   80     CONTINUE
   90   CONTINUE
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'LM=',LM
        WRITE (6,*) 'IELDOF',IELDOF,'LM =',(LM(I),I=1,IELDOF)
        CALL LMATRX(STIFF,NO,IELDOF,(L+1),'ELEMENT STIFFNESS')
        WRITE (6,*) 'KGDATA:',KGDATA,' PKG:',PKG
      ENDIF
C-----------------------------------------------------------------
C  ASSEMBLE GEOMETRIC STIFFNESS MATRIX FOR A UNIT LOAD...
C-----------------------------------------------------------------
      LKG=L
      IF ( PKG.EQ.0 .AND. KGDATA(1).NE.2 ) THEN
        AXIAL=.FALSE.
        REL=IBCLR(REL,6)
        GO TO 180
      ENDIF
C
C---------- DETERMINE IF AXIAL DEFORMATION IS CONSIDERED
C           BIT #6 OF REL IS TRUE IF AXIAL DEFORMATION IS CONSIDERED
      DO 999 I=1,5
        CS=CTS(I,1)
        CE=CTE(I,1)
        IF (ABS(CS).LE. .0001 .AND. ABS(CE).LE. .0001 ) GO TO 999
        IS=IDOF(JOINTI,I)
        IE=IDOF(JOINTJ,I)
        IF (IS.EQ.IE) GO TO 999
          REL=IBSET(REL,6)
  999   CONTINUE
C
        AXIAL=BTEST(REL,6)
      IF ( BTEST(IBUG,7) ) WRITE (6,*) 'AXIAL:',AXIAL
C
      IF (KGDATA(1).NE.0 .AND. KGDATA(2).NE.0 .AND. BTEST(REL,6)) THEN
C
C---------- DETERMINE THE AXIAL LOAD TO BE USED FOR KG...
        IF (KGDATA(1).EQ.1) THEN
            PGEOM =  PKG
        ELSE IF (KGDATA(1).EQ.2) THEN
            PGEOM =  0.50*(F(1)-F(7))
        ENDIF
C
C--ASSEMBLE LOCAL GEOMETRIC STIFFNESS MATRIX FOR A UNIT LOAD...
C         ZERO LOCAL KG MATRIX
          DO 998 I=1,12
          DO 998 J=1,12
  998       S(I,J)=0
C
C---------- LUMPED MASS FORM OF GEOMETRIC STIFFNESS
        IF (KGDATA(2).EQ.1) THEN
          F1=1/LENGTH
            S( 2, 2)=  F1
            S( 2, 8)= -F1
            S( 8, 2)= -F1
            S( 8, 8)=  F1
            S( 5, 5)=  F1
            S( 5, 9)= -F1
            S( 9, 5)= -F1
            S( 9, 9)=  F1
C
C---------- CONSISTENT MASS FORM OF GEOMETRIC STIFFNESS
        ELSE IF (KGDATA(2).EQ.2) THEN
            S( 2, 2)=  FZ
            S( 2, 6)=  QZ
            S( 2, 8)= -FZ
            S( 2,12)=  EZ
            S( 6, 2)=  QZ
            S( 6, 6)=  AZ
            S( 6, 8)= -QZ
            S( 6,12)=  CZ
            S( 8, 2)= -FZ
            S( 8, 6)= -QZ
            S( 8, 8)=  FZ
            S( 8,12)= -EZ
            S(12, 2)=  EZ
            S(12, 6)= -CZ
            S(12, 8)= -EZ
            S(12,12)=  BZ
C
            S( 5, 5)=  FV
            S( 5, 5)= -QV
            S( 5, 9)= -FV
            S( 5,11)= -EV
            S( 5, 5)= -QV
            S( 5, 5)=  AV
            S( 5, 9)=  QV
            S( 5,11)= -CV
            S( 9, 3)= -FV
            S( 9, 5)=  QV
            S( 9, 9)=  FV
            S( 9,11)=  EV
            S(11, 3)= -EV
            S(11, 5)= -CV
            S(11, 9)=  EV
            S(11,11)=  BV
        ENDIF
      IF (BTEST(IBUG,8)) CALL MMATRX( S    ,12,12,' KG -LOCAL COORD ')
C
C----- CALCULATE SAT
        DO 1060 I=1,12
          DO 1060 J=1,12
          SAT(I,J)=0
            DO 1060 K=1,12
 1060         SAT(I,J)=SAT(I,J)+S(I,K)*A(J,K)
C
C----- CALCULATE ASAT
        DO 1070 I=1,12
          DO 1070 J=1,12
          ASAT(I,J)=0
            DO 1070 K=1,12
 1070         ASAT(I,J)=ASAT(I,J)+A(I,K)*SAT(K,J)
        IF (BTEST(IBUG,8))  CALL MMATRX(ASAT  ,12,12,' A*KG*AT ')
C
C----- CONVERT TO HALF STORAGE MODE  BY COLUMNS
        KGDOF=0
        DO 1090 I=1,12
          IF (LM(I).EQ.0 .OR. ASAT(I,I).LE.0) GO TO 1090
          KGDOF=KGDOF+1
          LMKG(KGDOF)=LM(I)
          DO 1080 J=I,1,-1
            IF (LM(J).EQ.0 .OR. ASAT(J,J).LE.0 ) GO TO 1080
            L=L+1
            STIFF(L)=ASAT(I,J)
 1080       CONTINUE
 1090     CONTINUE
      IF ( BTEST(IBUG,7) ) THEN
          WRITE (6,*) 'KGDOF',KGDOF,'LMKG =',(LMKG(I),I=1,KGDOF)
          CALL LMATRX(STIFF(LKG+1),NO,KGDOF,(L-LKG),
     &       'ELEMENT GEOMETRIC STIFFNESS (UNIT LOAD)')
        ENDIF
      ENDIF
C
C----- RELEASE UNUSED STORAGE
  180 IREL=15-L
C
C     PRINT COLUMN DATA
      IF (FIRST .OR. BTEST(IBUG,7) .OR. BTEST(IBUG,8) ) WRITE (6,192)
  191 WRITE(6,195) NAME,IELNO,MAT,NODE1,NODE2,IREL,C,LENGTH,
     &     VX(1),VX(2),VX(3),XS,-XE,PKG
C
  192 FORMAT(/' ELEMENT 08.  3D BEAM ELEMENT'//
     &   T22,'= MATL START   END' ,' REL CD',5X,'LENGTH',5X,
     &   11('-'),' Y-AXIS ',
     &   12('-'),' START OIST    END OIST     PKG')
  195 FORMAT(1X,A15,4I6.2X,A,1P,G12.4,0P,
     &       F9.5,' I',SP,F9.5,' J',F9.5,' K',1P,5S,5G12.4)
      RETURN
C==================== DETERMINE GEOMETRIC STIFF ====================
  200 CONTINUE
C- KG IS DETERMINED FOR A UNIT LOAD WITH IOPT=1.
C- THE ACTUAL LOAD (PGEOM) IS DETERMINED HERE.
C- KG IS MULTIPLIED BY PGEOM WHEN THE TOTAL STIFFNESS IS ASSEMBLED
C- NEGATIVE PGEOM IS COMPRESSION....
      AXIAL=BTEST(REL,6)
C
C---------- DETERMINE THE AXIAL LOAD TO BE USED FOR KG...
      IF (AXIAL) THEN
        IF (KGDATA(1).EQ.1) THEN
            PGEOM =  PKG
        ELSE IF (KGDATA(1).EQ.2) THEN
            PGEOM =  0.50*(F(1)-F(7))
        ENDIF
      ENDIF
      IF (BTEST(IBUG,7)) WRITE (6,*) 'AXIAL ',AXIAL,'  PGEOM:',PGEOM
C
      RETURN
C==================== DETERMINE STIFFNESS ====================
  500 CONTINUE
C
C     FORCE-FRAME DISPLACEMENT MATRIX
```

```
C     THE STIFFNESS IS STORED IN STIFF, THE MAPING MATRIX IN LM
C     THESE HAVE THE GLOBAL ADDRESSES IZKE AND IZLM...
C
      ,RETURN
C======================== DETERMINE INCRIMENTAL FORCES ==================
  400 CONTINUE
       IF (PRINT.AND.FIRST) WRITE (6,491) STEPID
C
C------- CALL BMLOA TO DETERMINE THE FIXED END FORCES
      IF (MAXELD.GT.0) THEN
        CALL BMLOAC IELNO,FEMFLG,LENGTH,REL,MAXELD,NELD,NLOAD,
     &   FEM,IELD,ELD)
      ELSE
        FEMFLG=.FALSE.
      ENDIF
C
C------ LOOP FOR EACH LOAD
      DO 430 LOAD=1,NLOAD
C
C------- TRANSFORM FRAME DISPL INTO LOCAL DISPL
       DO 410 I=1,5
        RC I  )=0.
        RC I+5)=0.
        RC I+6)=0.
        RC I+9)=0.
        DO 410 J=1,5
           D1=DISPL(IDOF(JOINTI,J  ),LOAD)
           D2=DISPL(IDOF(JOINTI,J+5),LOAD)
           D3=DISPL(IDOF(JOINTJ,J  ),LOAD)
           D4=DISPL(IDOF(JOINTJ,J+5),LOAD)
        RC I  )=RC I  )+CTSC J,I)*D1 + BSC J,I)*D2
        RC I+5)=RC I+5)+CTSC J,I)*D2
        RC I+6)=RC I+6)+CTE C J,I)*D5 + BE C J,I)*D4
  410   RC I+9)=RC I+9)+CTE C J,I)*D4
C
C------ INCRIMENTAL FORCES
      FC 1)=   H*C RC 1)-RC 7))
      FC 2)= S22*C RC 2)-RC 8))+S26*RC 6)+S212*RC12)
      FC 3)= S33*C RC 3)-RC 9))-S35*RC 5)-S311*RC11)
      FC 4)=   Q*C RC 4)-RC10))
      FC 5)=-S35*C RC 3)-RC 9))+BI*RC 5)+D*RC11)
      FC 6)= S26*C RC 2)-RC 8))+AI*RC 6)+C*RC12)
C
      FC 7)=  -H*C RC 1)-RC 7))
      FC 8)=-S22*C RC 2)-RC 8))-S26*RC 6)-S212*RC12)
      FC 9)=-S33*C RC 3)-RC 9))+S35*RC 5)+S311*RC11)
      FC10)=  -Q*C RC 4)-RC10))
      FC11)=-S311*C RC 3)-RC 9))+ D*RC 5)+BJ*RC11)
      FC12)= S212*C RC 2)-RC 8))+ C*RC 6)+AJ*RC12)
C
C------ SUBTRACT OF FIXED END FORCES
      IF (FEMFLG) THEN
        DO 420 I=1,12
  420     F I)=F I)-FEM I,LOAD)
      ENDIF
C
C------- DETERMINE MAXIMUM VALUES FOR MULTIPLE LOAD CASES
      IF (NLOAD.GT.1) THEN
        DO 425 I=1,12
  425     IF ( ABS(F I)).GT.ABS(FMAX I)) )  FMAX I)=F I)
      ENDIF
C
C------- PRINT FORCES
      IF (BTEST(IBUG,0).AND.PRINT)
     &                      WRITE (6,495) IELNO,LOAD,NODEI,(RC J),J=1,6),
     &                                                NODEJ,(RC J),J=7,12)
      IF (          PRINT) WRITE (6,492) IELNO,LOAD,NODEI,(F(J),J=1,6),
     &                                                NODEJ,(F(J),J=7,12)
C
  430 CONTINUE
C
C------- CALCULATE MEMBER FORCE ON JOINT, AND ADD TO UNBALANCED FORCES
C     THIS IS USED TO CALC UNBALANCED FORCES.
       DO 440 I=1,3
        I1=IDOF(JOINTI,I  )
        I2=IDOF(JOINTI,I+5)
        J1=IDOF(JOINTJ,I  )
        J2=IDOF(JOINTJ,I+5)
        DO 440 J=1,5
          FUB(I1)=FUB(I1)+CTSC I,J)*(F(J  )+FT(J  ))
          FUB(I2)=FUB(I2)+ BSC I,J)*(F(J  )+FT(J  ))
          FUB(J1)=FUB(J1)+CTE C I,J)*(F(J+6)+FT(J+6))
          FUB(J2)=FUB(J2)+ BE C I,J)*(F(J+6)+FT(J+6))
  440     FUB(J2)=FUB(J2)+CTE C I,J)*(F(J+9)+FT(J+9))
C
  491 FORMAT (//'      3D BEAM FORCES...',5X,A/
     & '   ELEMENT LOAD        NODE',7X,'AXIAL',11X,'FV',15X,'FZ',
     & 11X,'TORSION',10X,'MV',15X,'MZ')
  492 FORMAT (/,1I0,I5,2X,'FORCE',I6,1P,6G15.6,
     &          /15X    ,2X,'FORCE',I6,   6G15.6)
  495 FORMAT (/,1I0,I5,2X,'DISPL',I6,1P,6G15.6,
     &          /15X    ,2X,'DISPL',I6,   6G15.6)
      RETURN
C======================== DETERMINE TOTAL FORCES ========================
  500 CONTINUE
       IF (PRINT.AND.FIRST) WRITE (6,491) STEPID
C
C------ TOTAL FORCES AND DISPLACEMENTS
      DO 510 I=1,12
           FT I)=FT I)+F(I)
           RT I)=RT I)+RC I)
  510      IF ( ABS(FT I)).GT.ABS(FMAX I)) )  FMAX I)=FT I)
  520 FORMAT ('    I:',I5,' R:',1P,G12.4,' RT:',G12.4,
     &                    ' F:',1P,G12.4,' FT:',G12.4)
      LOAD=1
      IF (BTEST(IBUG,0).AND.PRINT)
     &                      WRITE (6,495) IELNO,LOAD,NODEI,(RT(J),J=1,6),
     &                                                NODEJ,(RT(J),J=7,12)
      IF (          PRINT) WRITE (6,492) IELNO,LOAD,NODEI,(FT(J),J=1,6),
     &                                                NODEJ,(FT(J),J=7,12)
C
      RETURN
C
C======================== DETERMINE FIXED END FORCES, AND ADD TO FORCE=
  600 IF (MAXELD.LE.0) RETURN
C
C------- CALL BMLOA TO DETERMINE THE FIXED END FORCES
      CALL BMLOAC IELNO,FEMFLG,LENGTH,REL,MAXELD,NELD,NLOAD,
     &   FEM,IELD,ELD)
      FEMFLG=FEMFLG
      IF (.NOT. FEMFLG) RETURN
C
C------- ADD FIXED END FORCES TO LOAD MATRIX
      DO 620 LOAD=1,NLOAD
        DO 610 I=1,5
        GFEM I  )=0.
        GFEM I+5)=0.
        GFEM I+6)=0.
        GFEM I+9)=0.
        DO 610 J=1,5
           D1=FEM J  ,LOAD)
           D2=FEM J+5,LOAD)
           D3=FEM J+6,LOAD)
           D4=FEM J+9,LOAD)
        GFEM I  )=GFEM I  )+CTSC I,J)*D1
        GFEM I+5)=GFEM I+5)+ BSC I,J)*D1 + CTSC I,J)*D2
        GFEM I+6)=GFEM I+6)+CTE C I,J)*D3
  610   GFEM I+9)=GFEM I+9)+ BE C I,J)*D3 + CTE C I,J)*D4
      IF (BTEST(IBUG,7)) THEN
        WRITE(6,*) 'IELNO',IELNO
        CALL MMATRX( FEM  ,1  ,12,'LOCAL  FIXED END FORCES' )
        CALL MMATRX(GFEM  ,1  ,12,'GLOBAL FIXED END FORCES' )
      ENDIF
C
      DO 620 J=1,6
        JI=IDOF(JOINTI,J)
        JJ=IDOF(JOINTJ,J)
        FORCE JI,LOAD)=FORCE JI,LOAD)+GFEM J  )
  620   FORCE JJ,LOAD)=FORCE JJ,LOAD)+GFEM J+6)
      IF (BTEST(IBUG,7) ) THEN
        CALL MMATRX(FORCE ,NDOF,NLOAD,'GLOBAL LOAD MATRIX' )
      ENDIF
C
      RETURN
C
C====================== WRITE MEMBER FORCES TO OUTPUT FILE ==============
  700 CONTINUE
C
      WRITE (IUNIT,710)  NODEI,(FT(J),J=1,6),NODEJ,(FT(J),J=7,12)
  710 FORMAT (I6,1P,6G12.5)
C
      RETURN
C
C====================== READ AND PRINT MEMBER FORCES FROM OUTPUT FILE ====
  800 CONTINUE
      BACKSPACE(IUNIT)
      READ (IUNIT,*  ) ITYPE,IELNO,IWRITE,T0,DT
      WRITE (6,805)
      ISTEP=-IWRITE
  810 READ (IUNIT,815,END=850)
     &   NODEI,(FT(J),J=1,6),NODEJ,(FT(J),J=7,12)
  815 FORMAT (I6,6G12.5)
        ISTEP=ISTEP+IWRITE
        T=T0+DT*ISTEP
        IF (MOD(ISTEP,INC).EQ.0)
     &   WRITE (6,820) ISTEP,T,NODEI,(FT(J),J=1,6),
     &                         NODEJ,(FT(J),J=7,12)
        GO TO 810
C
  805 FORMAT (//'      3D BEAM FORCES...',/
     &  '   STEP      TIME          NODE',7X,'AXIAL',11X,'FV',15X,'FZ',
     &  11X,'TORSION',10X,'MV',15X,'MZ')
  820 FORMAT (/,I10,1P,G15.5,0P,I6,1P,6G15.6,
     &          /15X,          I6,   6G15.6)
C
  850 RETURN
C
C====================== DETERMINE THE LENGTH OF THE MATERIAL DATA
  900 CONTINUE
      NMYST=1
      ISE=0
C
        MAT   = RINPUT( 1)
        CALL MATLIB
     & ( 0    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE,DUCT,EXCR,
     &   ET   ,F      ,RT   ,R    ,V     ,VL   ,LELEM ,LMAT  ,
     &   MAT  ,MATTYP,SE,IELNO,DAMAGE)
        ISE=ISE+LELEM
      RETURN
C
C====================== DETERMINE THE TOTAL STRAIN ENERGY
 1000 CONTINUE
      EESE=0
      EPSE=0
C
      DO 1010 I=1,12
 1010   EESE=EESE + (FT(I)+F(I))*(R(I)+RT(I))/2.
C
      ENGDAT(1) = EESE
      ENGDAT(2) = 0
C
      RETURN
C
C====================== DUCTILITIES AND EXCURSION RATIOS
C                     NO D & E RATIOS FOR ELASTIC ELEMENT
 1100 RETURN
C
C
C====================== PRINT MAXIMUM ELEMENT LOADS
C
 1200 CONTINUE
      LOAD2=0
      IF (FIRST) WRITE (6,491) ' MAXIMUM VALUES FOR ALL STEPS  '
     &          //' NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'
                WRITE (6,492) IELNO,LOAD2,NODEI,(FMAX(J),J=1,6),
     &                                NODEJ,(FMAX(J),J=7,12)
C
      RETURN
C
C====================== RESET ELEMENT FORCES
C
 1500 CONTINUE
C----- ZERO MATRICIES
       DO 1540 I=1,12
         FMAX I)=0
         F I)=0
         FT I)=0
         R I)=0
 1540    RT I)=0
      WRITE (6,1550) IELNO
 1550 FORMAT (' 3D BEAM..... ELEMENT #',I6,
     &  ' INTERNAL FORCES AND HYSTERESIS MODELS ARE RESET TO ZERO')
        FALSE=.FALSE.
        LSTTYP=0
        CALL MATLIB
     & ( 2    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE,DUCT,EXCR,
```

```
     &   FT    ,F     ,RT    ,R     ,V     ,VL    ,LELEM ,LMAT  ,
     &   MAT   ,MATTYP,SE,IELNO,DAMAGE)
C
      RETURN
C
C==================== DAMAGE INDEX
C
 1400 DAMAGE = 0.
C
      RETURN
C
      END
C=======================================================================
      SUBROUTINE ELE02
     & (IOPT       ,IUNIT     ,NEMK      ,FIRST     ,PRINT     ,
     &  NDOF       ,NLOAD     ,MAXNOD    ,NNODE     ,NCOS      ,
     &  ID         ,IDOF      ,COORD     ,COSINE    ,JTCOS     ,
     &  CONST      ,DISPL     ,VELOC     ,FORCE     ,FUB       ,
     &  IREL       ,BUG       ,IBUG      ,ACCEL     ,KGDATA    ,
     &  GRAV       ,PGEOM     ,ELSTIC    ,NAME      ,STEPID    ,
     &  IELNO      ,RINPUT    ,EESE      ,EPSE      ,INC       ,
     &  DAMAGE     ,
     &  IELTYP     ,IELDOF    ,NODEI     ,NODEJ     ,JOINTI    ,
     &  JOINTJ     ,R         ,RT        ,F         ,FT        ,
     &  V          ,VT        ,ISE       ,H         ,LENGTH    ,
     &  MAT        ,A         ,LM        ,KTYPE     ,FMAX      ,
     &  SE         ,STIFF     )
C
      IMPLICIT REAL(A-H,O-Z)
      LOGICAL ERR,FIRST,PRINT,BUG,FALSE,NEMK,ELSTIC
      LOGICAL BTEST
      CHARACTER*80 NAME
      CHARACTER*(*) STEPID
      CHARACTER*9  TYPE(6)
      DIMENSION  IDOF(MAXNOD,6),COORD(MAXNOD,6),COSINE(3,3,NCOS)
      DIMENSION CONST(MAXNOD,6),   ID(MAXNOD)  ,JTCOS(MAXNOD)
      DIMENSION SAT(2,12),STIFF(78)
      DIMENSION AC(12,2),S(2,2),LM(12),FMAX(2)
      DIMENSION CTS( 3, 3),CTE( 3, 3),VX(3),VV(3),VZ(3)
      DIMENSION BS( 3, 3),BE( 3, 3),SE(ISE)
      DIMENSION DISPL(NDOF,NLOAD),FUB(NDOF)
      DIMENSION VELOC(NDOF)
      DIMENSION FORCE(NDOF,NLOAD)
      DIMENSION RINPUT(100),WORK(6),DUCT(3),EXCR(6)
      REAL LENGTH
      REAL IX,IV,IZ,KIIZ,KJJZ,KIJZ,KIIV,KJJV,KIJV,KTI,KVJ
      INTEGER REL,RELT
      DIMENSION MD(15)
      DATA MD/1,2,4,7,11,16,22,29,37,46,56,67,79/
      DATA TYPE/'    AXIAL','  SHEAR-V','  SHEAR-Z',
     &          '  TORSION','BENDING-V','BENDING-Z'/
C
C-----------------------------------------------------------------------
C VARIABLES:
C-----------------------------------------------------------------------
C---------- GLOBAL VARIABLES
C IOPT   = 1, INITIALIZE BEAM COLUMN ELEMENT
C        = 2, CALCULATE GEOMETRIC STIFFNESS
C        = 3, FORM STIFFNESS
C        = 4, CALCULATE INCREMENTAL FORCES
C        = 5, CALCULATE TOTAL FORCES AND ENERGIES
C IUNIT  = OUTPUT FILE UNIT # FOR PRINTING OUTPUT
C ERR    = ERR OR FLAG, IF AN ERR OR WAS OCCURED, ERR=.TRUE.
C FIRST  = FLAG, FIRST=.TRUE.. PRINT HEADERS
C PRINT  = FLAG, PRINT=.TRUE.. PRINT DATA
C NDOF   = # OF GLOBAL DOF
C NLOAD  = # OF LOAD COMBINATIONS
C MAXNOD = # ROW DIMENSION OF NODE ARRAY
C NNODE  = # OF NODES
C ID     = ARRAY OF EXTERNAL NODE NUMBERS
C IDOF   = ARRAY OF DEGREES OF FREEDOM
C COORD  = ARRAY OF COORDINATES
C COSINE = ARRAY OF DIRECTION COSINES OF NODES
C CONST  = ARRAY OF CONSTRAINT TRANSFORMATIONS
C DISPL  = GLOBAL DISPLACEMENT MATRIX
C Z      = REAL GLOBAL STORTAG VECTOR
C NZ     = INTEGER GLOBAL STORTAG VECTOR
C IZMAT  = LOCATION OF MATERIAL DATA IN GLOBAL STORAGE VECTOR
C NAME   = ELEMENT NAME
C IELNO  = ELEMENT NUMBER
C RINPUT = INPUT DATA
C STIFF  = OUTPUT ELEMENT STIFFNESS (UPPER TRIANGULAR FORM)
C LM     = OUTPUT LOCAL TO GLOBAL STIFFNESS MAPPING MATRIX
C---------- ELEMENT VARIABLES
C H,Q,AI,AJ,C,BI,BJ,S22,S26,S212,S33,S35,S311,D
C        = INDIVIDUAL TERMS IN THE ELEMENT STIFFNESS (S) MATRIX
C PKG    = INPUT LOAD FOR FORMING GEOMETRIC STIFFNESS MATRIX...
C R      = INCRIMENTAL DISPLACEMENTS
C RT     = TOTAL DISPLACEMENTS
C F      = INCRIMENTAL FORCES
C FT     = TOTAL FORCES
C LM     = LOCAL TO GLOBAL STIFFNESS MAPPING MATRIX
C CTS    = LOCAL TO GLOBAL ROTATION MATRIX START
C CTE    = LOCAL TO GLOBAL ROTATION MATRIX END
C BS     = LOCAL TO GLOBAL ROTATED ECCENTRICITY/CONSTRAINT  MATRIX START
C BE     = LOCAL TO GLOBAL ROTATED ECCENTRICITY/CONSTRAINT  MATRIX END
C STIFF  = ELEMENT STIFFNESS
C IELTYP = ELEMENT TYPE = 10
C NODEI  = EXTERNAL JOINT NUMBER, END I
C NODEJ  = EXTERNAL JOINT NUMBER, END J
C JOINTI = INTERNAL JOINT NUMBER, END I
C JOINTJ = INTERNAL JOINT NUMBER, END J
C---------- TEMPRORY VARIABLES
C VX     = VECTOR DEFINING THE ELEMENT X AXIS
C VV     = VECTOR DEFINING THE ELEMENT Y AXIS
C COSLOC = LOCAL COSINE MATRIX
C A      = GLOBAL TO LOCAL FORCE TRANSFORMATION MATRIX
C S      = ELEMENT STIFFNESS AT LOCAL COORD. AT ENDS OF FLEXABLE PART
C SAT    = PRODUCT OF S*A'
C ASAT   = PRODUCT OF A*SAT
C-----------------------------------------------------------------------
C
C
C------- CHOOSE OPTION
      IF (IOPT.LT.1 .OR.IOPT.GT.14)
     & WRITE (6,*) 'INVALID OPTION IN ELE-02, IOPT=',IOPT
C
      GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,
     &                                          1300,1400),IOPT
C
  100 CONTINUE
C
C----- INTERPERTATE INPUT DATA
      IELTYP= 2
      MAT   = RINPUT( 1)
      NODEI = RINPUT( 2)
      NODEJ = RINPUT( 3)
      KTYPE = RINPUT( 4)
      XLEN  = RINPUT( 5)
      VX(1) = RINPUT( 6)
      VX(2) = RINPUT( 7)
      VX(3) = RINPUT( 8)
      XS    = RINPUT( 9)
      XE    = RINPUT(10)
C
C----- GET INTERNAL NODE # ASSOCIATED WITH EXTERNAL NODE #'S
      JOINTI=IQUICK(NODEI,ID,NNODE)
      JOINTJ=IQUICK(NODEJ,ID,NNODE)
C
      IF (KTYPE.LT.1 .OR. KTYPE.GT.6) THEN
         WRITE(6,102) IELNO,NODEI,NODEJ,KTYPE
  102    FORMAT(1X,20('**'),'ER','ROR - INVALID KTYPE',/
     &    5X,'IELNO=',I5,5X,'JOINTI=',I5,5X,'JOINTJ=',I5,
     &    5X,'KTYPE=',I5,/
     &    5X,'REVISE INPUT  1 LE KTYPE LE 6 ..............')
         LENGTH=1.0
      ENDIF
C
C-----------------------------------------------------------------------
C  GET GLOBAL TO LOCAL TRANSFORMATION MATRIX
C-----------------------------------------------------------------------
C------ DEFINE VECTOR X, LENGTH
      VX(1)=COORD(JOINTJ,1)-COORD(JOINTI,1)
      VX(2)=COORD(JOINTJ,2)-COORD(JOINTI,2)
      VX(3)=COORD(JOINTJ,3)-COORD(JOINTI,3)
      LENGTH=SQRT(VX(1)**2+VX(2)**2+VX(3)**2)-XS-XE
      IF (LENGTH.EQ.0) THEN
        DO 105 II=1,3
          VX(II)= COSINE(1,II,JTCOS(JOINTI))
  105     VV(II)= COSINE(2,II,JTCOS(JOINTI))
        LENGTH=1.00
      ELSE
        LENGTH=LENGTH-XS-XE
      ENDIF
      IF (XLEN.NE.0) THEN
        LENGTH=XLEN
        XS=0
        XE=0
      ENDIF
      IF (LENGTH.LE.0) THEN
        WRITE(6,101) IELNO,JOINTI,JOINTJ,LENGTH
  101   FORMAT(1X,20('**'),'ER','ROR - LENGTH IS LE 0',/
     &    5X,'IELNO=',I5,5X,'JOINTI=',I5,5X,'JOINTJ=',I5,
     &    5X,'LENGTH',1P,G15.6/
     &    5X,'REVISE INPUT.............................')
        LENGTH=1.0
      ENDIF
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'JOINTS',JOINTI,JOINTJ
        WRITE (6,*) 'VX',VX
        WRITE (6,*) 'VV',VV
        WRITE (6,*) 'LENGTH  ',LENGTH
      ENDIF
C------ GET LOCAL TO GLOBAL TRANSFORMATION MATRICIES CT AND B
      ICS=JTCOS(JOINTI)
      ICE=JTCOS(JOINTJ)
      CALL ROTXYZ(VX,VV,COSINE(1,1,ICS),CTS, XS ,0. ,0. ,BS,
     &      CONST(JOINTI,1),MAXNOD)
      CALL ROTXYZ(VX,VV,COSINE(1,1,ICE),CTE,-XE ,0. ,0. ,BE,
     &      CONST(JOINTJ,1),MAXNOD)
      IF ( BTEST(IBUG,7) ) THEN
          CALL MMATRX( CTS  , 3, 3,'CTS   ')
          CALL MMATRX( CTE  , 3, 3,'CTE   ')
      ENDIF
C
C----- GET GLOBAL CONSTRAINT MATRIX BS, BE
C
      IF ( BTEST(IBUG,7) ) THEN
         CALL MMATRX( BS, 3, 3,'BS   ')
         CALL MMATRX( BE, 3, 3,'BE   ')
      ENDIF
C
C----- ZERO MATRICIES
         F =0
         FT=0
         R =0
         RT=0
         V =0
         VT=0
         FMAX(1)=0
         FMAX(2)=0
C
C----- ASSEMBLE TRANSFORMATION MATRIX A
C  NOTE: A IS DIMMENSIONED AS A 12X2 MATRIX, BUT ONLY THE FIRST IELDOF
C        ROW'S CONTAIN INFORMATION. THE BALANCE OF THE MATRIX IS
C        UNDEFINED.  THE GLOBAL DOF COORESPONDING TO THE ROW'S OF A
C        ARE STORED IN VECTOR LM.  AGAIN, ONLY THE FIRST IELDOF ROW'S
C        OF LM ARE DEFINED.
      IELDOF=0
      DO 50 I=1,12
C
C               REMOVE DOF'S THAT ARE CONSTRAINED TO THE SAME DOF...
         J=I
         IF (I.GT.6) J= I-6
         IF ( IDOF(JOINTI,J).EQ.IDOF(JOINTJ,J) ) GO TO 50
         IF (I.LE.6) THEN
              LMTI=IDOF(JOINTI,I)
         ELSE
              LMTI=IDOF(JOINTJ,J)
         ENDIF
C
         A1=0
         A2=0
         IF (I.LE.3) THEN
              IF (KTYPE.LE.3) A1=CTS(I,KTYPE)
         ELSE IF (I.LE.6) THEN
              IF (KTYPE.LE.3) THEN
                   A1=BS(I-3,KTYPE)
              ELSE
                   A1=CTS(I-3,KTYPE-3)
              ENDIF
         ELSE IF (I.LE.9) THEN
              IF (KTYPE.LE.3) A2=CTE(I-6,KTYPE)
         ELSE
              IF (KTYPE.LE.3) THEN
                   A2=BE(I-9,KTYPE)
              ELSE
```

```
                A2=CTE(I-9,KTYPE-5)
              ENDIF
            ENDIF
            IF (A1.EQ.0 .AND. A2.EQ.0) GO TO 50
            IELDOF=IELDOF+1
            A(IELDOF,1)=A1
            A(IELDOF,2)=A2
            LM(IELDOF)=LMI
C
   50     CONTINUE
       IF ( BTEST(IBUG,7) ) CALL HMTRX2( A     ,IELDOF,2,' A     ',12)
C
C----- GET SPRING STIFFNESS
          FALSE=.FALSE.
          LSTTYP=0
          CALL MATLIB
     &  ( 2     ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &    FT    ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &    MAT   ,MATTYP,SE,IELNO,DAMAGE)
C
  111 FORMAT (OP,' ZMATRXC',I6,'):',I15,1P,G20.10)
C------- H IS THE SPRING STIFFNESS
C
      H = SE(1)
      HH= SE(1)/LENGTH
      S(1,1)= HH
      S(1,2)=-HH
      S(2,1)=-HH
      S(2,2)= HH
C
C----- CALCULATE SAT
      DO 60 I=1,2
        DO 60 J=1,IELDOF
        SAT(I,J)=0
          DO 60 K=1,2
   60       SAT(I,J)=SAT(I,J)+S(I,K)*A(J,K)
C
C----- CALCULATE ASAT AND CONVERT TO HALF STORAGE MODE  BY COLUMNS
C
C         1  3  6 10 15 21 28 36 45 55 66 78  NUMBER INDICATES
C            2  5  9 14 20 27 35 44 54 65 77  LOCATION OF DATA
C               4  8 13 19 26 34 43 53 64 76  IN ARRAY STIFF
C                  7 12 18 25 33 42 52 63 75
C                   11 17 24 32 41 51 62 74
C                      16 23 31 40 50 61 73
C                         22 30 39 49 60 72
C                            29 38 48 59 71
C                               37 47 58 70
C                                  46 57 69
C                                     56 68
C                                        67
      L=0
      DO 70 J=1,IELDOF
        DO 70 I=J,1,-1
        L=L+1
        STIFF(L)=0
          DO 70 K=1,2
   70       STIFF(L)=STIFF(L)+A(I,K)*SAT(K,J)
C
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'IELDOF',IELDOF,'LM =',(LM(I),I=1,IELDOF)
        CALL LMATRX(STIFF,ND,IELDOF,(L+1),'SPRING STIFFNESS')
      ENDIF
C
C----- RELEASE UNUSED STORAGE
  180 IREL=78-L
C
C     PRINT COLUMN DATA
      IF (FIRST .OR. BTEST(IBUG,7) .OR. BTEST(IBUG,8) ) WRITE (6,192)
  191 WRITE(6,193) NAME,IELNO,MAT,NODEI,NODEJ,TYPE(KTYPE),LENGTH,
     &    VV(1),VV(2),VV(3),XS,-XE
C
  192 FORMAT(/' ELEMENT 02, ONE (LOCAL) DOF SPRING ELEMENT'//
     &  T22,'B  MATL START   END',2X,'--TYPE---',6X,'LENGTH',5X,
     &  11('-'),'  V-AXIS  ',
     &  12('-'),'  START DIST    END DIST')
  193 FORMAT(1X,A15,4I6,2H,A9,5X,1P,G12.4,0P,
     &       F9.5,' I',3P,F9.5,' J',F9.5,' K',1P,3S,3G12.4)
      RETURN
C======================== DETERMINE GEOMETRIC STIFF ====================
  200 RETURN
C    THE GEOMETRIC STIFFNESS MATRIX IS NOT AVAILABLE FOR THIS ELEMENT
C
C======================== DETERMINE STIFFNESS ==========================
  300 CONTINUE
C
C------ H CONTAINS THE SPRING STIFFNESS, WHICH WAS DETERMINED IN THE
C       LAST CALL TO MAT_LIB
C       IF H=SE(1) THEN THE STIFFNESS HAS NOT CHANGED, RETURN
C
      IF (H.EQ.SE(1)) RETURN
C
      H = SE(1)
      HH= SE(1)/LENGTH
      S(1,1)= HH
      S(1,2)=-HH
      S(2,1)=-HH
      S(2,2)= HH
C
C----- CALCULATE SAT
      DO 360 I=1,2
        DO 360 J=1,IELDOF
        SAT(I,J)=0
          DO 360 K=1,2
  360       SAT(I,J)=SAT(I,J)+S(I,K)*A(J,K)
C
C----- CALCULATE ASAT, STORE IN UPPER TRIANGULAR MATRIX
      L=0
      DO 370 J=1,IELDOF
        DO 370 I=J,1,-1
        L=L+1
        STIFF(L)=0
          DO 370 K=1,2
  370       STIFF(L)=STIFF(L)+A(I,K)*SAT(K,J)
C
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'IELDOF',IELDOF,'LM =',(LM(I),I=1,IELDOF)
        CALL LMATRX(STIFF,MD,IELDOF,(L+1),'SPRING STIFFNESS')
      ENDIF
C
      RETURN
C======================== DETERMINE INCREMENTAL FORCES =================
  400 CONTINUE
       IF (PRINT.AND.FIRST) WRITE (6,491) STEPID
C
C------ LOOP FOR EACH LOAD
      DO 450 LOAD=1,NLOAD
C
C------ TRANSFORM FRAME DISPL INTO LOCAL DISPL
C... LOCAL DISPL ARE AT THE END END OF THE SPRING. THUS
C... POSITIVE IS TENSION AND NEGATIVE IS COMPRESSION
        R=0
        V=0
        DO 410 I=1,IELDOF
          J=LM(I)
            A1= A(I,1)
            A2= A(I,2)
            DISP= DISPL(J,LOAD)
            VEL= VELOC(J)
C
          R= R + (A(I,2)-A(I,1)) * DISPL(J,LOAD)
  410     V= V + (A(I,2)-A(I,1)) * VELOC(J)
C
C------ INCREMENTAL FORCES
      F= H*R/LENGTH
C
C------ CHECK STIFFNESS AND CALCULATE UNBALANCED FORCES
C       THIS IS PREFORMED ONLY FOR INELASTIC ANALYSIS
      P =FT+F
      IF (.NOT. ELSTIC) THEN
C         -- SET UP TEMPORARY - TOTAL LOADS, DISPL'S AND VELOC
          PL=FT
          D =(RT+R)/LENGTH
          DL=RT /LENGTH
          VC=VT+V
          VL=VT
C         -- TRANSFER PERMANENT HYSTERESIS DATA TO TEMPROARY STORAGE...
          ISE2=ISE/2
          DO 415 I=1,ISE2
  415       SE(I+ISE2)=SE(I)
C         -- CALL HYST MODEL TO GET NEW STIFFNESS AND
C            THE TOTAL FORCE P AT DISPL D.
          MOPT=3
          CALL MATLIB
     &  (MOPT ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &    P     ,PL    ,D     ,DL    ,VC    ,VL    ,LELEM ,LMAT  ,
     &    MAT   ,MATTYP,SE(ISE2+1),IELNO,DAMAGE)
          IF (SE(ISE2+1).NE.SE(1)) NEWK=.TRUE.
      ENDIF
C
C------ SAVE PEAK VALUES FOR MULTIPLE LOAD CASES
      IF (NLOAD.GT.1 .AND.  ABS(F).GT.ABS(FMAX(1)) ) THEN
        FMAX(1)=F
        FMAX(2)=R
      ENDIF
C
C------ PRINT DATA
      HH= H/LENGTH
      IF (PRINT) WRITE (6,492) IELNO,LOAD,TYPE(KTYPE),NODEI,NODEJ,F,R,HH
C
  450 CONTINUE
C
C------ CALCULATE THE UNBALANCED MEMBER FORCE ON A JOINT.
          UBAL= F+FT-P
          F   = F
          FT  = FT
          P   = P
          IF (BTEST(IBUG,7)) WRITE (6,495) F,FT,P
          DO 440 I=1,IELDOF
            J=LM(I)
            A1= A(I,1)
            A2= A(I,2)
            FUB(J)= FUB(J) + (A(I,1)-A(I,2)) * UBAL
            IF (BTEST(IBUG,7))
     &      WRITE (6,496) (II,LM(II),FUB(II),II=1,IELDOF)
  440     CONTINUE
C
  491 FORMAT (/'       SPRING FORCES...',5X,A/
     &  '   ELEMENT LOAD',T17,'      TYPE',' NODEI',' NODEJ',
     &  '       FORCE     ',' DISPLACMENT  ','   STIFFNESS   ')
  492 FORMAT (I10,I5,2X,A9,2I6,1P,5G15.6)
  495 FORMAT (' F:',1P,G16.8,'  FT:',G16.8,'  P:',G16.8)
  496 FORMAT (' I:',I5,'  LM(I):',I6,'  FUB(I):',1P,G16.8)
C
C------ ADJUST F FOR UNBALANCED LOAD P
          F=P-FT
      RETURN
C======================== DETERMINE TOTAL FORCES, ENERGIES ECT ========
  500 CONTINUE
       IF (PRINT.AND.FIRST) WRITE (6,491) STEPID
C
C------ TOTAL FORCES, DISPLACEMENTS AND VELOCITIES
      FT=FT+F
      RT=RT+R
      VT=VT+V
      IF ( ABS(FT).GT.ABS(FMAX(1)) ) THEN
        FMAX(1)=FT
        FMAX(2)=RT
      ENDIF
C
C------ PRINT DATA
      HH= H/LENGTH
      IF (PRINT) WRITE(6,492) IELNO,1,TYPE(KTYPE),NODEI,NODEJ,FT,RT,HH
C
C------ TRANSFER TEMPROARY HYSTERESIS DATA TO PERMANENT STORAGE...
      ISE2=ISE/2
      IF (.NOT. ELSTIC) THEN
        DO 510 I=1,ISE2
  510     SE(I)=SE(I+ISE2)
      ENDIF
C
      RETURN
C
C======================== DETERMINE FIXED END FORCES, AND ADD TO FORCE=
  600 RETURN
C
C-------- FIXED END FORCES ARE NOT AVAILABLE FOR THE SPRING
C
C====================== WRITE MEMBER FORCES TO OUTPUT FILE ============
  700 CONTINUE
C
      CALL MATLIB
     &  ( 4     ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &    FT    ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &    MAT   ,MATTYP,SE,IELNO,DAMAGE)
C
```

```
      WRITE (IUNIT,710)  NODEI,NODEJ,KTYPE,FT,RT,H,EESE,EPSE,
     &                   (DUCT(I),EXCR(I),I=1,3)
  710 FORMAT (3I6,1P,5G12.4/6G12.4)
C
      RETURN
C
C=================== READ AND PRINT MEMBER FORCES FROM OUTPUT FILE ====
  800 CONTINUE
      BACKSPACE(IUNIT)
      READ (IUNIT,*  ) ITYPE,IELNO,INRITE,T0,DT
      ISTEP=-INRITE
C
  810 READ (IUNIT,815,END=850) NODEI,NODEJ,KTYPE,FT,RT,H,ESE,PSE,
     &                   (WORK(I),I=1,6)
  815 FORMAT (3I6,5G12.4/6G12.4)
        ISTEP=ISTEP+INRITE
        T=T0+DT*ISTEP
        IF (ISTEP.EQ.0) WRITE (6,805) NODEI,NODEJ,TYPE(KTYPE)
        IF (MOD(ISTEP,INC).EQ.0)
     &  WRITE (6,820) ISTEP,T,FT,RT,H,ESE,PSE
        GO TO 810
C
  805 FORMAT (//'       SPRING FORCES...'//
     & 5X,'NODEI:',I6,5X,'NODEJ:',I6,5X,'SPRING TYPE:',A,//
     & '     STEP       TIME        ',
     & '    FORCE     ',' DISPLACMENT ',' STIFFNESS',
     & '    ESE       ',',      PSE       ')
  820 FORMAT (I10,1P,G15.5,6G13.5)
C
  850 WRITE (6,840) (WORK(I),I=1,6)
  840 FORMAT (//10X,' MAXIMUN DUCTILITIES AND EXCURSION RATIOS'/
     &        T10,'==========================================='/
     & T15,'DISPLACEMENT DEFINATION: U1=',1P,G12.4,5X,'E1=',G12.4/
     & T15,'ENERGY DEFINATION #1:    U2=',1P,G12.4,5X,'E2=',G12.4/
     & T15,'ENERGY DEFINATION #2:    U3=',1P,G12.4,5X,'E3=',G12.4)
      RETURN
C
C=================== DETERMINE THE LENGTH OF THE MATERIAL DATA
  900 CONTINUE
      MMVST=1
      ISE=0
C
        MAT   = RINPUT( 1)
        CALL MATLIB
     & ( 0    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &   FT    ,F     ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MAT   ,MATTYP,SE,IELNO,DAMAGE)
        ISE=ISE+LELEM
C
C------- DOUBLE THE STORAGE, TO ALLOW FOR TEMPROARY VALUES...
        ISE=ISE*2
      RETURN
C
C=================== DETERMINE THE STRAIN ENERGY
 1000 CONTINUE
C
C=================== DUCTILITIES AND EXCURSION RATIO'S
 1100 CONTINUE
C.... OPTIONS 10 AND 11 HAVE THE SAME CODE FOR THIS ELEMENT...
C.... THE ENERGY HAS CALCULATED WITH THE LAST CALL FOR INCRIMENTAL FORCE
C     THE TOTAL FORCES HAVE NOT BEEN CALLED YET,THUS THE ENERGIES RESIDE
C     IN THE MATRIX SE BEGINING AT ADDRESS ISE21.
C
      ISE21=ISE/2 +1
      CALL MATLIB
     & ( 4    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &   FT    ,F     ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MAT   ,MATTYP,SE(ISE21),IELNO,DAMAGE)

      IF (ELSTIC) EESE=(FT+F)*(RT+R)*0.50

      IF (PRINT.AND.FIRST) WRITE (6,1191)
      IF (PRINT) WRITE (6,1192) IELNO,(DUCT(I),EXCR(I),I=1,3)
 1191 FORMAT (//'  SPRING DUCTILITIES AND EXCURSION RATIOS....',5X/
     & '    ELEMENT',T12,'|- DUCTILITY DEFINATION #1 -| ',
     &                   '|- DUCTILITY DEFINATION #2 -| ',
     &                   '|- DUCTILITY DEFINATION #3 -| ',// T12,
     & '    DUCTILITY    EXCURSION     ',
     & '    DUCTILITY    EXCURSION     ',
     & '    DUCTILITY    EXCURSION     ')
 1192 FORMAT (I10,1P,6G15.6)
C
      RETURN
C
C=================== MAXIMUN FORCES
 1200 CONTINUE
      IF (FIRST) WRITE (6,491) 'MAXIMUN LOAD AND DISPL AT MAXIMUN LOAD '
     &          //' NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'
      WRITE(6,492) IELNO,0,TYPE(KTYPE),NODEI,NODEJ,FMAX(1),FMAX(2)
      RETURN
C
C
C=================== RESET INTERNAL FORCES
 1300 CONTINUE
C
C----- ZERO MATRICIES
        F =0
        FT=0
        R =0
        RT=0
        V =0
        VT=0
        FMAX(1)=0
        FMAX(2)=0
      DO 1310 I=1,ISE
 1310   SE(I)=0
      WRITE (6,1330) IELNO
 1330 FORMAT (' SPRING..... ELEMENT #',I6,
     & '  INTERNAL FORCES AND HYSTERESIS MODELS ARE RESET TO ZERO')
        CALL MATLIB
     & ( 2    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &   FT    ,F     ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MAT   ,MATTYP,SE,IELNO,DAMAGE)
      RETURN
C
C
C=================== DAMAGE INDEX ====================
 1400 IF (ELSTIC) RETURN
C
      FMAX1=FMAX(1)
      FMAX2=FMAX(2)
      CALL MATLIB
     & ( 5    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,EESE,EPSE, DUCT,EXCR,
     &   FMAX1,0.    ,FMAX2,  0., 0.  ,0.    ,LELEM ,LMAT  ,
     &   MAT   ,MATTYP,SE,IELNO,DAMAGE)

      IF (PRINT.AND.FIRST) WRITE (6,1410)
      IF (PRINT) WRITE (6,1420) IELNO,DAMAGE,FMAX1,FMAX2,EESE,EPSE
 1410 FORMAT (/
     & '   ELEMENT',T12,' DAMAGE INDEX       F MAX      ',
     &                  '       D MAX          ESE        ',
     &                  '       PSE        ')
 1420 FORMAT (I10,1P,6G15.6)
C
      DAMAGE=DAMAGE*(EESE+EPSE)
      RETURN
C
      END
C=======================================================================
      SUBROUTINE ELE05
     & (IOPT    ,IUNIT   ,NEMK    ,FIRST   ,PRINT   ,
     &  NDOF    ,NLOAD   ,MAXNOD  ,NNODE   ,NCOS    ,
     &  ID      ,IDOF    ,COORD   ,COSINE  ,JTCOS   ,
     &  CONST   ,DISPL   ,VELOC   ,FORCE   ,FUB     ,
     &  IREL    ,BUG     ,IBUG    ,ACCEL   ,KGDATA  ,
     &  GRAV    ,PGEOM   ,ELSTIC  ,NAME    ,STEPID  ,
     &  IELNO   ,RINPUT  ,AXIAL   ,ESE     ,PSE     ,
     &  INC     ,DAMAGE  ,
     &  IELTYP  ,IELDOF  ,ISEB    ,ISES    ,ISEA    ,
     &  ISE     ,J1      ,J2      ,J3      ,J4      ,
     &  MATB    ,MATS    ,MATA    ,BLANK   ,R       ,
     &  RT      ,F       ,FT      ,V       ,VT      ,
     &  PKG     ,LKG     ,LM      ,LMKG    ,A       ,
     &  SB      ,SS      ,SA      ,TEMP1   ,TEMP2   ,
     &  KGDOF   ,LENGTH  ,FMAX    ,SE      ,STIFF   )
C
      IMPLICIT REAL(A-H,O-Z)
      DOUBLE PRECISION SX,SXX,SV,SXV,SN
      LOGICAL ERR,FIRST,PRINT,BUG,FALSE,NEMK,ELSTIC ,AXIAL,BTEST
      CHARACTER*80 NAME
      CHARACTER*(*) STEPID
      DOUBLE PRECISION  SAT,DSTIFF(600)
      DIMENSION  IDOF(MAXNOD,6),COORD(MAXNOD,6),COSINE(5,5,NCOS)
      DIMENSION CONST(MAXNOD,6),    ID(MAXNOD)  ,JTCOS(MAXNOD)
      DIMENSION SAT(5,24),STIFF(600),KGDATA(5) ,SAT2(6,24)
      DIMENSION A1(24,6),A(24,5),S(6,6),LM(24),ACC(6,3),LMKG(24)
      DIMENSION CT(5,5),BS(5,5),VX(5),VY(5),VXB(5),VXT(5),VZT(5),VZB(5)
      DIMENSION                                          VZ1(5),VZ2(5)
      DIMENSION SE(ISE),A11(6,3),A22(6,3),A44(6,3),A33(6,3)
      DIMENSION F(5),R(5),V(5),FT(5),RT(5),VT(5),PK(5)
      DIMENSION DISPL(NDOF,NLOAD),FUB(NDOF),EFUB(24),FMAX(6)
      DIMENSION VELOC(NDOF)
      DIMENSION FORCE(NDOF,NLOAD)
      DIMENSION RINPUT(100),WORK(1),DUCT(3,3),EXCR(6,3)
      REAL LENGTH
      DIMENSION MX(25)
      DATA MX/ 1.  2.  4.  7. 11. 16. 22. 29. 37. 46. 56. 67.
     &         79. 92,106,121,137,154,172,191,211,232,254,277,301/
C
C-----------------------------------------------------------------------
C VARIABLES:
C-----------------------------------------------------------------------
C---------- GLOBAL VARIABLES
C IOPT   = 1, INITIALIZE BEAM COLUMN ELEMENT
C        = 2, CALCULATE GEOMETRIC STIFFNESS
C        = 3, FORM STIFFNESS
C        = 4, CALCULATE INCREMENTAL FORCES
C        = 5, CALCULATE TOTAL FORCES AND ENERGIES
C IUNIT  = OUTPUT FILE UNIT # FOR PRINTING OUTPUT
C ERR    = ERR OR FLAG, IF AN ERR OR HAS OCCURED, ERR=.TRUE.
C FIRST  = FLAG, FIRST=.TRUE.. PRINT HEADERS
C PRINT  = FLAG, PRINT=.TRUE.. PRINT DATA
C NDOF   = # OF GLOBAL DOF
C NLOAD  = # OF LOAD COMBINATIONS
C MAXNOD = # ROW DIMENSION OF NODE ARRAY
C NNODE  = # OF NODES
C ID     = ARRAY OF ENTERNAL NODE NUMBERS
C IDOF   = ARRAY OF DEGREES OF FREEDOM
C COORD  = ARRAY OF COORDINATES
C COSINE = ARRAY OF DIRECTION COSINES OF NODES
C CONST  = ARRAY OF CONSTRAINT TRANSFORMATIONS
C DISPL  = GLOBAL DISPLACEMENT MATRIX
C Z      = REAL GLOBAL STORTAG VECTOR
C NZ     = INTEGER GLOBAL STORTAG VECTOR
C IZMAT  = LOCATION OF MATERIAL DATA IN GLOBAL STORAGE VECTOR
C NAME   = ELEMENT NAME
C IELNO  = ELEMENT NUMBER
C RINPUT = INPUT DATA
C STIFF  = OUTPUT ELEMENT STIFFNESS (UPPER TRIANGULAR FORM)
C LM     = OUTPUT LOCAL TO GLOBAL STIFFNESS MAPPING MATRIX
C---------- ELEMENT VARIABLES
C H,Q,AI,AJ,C,BI,BJ,S22,S26,S212,S33,S35,S311,0
C        = INDIVIDUAL TERMS IN THE ELEMENT STIFFNESS (S) MATRIX
C PKG    = INPUT LOAD FOR FORMING GEOMETRIC STIFFNESS MATRIX...
C R      = INCRIMENTAL DISPLACEMENTS
C RT     = TOTAL DISPLACEMENTS
C F      = INCRIMENTAL FORCES
C FT     = TOTAL FORCES
C LM     = LOCAL TO GLOBAL STIFFNESS MAPPING MATRIX
C CTS    = LOCAL TO GLOBAL ROTATION MATRIX START
C CTE    = LOCAL TO GLOBAL ROTATION MATRIX END
C BS     = LOCAL TO GLOBAL ROTATED ECCENTRICITY/CONSTRAINT  MATRIX START
C BE     = LOCAL TO GLOBAL ROTATED ECCENTRICITY/CONSTRAINT  MATRIX END
C STIFF  = ELEMENT STIFFNESS
C IELTYP = ELEMENT TYPE = 10
C NODEI  = EXTERNAL JOINT NUMBER, END I
C NODEJ  = EXTERNAL JOINT NUMBER, END J
C JOINTI = INTERNAL JOINT NUMBER, END I
C JOINTJ = INTERNAL JOINT NUMBER, END J
C---------- TEMPROARY VARIABLES
C VX     = VECTOR DEFINING THE ELEMENT X AXIS
C VY     = VECTOR DEFINING THE ELEMENT Y AXIS
C COSLOC = LOCAL COSINE MATRIX
C A      = GLOBAL TO LOCAL FORCE TRANSFORMATION MATRIX
C S      = ELEMENT STIFFNESS AT LOCAL COORD, AT ENDS OF FLEXABLE PART
C SAT    = PRODUCT OF S*A'
C ASAT   = PRODUCT OF A*SAT
C-----------------------------------------------------------------------
C
C
C-------- CHOOSE OPTION
        IF (IOPT.LT.1 .OR.IOPT.GT.14)
     & WRITE (6,*) 'INVALID OPTION IN ELE-02, IOPT=',IOPT
C
        GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,
     &                                          1300,1400),IOPT
C
```

```
  100 CONTINUE
C
C----- INTERPERTATE INPUT DATA
      IELTYP= 5
      MATB  = RINPUT( 1)
      MATS  = RINPUT( 2)
      MATA  = RINPUT( 3)
      NODE1 = RINPUT( 4)
      NODE2 = RINPUT( 5)
      NODE3 = RINPUT( 6)
      NODE4 = RINPUT( 7)
      ALPIN = RINPUT( 8)
      PKG   = RINPUT( 9)
C
C----- GET INTERNAL NODE # ASSOCIATED WITH EXTERNAL NODE #'S
      J1=IQUICK(NODE1,ID,NNODE)
      J2=IQUICK(NODE2,ID,NNODE)
      J3=IQUICK(NODE3,ID,NNODE)
      J4=IQUICK(NODE4,ID,NNODE)
C
C-------------------------------------------------------------
C  GET GLOBAL TO LOCAL TRANSFORMATION MATRIX
C-------------------------------------------------------------
C----- DEFINE VECTOR X, AT BOTTOM OF THE WALL
      VXB(1)=COORDX(J4,1)-COORDX(J3,1)
      VXB(2)=COORDX(J4,2)-COORDX(J3,2)
      VXB(3)=COORDX(J4,3)-COORDX(J3,3)
C
C----- DEFINE VECTOR Y, AT TOP OF THE WALL
      VXT(1)=COORDX(J1,1)-COORDX(J2,1)
      VXT(2)=COORDX(J1,2)-COORDX(J2,2)
      VXT(3)=COORDX(J1,3)-COORDX(J2,3)
C
C----- CALCULATE V VECTOR ...
      VV(1)=(COORDX(J1,1)+COORDX(J2,1)-COORDX(J3,1)-COORDX(J4,1))/2.
      VV(2)=(COORDX(J1,2)+COORDX(J2,2)-COORDX(J3,2)-COORDX(J4,2))/2.
      VV(3)=(COORDX(J1,3)+COORDX(J2,3)-COORDX(J3,3)-COORDX(J4,3))/2.
C
C----- CALCULATE AVERAGE LENGTH,ALPHA AND BETA...
      LENGTH=SQRT(VV(1)**2+VV(2)**2+VV(3)**2)
      IF (LENGTH.LE.0) THEN
         WRITE(6,101) IELNO,LENGTH
  101    FORMAT(1X,20('*'),'ER','ROR - LENGTH IS LE 0',/
     &    3X,'IELNO=',I5,3X,3X,'LENGTH',1P,G15.6/
     &    3X,'REVISE INPUT.............................'/)
         LENGTH=1.0
      ENDIF
C
      ALPHA=ALPIN*LENGTH
      BETA = LENGTH-ALPHA
C
C----- NORMALIZE V VECTOR ...
      VV(1)=VV(1)/LENGTH
      VV(2)=VV(2)/LENGTH
      VV(3)=VV(3)/LENGTH
C
C----- CALCULATE THE WIDTH ...      --- CHECK THE NEXT 7 LINES ---
      CALL VCROSS(VZT,VV,VXT)
      CALL VCROSS(VZ1,VZT,VV)
      CALL VCROSS(VZB,VV,VXB)
      CALL VCROSS(VZ2,VZB,VV)
      WT = SQRT(VZ1(1)**2 + VZ1(2)**2 + VZ1(3)**2)
      WB = SQRT(VZ2(1)**2 + VZ2(2)**2 + VZ2(3)**2)
      W  = (WT+WB)/2.0
C
C----- CHECK FOR TWIST
      TWIST=ACOS(VCOS(VZT,VZB))*57.2958
      IF (ABS(TWIST).GT. 5.0) WRITE (6,102) TWIST
  102 FORMAT(5X,10('*'),' WARNING THE WALL IS TWISTED=',1P,G12.4,
     &    ' DEGREES, REVISE INPUT AND RERUN')
C
C----- CALCULATE VN, AND NORMALIZE
      CALL VCROSS(VX,VV,VZB)
      WX=SQRT(VX(1)**2+VX(2)**2+VX(3)**2)
      VX(1)=VX(1)/WX
      VX(2)=VX(2)/WX
      VX(3)=VX(3)/WX
C
C----- CHECK FOR TWIST
      IF ( BTEST(IBUG,7) ) THEN
         WRITE (6,*) 'JOINTS',J1,J2,J3,J4
         WRITE (6,*) '                VX',VX
         WRITE (6,*) '                VV',VV
         WRITE (6,*) '                VZ',VZB
         WRITE (6,*) 'LENGTH,ALPHA,BETA',LENGTH,ALPHA,BETA
      ENDIF
C
C----- GET LOCAL TO GLOBAL TRANSFORMATION MATRICIES
C--- JOINT #1
         CALL ROTXYZ(VX,VV,COSINE(1,1,JTOSC(J1)),CT,
     &                        0.,0.,0.,BS,  CONST(J1,1),MAXNOD)
         DO 21 I=1,3
         DO 21  J=1,3
           A11(I   ,J)=CT(I,J)
   21      A11(I+3,J)=BS(I,J)
C--- JOINT #2
         CALL ROTXYZ(VX,VV,COSINE(1,1,JTOSC(J2)),CT,
     &                        0.,0.,0.,BS,  CONST(J2,1),MAXNOD)
         DO 22 I=1,3
         DO 22  J=1,3
           A22(I   ,J)=CT(I,J)
   22      A22(I+3,J)=BS(I,J)
C--- JOINT #3
         CALL ROTXYZ(VX,VV,COSINE(1,1,JTOSC(J3)),CT,
     &                        0.,0.,0.,BS,  CONST(J3,1),MAXNOD)
         DO 23 I=1,3
         DO 23  J=1,3
           A33(I   ,J)=CT(I,J)
   23      A33(I+3,J)=BS(I,J)
C--- JOINT #4
         CALL ROTXYZ(VX,VV,COSINE(1,1,JTOSC(J4)),CT,
     &                        0.,0.,0.,BS,  CONST(J4,1),MAXNOD)
         DO 24 I=1,3
         DO 24  J=1,3
           A44(I   ,J)=CT(I,J)
   24      A44(I+3,J)=BS(I,J)
C
         IF ( BTEST(IBUG,7) ) THEN
              WRITE (6,*) 'JOINT #1,  J1',J1
              CALL MMATRX(A11  , 6, 3,'A11  ')
              WRITE (6,*) 'JOINT #2,  J2',J2
              CALL MMATRX(A22  , 6, 3,'A22  ')
              WRITE (6,*) 'JOINT #3,  J3',J3
              CALL MMATRX(A33  , 6, 3,'A33  ')
              WRITE (6,*) 'JOINT #4,  J4',J4
              CALL MMATRX(A44  , 6, 3,'A44  ')
         ENDIF
C
C----- ZERO MATRICIES
         DO 25 I=1,3
           FMAX(I  )= 0
           FMAX(I+3)= 0
           F (I)=0
           FT(I)=0
           R (I)=0
           RT(I)=0
           V (I)=0
   25    VT(I)=0
         DO 27 I=1,24
           LM  (I)=0
           LMKG(I)=0
           DO 26 J=1,3
   26        A(I,J)=0.00
           DO 27 K=1,6
   27        AT(I,K)=0.00
C
C----- ASSEMBLE TRANSFORMATION MATRIX A = A3 X A4 X A3
C  NOTE: AT IS DIMMENSIONED AS A 24X6 MATRIX, BUT ONLY THE FIRST IELDOF
C        ROW'S CONTAIN INFORMATION. THE BALANCE OF THE MATRIX IS
C        UNDEFINED.  THE GLOBAL DOF COORESPONDING TO THE ROW'S OF A
C        ARE STORED IN VECTOR LM.  AGAIN, ONLY THE FIRST IELDOF ROW'S
C        OF LM ARE DEFINED.
         IELDOF=0
         DO 50 I=1,4
         DO 50 J=1,6
C
C               REMOVE DOF'S THAT ARE CONSTRAINED TO THE SAME DOF...
           IF ( (IDOF(J1,J).EQ.IDOF(J4,J)) .OR.
     &          (IDOF(J1,J).EQ.IDOF(J3,J)) .OR.
     &          (IDOF(J2,J).EQ.IDOF(J4,J)) .OR.
     &          (IDOF(J2,J).EQ.IDOF(J3,J)) ) GO TO 50
C
           IELDOF=IELDOF+1
           IF (I.EQ.1) THEN
               LMT1=IDOF(J1,J)
               AT(IELDOF,1)=A11(J,I)
               AT(IELDOF,2)=A11(J,2)
           ELSE IF (I.EQ.2) THEN
               LMT1=IDOF(J2,J)
               AT(IELDOF,3)=A22(J,2)
           ELSE IF (I.EQ.3) THEN
               LMT1=IDOF(J3,J)
               AT(IELDOF,4)=A33(J,2)
           ELSE IF (I.EQ.4) THEN
               LMT1=IDOF(J4,J)
               AT(IELDOF,5)=A44(J,1)
               AT(IELDOF,6)=A44(J,2)
           ENDIF
C
           LM(IELDOF)=LMT1
           IF((AT(IELDOF,1).EQ.0).AND.(AT(IELDOF,2).EQ.0) .AND.
     &        (AT(IELDOF,3).EQ.0).AND.(AT(IELDOF,4).EQ.0) .AND.
     &        (AT(IELDOF,5).EQ.0).AND.(AT(IELDOF,6).EQ.0)) THEN
               IELDOF=IELDOF-1
           ELSE IF (IELDOF.GT.1) THEN
              DO 49  II=1,IELDOF-1
                IF (LM(II).EQ.LM(IELDOF)) THEN
                   DO 48 JJ=1,6
                     AT(II,JJ)=AT(II,JJ)+AT(IELDOF,JJ)
   48              AT(IELDOF,JJ)=0
                   IELDOF=IELDOF-1
                   GO TO 50
                ENDIF
   49         CONTINUE
           ENDIF
C
   50    CONTINUE
C
C------ A2A1 FOR STIFFNESS ONLY...
         AC(1,1)=  0.
         AC(2,1)= -1./W
         AC(3,1)=  1./W
         AC(4,1)= -1./W
         AC(5,1)=  0.
         AC(6,1)=  1./W
         AC(1,2)=  1.
         AC(2,2)=  ALPHA/W
         AC(3,2)= -ALPHA/W
         AC(4,2)= -BETA /W
         AC(5,2)= -1.
         AC(6,2)=  BETA /W
         AC(1,3)=  0.
         AC(2,3)=  1./2.
         AC(3,3)=  1./2.
         AC(4,3)= -1./2.
         AC(5,3)=  0.
         AC(6,3)= -1./2.
C
         DO 55 I=1,IELDOF
         DO 55  J=1,3
           AC(I,J)=0.
           DO 55 K=1,6
   55      AC(I,J)=AC(I,J)+AT(I,K)*AC(K,J)
C
         IF ( BTEST(IBUG,7) ) THEN
              CALL MMTRX(AT   ,IELDOF,6,' AT   ',24)
              CALL MMATRX(AC   , 6,3,' AC   ')
              CALL MMTRX(A    ,IELDOF,3,' A    ',24)
         ENDIF
C
C----- GET SPRING STIFFNESS
         FALSE=.FALSE.
         LSTYP=0
         CALL MATLIB
     & ( 2    ,LSTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEB,PSEB,DUCT(1,1),EXCR(1,1),
     &   FT   ,F    ,RT   ,R    ,0.   ,0.   ,LELEM ,LMAT  ,
     &   MATB  ,MATYP,SEC(ISEB),IELNO,DAMAGE)
         CALL MATLIB
     & ( 2    ,LSTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESES,PSES,DUCT(1,2),EXCR(1,2),
     &   FT   ,F    ,RT   ,R    ,0.   ,0.   ,LELEM ,LMAT  ,
     &   MATS  ,MATYP,SEC(ISES),IELNO,DAMAGE)
         CALL MATLIB
     & ( 2    ,LSTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEA,PSEA,DUCT(1,3),EXCR(1,3),
     &   FT   ,F    ,RT   ,R    ,0.   ,0.   ,LELEM ,LMAT  ,
```

```
     &   MATA  ,MATTYP,SE(ISEA),IELNO,DAMAGE)
C
C
      ESE=0.
      PSE=0.
C
  111 FORMAT (OP,' ZNATRX(',I6,',):',I15,1P,G20.10)
C
C------- SET UP THE LOCAL STIFFNESS MATRIX
      SB= SE(ISEB)
      SS= SE(ISES)
      SA= SE(ISEA)
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'BENDING STIFFNESS KB=',SB
        WRITE (6,*) 'SHEAR   STIFFNESS KS=',SS
        WRITE (6,*) 'AXIAL   STIFFNESS KA=',SA
      ENDIF
C
C------ CALCULATE SAT
      SBL=SB/LENGTH
      SSL=SS/LENGTH
      SAL=SA/LENGTH.
      DO 60 J=1,IELDOF
        SAT(1,J)=SBL*AC(J,1)
        SAT(2,J)=SSL*AC(J,2)
   60   SAT(3,J)=SAL*AC(J,3)
C
C------ CALCULATE ASAT AND CONVERT TO HALF STORAGE MODE  BY COLUMNS
C
C         1  3  6 10 15 21 28 36 45 55 66 78    300.  NUMBER INDICATES
C            2  5  9 14 20 27 35 44 54 65 77      .   LOCATION OF DATA
C               4  8 13 19 26 34 43 53 64 76      .   IN ARRAY STIFF
C                  7 12 18 25 33 42 52 63 75      .
C                   11 17 24 32 41 51 62 74       .
C                      16 23 31 40 50 61 73       .
C                         22 30 39 49 60 72       .
C                            29 38 48 59 71       .
C                               37 47 58 70       .
C                                  46 57 69       .
C                                     56 68       .
C                                        67       .
C                                           .     .
C                                           .     .
C                                           .     .
      L=0
      DO 70 J=1,IELDOF
        DO 70 I=J,1,-1
        L=L+1
        DSTIFF(L)=0
        DO 70 K=1,5
   70     DSTIFF(L)=DSTIFF(L)+AC(I,K)*SAT(K,J)
      DO 71 J=1,L
   71     STIFF(J)=DSTIFF(J)
C
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'IELDOF',IELDOF,'LM =',(LM(I),I=1,IELDOF)
        CALL LMATRX(STIFF,MD,IELDOF,(L+1),'HALL STIFFNESS')
      ENDIF
C
C-----------------------------------------------------------------------
C  ASSEMBLE GEOMETRIC STIFFNESS MATRIX FOR A UNIT LOAD...
C-----------------------------------------------------------------------
      LKG=L+1
      IF ( PKG.EQ.0 .AND. KGDATA(1).NE.2 ) GO TO 180
      IF (KGDATA(1).EQ.0 .OR. KGDATA(2).EQ.0 ) GO TO 180
      AXIAL=.TRUE.
C
C---------- DETERMINE THE AXIAL LOAD TO BE USED FOR KG...
        IF (KGDATA(1).EQ.1) THEN
            PGEOM =  PKG
        ELSE IF (KGDATA(1).EQ.2) THEN
            PGEOM = -F(3)
        ENDIF
C
C----- ASSEMBLE TRANSFORMATION MATRIX A
C  NOTE: AT IS DIMMENSIONED AS A 12X6 MATRIX, BUT ONLY THE FIRST IELDOF
C        ROW'S CONTAIN INFORMATION.  THE BALANCE OF THE MATRIX IS
C        UNDEFINED.  THE GLOBAL DOF COORESPONDING TO THE ROW'S OF A
C        ARE STORED IN VECTOR LM.  AGAIN, ONLY THE FIRST IELDOF ROW'S
C        OF LM ARE DEFINED.
      DO 140 I=1,24
      DO 140 K=1,6
  140   A(I,K)=0.00
      KGDOF=0
      DO 145 I=1,6
      DO 145 J=1,6
C
        KGDOF =KGDOF +1
        IF (I.EQ.1) THEN
            LMTI=IDOF(J1,J)
            A(KGDOF,1)=A11(J,1)
            A(KGDOF,2)=A11(J,5)
        ELSE IF (I.EQ.2) THEN
            LMTI=IDOF(J2,J)
            A(KGDOF,3)=A22(J,3)
        ELSE IF (I.EQ.5) THEN
            LMTI=IDOF(J5,J)
            A(KGDOF,4)=A55(J,5)
        ELSE IF (I.EQ.6) THEN
            LMTI=IDOF(J6,J)
            A(KGDOF,5)=A66(J,1)
            A(KGDOF,6)=A66(J,5)
        ENDIF
        LM(KGDOF)=LMTI
        IF((A(KGDOF,1).EQ.0).AND.(A(KGDOF,2).EQ.0).AND.
     &     (A(KGDOF,3).EQ.0).AND.(A(KGDOF,4).EQ.0).AND.
     &     (A(KGDOF,5).EQ.0).AND.(A(KGDOF,6).EQ.0)) THEN
            KGDOF=KGDOF-1
        ELSE IF (KGDOF.GT.1) THEN
           DO 149  II=1,KGDOF-1
            IF (LM(II).EQ.LM(KGDOF)) THEN
              DO 148 JJ=1,6
                A(II,JJ)=A(II,JJ)+A(KGDOF,JJ)
  148           A(KGDOF,JJ)=0
              KGDOF=KGDOF-1
              GO TO 145
            ENDIF
  149     CONTINUE
        ENDIF
C
  145   CONTINUE
C
C------ LOCAL GEOMETRIC STIFFNESS MATRIX
      DO 150 I2=1,12
      DO 150 JI=1,12
  150   SC(I2,JI)=0.0
        SC(1,1)=  1.0/LENGTH
        SC(2,2)=  0.5/LENGTH
        SC(3,3)=  0.5/LENGTH
        SC(4,4)=  0.5/LENGTH
        SC(5,5)=  1.0/LENGTH
        SC(6,6)=  0.5/LENGTH
        SC(1,5)= -1.0/LENGTH
        SC(2,6)= -0.5/LENGTH
        SC(3,4)= -0.5/LENGTH
        SC(5,1)= -1.0/LENGTH
        SC(6,2)= -0.5/LENGTH
        SC(4,3)= -0.5/LENGTH
C
C------ CALCULATE SAT
        DO 160 I=1,6
          DO 160 J=1,KGDOF
          SAT(I,J)=0
          DO 160 K=1,6
  160       SAT(I,J)=SAT(I,J)+SC(I,K)*AC(J,K)
C
C------ CALCULATE ASAT AND CONVERT TO HALF STORAGE MODE  BY COLUMNS
        DO 170 J=1,KGDOF
          DO 170 I=J,1,-1
          L=L+1
          STIFF(L)=0
          DO 170 K=1,6
  170       STIFF(L)=STIFF(L)+AC(I,K)*SAT(K,J)
C
C------ PRINT GEOMETRIC STIFF DATA FOR DEB_UGGING
        IF ( BTEST(IBUG,7) ) THEN
          CALL HMTRX2C (AT   ,KGDOF,6,' AT   ',24)
          CALL HMATRX( S     , 6,6,' KG-S ')
          CALL HMATRX(SAT2   ,6,KGDOF,'SAT2  ')
          WRITE (6,*) 'KGDOF',KGDOF,'LMKG =',(LMKG(I),I=1,KGDOF)
          CALL LMATRX(STIFF(LKG),MD,KGDOF,300,'HALL GEOMETRIC STIFF.')
        ENDIF
C
C------ RELEASE UNUSED STORAGE
  180 IREL=600-L
C
C     PRINT COLUMN DATA
      IF (FIRST .OR. BTEST(IBUG,7) .OR. BTEST(IBUG,8) ) WRITE (6,192)
  191 WRITE(6,195) NAME,IELNO,MATB,MATS,MATA,
     &           IDX(1),IDX(2),IDX(5),IDX(4),
     &     LENGTH,W,ALPIN,PKG
C
  192 FORMAT(/' ELEMENT 05, R/C SHEAR WALL ELEMENT'// T17,
     &     '  ELEM',' BEND',' SHEAR',' AXIAL',' JOINT',' JOINT',
     &     ' JOINT',' JOINT',    LENGTH   ','    WIDTH   ',
     &     '    ALPHA    ','      PKG      ','/,T17,
     &     '     a',' MATL',' MATL',' MATL',     a1',     a2',
     &     '    a5',     a4'//)
C
  195 FORMAT(1X,A15,8I6,2X,1P,4G12.4)
      IOPT=IOPT
      RETURN
C========================== DETERMINE GEOMETRIC STIFF ==================
  200 CONTINUE
C- KG IS DETERMINED FOR A UNIT LOAD WITH IOPT=1,
C- THE ACTUAL LOAD (PGEOM) IS DETERMINED HERE.
C- KG IS MULTIPLIED BY PGEOM WHEN THE TOTAL STIFFNESS IS ASSEMBLED
C- NEGATIVE PGEOM IS COMPRESSION....
C
      AXIAL=.TRUE.
C---------- DETERMINE THE AXIAL LOAD TO BE USED FOR KG...
        IF (KGDATA(1).EQ.1) THEN
            PGEOM =  PKG
        ELSE IF (KGDATA(1).EQ.2) THEN
            PGEOM = -F(3)
        ENDIF
        IF (BTEST(IBUG,7)) WRITE (6,*) 'AXIAL ',AXIAL,'  PGEOM:',PGEOM
C
      RETURN
C
C========================== DETERMINE STIFFNESS ========================
  300 CONTINUE
      SB=SB
      SS=SS
      SA=SA
      IF (BTEST(IBUG,7)) WRITE (6,402) IELNO,SB,SS,SA
      IF (BTEST(IBUG,7)) WRITE (6,402) IELNO,SE(ISEB),SE(ISES),SE(ISEA)
C
C------- SB, SS AND SA CONTAIN THE STIFFNESSES. WHICH WERE DETERMINED IN
C        THE LAST CALL TO MA_TLIB
C        IF SB=SE(ISEB) AND SS=SE(ISES) AND SA=SE(ISEA)
C        THEN THE STIFFNESS HAS NOT CHANGED, RETURN
C
      IF (SB.EQ.SE(ISEB).AND.SS.EQ.SE(ISES).AND.SA.EQ.SE(ISEA)) RETURN
C
C------- SET UP THE LOCAL STIFFNESS MATRIX
      SB= SE(ISEB)
      SS= SE(ISES)
      SA= SE(ISEA)
      IF (BTEST(IBUG,7)) WRITE (6,402) IELNO,SB,SS,SA
      SBL= SE(ISEB)/LENGTH
      SSL= SE(ISES)/LENGTH
      SAL= SE(ISEA)/LENGTH
C
C------ CALCULATE SAT
      DO 560 J=1,IELDOF
        SAT(1,J)=SBL*AC(J,1)
        SAT(2,J)=SSL*AC(J,2)
  560   SAT(3,J)=SAL*AC(J,3)
C
C------ CALCULATE ASAT, STORE IN UPPER TRIANGULAR MATRIX
      L=0
      DO 570 J=1,IELDOF
        DO 570 I=J,1,-1
        L=L+1
        DSTIFF(L)=0
        DO 570 K=1,5
  570     DSTIFF(L)=DSTIFF(L)+AC(I,K)*SAT(K,J)
      DO 571 J=1,L
  571     STIFF(J)=DSTIFF(J)
C
      IF ( BTEST(IBUG,7) ) THEN
        WRITE (6,*) 'IELDOF',IELDOF,'LM =',(LM(I),I=1,IELDOF)
        CALL LMATRX(STIFF,MD,IELDOF,(L+1),'HALL STIFFNESS')
      ENDIF
C
      RETURN
C========================== DETERMINE INCREMENTAL FORCES ===============
```

```
  400 CONTINUE                                                          ELE06100
      IF (PRINT.AND.FIRST) WRITE (6,491) STEPID                         ELE06110
      IF (BTEST(IBUG,7)) THEN                                           ELE06120
        WRITE (6,401) IELNO                                             ELE06130
  401   FORMAT (1X,30('='),'SHEAR WALL, ELEMENT#',I6,1X,30('=') )       ELE06140
      ENDIF                                                             ELE06150
C                                                                       ELE06160
C------- SET UP THE COMPONENT STIFFNESSES                               ELE06170
      SB= SE(ISEB)                                                      ELE06180
      SS= SE(ISES)                                                      ELE06190
      SA= SE(ISEA)                                                      ELE06200
      IF (BTEST(IBUG,7)) WRITE (6,402) IELNO,SB,SS,SA                   ELE06210
  402 FORMAT(' IELNO:',I1,' SB:',1P,G12.5,' SS:',G12.5,' SA:',G12.5)    ELE06220
C                                                                       ELE06230
C------- LOOP FOR EACH LOAD                                            ELE06240
      DO 450 LOAD=1,NLOAD                                               ELE06250
C                                                                       ELE06260
C------- TRANSFORM FRAME DISPL INTO LOCAL DISPL                         ELE06270
      DO 410 I=1,5                                                      ELE06280
        R(I)=0                                                          ELE06290
        V(I)=0                                                          ELE06300
        DO 410 J=1,IELDOF                                               ELE06310
          K=LM(J)                                                       ELE06320
          R(I)=R(I)+A(J,I)*DISPL(K,LOAD)                                ELE06330
  410     V(I)=V(I)+A(J,I)*VELOC(K)                                     ELE06340
C                                                                       ELE06350
C------- INCRIMENTAL FORCES                                            ELE06360
      F(1)=SB*R(1)/LENGTH                                               ELE06370
      F(2)=SS*R(2)/LENGTH                                               ELE06380
      F(5)=SA*R(5)/LENGTH                                               ELE06390
C                                                                       ELE06400
C------- CHECK STIFFNESS AND CALCULATE UNBALANCED FORCES                ELE06410
C         THIS IS PREFORMED ONLY FOR INELASTIC ANALYSIS                 ELE06420
      P(1) =FT(1)+F(1)                                                  ELE06430
      P(2) =FT(2)+F(2)                                                  ELE06440
      P(5) =FT(5)+F(5)                                                  ELE06450
      IF (.NOT. ELSTIC) THEN                                            ELE06460
C         -- TRANSFER PERMANENT HYSTERESIS DATA TO TEMPROARY STORAGE...  ELE06470
          ISE2=ISE/2                                                    ELE06480
          DO 415 I=1,ISE2                                               ELE06490
  415       SE(I+ISE2)=SE(I)                                            ELE06500
C         -- BENDING ----------------                                   ELE06510
C         -- SET UP TEMPORARY - TOTAL LOADS, DISPL'S AND VELOC          ELE06520
          PL=F(1)                                                       ELE06530
          D =(RT(1)+R(1))/LENGTH                                        ELE06540
          DL=RT(1) /LENGTH                                              ELE06550
          VC=VT(1)+V(1)                                                 ELE06560
          VL=VT(1)                                                      ELE06570
C         -- CALL HYST MODEL TO GET NEW STIFFNESS AND                    ELE06580
C             THE TOTAL FORCE P AT DISPL D.                             ELE06590
          MOPT=5                                                        ELE06600
          CALL MATLIB                                                   ELE06610
     &     (MOPT,LSTTYP,FALSE ,IREL,FALSE,NAME,                         ELE06620
     &    ESEB,PSEB,DUCT(1,1),EXCR(1,1),                                ELE06630
     &      P(1),PL,D,DL,VC,VL,LELEM ,LMAT  ,                           ELE06640
     &      MATB,MATTYP,SE(ISE2+ISEB),IELNO,DAMAGE)                     ELE06650
          IF (SE(ISE2+ISEB).NE.SE(ISEB)) MEMK=.TRUE.                    ELE06660
C         -- SHEAR ----------------                                     ELE06670
C         -- SET UP TEMPORARY - TOTAL LOADS, DISPL'S AND VELOC          ELE06680
          PL=F(2)                                                       ELE06690
          D =(RT(2)+R(2))/LENGTH                                        ELE06700
          DL=RT(2) /LENGTH                                              ELE06710
          VC=VT(2)+V(2)                                                 ELE06720
          VL=VT(2)                                                      ELE06730
C         -- CALL HYST MODEL TO GET NEW STIFFNESS AND                    ELE06740
C             THE TOTAL FORCE P AT DISPL D.                             ELE06750
          MOPT=5                                                        ELE06760
          CALL MATLIB                                                   ELE06770
     &     (MOPT,LSTTYP,FALSE ,IREL,FALSE,NAME,                         ELE06780
     &    ESES,PSES,DUCT(1,2),EXCR(1,2),                                ELE06790
     &      P(2),PL,D,DL,VC,VL,LELEM ,LMAT  ,                           ELE06800
     &      MATS,MATTYP,SE(ISE2+ISES),IELNO,DAMAGE)                     ELE06810
          IF (SE(ISE2+ISES).NE.SE(ISES)) MEMK=.TRUE.                    ELE06820
C         -- AXIAL ----------------                                     ELE06830
C         -- SET UP TEMPORARY - TOTAL LOADS, DISPL'S AND VELOC          ELE06840
          PL=FT(5)                                                      ELE06850
          D =(RT(5)+R(5))/LENGTH                                        ELE06860
          DL=RT(5) /LENGTH                                              ELE06870
          VC=VT(5)+V(5)                                                 ELE06880
          VL=VT(5)                                                      ELE06890
C         -- CALL HYST MODEL TO GET NEW STIFFNESS AND                    ELE06900
C             THE TOTAL FORCE P AT DISPL D.                             ELE06910
          MOPT=5                                                        ELE06920
          CALL MATLIB                                                   ELE06930
     &     (MOPT,LSTTYP,FALSE ,IREL,FALSE,NAME,                         ELE06940
     &    ESEA,PSEA,DUCT(1,5),EXCR(1,5),                                ELE06950
     &      P(5),PL,D,DL,VC,VL,LELEM ,LMAT  ,                           ELE06960
     &      MATA  ,MATTYP,SE(ISE2+ISEA),IELNO,DAMAGE)                   ELE06970
          IF (SE(ISE2+ISEA).NE.SE(ISEA)) MEMK=.TRUE.                    ELE06980
      ENDIF                                                             ELE06990
C                                                                       ELE07000
      IF (ELSTIC) THEN                                                  ELE07010
        ESEB=0.50*(FT(1)+F(1))*(RT(1)+R(1))                             ELE07020
        ESES=0.50*(FT(2)+F(2))*(RT(2)+R(2))                             ELE07030
        ESEA=0.50*(FT(5)+F(5))*(RT(5)+R(5))                             ELE07040
        ESE= ESEB+ESES+ESEA                                             ELE07050
        PSE=0.0                                                         ELE07060
      ELSE                                                              ELE07070
        ESE=(ESEB+ESES+ESEA)*LENGTH                                     ELE07080
        PSE=(PSEB+PSES+PSEA)*LENGTH                                     ELE07090
      ENDIF                                                             ELE07100
C                                                                       ELE07110
C------- SAVE MAXIMUM LOADS                                            ELE07120
      IF (NLOAD.GT.1) THEN                                              ELE07130
        DO 589 I=1,5                                                    ELE07140
          IF (ABS(F(I)).GT.ABS(FMAX(I)) ) THEN                          ELE07150
            FMAX(I  )= F(I)                                             ELE07160
            FMAX(I+5)= R(I)                                             ELE07170
          ENDIF                                                         ELE07180
  589   CONTINUE                                                        ELE07190
      ENDIF                                                             ELE07200
C                                                                       ELE07210
C------- PRINT DATA                                                     ELE07220
      IF (PRINT) WRITE (6,592) IELNO,LOAD,                              ELE07230
     &           (IDX(J1),IDX(J2),IDX(J5),IDX(J6),                      ELE07240
     &                                    (F(I),R(I),I=1,5)             ELE07250
  592 FORMAT (1I0,I5,2X,4I5,1P,6G15.6)                                  ELE07260
C                                                                       ELE07270
C------- CALCULATE THE UNBALANCED MEMBER FORCE ON A JOINT.              ELE07280
      IF (LOAD.EQ.1) THEN                                               ELE07290
            UBM = -( F(1)+FT(1)-P(1) )                                  ELE07300
            UBS = -( F(2)+FT(2)-P(2) )                                  ELE07310
            UBA = -( F(5)+FT(5)-P(5) )                                  ELE07320
            DO 440 I=1,IELDOF                                           ELE07330
            K=LM(I)                                                     ELE07340
            EFUB(I)= A(I,1)*UBM + A(I,2)*UBS + A(I,5)*UBA               ELE07350
  440       FUB(K)= FUB(K) + EFUB(I)                                    ELE07360
          IF (BTEST(IBUG,7)) THEN                                       ELE07370
            WRITE (6,493) (I,F(I),FT(I),P(I),I=1,5)                     ELE07380
            WRITE (6,*) 'UB: M,S,A =',UBM,UBS,UBA                       ELE07390
            WRITE (6,494) (I,LM(I),EFUB(I),I=1,IELDOF)                  ELE07400
          ENDIF                                                         ELE07410
      ENDIF                                                             ELE07420
      F(1)=P(1)-FT(1)                                                   ELE07430
      F(2)=P(2)-FT(2)                                                   ELE07440
      F(5)=P(5)-FT(5)                                                   ELE07450
  450 CONTINUE                                                          ELE07460
C                                                                       ELE07470
  491 FORMAT (/'    R/C SHEAR WALL FORCES...',5X,A/                     ELE07480
     & '  ELEMENT LOAD',T17,                                            ELE07490
     & ' JNT-1',' JNT-2',' JNT-3',' JNT-4',                             ELE07500
     & '    MOMENT      ',' ROTATION    ',                             ELE07510
     & '     SHEAR      ',' SHEAR DISPL.  ',                           ELE07520
     & '     AXIAL      ',' AXIAL DISPL.  '/)                          ELE07530
  492 FORMAT (I10,I5,2X,4I5,1P,6G15.6)                                  ELE07540
  493 FORMAT (' I:',I1,'  F:',1P,G12.4,'  FT:',G12.4,'  P:',G12.4)      ELE07550
  494 FORMAT (' I:',I5,'  LM(I):',I6,'  FUB(I):',1P,G12.4)              ELE07560
C                                                                       ELE07570
      IOPT=IOPT                                                         ELE07580
      RETURN                                                            ELE07590
C========================== DETERMINE TOTAL FORCES, ENERGIES ECT ========ELE07600
  500 CONTINUE                                                          ELE07610
       IF (PRINT.AND.FIRST) WRITE (6,491) STEPID                        ELE07620
C                                                                       ELE07630
C------- TOTAL FORCES, DISPLACEMENTS AND VELOCITIES                     ELE07640
      DO 510 I=1,5                                                      ELE07650
      FT(I) = FT(I) + F(I)                                              ELE07660
      RT(I) = RT(I) + R(I)                                              ELE07670
      VT(I) = VT(I) + V(I)                                              ELE07680
      IF ( ABS(FT(I)).GT.ABS(FMAX(I)) ) THEN                            ELE07690
        FMAX(I  )=FT(I)                                                 ELE07700
        FMAX(I+5)=RT(I)                                                 ELE07710
      ENDIF                                                             ELE07720
  510 CONTINUE                                                          ELE07730
C                                                                       ELE07740
C------- PRINT DATA                                                     ELE07750
      IF (PRINT) WRITE (6,492) IELNO,LOAD,                              ELE07760
     &           (IDX(J1),IDX(J2),IDX(J5),IDX(J6),                      ELE07770
     &                                    (FT(I),RT(I),I=1,5)           ELE07780
C                                                                       ELE07790
C------- TRANSFER TEMPROARY HYSTERESIS DATA TO PERMANENT STORAGE...     ELE07800
      ISE2=ISE/2                                                        ELE07810
      IF (.NOT.ELSTIC) THEN                                             ELE07820
        DO 520 I=1,ISE2                                                 ELE07830
  520     SE(I)=SE(I+ISE2)                                              ELE07840
      ENDIF                                                             ELE07850
C                                                                       ELE07860
      RETURN                                                            ELE07870
C                                                                       ELE07880
C========================== DETERMINE FIXED END FORCES, AND ADD TO FORCE=ELE07890
  600 RETURN                                                            ELE07900
C                                                                       ELE07910
C------- FIXED END FORCES ARE NOT AVAILABLE FOR R/C SHEAR WALL          ELE07920
C                                                                       ELE07950
C========================== WRITE MEMBER FORCES TO OUTPUT FILE ==========ELE07970
  700 CONTINUE                                                          ELE07950
C                                                                       ELE07960
C.......... BENDING ENERGY, DUCTILITY AND EXCURSION .....              ELE07970
      IF (ELSTIC) THEN                                                  ELE07980
        ESEB=0.50*(FT(1)+F(1))*(RT(1)+R(1))                             ELE07990
        ESES=0.50*(FT(2)+F(2))*(RT(2)+R(2))                             ELE08000
        ESEA=0.50*(FT(5)+F(5))*(RT(5)+R(5))                             ELE08010
        ESE= ESEB+ESES+ESEA                                             ELE08020
        PSE=0.0                                                         ELE08030
      ELSE                                                              ELE08040
        CALL MATLIB                                                     ELE08050
     &  ( 4    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,                         ELE08060
     &    ESEB,PSEB,DUCT(1,1),EXCR(1,1),                               ELE08070
     &    FT    ,F     ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,     ELE08080
     &    MATB  ,MATTYP,SE(ISEB),IELNO,DAMAGE)                         ELE08090
C.......... SHEAR ENERGY, DUCTILITY AND EXCURSION .....                ELE08100
        CALL MATLIB                                                     ELE08110
     &  ( 4    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,                         ELE08120
     &    ESES,PSES,DUCT(1,2),EXCR(1,2),                               ELE08130
     &    FT    ,F     ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,     ELE08140
     &    MATS  ,MATTYP,SE(ISES),IELNO,DAMAGE)                         ELE08150
C.......... AXIAL ENERGY, DUCTILITY AND EXCURSION .....                ELE08160
        CALL MATLIB                                                     ELE08170
     &  ( 4    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,                         ELE08180
     &    ESEA,PSEA,DUCT(1,5),EXCR(1,5),                               ELE08190
     &    FT    ,F     ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,     ELE08200
     &    MATA  ,MATTYP,SE(ISEA),IELNO,DAMAGE)                         ELE08210
C                                                                       ELE08220
        ESE=(ESEB+ESES+ESEA)*LENGTH                                     ELE08230
        PSE=(PSEB+PSES+PSEA)*LENGTH                                     ELE08240
      ENDIF                                                             ELE08250
C                                                                       ELE08260
      IF (PRINT) WRITE (IUNIT,710) IDX(J1),IDX(J2),IDX(J5),IDX(J6)      ELE08270
      WRITE (IUNIT,711) (FT(I),RT(I),I=1,5),ESE,PSE                     ELE08280
  710 FORMAT (4I5,1P,6E10.5/(8E10.5) )                                  ELE08290
  711 FORMAT (1P,8E10.5)                                                ELE08300
C                                                                       ELE08310
      RETURN                                                            ELE08320
C                                                                       ELE08330
C========================== READ AND PRINT MEMBER FORCES FROM OUTPUT FILE ====ELE08350
  800 CONTINUE                                                          ELE08360
      SX =0                                                             ELE08370
      SXX=0                                                             ELE08380
      SV =0                                                             ELE08390
      SXV=0                                                             ELE08400
      SN =0                                                             ELE08410
      BACKSPACE(IUNIT)                                                  ELE08420
      READ (IUNIT,*  ) ITYPE,IELNO,INRITE,TO,DT                        ELE08430
      READ (IUNIT,710) NODE1,NODE2,NODE3,NODE4                          ELE08440
      ISTEP=-INRITE                                                     ELE08450
      AVEMV=0                                                           ELE08460
      ICOUNT=0                                                          ELE08470
C                                                                       ELE08480
  810 READ (IUNIT,815,END=850) (F(I),R(I),I=1,5),ESE,PSE                ELE08490
  815 FORMAT (8E10.5)                                                   ELE08500
        ISTEP=ISTEP+INRITE                                              ELE08510
        T=TO+DT*ISTEP                                                   ELE08520
        IF (ISTEP.EQ.0) WRITE (6,805) NODE1,NODE2,NODE3,NODE4           ELE08530
        IF (MOD(ISTEP,INC).EQ.0)                                        ELE08540
     &  WRITE (6,820) ISTEP,T,(F(I),R(I),I=1,5),ESE,PSE                 ELE08550
        SX  = SX  + F(2)                                                ELE08560
        SXX = SXX + F(2)*F(2)                                           ELE08570
```

```
         SV  = SV  + FC1)
         SXV = SXV + FC1)*FC2)
         SN  = SN  + 1
         IF (FC2).NE.0) THEN
              RATHV= FC1)/FC2)
                AVEMV=AVEMV+RATHV
                ICOUNT=ICOUNT+1
         ENDIF
         GO TO 810
C
  850 CONTINUE
      AVEMV=AVEMV/ MAXCICOUNT,1)
      WRITE (6,851) AVEMV,ICOUNT
      IF (SV.NE.0 ) THEN
        RM=SV/SX
        ICOUNT=SN
        WRITE (6,855) RM,ICOUNT
      ENDIF
      DM = SN*SXX - SX*SX
      DB = SX*SX  - SXX*SN
      IF (DM.NE.0 .AND. DB.NE.0) THEN
        RB = ( SXV*SX - SV*SXX ) / DB
        RM = ( SN*SXV - SX*SV  ) / DM
        ICOUNT=SN
        WRITE (6,852) RM,RB,ICOUNT
      ENDIF
  851 FORMAT(/T10,'AVERAGE MOMENT/SHEAR RATIO=',1P,G12.4,5X,OP,I6,
     & ' POINTS WERE USED. ',
     & ' POINTS WITH V=0 ARE EXCLUDED')
  852 FORMAT( T10,'REGRESSION MOMENT-SHEAR EQUN:   ',
     &          ' M =', 1P,G12.4, '* V +', G12.4,5X,
     & OP,I6,' POINTS WERE USED')
  853 FORMAT( T10,'REGRESSION MOMENT-SHEAR EQUN:   ',
     &          ' M =', 1P,G12.4, '* V  ', 12X  ,5X,
     & OP,I6,' POINTS WERE USED')


C
C
  805 FORMAT (/'    R/C SHEAR WALL FORCES...'./
     & 5X,'NODE1:',I6,5X,'NODE2:',I6,5X,'NODE3:',I6,5X,'NODE4:',I6,//
     &     ' STEP    TIME   ',
     & '    MOMENT      ','   ROTATION   ',
     & '     SHEAR      ',' SHEAR DISPL. ',
     & '     AXIAL      ',' AXIAL DISPL. ',
     & '   ESE          ','   PSE        '/)
  820 FORMAT (I6.1P,G12.4,5G14.5,2G12.4)
  835 FORMAT (/T10,' MAXIMUN DUCTILITIES AND EXCURSION RATIOS'/
     &        T10,'=======================================')
  840 FORMAT (/T10,A/
     &  T15,'DISPLACEMENT DEFINATION: U1=',1P,G12.4,5X,'E1=',G12.4/
     &  T15,'ENERGY DEFINATION #1:    U2=',1P,G12.4,5X,'E2=',G12.4/
     &  T15,'ENERGY DEFINATION #2:    U3=',1P,G12.4,5X,'E3=',G12.4)
      RETURN
C
C==================== DETERMINE THE LENGTH OF THE MATERIAL DATA
  900 CONTINUE
      ISE=0
C------- LOOP FOR EACH MATEIAL -------
      DO 910 I=1,3
        IF (I.EQ.1) THEN
          ISEB=ISE+1
        ELSE IF (I.EQ.2) THEN
          ISES=ISE+1
        ELSE IF (I.EQ.3) THEN
          ISEA=ISE+1
        ENDIF
        MATI = RINPUT(I)
        CALL MATLIB
     & ( 0   ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESE, PSE,DUCT(1,2),EXCR(1,2),
     &  FT   ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &  MATI ,MATTYP,SE,IELNO,DAMAGE)
  910   ISE=ISE + LELEM
C
C------- DOUBLE THE STORAGE, TO ALLOW FOR TEMPROARY VALUES...
        ISE=ISE*2
C
      RETURN
C
C==================== DETERMINE THE STRAIN ENERGY
 1000 CONTINUE
C==================== DUCTILITEIS AND EXCURSION RATIOS
 1100 CONTINUE
C      ... NOTE ELEOS WAS LAST CALLED TO CALC INCIRMENTAL DISPL.
C               THE ENERGIES ASSOCIATED WITH INCR DISPL ARE STORED IN
C               THE ADDRESSES: ISEB+ISE/2
C                              ISES+ISE/2
C                              ISEA+ISE/2
C.......... BENDING ENERGY, DUCTILITY AND EXCURSION .....
        ISE2=ISE/2
        CALL MATLIB
     & ( 4   ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEB,PSEB,DUCT(1,1),EXCR(1,1),
     &   FT   ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MATB  ,MATTYP,SE(ISEB+ISE2),IELNO,DAMAGE)
C.......... SHEAR ENERGY, DUCTILITY AND EXCURSION .....
        CALL MATLIB
     & ( 4   ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESES,PSES,DUCT(1,2),EXCR(1,2),
     &   FT   ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MATS  ,MATTYP,SE(ISES+ISE2),IELNO,DAMAGE)
C.......... AXIAL ENERGY, DUCTILITY AND EXCURSION .....
        CALL MATLIB
     & ( 4   ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEA,PSEA,DUCT(1,3),EXCR(1,3),
     &   FT   ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MATA  ,MATTYP,SE(ISEA+ISE2),IELNO,DAMAGE)
C
        IF (ELSTIC) THEN
           ESEB=0.50*(FT(1)+F(1))*(RT(1)+R(1))
           ESES=0.50*(FT(2)+F(2))*(RT(2)+R(2))
           ESEA=0.50*(FT(3)+F(3))*(RT(3)+R(3))
           ESE= ESEB+ESES+ESEA
           PSE=0.0
        ELSE
           ESE=(ESEB+ESES+ESEA)*LENGTH
           PSE=(PSEB+PSES+PSEA)*LENGTH
        ENDIF
C
        IF (BTEST(IBUG,7)) THEN
            WRITE (6,*) 'ESEB ESES ESEA  ',ESEB,ESES,ESEA
            WRITE (6,*) 'PSEB PSES PSEA  ',PSEB,PSES,PSEA
            WRITE (6,*) 'ESE  PSE  LENGTH',ESE ,PSE ,LENGTH
        ENDIF
        IF (PRINT.AND.FIRST) WRITE (6,1191)
      IF (PRINT) WRITE (6,1192)IELNO,((DUCT(I,J),EXCR(I,J),I=1,5),J=1,5)
 1192 FORMAT (1X,I5,18F7.2)
 1191 FORMAT (/'    R/C SHEAR WALL DUCTILITY AND EXCURSION RATIOS'/6X,
     &  ' |-------- BENDING COMPONENT ----------|',
     &  ' |---------- SHEAR COMPONENT ----------|',
     &  ' |---------- AXIAL COMPONENT ----------|'/,' ELEM#',
     &  ' |-- DEFN 1---  --- DEFN 2---  --- DEFN 5--|',
     &  ' |-- DEFN 1---  --- DEFN 2---  --- DEFN 5--|'/6X,
     &  '  DUCT.  EXCR.  DUCT.  EXCR.  DUCT.  EXCR.',
     &  '  DUCT.  EXCR.  DUCT.  EXCR.  DUCT.  EXCR.',
     &  '  DUCT.  EXCR.  DUCT.  EXCR.  DUCT.  EXCR.')
      RETURN
C
C==================== WRITE MAXIMUM AND MINIMUM VALUES
 1200 CONTINUE
      IF (FIRST) WRITE(6,491)'MAXIMUM LOADS AND DISPL AT MAXIMUM LOADS '
     &          //' NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'
     &          WRITE (6,492) IELNO,LOAD,
     &          IDCJ1),IDCJ2),IDCJ3),IDCJ4),
     &             (FMAXCI),FMAXC5+I),I=1,5)
C
      RETURN
C
C==================== RESET INTERNAL FORCES
 1500 CONTINUE
C
C------ ZERO MATRICIES
      DO 1510 I=1,ISE
 1510   SECI)=0
      DO 1525 I=1,5
        FMAXCI)=0
        FMAXCI+5)=0
        F CI)=0
        FTCI)=0
        R CI)=0
        RTCI)=0
        V CI)=0
 1525   VTCI)=0
      WRITE (6,1550) IELNO
 1550 FORMAT (' SHEAR WALL..... ELEMENT #',I6,
     & '  INTERNAL FORCES AND HYSTERESIS MODELS ARE RESET TO ZERO')

        CALL MATLIB
     & ( 2    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEB,PSEB,DUCT(1,1),EXCR(1,1),
     &  FT    ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &  MATB  ,MATTYP,SE(ISEB),IELNO,DAMAGE)
      CALL MATLIB
     & ( 2    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESES,PSES,DUCT(1,2),EXCR(1,2),
     &  FT    ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &  MATS  ,MATTYP,SE(ISES),IELNO,DAMAGE)
      CALL MATLIB
     & ( 2    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEA,PSEA,DUCT(1,3),EXCR(1,3),
     &  FT    ,F      ,RT    ,R     ,0.    ,0.    ,LELEM ,LMAT  ,
     &  MATA  ,MATTYP,SE(ISEA),IELNO,DAMAGE)
      RETURN
C==================== DAMAGE INDEX
 1400 IF (ELSTIC) RETURN
      FMAX1=FMAXCI)
      FMAX2=FMAXC2)
      FMAX3=FMAXC3)
      FMAX4=FMAXC4)
      FMAX5=FMAXC5)
      FMAX6=FMAXC6)
C.......... BENDING ENERGY, AND DAMAGE INDEX
        CALL MATLIB
     & ( 5    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEB,PSEB,DUCT(1,1),EXCR(1,1),
     &   FMAX1,0.     ,FMAX4,0.     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MATB  ,MATTYP,SE(ISEB),IELNO,DAMB)
C.......... BENDING ENERGY, AND DAMAGE INDEX
        CALL MATLIB
     & ( 5    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESES,PSES,DUCT(1,2),EXCR(1,2),
     &   FMAX2,0.     ,FMAX5,0.     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MATS  ,MATTYP,SE(ISES),IELNO,DAMS)
C.......... AXIAL ENERGY, AND DAMAGE INDEX
        CALL MATLIB
     & ( 5    ,LSTTYP ,FALSE ,IREL,FALSE,NAME,
     &   ESEA,PSEA,DUCT(1,3),EXCR(1,3),
     &   FMAX3,0.     ,FMAX6,0.     ,0.    ,0.    ,LELEM ,LMAT  ,
     &   MATA  ,MATTYP,SE(ISEA),IELNO,DAMA)
C
      ESE=(ESEB+ESES+ESEA)*LENGTH
      PSE=(PSEB+PSES+PSEA)*LENGTH
      DAMAGE=(DAMB*(ESEB+PSEB)+DAMS*(ESES+PSES)+DAMA*(ESEA+PSEA))*LENGTH
      IF (FIRST) WRITE (6,1420)
      WRITE (6,1410)IELNO,DAMB,ESEB,PSEB,DAMS,ESES,PSES,DAMA,ESEA,PSEA
 1410 FORMAT (1X,I5,3(OP,F14.5,1P,2G14.5))
 1420 FORMAT (/6X,
     &  ' |-------- BENDING COMPONENT ----------|',
     &  ' |---------- SHEAR COMPONENT ----------|',
     &  ' |---------- AXIAL COMPONENT ----------|'/,' ELEM#',
     &  ' |-- DAMAGE -- ---- ESE ---- ---- PSE ---|',
     &  ' |-- DAMAGE -- ---- ESE ---- ---- PSE ---|',
     &  ' |-- DAMAGE -- ---- ESE ---- ---- PSE ---|')
      RETURN
      END
C=======================================================================
C   ENERGY BALANCE
C=======================================================================
      SUBROUTINE ENERGY(L1,L2,L3,L4,L5,L6,NDOF,MAXNOD,NCOS,PT2,COND,
     & IND,IMDMS,IMDKG,MD,MDMS,MDKG,DYN,WRITE,ABORT,
     & STIFF,KG,MASS,
     & EIE,ESE,PSE,EKE,EDD,
     & ALPHA,BETA,KGFORM,NELMT,NNODE,DT,SUMUB,
     &  SUMRC,CSUMRC, SUMFD,DSUMFD,FDAMP,DFDAMP, FZ,
     & GA,GV,GD,GAT,GVT,GDT,  WORK,WORK2,FG,DFG,
     & DACC,DVEL,DDISP,ACC,VEL,DISP, LOAD,DLOAD,
     & ID      , IDOF     , CNST     , JTFLG    , COORD    ,
     & COSINE  , JTCOS    )
C
      DOUBLE PRECISION  EIE,ESE,DEIE,PSE,DEDD,EDD,EKE
      DIMENSION COORD(MAXNOD,3),ID(MAXNOD),
     &          IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CNST(MAXNOD,6),
     &          JTFLG(MAXNOD),JTCOS(MAXNOD)
C
      REAL MASS,KG,LOAD
C
```

```
      CHARACTER*80 NAME                                                 ENE00250
      DIMENSION STIFF(IMD),MDK(L6+1)                                    ENE00260
      DIMENSION MASS(IMMS),MMSK(L6+1)                                   ENE00270
      DIMENSION KG(IMDKG),MDKG(L6+1)                                    ENE00280
      DIMENSION RINPUT(100),FDAMP(L6),DFDAMP(L6),FG(6),DFG(6)           ENE00290
      DIMENSION  SUMRC(6),DSUMRC(6),SUMFD(6),DSUMFD(6),SUMUB(6)         ENE00300
      LOGICAL BTEST,PRINT,HEAD,PT2,AXIAL,COND,F,T,DYN,WRITE,ABORT       ENE00310
C                                                                       ENE00320
      DIMENSION  GD(3), GV(3), GA(3)                                    ENE00330
      DIMENSION GDT(3),GVT(3),GAT(3)                                    ENE00340
      DIMENSION DDISP(L6),DVEL(L6),DACC(L6)                             ENE00350
      DIMENSION  DISP(L6), VEL(L6), ACC(L6)                             ENE00360
      DIMENSION WORK(L6),WORK2(L6),LOADK(L6),DLOADK(L6)                 ENE00370
      DATA F/.FALSE./, T/.TRUE./                                        ENE00380
C                                                                       ENE00390
C------------------ NOMENCLATURE ------------------------------------CENE00400
C  DACC  INCRIMENTAL RELATIVE ACCELERATION                            CENE00410
C  DVEL  INCRIMENTAL RELATIVE VELOCITY                                CENE00420
C  DDISP INCRIMENTAL RELATIVE DISPLACEMENT                            CENE00430
C  ACC   PREVIOUS STEP'S RELATIVE ACCELERATION                        CENE00440
C  VEL   PREVIOUS STEP'S RELATIVE VELOCITY                            CENE00450
C  DISP  PREVIOUS STEP'S RELATIVE DISPLACEMENT                        CENE00460
C  DLOAD INCRIMENTAL APPLIED LOAD, NOT M*AG                           CENE00470
C  LOAD  PREVIOUS STEP'S APPLIED LOAD, NOT M*AG                       CENE00480
C  DFDAMP INCRIMENTAL DAMPING FORCE                                   CENE00490
C  FDAMP PREVIOUS STEP'S DAMPING FORCE                                CENE00500
C                                                                     CENE00510
C  GD    INCRIMENTAL GROUND DISPLACEMENT                              CENE00520
C  GV    INCRIMENTAL GROUND VELOCITY                                  CENE00530
C  GA    INCRIMENTAL GROUND ACCELERATION                              CENE00540
C  GDT   TOTAL GROUND DISPLACEMENT                                    CENE00550
C  GVT   TOTAL GROUND VELOCITY                                        CENE00560
C  GAT   TOTAL GROUND ACCELERATION                                    CENE00570
C  WORK  TEMP FORCE VECTOR                                            CENE00580
C  WORK2 TEMP DISPL OR VELOCITY VECTOR                                CENE00590
C-------------------------------------------------------------------CENE00600
C                                                                       ENE00610
      L4=L+L4+1                                                         ENE00620
C                                                                       ENE00630
C=======================================                                ENE00640
C===  INTEGRATE GROUND ACCELERATION  ===                                ENE00650
C=======================================                                ENE00660
      IF (DYN) THEN                                                     ENE00670
      DO 10 I=1,3                                                       ENE00680
        GA(I)=GA(I)                                                     ENE00690
        GV(I)=(2*GAT(I)+GA(I)) *DT /2                                   ENE00700
        GD(I)=GVT(I)*DT + (3*GAT(I)+GA(I)) *DT*DT /6                    ENE00710
        GAT(I)=GA(I)+GAT(I)                                             ENE00720
        GVT(I)=GV(I)+GVT(I)                                             ENE00730
   10   GDT(I)=GD(I)+GDT(I)                                             ENE00740
      ENDIF                                                             ENE00750
       IF (PT2) THEN                                                    ENE00760
        IF (DYN) THEN                                                   ENE00770
             WRITE (6,22) (I,ACC(I),VEL(I),DISP(I),LOADK(I),I=1,L4)     ENE00780
             WRITE (6,25) (I,DACC(I),DVEL(I),DDISP(I),DLOADK(I),I=1,L4) ENE00790
             WRITE (6,24) (I,FG(I),DFG(I),FDAMP(I),DFDAMP(I),I=1,L6)    ENE00800
        ELSE                                                            ENE00810
             WRITE (6,22) (I,ACC(I),VEL(I),DISP(I),LOADK(I),I=1,L6)     ENE00820
             WRITE (6,25) (I,DACC(I),DVEL(I),DDISP(I),DLOADK(I),I=1,L6) ENE00830
             WRITE (6,24) (I,FG(I),DFG(I),FDAMP(I),DFDAMP(I),I=1,L6)    ENE00840
        ENDIF                                                           ENE00850
   22   FORMAT(/T10,'TOTAL RELATIVE RESPONSE FOR PREVIOUS TIME STEP'/   ENE00860
     &  (T10,'DOF:',I6,' ACC:',1P,G15.5,' VEL:',G15.5,' DSP:',G15.5,    ENE00870
     &       'LOAD:',G15.5))                                            ENE00880
   25   FORMAT(/T10,'INCRIMENTAL RELATIVE RESPONSE'/                    ENE00890
     &  (T10,'DOF:',I6,' ACC:',1P,G15.5,' VEL:',G15.5,' DSP:',G15.5,    ENE00900
     &       'LOAD:',G15.5))                                            ENE00910
   24   FORMAT(/T10,'SPECIAL FORCES '/                                  ENE00920
     &  (T10,'DOF:',I6,' FD:',1P,G15.5,' DFD:',G15.5,' FDAMP:',G15.5,   ENE00930
     &       ' DFDAMP:',G15.5))                                         ENE00940
       ENDIF                                                            ENE00950
C                                                                       ENE00960
C=========================================                              ENE00970
C===  CALCULATE INPUT EARTHQUAKE ENERGY  ===                            ENE00980
C=========================================                              ENE00990
C                                                                       ENE01000
      DEIE=0                                                            ENE01010
C                                                                       ENE01020
C..... CALCULATE THE INCR. INPUT ENERGY DUE TO GRND MOTION              ENE01030
       IF (DYN) THEN                                                    ENE01040
          IF (PT2) WRITE(6,*) ' SUMRC', SUMRC                           ENE01050
          IF (PT2) WRITE(6,*) 'DSUMRC',DSUMRC                           ENE01060
          IF (PT2) WRITE(6,*) ' SUMFD', SUMFD                           ENE01070
          IF (PT2) WRITE(6,*) 'DSUMFD',DSUMFD                           ENE01080
          IF (PT2) WRITE(6,*) ' SUMUB', SUMUB                           ENE01090
          DO 48 I=1,3                                                   ENE01100
            FORCEI= SUMRC(I)+ 0.5*DSUMRC(I) + SUMFD(I) + 0.5*DSUMFD(I)  ENE01110
     &            + SUMUB(I)                                            ENE01120
            ENER=GD(I)*FORCEI                                           ENE01130
            IF (PT2) WRITE (6,*) I,                                     ENE01140
     &        'FORCEI',FORCEI,'  GD',GD(I),'  ENER',ENER                ENE01150
   48       DEIE=DEIE + ENER                                            ENE01160
       ENDIF                                                            ENE01170
C                                                                       ENE01180
C..... CALCULATE THE INCR. INPUT ENERGY DUE TO APPLIED LOADS...         ENE01190
          N2=L6                                                         ENE01200
          IF (DYN) N2=L4                                                ENE01210
          DO 50 I=1,N2                                                  ENE01220
            FORCEI=LOADK(I)+DLOADK(I)/2                                 ENE01230
            ENER=FORCEI*DDISP(I)                                        ENE01240
            IF (PT2.AND.(FORCEI.NE.0 .OR. DDISP(I).NE.0))               ENE01250
     &         WRITE (6,*) I,                                           ENE01260
     &        'FORCEI',FORCEI,'  DDISP',DDISP(I),'  ENER',ENER          ENE01270
   50       DEIE=DEIE + ENER                                            ENE01280
C                                                                       ENE01290
C..... CALCULATE THE ENERGY DUE TO GEOMETRIC STIFFNESS                  ENE01300
       IF (KGFORM.EQ.2 ) THEN                                           ENE01310
C....    GET THE INCRIMENTAL DISPL AND STORE IN VECTOR WORK2.....       ENE01320
C...     SAVE INCRIMENTAL IN VECTOR WORK2                               ENE01330
         DO 51 I=1,L4                                                   ENE01340
   51      WORK(I)=DDISP(I)                                             ENE01350
C...     ADD GROUND RESPONSE IN JOINT CORRDINATES TO RELATIVE RESP      ENE01360
C          MULTIPLY GROUND DISPL * DIRECTION COSINE                     ENE01370
C          OMIT ALL DOF THAT ARE CONSTRAINED TO A MASTER DOF...         ENE01380
         IF (GDK(1).NE.0 .OR. GDK(2).NE.0 .OR. GDK(3).NE.0) THEN        ENE01390
           DO 52 NODE=1,NNODE                                           ENE01400
             ICOS=JTCOS(NODE)                                           ENE01410
             DO 52 K=1,3                                                ENE01420
             IF ( .NOT. BTEST(JTFLG(NODE),K+11)) THEN                   ENE01430
               II=IDOF(NODE,K)                                          ENE01440
               IF (II.LE.L4 .AND. II.GE.L3)                             ENE01450
     &           WORK(II)=WORK(II)      + GD(1)*COSINE(1,K,ICOS)        ENE01460
     &         + GD(2)*COSINE(2,K,ICOS) + GD(3)*COSINE(3,K,ICOS)        ENE01470
             ENDIF                                                      ENE01480
   52      CONTINUE                                                     ENE01490
         ENDIF                                                          ENE01500
C                                                                       ENE01510
C..... CALCULATE THE INCRIMENTAL INPUT ENERGY DUE TO KG...              ENE01520
         DIKG=0                                                         ENE01530
         DO 55 I=L3,L4                                                  ENE01540
           ENER= (FG(I)+DFG(I)/2)*WORK(I)                               ENE01550
           IF (PT2) WRITE (6,*) I,                                      ENE01560
     &       'FG,DFG',FG(I),DFG(I),'  DDISP',WORK(I),'  ENER',ENER      ENE01570
   55    DIKG=DIKG+ENER                                                 ENE01580
         DEIE=DEIE+DIKG                                                 ENE01590
       ENDIF                                                            ENE01600
C                                                                       ENE01610
C..... CALCULATE THE TOTAL INPUT ENERGY                                 ENE01620
       EIE=EIE + DEIE                                                   ENE01630
C                                                                       ENE01640
C=========================================                              ENE01650
C===  CALCULATE TOTAL STRAIN ENERGY   ===                               ENE01660
C=========================================                              ENE01670
C..... CALCULATE THE TOTAL STRAIN ENERGY                                ENE01680
       DESE=ESE                                                         ENE01690
       DPSE=PSE                                                         ENE01700
       ESE=0                                                            ENE01710
       PSE=0                                                            ENE01720
C                                                                       ENE01730
C      ------- GET ELASTIC AND PLASTIC STRAIN ENERGY FOR EACH ELEMENT   ENE01740
       LSTTYP=0                                                         ENE01750
       PRINT=.FALSE.                                                    ENE01760
       HEAD =.FALSE.                                                    ENE01770
       DO 60 IELNO=1,NELMT                                              ENE01780
         CALL ELELIB                                                    ENE01790
     &  (10,LSTTYP ,.FALSE.,IREL,.FALSE.,  NAME ,EESE,EPSE,DAMAGE,      ENE01800
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)    ENE01810
         IF (PT2) WRITE (6,55) IELNO,EESE,EPSE                          ENE01820
   55    FORMAT('  ELEMENT=',I5,'   ELASTIC SE=',1P,G13.4,             ENE01830
     &                      '   PLASTIC SE=',1P,G13.4)                 ENE01840
         ESE = ESE + EESE                                               ENE01850
   60    PSE = PSE + EPSE                                               ENE01860
C                                                                       ENE01870
C..... CALCULATE THE INCRIMENTAL STRAIN ENERGY                          ENE01880
       DESE=ESE-DESE                                                    ENE01890
       DPSE=PSE-DPSE                                                    ENE01900
C                                                                       ENE01910
C=================================                                     ENE01920
C===  CALCULATE KINETIC ENERGY  ===                                     ENE01930
C=================================                                     ENE01940
       IF (DYN) THEN                                                    ENE01950
C..... GET THE ABSOLUTE VELOCITY AND STORE IN VECTOR WORK2.....         ENE01970
C...     SAVE RELATIVE RESPONSE IN VECTOR WORK2                         ENE01980
         DO 15 I=1,L4                                                   ENE01990
   15      WORK2(I)=VEL(I)+OVEL(I)                                      ENE02000
C...     ADD GROUND RESPONSE IN JOINT CORRDINATES TO RELATIVE RESP      ENE02010
C          MULTIPLY GROUND VELOCITY * DIRECTION COSINE                  ENE02020
C          OMIT ALL DOF THAT ARE CONSTRAINED TO A MASTER DOF...         ENE02030
         IF (GVT(1).NE.0 .OR. GVT(2).NE.0 .OR. GVT(3).NE.0) THEN        ENE02040
           DO 20 NODE=1,NNODE                                           ENE02050
             ICOS=JTCOS(NODE)                                           ENE02060
             DO 20 K=1,3                                                ENE02070
             IF ( .NOT. BTEST(JTFLG(NODE),K+11)) THEN                   ENE02080
               II=IDOF(NODE,K)                                          ENE02090
               IF (II.LE.L4)                                            ENE02100
     &         WORK2(II)=WORK2(II)      + GVT(1)*COSINE(1,K,ICOS)       ENE02110
     &         + GVT(2)*COSINE(2,K,ICOS) + GVT(3)*COSINE(3,K,ICOS)      ENE02120
             ENDIF                                                      ENE02130
   20        CONTINUE                                                   ENE02140
         ENDIF                                                          ENE02150
C                                                                       ENE02160
C..... CALCULATE THE TOTAL KE                                           ENE02170
       DEKE=EKE                                                         ENE02180
       CALL ETMPLY(PT2,L1,L2,L3,L4,L41,L4,                              ENE02190
     &    IMDMS,MDMS,MASS,WORK,WORK2,WORK2,EKE)                         ENE02200
       EKE=EKE/2.0                                                      ENE02210
C                                                                       ENE02220
C..... CALCULATE THE INCRIMENTAL KE                                     ENE02230
       DEKE=EKE-DEKE                                                    ENE02240
      ELSE                                                              ENE02250
C                                                                       ENE02260
C..... ZERO KE FOR STATIC CASE.                                         ENE02270
       DEKE=0                                                           ENE02280
       EKE =0                                                           ENE02290
      ENDIF                                                             ENE02300
C                                                                       ENE02310
C=================================                                     ENE02320
C===  CALCULATE DAMPINED ENERGY   ===                                   ENE02330
C=================================                                     ENE02340
      IF (DYN) THEN                                                     ENE02350
C                                                                       ENE02360
C..... CALCULATE THE INCRIMENTAL DAMPING ENERGY                         ENE02370
       DEDD =0                                                          ENE02380
       IF (ALPHA.NE.0 .OR. BETA.NE.0) THEN                              ENE02390
         DO 85 I=L3,L4                                                  ENE02400
           FORCEI= FDAMP(I)+DFDAMP(I)/2                                 ENE02410
           VMI = VEL(I)+ OVEL(I)/2                                      ENE02420
           ENER=FORCEI*VMI                                              ENE02430
           IF (PT2) WRITE (6,*) I,                                      ENE02440
     &       'FORCEI',FORCEI,'  VMI',VMI,'  ENER',ENER                  ENE02450
   85      DEDD=DEDD + ENER                                             ENE02460
           DEDD=DEDD*DT                                                 ENE02470
       ENDIF                                                            ENE02480
C                                                                       ENE02490
C..... CALCULATE THE TOTAL DAMPINED ENERGY                              ENE02500
       EDD=EDD+DEDD                                                     ENE02510
      ELSE                                                              ENE02520
C                                                                       ENE02530
C..... ZERO DE FOR STATIC CASE.                                         ENE02540
       EDD=0                                                            ENE02550
       DEDD=0                                                           ENE02560
      ENDIF                                                             ENE02570
C                                                                       ENE02580
C============================                                           ENE02590
C===  CHECK ENERGY BALANCE  ===                                         ENE02600
C============================                                           ENE02610
      DIFF=EIE-ESE-EKE-PSE-EDD                                          ENE02620
      IF (EIE.NE.0) THEN                                                ENE02630
        RE=DIFF/EIE * 100                                               ENE02640
      ELSE                                                              ENE02650
        RE=100.00                                                       ENE02660
      ENDIF                                                             ENE02670
C                                                                       ENE02680
C=================                                                     ENE02690
C===  PRINT DATA ===                                                    ENE02700
C=================                                                     ENE02710
      IF (WRITE)                                                        ENE02720
```

```
      &WRITE (6,100) DEIE,ETE,DESE,ESE,DEKE,EKE,DPSE,PSE,DEDD,EDD,RE
  100 FORMAT (/,T10,' ENERGY BALANCE'/
     &            T10,'==============',
     & T40,'INCRIMENTAL',T58,'TOTAL'/
     & T15,'INPUT ENERGY'              ,T40,1P,G12.5,T55,G12.5/
     & T15,'ELASTIC STRAIN ENERGY'     ,T40,1P,G12.5,T55,G12.5/
     & T15,'KINETIC ENERGY '           ,T40,1P,G12.5,T55,G12.5/
     & T15,'PLASTIC STRAIN ENERGY'     ,T40,1P,G12.5,T55,G12.5/
     & T15,'DAMPING ENERGY'            ,T40,1P,G12.5,T55,G12.5/
     & T15,'RELATIVE ER','ROR'         ,                T55,G12.5,'%')

      IF (DYN.AND.WRITE) WRITE (6,110) ( GD(I), GV(I), GA(I),I=1,3),
     &                                 (GDT(I),GVT(I),GAT(I),I=1,3)
  110 FORMAT (/,T10,' GROUND RESPONSE' /
     &            T10,'================'/
     & T10,'INCRIMENTAL VALUES',T46,'DISPLACEMENT',
     &     T65,'VELOCITY',T76,'ACCELERATION'/
     & T15,'GLOBAL X AXIS TRANSLATION...',T45,1P,3G15.5,/
     & T15,'GLOBAL Y AXIS TRANSLATION...',T45,1P,3G15.5,/
     & T15,'GLOBAL Z AXIS TRANSLATION...',T45,1P,3G15.5,/
     & T10,'TOTAL VALUES'/
     & T15,'GLOBAL X AXIS TRANSLATION...',T45,1P,3G15.5,/
     & T15,'GLOBAL Y AXIS TRANSLATION...',T45,1P,3G15.5,/
     & T15,'GLOBAL Z AXIS TRANSLATION...',T45,1P,3G15.5)
C
C=====================================
C===  DUCTILITIES AND EXCURSION RATIOS ===
C=====================================
      LSTTYP=0
      PRINT=WRITE
      HEAD =WRITE
      DO 200 IELNO=1,NELMT
  200   CALL ELELIB
     &   (11,LSTTYP ,.FALSE..IREL,.FALSE., NAME ,EESE,EPSE,DAMAGE,
     &    IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)
C
      RETURN
      END
C=======================================================================
C   CALC THE ENERGY (E) FOR UCTRANSP) * P = UCTRANSP) * A * X
C
C - MATRIX A IS AN UPPER TRIANGULAR MATRIX, WITH MAIN DIAGONAL ADDRESS
C     MATRIX MD.
C     DOF L1 TO L2 HAVE BEEN CONDENSED OUT
C     DOF L3 TO L4 HAVE BEEN MODIFIED FOR THE CONDENSATION OF L1 TO L2
C     DOF L5 TO L6 HAVE NOT BEEN MODIFIED
C - X IS A VECTOR.
C - P IS A DUMMY INPUT VECTOR.
C=======================================================================
      SUBROUTINE ETMPLV(PRINT,L1,L2,L3,L4,L5,L6,IMD,MD,A,P,X,U,E)
      DOUBLE PRECISION E,E1
      DIMENSION A(IMD),X(L6),P(L6),MD(L6+1),U(L6)
      LOGICAL BTEST,PRINT,T,F
      IJI(I,J)=MD(J)+J-I
C
      T=.TRUE.
      F=.FALSE.
C
      IF (PRINT) THEN
        WRITE (6,500) 'CALC ENERGY FOR PARTIALLY REDUCED A MATRIX',
     &          L1,L2,L3,L4,L5,L6,ZERO,X5
        CALL LMATRX(A,MD,L6,IMD,'MATRIX A ')
        DO 515 I=L1,L6
  515     WRITE(6,520) I,X(I),U(I)
      ENDIF
C
C------------ CALC PF* = KFF**X1
      CALL TMPLV(F,T,IMD,L6,L3,L4,L5,L4,MD,A,P,X)
C
C------------ CALC E1 = XF' * PF*
      E=0
      DO 10 L=L3,L4
   10   E = E + U(L)*P(L)
      E1=E
C
C------------ CALC PF  = KFR *XR
      CALL TMPLV(F,T,IMD,L6,L1,L4,L5,L6,MD,A,P,X)
C
C------------ CALC PF  = KFR *XR
      CALL TMPLV(F,T,IMD,L6,L5,L6,L1,L6,MD,A,P,X)
C
C------------ CALC E2 + E3 + E4 = XF * PF + XR * PR
      DO 20 L=L1,L6
   20   E = E + U(L)*P(L)
C
C------------ PRINT ANSWER ...
      IF (PRINT) WRITE (6,510) E,E1
C
  500 FORMAT (/2X,A,/2X,'L1:',I5,'   L2:',I5,
     &'   L3:',I5,'   L4:',I5,
     &'   L5:',I5,'   L6:',I5,'  ZERO: ',L1,'    X5: ',L1)
  510 FORMAT (/2X,' TOTAL ENERGY:',1P,G15.5,'  E1:',G15.5)
  520 FORMAT (2X,'DOF:',I6,'  X(DOF)',1P,G14.5,'  U(DOF):',G14.5)
      RETURN
      END
C=======================================================================
      SUBROUTINE FEMREL(MA,MB,RELA,RELB,ITORQ,IELNO)
      REAL    MA,MB
      INTEGER REL
      LOGICAL RELA,RELB,BTEST
C
      IF (ITORQ.EQ.0) THEN
          IF (RELA .AND. RELB) THEN
            MA =0
            MB=0
          ELSE IF (RELB) THEN
            MA = MA+MB/2
            MB=0
          ELSE IF (RELA) THEN
            MB= MB+MA/2
            MA =0
          ENDIF
      ELSE
          IF (RELA .AND. RELB) THEN
            WRITE (6,900) IELNO,'TORSION WITH BOTH ENDS RELEASED'
            MA=0
            MB=0
          ELSE IF (RELB) THEN
            MA=MA+MB
            MB=0
          ELSE IF (RELA) THEN
            MB=MB+MA
            MA =0
          ENDIF
```

```
      ENDIF
C=======================================================================
  900 FORMAT(' ER','ROR CALCULATING FIXED END FORCES FOR ELEMENT #',
     &  I6,' ',A,' IS NOT AVAILABLE')
      RETURN
      END
C=======================================================================
C             STRIPS PRECEEDING BLANK SPACES OFF A CHARACTER STRING
      SUBROUTINE FIRST(WORD)
      LOGICAL BTEST
      CHARACTER*(*) WORD
          IF (INDEX(WORD,' ').EQ.0) RETURN
      DO 10 I=1,(LEN(WORD))
           J=INDEX(WORD(I:),' ')
           IF (J.EQ.1) GO TO 10
               WORD=WORD(I:)
               RETURN
   10      CONTINUE
   20 RETURN
      END
C=======================================================================
      SUBROUTINE FORM(IOPT)
      CHARACTER*80  NAME
      CHARACTER*150 ELNO
      LOGICAL PRINT,HEAD,FALSE,AXIAL,BTEST,BUG5,BUG57
      INCLUDE (ZCOMM)
C                                                                       FOR00100
C-----------------------------------------------------------------------FOR00110
C  MATRIX STORAGE SCHEME:                                               FOR00120
C -THE STIFFNESS MATRIX (K) IS BANDED AND SYMMETRIC.                    FOR00130
C  THUS ONLY THE SKYLINE ABOVE THE MAIN DIAGONAL NEEDS TO BE STORED.    FOR00140
C  THE MATRIX K IS STORED IN A LINEAR ARRAY BY COLUMNS.                 FOR00150
C  THE VALUE ON THE MAIN DIAGONAL IS STORED IN THE LINEAR ARRAY FIRST.  FOR00160
C  RELATIVE ADDRESSES OF THE LINEAR ARRAY ELEMENTS CONTAINING THE       FOR00170
C  MAIN DIAGONAL TERMS ARE STORED IN MD.                                FOR00180
C  THE ABSOLUTE ADDRESS OF THE FIRST ELEMENT IN THE K MATRIX IS IZSTIF. FOR00190
C  THE STIFFNESS MATRIX K IS STORED IN THE Z ARRAY, Z(IZSTIF)=K(1,1)    FOR00200
C  THE ABSOLUTE ADDRESS OF THE FIRST ELEMENT IS THE MD MATRIX IS IZMD.  FOR00210
C  THE STIFFNESS MATRIX K IS STORED IN THE NZ ARRAY, NZ(IZMD)=MD(1)     FOR00220
C  THUS: IF I LE J (UPPER TRIANGULAR MATRIX)                            FOR00230
C          IF IZMD-1+J < IZMD-1+J+1                                     FOR00240
C             IJ=NZ(IZMD-1+J)+J-I ; RELATIVE ADDRESS IF ELEMENT (I,J)   FOR00250
C             K(I,J)=Z(IZSTIF-1 + IJ)                                   FOR00260
C          ELSE                                                         FOR00270
C             K(I,J)=0             ; (I,J) IS ABOVE THE SKYLINE        FOR00280
C                                                                       FOR00290
C  EXAMPLE: LET K BE THE 5X5 FULL SYMETRIC MATRIX BELOW...              FOR00300
C     | 1 | 3 |   |   | 12 |                                           FOR00310
C     |   | 2 | 5 |   | 11 |  NOTE: K(1,3),K(1,4) AND K(2,4) ARE ZERO. FOR00320
C  K= |   | 4 | 7 | 10 |                                              FOR00330
C     |  SYM. |   6 |  9 |  MD= 1, 2, 4, 6, 8, 13                      FOR00340
C     |       |     8 |                                               FOR00350
C  SAY IZMD=121,IZSTIF=200                                              FOR00360
C  NZ(121)=1          Z(200)=1                                          FOR00370
C  NZ(122)=2          Z(201)=2                                          FOR00380
C  NZ(123)=4          Z(202)=3                                          FOR00390
C  NZ(124)=6          Z(203)=4                                          FOR00400
C  NZ(125)=8          Z(204)=5                                          FOR00410
C  NZ(126)=13         Z(205)=6                                          FOR00420
C                     Z(206)=7                                          FOR00430
C                     Z(206)=8                                          FOR00440
C                     Z(206)=9                                          FOR00450
C                     Z(206)=10                                         FOR00460
C                     Z(206)=11                                         FOR00470
C                     Z(206)=12                                         FOR00480
C  THUS, FOR K(3,4)  I=3, J=4                                           FOR00490
C            IJ=NZ(IZMD-1+J)+J-I = NZ(121-1+4) +4-3 = 6+4-3 =7         FOR00500
C            K(I,J)=Z(IZSTIF-1 + IJ) = Z(200-1+7) = Z(206) = 7  OK     FOR00510
C  THUS, FOR K(2,4)  I=2, J=4                                           FOR00520
C            IJ=NZ(IZMD-1+J)+J-I = NZ(121-1+4) +4-2 = 6+4-2 =8         FOR00530
C            NZ(IZMD -1+J-1)=NZ(121-1+4-1)=8                           FOR00540
C            SINCE IJ GE NZ(IZMD-1+J-1)                                FOR00550
C            K(I,J)=0  (I,J) IS ABOVE THE SKYLINE....                  FOR00560
C-----------------------------------------------------------------------FOR00570
C                                                                       FOR00580
      BUG5 = BTEST(IBUG,5)
      BUG57= BTEST(IBUG,7) .AND. BUG5
                                                                        FOR00620
      LSTTYP=0                                                          FOR00630
      PRINT=.FALSE.                                                     FOR00640
      HEAD =.FALSE.                                                     FOR00650
C                                                                       FOR00660
C-----------------------------------------------------------------------FOR00670
C IOPT=1 FORM STIFFNESS MATRIX                                          FOR00680
C IOPT=2 FORM GEOMETRIC STIFFNESS MATRIX                                FOR00690
C-----------------------------------------------------------------------FOR00700
C                                                                       FOR00710
       IF (BUG5)  WRITE (6,*) ' IN ROUTINE FORM IOPT:',IOPT             FOR00720
C                                                                       FOR00730
   49 FORMAT (1X,A/('  I:',I5,'  MONASS(I):',1I1))                      FOR00740
C                                                                       FOR00750
      GO TO (100,200),IOPT                                              FOR00760
C                                                                       FOR00770
C=======================                                                FOR00780
C= ASSEMBLE STIFFNESS  =====                                            FOR00790
C=======================                                                FOR00800
  100 CONTINUE                                                          FOR00810
          IST =IZSTIF                                                   FOR00820
          IEND=IZSTIF +NZ(IZMD+NDOF) -1                                 FOR00830
C                                                                       FOR00840
C---------- ZERO STIFFNESS                                              FOR00850
       DO 110 I=IST,IEND                                                FOR00860
  110     Z(I)=0                                                        FOR00870
C                                                                       FOR00880
C---------- LOOP FOR EACH ELEMENT                                       FOR00890
      DO 120 IELNO=1,NELMT                                              FOR00900
C                                                                       FOR00910
C---------- GET STIFFNESS OF EACH ELEMENT                               FOR00920
       FALSE=.FALSE.                                                    FOR00930
       CALL ELELIB                                                      FOR00940
     &   (5,LSTTYP ,FALSE ,IREL,FALSE ,NAME ,EESE,EPSE,DAMAGE,          FOR00950
     &    IELNO ,IELDOF,KGDOF,PGEOM,Z  ,AXIAL,IZDISP,IZLOAD,MSTOR)     FOR00960
C                                                                       FOR00970
       IF (BUG57) THEN                                                  FOR00980
         WRITE (ELNO,15)  IELNO,(NZ(I+IZLM-1),I=1,MIN(IELDOF,18))      FOR00990
   15    FORMAT ('ELEMENT #',I6,' LM:',18I6)                           FOR01000
         CALL LMATRX(Z(IZKE),MD,IELDOF,MD(IELDOF+1),ELNO)              FOR01010
       ENDIF                                                            FOR01020
C                                                                       FOR01030
      CALL FORML(Z(IZSTIF),NZ(IZMD),NDOF,NZ(IZMD+NDOF),                 FOR01040
```

```
     &     Z(IZKE),MD,IELDOF,MDK(IELDOF+1),NZ(IZLM),1.)
C
C---------- SUBTRACT GEOMETRIC STIFFNESS
      NZ(IZKGDT+5)=NZ(IZKGDT+5)
C
      IF (NZ(IZKGDT+5).EQ.1) THEN
        FALSE=.FALSE.
        CALL ELELIB
     &  (2,LSTTYP ,FALSE ,IREL,FALSE, NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,Z  ,AXIAL,IZDISP,IZLOAD,MSTOR)
C
        KGDOF=KGDOF
C
        IF (BUG57) THEN
          WRITE (ELNO,15)  IELNO,(NZ(I+IZLMKG-1),I=1,MIN(KGDOF,18))
          CALL LMATRX(Z(IZKEKG),MD,KGDOF,MDK(KGDOF+1),ELNO)
        ENDIF
C
        IF (.NOT. AXIAL) GO TO 120
C           DETERMINE THE FACTOR FOR KG
              IF (NZ(IZKGDT+1).EQ.1) THEN
                    ACCELZ= Z(IZKGDT)
                    PGEOM=PGEOM*(1+ACCELZ)
              ENDIF
C     SWAP THE SIGN ON PGEOM FOR SUBTRACTION.....
      PGEOM=-PGEOM
C
      CALL FORML(Z(IZSTIF),NZ(IZMD),NDOF,NZ(IZMD+NDOF),
     &     Z(IZKEKG),MD,KGDOF,MDK(KGDOF+1),NZ(IZLMKG),PGEOM)
      ENDIF
C
  120   CONTINUE
C
      RETURN
C
C---------- ASSEMBLE GEOMETRIC STIFNESS
  200 CONTINUE
      IF (NZ(IZKGDT+5).NE.2 ) RETURN
          IST =IZKG
          IEND=IZKG + NZ(IZMDKG+NDOF) - 1
C
C---------- ZERO STIFFNESS
      DO 210 I=IST,IEND
  210   Z(I)=0
C
C---------- LOOP FOR EACH ELEMENT
      DO 220 IELNO=1,NELMT
C
C---------- GET STIFFNESS OF EACH ELEMENT
        FALSE=.FALSE.
        CALL ELELIB
     &  (2,LSTTYP ,FALSE ,IREL,FALSE, NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,Z  ,AXIAL,IZDISP,IZLOAD,MSTOR)
C
        IF (.NOT. AXIAL) GO TO 220
C
        IF (BUG57) THEN
          WRITE (6,*)  'PGEOM=',PGEOM
          WRITE (ELNO,15)  IELNO,(NZ(I+IZLMKG-1),I=1,MIN(KGDOF,18))
          CALL LMATRX(Z(IZKEKG),MD,KGDOF,MDK(KGDOF+1),ELNO)
        ENDIF
C
      CALL FORML(Z(IZKG),NZ(IZMDKG),NDOF,NZ(IZMDKG+NDOF),
     &     Z(IZKEKG),MD,KGDOF,MDK(KGDOF+1),NZ(IZLMKG),PGEOM)
  220 CONTINUE
C
      RETURN
      END
C=======================================================================
      SUBROUTINE FORML(K,MD,NDOF,IMD,KE,MDKE,IELDOF,IMDKE,LM,FACTOR)
      LOGICAL BTEST
      DIMENSION MDK(NDOF+1),MDKE(IELDOF+1),LM(IELDOF)
      REAL K(IMD),KE(IMDKE)
C
C-----------------------------------------------------------------------
C   MATRIX STORAGE SCHEME:
C  -THE STIFFNESS MATRIX (K) IS BANDED AND SYMMETRIC.
C   THUS ONLY THE SKYLINE ABOVE THE MAIN DIAGONAL NEEDS TO BE STORED.
C   THE MATRIX K IS STORED IN A LINEAR ARRAY BY COLUMNS.
C   THE VALUE ON THE MAIN DIAGONAL IS STORED IN THE LINEAR ARRAY FIRST.
C   RELATIVE ADDRESSES OF THE LINEAR ARRAY ELEMENTS CONTAINING THE
C   MAIN DIAGONAL TERMS ARE STORED IN MD.
C   THUS: FOR K(I,J)
C         IF I LE J (UPPER TRIANGULAR MATRIX)
C         IF MDK(J)+J-I < MDK(J+1)
C                 IJ=MDK(J)+J-I      : RELATIVE ADDRESS IF ELEMENT (I,J)
C                 K(I,J)=K(IJ)
C          ELSE
C                 K(I,J)=0           : (I,J) IS ABOVE THE SKYLINE
C
C   EXAMPLE: LET B BE THE SXS FULL SYMETRIC MATRIX BELOW...
C      | 1 | 3 |   |  12 |
C      |   | 2 | 5 |  11 |   NOTE: K(1,3),K(1,4) AND K(2,4) ARE ZERO.
C   K= |   |   | 4 |  7  |  10 |
C      |  SYMM.    | 6 |  9 |  MD= 1, 2, 4, 6, 8, 13
C      |           |   |  8 |
C   SAY IZMD=121,IZSTIF=200
C   MDK(1)=1          K( 1)=1
C   MDK(2)=2          K( 2)=2
C   MDK(3)=4          K( 3)=3
C   MDK(4)=6          K( 4)=4
C   MDK(5)=8          K( 5)=5
C   MDK(6)=13         K( 6)=6
C                     K( 7)=7
C                     K( 8)=8
C                     K( 9)=9
C                     K(10)=10
C                     K(11)=11
C                     K(12)=12
C   THUS, FOR K(3,4)  I=3, J=4
C             IJ=MDK(J)+J-I = MDK(4)+4-3 = 6+4-3 =7
C             K(I,J)=K(IJ) = K(7) =  7  OK
C   THUS, FOR K(2,4)  I=2, J=4
C             IJ=MDK(J)+J-I = MDK(4) +4-2 = 6+4-2 =8
C             MDK(J+1)=MDK(4+1)=8
C             SINCE IJ GE MDK(J+1)
C             K(I,J)=0  (I,J) IS ABOVE THE SKYLINE....
C-----------------------------------------------------------------------
C   THE ELEMENT STIFFNESS IS STORED IN A SIMILAR MANNER.
C   KE IS THE LINEAR ARRAY CONTAINING THE ELEMENT STIFFNESS
C   MDKE IS THE ARRAY CONTAINING THE MAIN DIAGONAL ADDRESSES FOR KE
C-----------------------------------------------------------------------
C
C---------- MAP ELEMENT STIFFNESS INTO GLOBAL STIFFNESS
C LI    = GLOBAL DOF ASSOCIATED WITH THE I'TH ROW OF THE MEMBER STIFF.
C LJ    = GLOBAL DOF ASSOCIATED WITH THE I'TH ROW OF THE MEMBER STIFF.
C IJELM = ABSOLUTE ADDRESS OF THE MEMBER STIFFNESS TERM (I,J)
C IJGLOB= ABSOLUTE ADDRESS OF THE GLOBAL STIFFNESS TERM (LI,LJ)
C
C*    MAXIJ = MDK(NDOF+1)
C
      DO 30 I=1,IELDOF
        LI=LM(I)
C*      IF (LI.LE.0 .OR. LI.GT.NDOF) GO TO 30
C*      DO 20 J=I,IELDOF
          LJ=LM(J)
C*        IF (LJ.LE.0 .OR. LJ.GT.NDOF) GO TO 20
          IJELM =MDKE(J)+J-I
          IF (KE(IJELM).EQ.0) GO TO 20
          IF (LI.LE.LJ) THEN
              IJGLOB=MDK(LJ)+LJ-LI
          ELSE
              IJGLOB=MDK(LI)+LI-LJ
          ENDIF
C*        IF ( IJGLOB.GE.MAXIJ ) THEN
C*          WRITE (6,*) 'ER','ROR IN MAPPING ELEMENT STIFFNESS FOR',
C*   &       ' ELEMENT #',IELNO,LI,LJ,IJGLOB,MAXIJ
C*        ELSE
            K(IJGLOB)=K(IJGLOB)+KE(IJELM) * FACTOR
C*        ENDIF
   20   CONTINUE
   30 CONTINUE
C
C------ NOTE: LINES COMMENTED OUT C* ARE FOR ER_ROR CHECKING. THE VALUES
C           IN LM ARE ASSUMED TO BE CORRECT...
C
      RETURN
      END
C=======================================================================
C
C SUBROUTINE GETCHR DETERMINES THE VALUE OF THE FIRST CHARACTER
C                  AFTER AN OPTIONAL (LWORD=.TRUE.) KEYWORD (WORD).
C                  AND REMOVES THE STRING (WORD)//CHAR    IF
C                  MODIFY=.TRUE.
C
      SUBROUTINE GETCHR(VERB,WORD,CHAR,LWORD,MODIFY)
C
      CHARACTER*(*) VERB,WORD,CHAR
      LOGICAL MODIFY, LWORD ,BTEST
C
  999 CHAR=' '
      IF (LWORD) THEN
          ISTART=INDEX(VERB,WORD)
          IF (ISTART.LE.0) RETURN
          I=ISTART+LEN(WORD)
      ELSE
          I=1
          ISTART=1
          CALL FIRST(VERB)
      ENDIF
C
      IF (VERB(I:I).EQ.'''' ) THEN
         I=I+1
         IJ=I
   10    J=INDEX( VERB(IJ:),'''')-2  + IJ
         IF (VERB(J+2):(J+2)).EQ.'''') THEN
             IJ=J+3
             GO TO 10
         ENDIF
         IF (J.EQ.0) THEN
             CHAR=VERB(I: )
             GO TO 10
         ELSE
             CHAR=VERB(I:J)
         ENDIF
   20    IJ=INDEX(CHAR,'''''')
         IF (IJ.NE.0) THEN
             CHAR=CHAR(:IJ)//CHAR((IJ+2):)
             GO TO 20
             ENDIF
      ELSE
         J=INDEX(VERB(I:),' ') + I -2
         IF (J.GE.I) CHAR=VERB(I:J)
      ENDIF
C
      IF (MODIFY) THEN
         IF (VERB(J+2:J+2).EQ.',') J=J+1
         VERB(ISTART: )=VERB(J+2: )
         CALL FIRST(VERB)
      ENDIF
C
      RETURN
      END
C=======================================================================
C
C SUBROUTINE GETINT DETERMINES THE VALUE OF THE FIRST INTEGER (INTEG)
C                  AFTER AN OPTIONAL (LWORD=.TRUE.) KEYWORD (WORD).
C                  AND REMOVES THE STRING (WORD)//INTEGER IF
C                  MODIFY=.TRUE.  INTEG CAN BE ANY LENGTH REPRESENTABLE
C                  BY SINGLE PRECESION.
C
      SUBROUTINE GETINT(VERB,WORD,INTEG,DEFULT,LWORD,MODIFY)
C
      CHARACTER*(*) VERB,WORD
      CHARACTER*10  NUMB
      LOGICAL MODIFY, LWORD, HIT, BTEST
      INTEGER DEFULT
      DATA NUMB/'0123456789'/
C
      INTEG=DEFULT
      HIT=.TRUE.
      IF (LWORD) THEN
          ISTART=INDEX(VERB,WORD)
          IF (ISTART.LE.0) RETURN
          I=ISTART+LEN(WORD)
      ELSE
          I=1
          ISTART=1
      ENDIF
C
      ISIGN = 1
      L=LEN(VERB)
    5 IF (I.GT.L) RETURN
      IF (VERB(I:I).EQ.'-') THEN
         ISIGN =-1
         I=I+1
         GO TO 5
      ELSE IF (VERB(I:I).EQ.'+' .OR.VERB(I:I).EQ.' ') THEN
```

```
      I=I+1                                                             GET00580
      GO TO 5                                                           GET00590
    ENDIF                                                               GET00400
C                                                                       GET00410
 10 IF (LGE(VERBK(I:I),'0') .AND. LLE(VERBK(I:I),'9') ) THEN            GET00420
      IF ( HIT) THEN                                                    GET00430
        HIT=.FALSE.                                                     GET00440
        INTEG=0                                                         GET00450
      ENDIF                                                             GET00460
      DO 20 J=1,10                                                      GET00470
       IF(VERBK(I:I).EQ. NUMBK(J:J) ) THEN                              GET00480
         INTEG=10*INTEG+ (J-1)                                          GET00490
         I=I+1                                                          GET00500
         GO TO 10                                                       GET00510
       ENDIF                                                            GET00520
 20   CONTINUE                                                          GET00530
    ENDIF                                                               GET00540
C                                                                       GET00550
    INTEG=INTEG*ISIGN                                                   GET00560
C                                                                       GET00570
    IF (MODIFY) THEN                                                    GET00580
      IF (VERBK(I:I).EQ.',') I=I+1                                      GET00590
      VERBK(ISTART: )=VERBK(I: )                                        GET00600
    ENDIF                                                               GET00610
C                                                                       GET00620
    RETURN                                                              GET00630
    END                                                                 GET00640
C=======================================================================HVS00040
C========== HVST02 ====================================================HVS00010
C     SU_BROUTINE AXLMOD - AXIAL HVSTERESIS MODEL                       HVS00050
C                          REFERANCE: ANALYSIS OF THE FULL SCALE SEVEN   HVS00060
C                                     STORY R/C TEST STRUCTURE.         HVS00070
C                                                                       HVS00080
C     P     = CURRENT LOAD. POSITIVE IS TENSION                         HVS00090
C     O     = CURRENT DISPLACMENT, POSITIVE IS ELONGATION               HVS00100
C     S     = STIFFNESS                                                 HVS00110
C     KC    = VIRGIN COMPRESSION STIFFNESS, EC                          HVS00120
C     KT1   = PRE-YEILD VIRGIN TENSILE STIFFNESS, 0.9*KC                HVS00130
C     KT2   = POST-YEILD VIRGIN TENSILE STIFFNESS, 0.001*KC             HVS00140
C     PV    = TENSILE YEILD POINT, PHO*FV                               HVS00150
C     FMAX  = MAXIMUN TENSILE FORCE BEFORE UNLOADING                    HVS00160
C     DMAX  = MAXIMUN TENSILE DISPLACMENT BEFORE UNLOADING              HVS00170
C     FX    = FORCE AT STIFFNESS CHANGE POINT ON THE UNLOADING CURVE     HVS00180
C     DX    = DISPL. AT STIFFNESS CHANGE POINT ON THE UNLOADING CURVE    HVS00190
C     PI,DI = INTERSECTION OF TWO LINES                                 HVS00200
C     DVT   = TENSILE YEILD DISPLACMENT = PV/KT1                        HVS00210
C     DVC   = COMPRESSION YEILD DISPLACMENT = PV/KC                     HVS00220
C     FP    = FORCE AT COMPRESSION STIFFNESS CHANGE POINT               HVS00230
C     DP    = COMPRESSION  AT COMPRESSION STIFFNESS CHANGE POINT        HVS00240
C     KR    = UNLOADING STIFFNESS = KC/(DMAX/DVT)**C,20)                HVS00250
C     KS    = LOADING STIFFNESS, LINE BETWEEN (FX,DX) AND (FP,DP)       HVS00260
C     KT    = LOADING STIFFNESS, LINE BETWEEN (FP,DP) AND (-2PV,-2DV)   HVS00270
C                                                                       HVS00280
C                                                                       HVS00290
      SUBROUTINE HVST02(P,D,PL,DL,V,VL,S,HRLN,KC,KT1,KT2,PV,ALPHA,BETA, HVS00300
     &   DVT,DVC,CSE,FMAX,DMAX,FX,DX,FP,DP,KR,KS,KT,NRL,A,PRINT,        HVS00310
     &   SEQ,ESE,PSE,PSEOLD,UPOS,UNEG,EXCR,IDR)                         HVS00320
C                                                                       HVS00330
      REAL*4 KC,KT1,KT2,KR,KS,KT,NRL                                    HVS00340
      DIMENSION UPOS(5),UNEG(5),EXCR(6),ESE(5)                          HVS00350
      INTEGER RULEI                                                     HVS00360
      LOGICAL LT,PRINT,PGTPL,PLTPL                                      HVS00370
C                                                                       HVS00380
 8000 IF (PRINT) WRITE (6,8010) P,D,PL,DL,V,VL,S,HRLN,KC,KT1,KT2,PV,    HVS00390
     &   ALPHA,BETA,DVT,DVC,FMAX,DMAX,FX,DX,FP,DP,KR,KS,KT,NRL,A,       HVS00400
     &      SEQ,ESE(5),PSE,PSEOLD,CSE,UPOS,UNEG,EXCR                   HVS00410
 8010 FORMAT(                                                           HVS00420
     &' AXLMOD=P=',G15.5,'    O=',G15.5,'   PL=',G15.5,'   DL=',G15.5,/ HVS00430
     &' AX=    V=',G15.5,'   VL=',G15.5,'    S=',G15.5,' HRLN=',G15.5,/ HVS00440
     &' AX=   KC=',G15.5,'  KT1=',G15.5,'  KT2=',G15.5,'   PV=',G15.5,/ HVS00450
     &' AX=ALPHA=',G13.5,' BETA=',G15.5,'  DVT=',G13.5,'  DVC=',G15.5,/ HVS00460
     &' AX= FMAX=',G15.5,' DMAX=',G15.5,'   FX=',G13.5,'   DX=',G15.5,/ HVS00470
     &' AX=   FP=',G15.5,'   DP=',G15.5,'   KR=',G13.5,'   KS=',G15.5,/ HVS00480
     &' AX=   KT=',G15.5,'  NRL=',G15.5,'    A=',G15.5,'  SEQ=',G15.5,/ HVS00490
     &' AX=  ESE=',G15.5,'  PSE=',G15.5,' PSEO=',G13.5,'  CSE=',G15.5,/ HVS00500
     &' AX=  UP1=',G13.5,'  UP2=',G15.5,'  UP5=',G13.5/                 HVS00510
     &' AX=  UN1=',G13.5,'  UN2=',G15.5,'  UN5=',G13.5/                 HVS00520
     &' AX=  EX1=',G13.5,'  EX2=',G15.5,'  EX5=',G13.5/                 HVS00530
     &' AX=  EX4=',G13.5,'  EX5=',G15.5,'  EX6=',G13.5)                 HVS00540
C                                                                       HVS00550
C..... SET INITIAL STIFFNESS AND RETURN                                 HVS00560
      IF (NRL.EQ.0) THEN                                                HVS00570
                IF (P.LE.0) THEN                                        HVS00580
                    NRL=1                                               HVS00590
                    HRLN=1.1                                            HVS00600
                    S=KC                                                HVS00610
                    SEQ=KC                                              HVS00620
                    A=P                                                 HVS00630
                ELSE                                                    HVS00640
                    NRL=2                                               HVS00650
                    HRLN=2.1                                            HVS00660
                    S=KT1                                               HVS00670
                    SEQ=KC                                              HVS00680
                    A=P                                                 HVS00690
                ENDIF                                                   HVS00700
                RETURN                                                  HVS00710
      ENDIF                                                             HVS00720
C                                                                       HVS00730
C..... RETURN IF NO STEP HAS BEEN TAKEN                                 HVS00740
      IF (P.EQ.PL .OR. D.EQ.DL ) RETURN                                 HVS00750
C                                                                       HVS00760
C..... SET LOADING FLAGS                                                HVS00770
      PGTPL=P.GT.PL                                                     HVS00780
      PLTPL=P.LT.PL                                                     HVS00790
C                                                                       HVS00800
C..... CALC EXCURSION RATIO AT EVERY ZERO CROSSING...                   HVS00810
      IF (P*PL.LE.0 .AND. PL.GT.0)                                      HVS00820
     &    CALL ONEC(0,UPOS,EXCR,DVT,D,ESE,PSE,PSEOLD,CSE)               HVS00830
      IF (P*PL.LE.0 .AND. PL.LE.0) THEN                                 HVS00840
          EXCR(4)=0                                                     HVS00850
          EXCR(5)=0                                                     HVS00860
          EXCR(6)=0                                                     HVS00870
      ENDIF                                                             HVS00880
C                                                                       HVS00890
C..... CALC DUCTILITY IF LOAD GT PAST LOAD                              HVS00900
C                                                                       HVS00910
C     DUCTILITY FOR POSITIVE LOADS ONLY                                 HVS00920
      IF (P.GT.0 .AND. PGTPL)                                           HVS00930
     &    CALL ONEC(1,UPOS,EXCR,DVT,D ,ESE(2),PSE,PSEOLD,CSE)           HVS00940
C                                                                       HVS00950
 999  CALL IDRECT(IDR,IDRV,IDRV0,P,PL,V,VL)                             HVS00960
      IDR=IDR                                                           HVS00970
                                                                        HVS00980
   1  IF (NRL.EQ.0) NRL=1                                               HVS00990
C                                                                       HVS01000
      RULEI=NRL                                                         HVS01010
C                                                                       HVS01020
      GO TO (100,200,300,400,500,600,700,800,900,                       HVS01030
     &                    1000,1100,1200,1300),RULEI                    HVS01040
C                                                                       HVS01050
C RULE 1 - LOADING AND UNLOADING ON THE COMPRESSION BRANCH BEFORE YEILD HVS01060
C                                                                       HVS01070
 100  SEQ=KC                                                            HVS01080
      IF (P.GT.0) GO TO 200                                             HVS01090
      S=KC                                                              HVS01100
      NRL=1                                                             HVS01110
      HRLN=1.1                                                          HVS01120
      IF (PLTPL) A=MIN(10.*P,-2.*PV)                                    HVS01130
      IF (PGTPL) A=0.                                                   HVS01140
      RETURN                                                            HVS01150
C                                                                       HVS01160
C RULE 2 - TENSION LOADING BEFORE YEILD                                 HVS01170
C                                                                       HVS01180
 200  IF (DR) 250,250,210                                               HVS01190
 210  IF (P.GE.PV) GO TO 600                                            HVS01200
      S=KT1                                                             HVS01210
      NRL=2                                                             HVS01220
      HRLN=2.1                                                          HVS01230
      A=PV                                                              HVS01240
      RETURN                                                            HVS01250
 250  IF (PL.LE.0) GO TO 210                                            HVS01260
      FMAX=AMAX1(P,FMAX)                                                HVS01270
      DMAX=AMAX1(D,DMAX)                                                HVS01280
      FX  =FMAX-PV                                                      HVS01290
      DX  =DMAX-PV/KC                                                   HVS01300
      S=KC                                                              HVS01310
      NRL=3                                                             HVS01320
      HRLN=2.2                                                          HVS01330
      A=FX                                                              HVS01340
      RETURN                                                            HVS01350
C                                                                       HVS01360
C RULE 3 - UNLOADING ON TENSION BEFORE YEILD                            HVS01370
C                                                                       HVS01380
 500  IF (PGTPL) GO TO 500                                              HVS01390
      IF (P.GT.0)  GO TO 550                                            HVS01400
 510  IF (KC.NE.KT1) THEN                                               HVS01410
        CALL INTSCT(DI,PI,D,P,KC,DX,FX,KT1)                             HVS01420
      ELSE                                                              HVS01430
        PI=P                                                            HVS01440
        DI=D                                                            HVS01450
      ENDIF                                                             HVS01460
      IF (P.LE.PI) GO TO 520                                            HVS01470
      S=KC                                                              HVS01480
      NRL=3                                                             HVS01490
      HRLN=5.11                                                         HVS01500
      A=PI                                                              HVS01510
      RETURN                                                            HVS01520
 520  S=KT1                                                             HVS01530
      NRL=4                                                             HVS01540
      HRLN=5.12                                                         HVS01550
      A=PV                                                              HVS01560
      RETURN                                                            HVS01570
 550  IF (P.GE.FMAX) GO TO 200                                          HVS01580
      S=KC                                                              HVS01590
      NRL=5                                                             HVS01600
      HRLN=5.2                                                          HVS01610
      A=FX                                                              HVS01620
      RETURN                                                            HVS01630
C                                                                       HVS01640
C RULE 4 - LOADING TOWARDS Y' FROM FX.                                  HVS01650
C                                                                       HVS01660
 400  IF (DR) 500,500,410                                               HVS01670
 410  IF (ABS(P).GE.PV) GO TO 100                                       HVS01680
      S=KT1                                                             HVS01690
      NRL=4                                                             HVS01700
      HRLN=4.1                                                          HVS01710
      A=-PV                                                             HVS01720
      RETURN                                                            HVS01730
C                                                                       HVS01740
C RULE 5 - LOADING AND UNLOADING BETWEEN BRANCHES O-Y AND FX-Y'         HVS01750
C                                                                       HVS01760
 500  IF (KC.NE.KT1) THEN                                               HVS01770
        CALL INTSCT(OITENS,PITENS,O,P,KC,DVT,PV,KT1)                    HVS01780
        CALL INTSCT(OICOMP,PICOMP,O,P,KC,DX ,FX,KT1)                    HVS01790
      ELSE                                                              HVS01800
        PITENS=P                                                        HVS01810
        OITENS=O                                                        HVS01820
        PICOMP=P                                                        HVS01830
        OICOMP=O                                                        HVS01840
      ENDIF                                                             HVS01850
      IF (P.LE.PICOMP) GO TO 550                                        HVS01860
      IF (P.GE.PITENS) GO TO 210                                        HVS01870
      S=KC                                                              HVS01880
      NRL=5                                                             HVS01890
      HRLN=5.1                                                          HVS01900
      IF (PGTPL ) A=PITENS                                              HVS01910
      IF (PLTPL ) A=PICOMP                                              HVS01920
      RETURN                                                            HVS01930
C                                                                       HVS01940
C RULE 6 - LOADING AFTER TENSION YEILD                                  HVS01950
C                                                                       HVS01960
 600  IF (DR) 650,650,610                                               HVS01970
 610  S=KT2                                                             HVS01980
      NRL=6                                                             HVS01990
      HRLN=6.1                                                          HVS02000
      A=5*P                                                             HVS02010
C.......... CALC UNLOADING CURVE FOR ENERGY INFO                        HVS02020
          FMAX=AMAX1(P,FMAX)                                            HVS02030
          DMAX=AMAX1(D,DMAX)                                            HVS02040
          KR=KC*MIN(1.,(DVT/DMAX)**ALPHA)                               HVS02050
          FX  =FMAX-PV                                                  HVS02060
          DX  =DMAX-PV/KR                                               HVS02070
          KS  =(PV+FX)/(DX-DVC)                                         HVS02080
C-------- FORCE KR GE KS, TO PREVENT UNSTABLE LOOPS...                  HVS02090
          IF (KS,GT.KR)  KR=(FMAX+PV)/(DMAX-DVC)                       HVS02100
          SEQ=KR                                                        HVS02110
      RETURN                                                            HVS02120
 650  FMAX=AMAX1(P,FMAX)                                                HVS02130
      DMAX=AMAX1(D,DMAX)                                                HVS02140
      KR=KC*MIN(1.,(DVT/DMAX)**ALPHA)                                   HVS02150
      FX  =FMAX-PV                                                      HVS02160
      DX  =DMAX-PV/KR                                                   HVS02170
      KS  =(PV+FX)/(DX-DVC)                                             HVS02180
C-------- FORCE KR GE KS, TO PREVENT UNSTABLE LOOPS...                  HVS02190
      IF (KS,GT.KR) THEN                                                HVS02200
        KR=(FMAX+PV)/(DMAX-DVC)                                         HVS02210
        KS=KR                                                           HVS02220
```

```
        DX=DMAX-PV/KR
      ENDIF
      DP  =DVC + BETA*(DX-DVC)
      FP  =FX - (DX-DP)*KS
      KT  =(2*PV+FP)/(DP-2*DVC)
      DQ  =DVC + PV/KR
      S=KR
      NRL=7
      HRLN=6.2
      A=FX
      RETURN
C
C RULE 7 - UNLOADING AFTER YEILD
C
 700  SEQ=KR
      IF (IDR) 750,750,710
 710  IF (P.GE.FMAX) GO TO 600
      S=KR
      NRL=7
      HRLN=7.1
      A=FMAX
      RETURN
 750  IF (P.LE.FX) GO TO 800
      S=KR
      NRL=7
      HRLN=7.2
      A=FX
      RETURN
C
C RULE 8 - ON BRANCH BETWEEN DX & DP
C
 800  IF (PGTPL) THEN
        IF (KR.EQ.KS) GO TO 1200
        GO TO 1500
      ENDIF
      IF (P.LE.FP) GO TO 900
      S=KS
      NRL=8
      HRLN=8.1
      A=FP
      RETURN
C
C RULE 9 - ON BRANCH FP-Y*
C
 900  IF (PGTPL) GO TO 1500
      IF (P.LE.(-2*PV) ) GO TO 1000
      S=KT
      NRL=9
      HRLN=9.1
      A=-2*PV
      RETURN
C
C RULE 10 - ON THE COMPRESSION CURVE AFTER YELDING
C
 1000 SEQ=  1. / ( 1./KC + (4./KR - 4./KC)*(PV/P)**2 )
C
      IF (IDR) 1050,1050,1010
 1010 S=KC
      NRL=10
      HRLN=10.1
      A=10*P
      RETURN
 1050 IF (ABS(P).LE.PV) GO TO 1100
      S=KC
      NRL=10
      HRLN=10.2
      A=-PV
      RETURN
C
C RULE 11 - UNLOADING BETHEEN Y* & Q
C
 1100 SEQ=KR
      IF (ABS(P).GT.PV ) GO TO 1000
      IF (    P .GE. 0.) GO TO 1200
      S=KR
      NRL=11
      HRLN=11.1
      IF (PGTPL) A=0
      IF (PLTPL) A=-PV
      RETURN
C
C RULE 12 - LOADING FORM Q TO M
C
 1200 IF (PLTPL) THEN
        IF (KS.EQ.KR .AND. P.GT.FP) GO TO 800
        GO TO 1500
      ENDIF
      IF ( P .GE.FMAX) GO TO 600
      S=KS
      NRL=12
      HRLN=12.1
      A=FMAX
      RETURN
C
C RULE 13 - LOADING AND UNLOADING INSIDE Q-M-D-P-Y*-Y*
C
1500  IF (KR.NE.KS) THEN
        CALL INTSCT(DITENS,PITENS,D,P,KR,DMAX,FMAX,KS)
        CALL INTSCT(DICOMP,PICOMP,D,P,KR,DX ,FX,KS )
      ELSE
        IF (PGTPL) GO TO 1200
        PITENS=P
        OITENS=0
        PICOMP=P
        OICOMP=0
      ENDIF

C--------IS POINT WITHIN PY'-FP-PY* ?
      IF (PICOMP.LE.FP)  THEN
        IF (KC.NE.KT) THEN
            CALL INTSCT(DICOMP,PICOMP,D,P,KC,DP,FP,KT )
        ELSE
            IF (PLTPL) GO TO  900
            PICOMP=P
            DICOMP=0
        ENDIF
        IF (KR.EQ.KS .AND. PLTPL) GO TO  800
        S2=KT
      ELSE
        S2=KR
      ENDIF
       IF (PGTPL) THEN
         IF (P.GT.PITENS) GO TO 1200
         A=PITENS
         HRLN=15.1
       ENDIF
       IF (PLTPL) THEN
         IF (P.LT.PICOMP) GO TO 800
         A=PICOMP
         HRLN=15.2
       ENDIF
      S=MAX(KR,S2)
      NRL=15
      RETURN
C
      END
C=======================================================================
C==  BENDING HYSTERESIS MODEL..... HYST05 ..............................
C=======================================================================
C
C     INPUT VARIABLES
C     P       = CURRENT LOAD
C     D       = CURRENT DISPLACEMENT
C     PL      = LAST LOAD
C     VEL     = PRODUCT OF LAST AND CURRENT VELOCITY
C     NB      = NUMBER OF BACKBONE CURVE POINTS
C     NI      = MAXIMUN NUMBER OF SECONDARY LOOPS
C     PC      = CRACKING LOAD
C     DC      = CRACKING DISPLACEMENT
C     PB(j)   = BACKBONE CURVE LOADING POINT j, j=1,NB
C     DB(j)   = BACKBONE CURVE DISPLACEMENT POINT j,1,NB
C     SI      = INITIAL STIFFNESS, PC/DC
C
C     OUTPUT VARIABLES
C     ....AN INITIAL VALUE OF ZERO , OR FALSE
C               IS REQUIRED FOR ALL OUTPUT VARIABLES...
C     K       = STIFFNESS
C     RULE    = RULE NUMBER, STORED IN THE FORMAT RN.DIR UPON EXIT OF
C               THIS SU_BROU_TINE
C     A       = LOAD LIMIT BEFORE NEXT RULE CHANGE, IN THE CURRENT DIR
C     PMG     = MAXIMUN PAST LOAD IN THE POSITIVE DIRECTION
C     PMM     = MAXIMUN PAST LOAD IN THE NEGATIVE DIRECTION
C     DMG     = MAXIMUN PAST DISPLACEMENT IN THE POSITIVE DIRECTION
C     DMM     = MAXIMUN PAST DISPLACEMENT IN THE NEGATIVE DIRECTION
C     PR(I)   = SECONDARY LOOP LOADING POINT, I
C     DR(I)   = SECONDARY LOOP DISPLACEMENT POINT, I
C     FR(I)   = SECONDARY LOOP FLAG
C     IR      = NUMBER OF SECONDARY LOOPS ACTIVE
C     ALPHA1  = POSITIVE LOAD DEGRADING STIFFNESS FACTOR
C     ALPHA2  = NEGATIVE LOAD DEGRADING STIFFNESS FACTOR
C     BACKB   = BACKBONE LOADING FLAG
C
C     INTERNAL VARIABLES
C     J       = BACKBONE CURVE POINT # IN THE CURRENT DIRECTION
C     JD      = TEMPORARY VARIABLE
C     I,IT    = SECONDARY LOOP NUMBER
C     DIR     = CURRENT DIRECTION
C               DIR=1, POSITIVE LOADING
C               DIR=2, POSITIVE UNLOADING
C               DIR=3, NEGATIVE LOADING
C               DIR=4, NEGATIVE UNLOADING
C     DIRL    = LAST DIRECTION
C     IRULE   = INTEGER VALUE OF RULE #
C     LRULE   = INTEGER VALUE OF LAST RULE #
C     REVRSL  = LOAD REVERSAL FLAG
C     PRINT   = PRINT FLAG, USED FOR DEBUGGING
C     AP      = ABSOLUTE VALUE OF THE LOAD, P
C     PM      = MAXIMUM PAST LOAD IN THE CURRENT DIRECTION
C     DM      = MAXIMUM PAST DISPLACEMENT IN THE CURRENT DIRECTION
C     DMAX    = MAXIMUM PAST DISPLACEMENT IN BOTH DIRECTIONS
C     Q1      = ONE QUARTER OF PM, USED BY UNLOADING CURVES
C     Q3      = THREE QUARTERS OF PM, USED BY UNLOADING CURVES
C     DQ1     = DISPLACEMENT AT Q1 ON UNLOADING BRANCH
C     DQ3     = DISPLACEMENT AT Q3 ON UNLOADING BRANCH
C     DD1     = INTERMEDIATE DISPLACEMENT INTERCEPT OF UNLOADING BRANCH
C     DD2     = INTERMEDIATE DISPLACEMENT INTERCEPT OF UNLOADING BRANCH
C     DD      = DISPLACEMENT INTERCEPT OF UNLOADING BRANCH
C     DP      = DIFFERANCE IN LOAD, USED FOR CALCULATING K
C     DD      = DIFFERANCE IN DISPLACEMENT, USED FOR CALCULATING K
C     PK      = LOAD INTERCEPT OF LOADING AND BACKBONE CURVE
C     DK      = DISPLACEMENT INTERCEPT OF LOADING AND BACKBONE CURVE
C     PC5     = PC/5
C     DC5     = DISPLACEMENT AT PC5
C     P2,D2   = INTERSECTION OF LOADING AND UNLOADING CURVES
C     PI,DI   = SMALL LOOP UNLOADING POINTS
C     X       = MAXIMUN DISPLACEMENT
C     K8      = STIFFNESS FROM RULE 8
C     FLAG1   = UPPER SECONDARY LOOP FLAG
C     FLAG2   = LOWER SECONDARY LOOP FLAG
C     ALPHAA  = DEGRADING FACTOR FOR FIRST CYCLE
C     ALPHAB  = DEGRADING FACTOR FOR SUBSEQUENT CYCLES
C     S       = TEMPORARY STIFFNESS
C     S1      = UNLOADING STIFFNESS FROM FS1
C     S2      = UNLOADING STIFFNESS FROM FS2
C     S3      = UNLOADING STIFFNESS FROM FS3
C     SU      = STIFFNESS BETWEEN CURRENT AND RELOADING POINT
C     S02     = STIFFNESS BETWEEN PA AND DD
C     S05     = STIFFNESS BETWEEN PB AND DD
C
C     INTERNAL FUNCTIONS
C     FD2     = DISPLACEMENT INTERCEPT OF LOADING AND UNLOADING CURVE
C     FS1     = UNLOADING STIFFNESS WHEN P>.75 PMAX
C     FS2     = UNLOADING STIFFNESS WHEN .75 PMAX > P > .25 PMAX
C     FS3     = UNLOADING STIFFNESS WHEN .25 PMAX > P
C     FSL     = LOADING STIFFNESS WHEN .P < PC/5 WITHOUT REVERSAL...
C
C     SU_BROU_TINES CALLED
C     DIRECT  = DETERMINES THE VALUES OF DIR AND DIRL
C     INTSCT  = DETERMINES THE INTERSECTION OF TWO LINES IN POINT-SLOPE
C               FORM
C     KMIN    = DETERMINES THE STIFFNESS BASED ON DP,DD AND K_DEFULT
C
      SUBROUTINE HYST05(P,D,PL,DL,VEL,NB,NI,PC,DC,PB,DB,SI,CSE,
     & K,RULE,RDIR,A,PMG,PMM,DMG,DMM,BACKB,ALPHA1,ALPHA2,
     & IR,SR1,SR2,SR5,SRM,PR,DR,FR,PRINT,
     & SE,DV,ESE,PSE,PSEOLD,UPOS,UNEG,EXCR)
C
      IMPLICIT REAL(A-H,K,O-Z)
      IMPLICIT INTEGER(I,J,L-N)
      INTEGER DIR,DIRL
      CHARACTER*4  FR(1)
      LOGICAL  BACKB,REVRSL,PRINT,BTEST
      DIMENSION PB(1),DB(1),PR(1),DR(1),UPOS(5),UNEG(5),EXCR(6),ESE(5)
      DATA ALPHAA/1.129/,ALPHAB/1.029/
C
      FS1(X)= SI *(DC/X)**.294
      FS2(X)= SI *(0.8544*(DC/X-1) +1)
      FS3(X)= SI *(0.9092*(DC/X-1) +1)
```

```
      FSL(K)= SI *(DC/X)**.285                                          HVS01160
      FD2(K)= DM-.05*PM/(FSL(K))                                        HVS01170
C                                                                       HVS01180
C                                                                       HVS01190
 1000 IF (PRINT) WRITE (6,1010) P,D,PL,DL,NB,NI,PC,DC,SI,K,             HVS01200
     &      RULE,A,PMG,PMM,DMG,DMM,BACKB,IR,DIR,                       HVS01210
     & SE,DV,ESE(1),PSE,PSEOLD,UPOS,CSE,UNEG,EXCR,VEL                  HVS01220
 1010 FORMAT(                                                           HVS01230
     &' B1-    P=',G13.5,'    D=',G13.5,'    PL=',G13.5,'    DL=',G13.5/ HVS01240
     &' B1-   NB=',G13.5,'   NI=',G13.5,'    PC=',G13.5,'    DC=',G13.5/ HVS01250
     &' B1-   SI=',G13.5,'    K=',G13.5,' RULE=',G13.5,'    A=',G13.5/  HVS01260
     &' B1-  PMG=',G13.5,'  PMM=',G13.5,'   DMG=',G13.5,'  DMM=',G13.5/  HVS01270
     &' B1-BACKB=',G13.5,'   IR=',G13.5,'   DIR=',G13.5,'   SE=',G13.5/  HVS01280
     &' B1-   DV=',G13.5,'  ESE=',G13.5,'   PSE=',G13.5,' PSE0=',G13.5/  HVS01290
     &' B1-  UP1=',G13.5,'  UP2=',G13.5,'   UP3=',G13.5,'  CSE=',G13.5/  HVS01300
     &' B1-  UN1=',G13.5,'  UN2=',G13.5,'   UN3=',G13.5/                HVS01310
     &' B1-  EX1=',G13.5,'  EX2=',G13.5,'   EX3=',G13.5/                HVS01320
     &' B1-  EX4=',G13.5,'  EX5=',G13.5,'   EX6=',G13.5/'  VEL=',G13.5) HVS01330
      IF (IR.GT.0.AND.PRINT) WRITE (6,1020) (I,PR(I),DR(I),FR(I),I=1,IR) HVS01340
 1020 FORMAT                                                            HVS01350
     & (' B1-    I=',G13.5,'    PR=',G13.5,'    DR=',G13.5,'    FR=',A) HVS01360
C                                                                       HVS01370
      AP=ABS(P)                                                         HVS01380
C..... DETERMINE IF ELASTIC                                             HVS01390
      IF (RULE.EQ.0 .AND. AP.LT.PC ) THEN                               HVS01400
            K=SI                                                        HVS01410
            SE=K                                                        HVS01420
            RETURN                                                      HVS01430
      ENDIF                                                             HVS01440
C                                                                       HVS01450
C..... DETERMINE IF WALL IS INACTIVE. IF SO RETURN                      HVS01460
      IF (RULE.LT.0.0) RETURN                                           HVS01470
C                                                                       HVS01480
C..... DETERMINE CURRENT DIRECTION                                      HVS01490
      LRULE=INT(RULE)                                                   HVS01500
      DIR =RDIR                                                         HVS01510
      CALL DIRECT(DIR,DIRL,P,PL,VEL)                                    HVS01520
      RDIR =DIR                                                         HVS01530
      IF (PRINT) WRITE (6,1050) DIR,DIRL                                HVS01540
 1050 FORMAT                                                            HVS01550
     & (' B1-  DIR=',I8  ,'  DIRL=',I8)                                 HVS01560
C                                                                       HVS01570
C..... DETERMINE MAXIMUM AND MINIMUM VALUES                             HVS01580
            PMG=MAX(P,PMG, PC)                                          HVS01590
            DMG=MAX(D,DMG, DC)                                          HVS01600
            PMM=MIN(P,PMM,-PC)                                          HVS01610
            DMM=MIN(D,DMM,-DC)                                          HVS01620
            DMAX=MAX(DMG,-DMM)                                          HVS01630
            PMAX=MAX(PMG,-PMM)                                          HVS01640
C                                                                       HVS01650
C..... SET MAXIMUM AND MINIMUM VALUES...                                HVS01660
      IF (DIR.EQ.1 .OR. DIR.EQ.2 ) THEN                                 HVS01670
            PM=PMG                                                      HVS01680
            DM=DMG                                                      HVS01690
      ELSE IF (DIR.EQ.3 .OR. DIR.EQ.4) THEN                             HVS01700
            PM=PMM                                                      HVS01710
            DM=DMM                                                      HVS01720
      ENDIF                                                             HVS01730
C                                                                       HVS01740
C..... CALCULATE EQUIVALENT UNLOADING STIFFNESS FOR ENERGY...           HVS01750
         IF (BACKB .OR. RULE.EQ.1.0) THEN                               HVS01760
            S1=FS1(DMAX)                                                HVS01770
            S2=FS2(DMAX)                                                HVS01780
            S3=FS3(DMAX)                                                HVS01790
            DQ =DM - PM*(DMM-DMG)/(PMM-PMG)                             HVS01800
            Q1=PM/4                                                     HVS01810
            Q5=.5*Q1                                                    HVS01820
            DQ5=DM -0.25*PM/S1                                          HVS01830
            S02=KMIN(<DQ5-D0),Q5,S1)                                    HVS01840
            DQ1=DQ5-0.50*PM/MAX(S2,S02)                                 HVS01850
            S03=KMIN(<DQ1-D0),Q1,S1)                                    HVS01860
            SE= 0.5 * PM**2 / (                                         HVS01870
     &       (PM+Q5)*0.5*(DM-DQ5)                                       HVS01880
     &      +(Q5+Q1)*0.5*(DQ5-DQ1)                                      HVS01890
     &      +(   Q1)*0.5*(   Q1)/MAX(S3,S05) )                         HVS01900
         ENDIF                                                          HVS01910
C                                                                       HVS01920
C..... CHECK TO SEE IF LAST RULE IS STILL IN EFFECT                     HVS01930
      IF (DIR.EQ.DIRL .AND.                                             HVS01940
     &  ((DIR.EQ.1 .AND. P.LT.A) .OR. (DIR.EQ.2 .AND. P.GT.A)  .OR.     HVS01950
     &   (DIR.EQ.3 .AND. P.GT.A) .OR. (DIR.EQ.4 .AND. P.LT.A))) THEN    HVS01960
         RETURN                                                         HVS01970
       ENDIF                                                            HVS01980
C                                                                       HVS01990
C..... CALC EXCURSION RATIO AT EVERY ZERO CROSSING...                   HVS02000
      IF (P*PL.LE.0) CALL DNE(0,UPOS,EXCR, DV,DLI,ESE,PSE,PSEOLD,CSE)   HVS02010
C                                                                       HVS02020
C..... ERASE SMALL LOOP FLAGS                                           HVS02030
      IF (IR.GT.0) THEN                                                 HVS02040
        IT=IR                                                           HVS02050
        DO 10 I=1,IT                                                    HVS02060
          IF ((FR(I).EQ.'L' .AND. ( P.LE.PR(I) .OR. D.LE.DR(I) ))  .OR. HVS02070
     &        (FR(I).EQ.'7' .AND. ( P.GE.PR(I) .OR. D.GE.DR(I) ))) THEN HVS02080
            IR=I-1                                                      HVS02090
            GO TO 15                                                    HVS02100
          ENDIF                                                         HVS02110
   10   CONTINUE                                                        HVS02120
        ENDIF                                                           HVS02130
   15 I=IR+1                                                            HVS02140
C                                                                       HVS02150
      IF (DIR.EQ.1 .OR. DIR.EQ.3 )  THEN                                HVS02160
C..... LOADING RULES ..................................... RULES 1,6-11  HVS02170
C                                                                       HVS02180
C..... LOADING ON BACKBONE                                      RULE 1  HVS02190
    1     IF (BACKB .OR. RULE.EQ.0.0) THEN                              HVS02200
            K=0.                                                        HVS02210
            DO 18 J=2,NB                                                HVS02220
               DD=DB(J)-ABS(D)                                          HVS02230
               DP=PB(J)-AP                                              HVS02240
               IF (AP.LT.PB(J) .AND. (DD*DP).GT.0.) THEN                HVS02250
                 S=DP/DD                                                HVS02260
                 IF (S.GT.K) THEN                                       HVS02270
                   K=S                                                  HVS02280
                   A=SIGN(PB(J),P)                                      HVS02290
                 ENDIF                                                  HVS02300
               ENDIF                                                    HVS02310
   18       CONTINUE                                                    HVS02320
            RULE=1                                                      HVS02330
            BACKB=.TRUE.                                                HVS02340
            IR=0                                                        HVS02350
            IF (DIR.EQ.1) THEN                                          HVS02360
               ALPHA1=ALPHAA                                            HVS02370
            ELSE                                                        HVS02380
               ALPHA2=ALPHAA                                            HVS02390
            ENDIF                                                       HVS02400
C                                                                       HVS02410
C..... LOADING AFTER UNLOADING                                          HVS02420
         ELSE                                                           HVS02430
C..............  RELOADING INSIDE SMALL LOOPS                  RULE 11  HVS02440
   20       I=I-1                                                       HVS02450
            IF (I.GT.0 ) THEN                                           HVS02460
               IF (.NOT.(( FR(I).EQ.'7' .AND. DIR.EQ.1 ) .OR.           HVS02470
     &                   ( FR(I).EQ.'L' .AND. DIR.EQ.3 )))  GO TO 20   HVS02480
               K=KMIN(<D-DR(I)),(P-PR(I)),SI)                           HVS02490
               A=PR(I)                                                  HVS02500
               RULE=11+I/1000.                                          HVS02510
C                                                                       HVS02520
C..............  INITIAL RELOADING BELOW PC/S                           HVS02530
C..............................  WITH REVERSAL                 RULE 8   HVS02540
            ELSE IF (AP.LT.(PC/S) )THEN                                 HVS02550
              BACKB=.FALSE.                                             HVS02560
              IF ((DIR.EQ.1 .AND. DIRL.EQ.4) .OR.                       HVS02570
     &            (DIR.EQ.3 .AND. DIRL.EQ.2)) THEN                      HVS02580
               P2=.95*PM                                                HVS02590
               D2=FD2(DMAX)                                             HVS02600
               K=KMIN(<D-D2),(P-P2),SI)                                 HVS02610
               K8=K                                                     HVS02620
               A= P2                                                    HVS02630
               RULE=8                                                   HVS02640
C                                                                       HVS02650
               PCS=SIGN(PC/S,PM)                                        HVS02660
               DD1=DM-PM*(1./FS1(DMAX)+2./FS2(DMAX)+1./FS3(DMAX))/4.    HVS02670
               DD2=DM - PM*(DMM-DMG)/(PMM-PMG)                          HVS02680
               IF (DIR.EQ.1) DD=MAX(DD1,DD2)                            HVS02690
               IF (DIR.EQ.3) DD=MIN(DD1,DD2)                            HVS02700
               DCS=DD+PCS/FSL(DMAX)                                     HVS02710
               S=KMIN(<D-DCS),(P-PCS),SI)                               HVS02720
               IF (S.GT.K) THEN                                         HVS02730
                   K=S                                                  HVS02740
                   A= PCS                                               HVS02750
                   RULE=8.1                                             HVS02760
               ENDIF                                                    HVS02770
C                                                                       HVS02780
C..............................  WITH REVERSAL, FAR SIDE LOADING RULE 9  HVS02790
               IF ((DIR.EQ.1 .AND. D.LT.0) .OR.                         HVS02800
     &             (DIR.EQ.3 .AND. D.GT.0)) THEN                        HVS02810
               K=K8                                                     HVS02820
               DO 40 J=2,NB                                             HVS02830
                  IF ( PB(J) .LT.ABS(PM) ) GO TO 40                     HVS02840
                  DD= -SIGN(DB(J),D)-D                                  HVS02850
                  DP= -SIGN(PB(J),D)-P                                  HVS02860
                  IF ( (DD*DP).LE.0.) GO TO 40                          HVS02870
                    S=DP/DD                                             HVS02880
                    IF (S.GT.K .AND. S.LE.SI) THEN                      HVS02890
                      K=S                                               HVS02900
                      A=SIGN(PB(J),PM)                                  HVS02910
                      RULE=9                                            HVS02920
                      BACKB=.TRUE.                                      HVS02930
                    ENDIF                                               HVS02940
   40             CONTINUE                                             HVS02950
               ENDIF                                                    HVS02960
C                                                                       HVS02970
C..............................  WITHOUT REVERSAL              RULE 6   HVS02980
            ELSE                                                        HVS02990
              K=FSL(DMAX)                                               HVS03000
              A= SIGN(PC/S),PM)                                         HVS03010
              RULE=6                                                    HVS03020
            ENDIF                                                       HVS03030
C                                                                       HVS03040
C..............  MIDRANGE RELOADING CURVE                      RULE 7   HVS03050
         ELSE IF (AP.LT.ABS(.95*PM) ) THEN                              HVS03060
            BACKB=.FALSE.                                               HVS03070
            P2=.95*PM                                                   HVS03080
            D2=FD2(DMAX)                                                HVS03090
            K=KMIN(<D-D2),(P-P2),SI)                                    HVS03100
            A=P2                                                        HVS03110
            RULE=7                                                      HVS03120
C                                                                       HVS03130
C..............  LOADING TOWARDS THE BACKBONE CURVE             RULE 10  HVS03140
         ELSE                                                           HVS03150
            BACKB=.TRUE.                                                HVS03160
            IF (AP.GE.ABS(PM) .OR. (ABS(DM)).LE.DC) GO TO 1             HVS03170
            IR=0                                                        HVS03180
            IF (DIR.EQ.1) THEN                                          HVS03190
               IF (ALPHA1.NE.ALPHAA .AND. ALPHA1.NE.ALPHAB)             HVS03200
     &                ALPHA1=ALPHAA                                     HVS03210
               DD=ALPHA1*DM-D                                           HVS03220
            ELSE                                                        HVS03230
               IF (ALPHA2.NE.ALPHAA .AND. ALPHA2.NE.ALPHAB)             HVS03240
     &                ALPHA2=ALPHAA                                     HVS03250
               DD=ALPHA2*DM-D                                           HVS03260
            ENDIF                                                       HVS03270
            DP=PM-P                                                     HVS03280
            IF ((DP*DD).GT.0) THEN                                      HVS03290
              K=MIN( (DP/DD),SI)                                        HVS03300
            ELSE                                                        HVS03310
              K=SI                                                      HVS03320
            ENDIF                                                       HVS03330
            DO 50 J=2,NB                                                HVS03340
              IF ( PB(J).LE.ABS(PM) ) GO TO 50                          HVS03350
              JO=J-1                                                    HVS03360
              DD=DB(J)-DB(JO)                                           HVS03370
              DP=PB(J)-PB(JO)                                           HVS03380
              IF ((DP*DD).LE.0.) GO TO 50                               HVS03390
               CALL INTSCT(DX,PX,D,P,K,SIGN(DB(J),P),                   HVS03400
     &                     SIGN(PB(J),P),(DP/DD) )                      HVS03410
               PX=PX                                                    HVS03420
               DX=DX                                                    HVS03430
               IF (ABS(DX).GE.DB(JO) .AND. ABS(DX).LE.DB(J).AND.        HVS03440
     &            ABS(PX).GT.ABS(PM)  .AND.  PM*PX.GT.0) THEN           HVS03450
                 IF (AP.GT.ABS(PX)) GO TO 1                             HVS03460
                 A= PX                                                  HVS03470
                 RULE=10                                                HVS03480
                 IF (DIR.EQ.1)  THEN                                    HVS03490
                    ALPHA1=ALPHAB                                       HVS03500
                 ELSE                                                   HVS03510
                    ALPHA2=ALPHAB                                       HVS03520
                 ENDIF                                                  HVS03530
                 GO TO 100                                              HVS03540
               ENDIF                                                    HVS03550
   50        CONTINUE                                                   HVS03560
             GO TO 1                                                    HVS03570
           ENDIF                                                        HVS03580
         ENDIF                                                          HVS03590
C                                                                       HVS03600
C                                                                       HVS03610
      ELSE                                                              HVS03620
C..... UNLOADING RULES ..................................... RULES 2-5   HVS03630
```

```
C                                                                       HVS00540
C..... INITILIZE VALUES                                                 HVS00550
              BACKB=.FALSE.                                             HVS00560
              S1=FS1(DMAX)                                              HVS00570
              S2=FS2(DMAX)                                              HVS00580
              S3=FS3(DMAX)                                              HVS00590
              DO =DM - PM*(DMM-DMG)/(PMM-PMG)                           HVS00600
              Q1=PM/4                                                   HVS00610
              Q3=3*Q1                                                   HVS00620
              DQ3=DM -0.25*PM/S1                                        HVS00630
              SO2=KMINK(DQ3-DO),Q3,S1)                                  HVS00640
              DQ1=DQ3-0.50*PM/MAX(S2,SO2)                               HVS00650
              SO3=KMINK(DQ1-DO),Q1,S1)                                  HVS00660
              SU=0.                                                     HVS00670
C                                                                       HVS00680
C                                                                       HVS00690
C.............. UNLOADING INSIDE SMALL POSITIVE LOOPS            RULE 5 HVS00700
              I=IR+1                                                    HVS00710
   70         I=I-1                                                     HVS00720
              IF (I.GT.0) THEN                                          HVS00730
                IF (.NOT.((DIR.EQ.2 .AND. FR(I).EQ.'L')    .OR.         HVS00740
     &                   (DIR.EQ.4 .AND. FR(I).EQ.'7')) ) GO TO 70     HVS00750
                IF ((DIR.EQ.2 .AND.( (P.GT.Q3 .AND. PR(I).GT.Q3) .OR.   HVS00760
     &               (Q3.GE.P .AND.   P.GT.Q1 .AND. PR(I).GT.Q1) .OR.   HVS00770
     &               (Q1.GE.P .AND.   PR(I).GT.0 )) .OR.                HVS00780
     &              (DIR.EQ.4 .AND.( (P.LT.Q3 .AND. PR(I).LT.Q3) .OR.   HVS00790
     &               (Q3.LE.P .AND.   P.LT.Q1 .AND. PR(I).LT.Q1) .OR.   HVS00800
     &               (Q1.LE.P .AND.   PR(I).LT.0 )) ) THEN              HVS00810
                  K=(PR(I)-P)/(DR(I)-0)                                 HVS00820
                  IF (K.LE.0 .OR. K.GT.SI) GO TO 70                     HVS00830
                  A=PR(I)                                               HVS00840
                  RULE=5+I/1000.                                        HVS00850
                  GO TO 100                                             HVS00860
                ELSE                                                    HVS00870
                  SU=KMINK(DR(I)-0),(PR(I)-P),SI)                       HVS00880
                ENDIF                                                   HVS00890
              ENDIF                                                     HVS00900
C                                                                       HVS00910
C.............. CALC DUCTILITY AND EXCURSION RATIO...                  HVS00920
              IF  (RULE.EQ.1 .OR. RULE.GT.5)  THEN                      HVS00930
                IF (DIR.EQ.2) THEN                                      HVS00940
                  DLI= MAX( D , DL )                                    HVS00950
                  CALL DME(1,UPOS,EXCR, DY,DLI,ESE(2),PSE,PSEOLD,CSE)   HVS00960
                ELSE IF (DIR.EQ.4) THEN                                 HVS00970
                  DLI= MIN( D , DL )                                    HVS00980
                  CALL DME(1,UNEG,EXCR,-DY,DLI,ESE(3),PSE,PSEOLD,CSE)   HVS00990
                ENDIF                                                   HVS01000
              ENDIF                                                     HVS01010
C                                                                       HVS01020
C.............. UNLOADING ON THE TOP SEGMENT                    RULE 2HVS01030
              IF(AP.GT.ABS(Q3)) THEN                                    HVS01040
                K=MAX(S1,SU)                                            HVS01050
                A=Q3                                                    HVS01060
                RULE=2                                                  HVS01070
C.............. UNLOADING ON THE MIDDLE SEGMENT                 RULE 3HVS01080
              ELSE IF(AP.GT.ABS(Q1)) THEN                               HVS01090
                K=MAX(S2,SO2,SU)                                        HVS01100
                A=Q1                                                    HVS01110
                RULE=3                                                  HVS01120
C.............. UNLOADING ON THE BOTTOM SEGMENT                 RULE 4HVS01130
              ELSE                                                      HVS01140
                K=MAX(S3,SO3,SU)                                        HVS01150
                A=0.                                                    HVS01160
                RULE=4                                                  HVS01170
              ENDIF                                                     HVS01180
      ENDIF                                                             HVS01190
C..... SAVE REVERSAL POINTS FOR SMALL LOOPS.....                        HVS01200
  100 IRULE=INT(RULE)                                                   HVS01210
      REVRSL=IRULE.EQ.1.OR.IRULE.EQ.10.OR.LRULE.EQ.1.OR.LRULE.EQ.10     HVS01220
      IF      (<DIRL.EQ.1 .AND. DIR.EQ.2 .AND. .NOT. REVRSL)            HVS01230
     &         .OR. (DIRL.EQ.4 .AND. DIR.EQ.3) ) THEN                   HVS01240
            IF (IR.LT.NI)  THEN                                         HVS01250
              IR=IR+1                                                   HVS01260
              PR(IR)=P                                                  HVS01270
              DR(IR)=D                                                  HVS01280
              FR(IR)='7'                                                HVS01290
            ENDIF                                                       HVS01300
      ELSE IF ((DIRL.EQ.5 .AND. DIR.EQ.4 .AND. .NOT. REVRSL)            HVS01310
     &         .OR. (DIRL.EQ.2 .AND. DIR.EQ.1) ) THEN                   HVS01320
            IF (IR.LT.NI)  THEN                                         HVS01330
              IR=IR+1                                                   HVS01340
              PR(IR)=P                                                  HVS01350
              DR(IR)=D                                                  HVS01360
              FR(IR)='L'                                                HVS01370
            ENDIF                                                       HVS01380
      ENDIF                                                             HVS01390
      IF (PRINT) WRITE (6,1040) S1,S2,S3                                HVS01400
 1040 FORMAT(                                                           HVS01410
     &' B1=   S1=',G15.5,'   S2=',G15.5,'    S3=',G15.5,'     =',G15.5) HVS01420
C                                                                       HVS01430
      IF ((1.05*SI).GT.K .AND. K.GT.0 ) RETURN                          HVS01440
C                                                                       HVS01450
      K=0                                                               HVS01460
      RULE=-RULE                                                        HVS01470
      RETURN                                                            HVS01480
C                                                                       HVS01490
      END                                                               HVS01500
C=======================================================================HVS01510
C=== SHEAR HYSTERESIS MODEL..... HVST04 ...............................HVS01520
C=======================================================================HVS01530
C                                                                       HVS01540
C     INPUT VARIABLES                                                   HVS01550
C     P       = CURRENT LOAD                                            HVS01560
C     D       = CURRENT DISPLACEMENT                                    HVS01570
C     PL      = LAST LOAD                                               HVS01580
C     VEL     = PRODUCT OF LAST AND CURRENT VELOCITY                    HVS01590
C     NS      = NUMBER OF BACKBONE CURVE POINTS                         HVS01600
C     NI      = MAXIMUM NUMBER OF SECONDARY LOOPS                       HVS01610
C     PC      = CRACKING LOAD                                           HVS01620
C     DC      = CRACKING DISPLACEMENT                                   HVS01630
C     SI      = INITIAL STIFFNESS PC/DC                                 HVS01640
C     PS      = BACKBONE CURVE LOADING POINTS                           HVS01650
C     DS      = BACKBONE CURVE DISPLACEMENT POINTS                      HVS01660
C                                                                       HVS01670
C     OUTPUT VARIABLES                                                  HVS01680
C      AN INITIAL VALUE OF ZERO IS REQUIRED FOR ALL OUTPUT VARIABLES... HVS01690
C     K       = STIFFNESS                                               HVS01700
C     RULE    = RULE NUMBER, STORED IN THE FOR_MAT RN.DIR UPON EXIT OF  HVS01710
C               THIS SU_BROU_TINE                                       HVS01720
C     A       = LOAD LIMIT BEFORE NEXT RULE CHANGE                      HVS01730
C     PMG     = MAXIMUN PAST LOAD IN THE POSITIVE DIRECTION             HVS01740
C     PMN     = MAXIMUN PAST LOAD IN THE NEGATIVE DIRECTION             HVS01750
C     DMG     = MAXIMUN PAST DISPLACEMENT IN THE POSITIVE DIRECTION     HVS01760
C     DMN     = MAXIMUN PAST DISPLACEMENT IN THE NEGATIVE DIRECTION     HVS01770
C     PR      = SECONDARY LOOP LOADING POINT                            HVS01780
C     DR      = SECONDARY LOOP DISPLACEMENT POINT                       HVS01790
C     FR      = SECONDARY LOOP FLAG                                     HVS01800
C     IR      = SECONDARY LOOP NUMBER                                   HVS01810
C     BACKB   = BACKBONE LOADING FLAG                                   HVS01820
C     SR1     = RELOADING STIFFNESS FROM RULE 8 - USED BY RULE 11       HVS01830
C     SR2     = RELOADING STIFFNESS FROM RULE 9 - USED BY RULE 11       HVS01840
C     SR3     = RELOADING STIFFNESS FROM RULE 7 - USED BY RULE 11       HVS01850
C     SRM     = RELOADING STIFFNESS FROM RULES 8 & 9 - USED BY RULE 11  HVS01860
C                                                                       HVS01870
C     INTERNAL VARIABLES                                                HVS01880
C     J       = BACKBONE CURVE POINT # IN THE CURRENT DIRECTION         HVS01890
C     DIR     = CURRENT DIRECTION                                       HVS01900
C               DIR=1, POSITIVE LOADING                                 HVS01910
C               DIR=2, POSITIVE UNLOADING                               HVS01920
C               DIR=3, NEGATIVE LOADING                                 HVS01930
C               DIR=4, NEGATIVE UNLOADING                               HVS01940
C     DIRL    = LAST DIRECTION                                          HVS01950
C     IRULE   = INTERGER VALUE OF RULE                                  HVS01960
C     LRULE   = INTEGER VALUE OF LAST RULE #                            HVS01970
C     CRACKD  = CRACKED WALL FLAG                                       HVS01980
C     REVRSL  = LOAD REVERSAL FLAG                                      HVS01990
C     PRINT   = PRINT FLAG, USED FOR DEBU_GGING                         HVS02000
C     PC2     = PC/2, USED BY LOADING CURVES                            HVS02010
C     DC2     = DISPLACEMENT CORRESPONDING TO PC2 ON THE LOADING BRANCH  HVS02020
C     S       = INTERMEDIATE STIFFNESS                                  HVS02030
C     SR      = RELOADING STIFFNESS                                     HVS02040
C     SRP     = MINIMUN RELOADING STIFFNESS                             HVS02050
C     DO      = DISPLACEMENT INTERSEPT OF PEAK DISPLACEMENTS            HVS02060
C     DOP     = DISPLACEMENT INTERSEPT OF THE UNLOADING BRANCH          HVS02070
C     AP      = ABSOLUTE VALUE OF CURRENT LOAD P.                       HVS02080
C     PM      = MAXIMUN PAST LOAD IN THE CURRENT DIRECTION              HVS02090
C     DM      = MAXIMUN PAST DISPLACEMENT IN THE CURRENT DIRECTION      HVS02100
C     DMAX    = MAXIMUN PAST DISPLACEMENT IN EITHER DIRECTION           HVS02110
C     PMAX    = MAXIMUN PAST LOAD IN EITHER DIRECTION                   HVS02120
C     PA      = PM-PC, USED BY UNLOADING CURVES                         HVS02130
C     PB      = PC/2, USED BY UNLOADING CURVES                          HVS02140
C     DA      = DISPLACEMENT AT PA, USED BY UNLOADING CURVES            HVS02150
C     DB      = DISPLACEMENT AT PB, USED BY UNLOADING CURVES            HVS02160
C     DP      = DIFFERANCE IN LOAD, USED FOR CALCULATING K              HVS02170
C     DD      = DIFFERANCE IN DISPLACEMENT, USED FOR CALCULATING K      HVS02180
C     PX      = LOAD INTERCEPT OF LOADING AND BACKBONE CURVE            HVS02190
C     DX      = DISPLACEMENT INTERCEPT OF LOADING AND BACKBONE CURVE    HVS02200
C     P2,D2   = POINT ON THE LOADING CURVE BETWEEN RULES 7 AND 10       HVS02210
C     X       = MAXIMUN DISPLACEMENT                                    HVS02220
C               = DISPLACEMENT INTERCEPT FOR RULE 11                    HVS02230
C     X1      = 75% PC DISPLACEMNT  FOR RULE 11                         HVS02240
C     X2      = 25% PC DISPLACEMNT  FOR RULE 11                         HVS02250
C     I_IT    = TEMPORARY LABEL FOR SMALL LOOP LOADING AND UNLOADING PTSHVS02260
C     JO      = J-1                                                     HVS02270
C     ALPHA   = DEGRADING STIFFNESS FACTOR                              HVS02280
C     ALPHAP  = DEGRADING STIFFNESS FACTOR USED BY RULE 10              HVS02290
C     S1      = STIFFNESS FROM FUNCTION FS1                             HVS02300
C     S2      = STIFFNESS FROM FUNCTION FS2                             HVS02310
C     S3      = STIFFNESS FROM FUNCTION FS3                             HVS02320
C     SU      = STIFFNESS BETWEEN CURRENT AND RELOADING POINT           HVS02330
C     SO2     = STIFFNESS BETWEEN PA AND DO                             HVS02340
C     SO3     = STIFFNESS BETWEEN PB AND DO                             HVS02350
C                                                                       HVS02360
C     INTERNAL FUNCTIONS                                                HVS02370
C     FD2     = DISPLACEMENT INTERCEPT OF LOADING AND UNLOADING CURVE    HVS02380
C     FS1     = UNLOADING STIFFNESS WHEN P<(PM-PC)=PA                   HVS02390
C     FS2     = UNLOADING STIFFNESS WHEN PA>P>(PC/2)=PB                 HVS02400
C     FS3     = UNLOADING STIFFNESS WHEN     P>PB                       HVS02410
C     FSR     = LOADING BELOW PC/2, WITH PINCHING                       HVS02420
C                                                                       HVS02430
C     SU_BROU_TINES CALLED                                              HVS02440
C     DIRECT  = DETERMINES THE VALUES OF DIR AND DIRL                   HVS02450
C     INTSCT  = DETERMINES THE INTERSECTION OF TWO LINES IN POINT-SLOPE  HVS02460
C               FORM                                                    HVS02470
C     KMIN    = DETERMINES THE STIFFNESS BASED ON DP,DD AND K_DEFULT     HVS02480
C                                                                       HVS02490
      SUBROUTINE HVST04(P,D,PL,DL,VEL,NS,NI,PRINT,                      HVS02500
     &    PC    ,DC    ,PS    ,DS    ,SI    ,CSE   ,DY    ,           HVS02510
     &    K     ,RULE  ,RDIR  ,A     ,PMG   ,PMN   ,                  HVS02520
     &    DMG   ,DMN   ,BACKB ,SR1   ,SR2   ,SR3   ,                  HVS02530
     &    SRM   ,IR    ,SE    ,UPOS  ,UNEG  ,EXCR  ,                  HVS02540
     &    ESE   ,PSE   ,PSEOLD,PR    ,DR    ,FR    )                  HVS02550
C                                                                       HVS02560
      IMPLICIT REAL(A-H,K,O-Z)                                          HVS02570
      IMPLICIT INTEGER(I,J,L-N)                                         HVS02580
      INTEGER DIR,DIRL                                                  HVS02590
      LOGICAL  BACKB,CRACKD,REVRSL,PRINT,BTEST                          HVS02600
      CHARACTER*4  FR(NI)                                               HVS02610
      DIMENSION PS(NS),DS(NS),PR(NI),DR(NI)                             HVS02620
      DIMENSION UPOS(3),UNEG(3),EXCR(6),ESE(3)                          HVS02630
      DATA ALPHA/1.0/                                                   HVS02640
C                                                                       HVS02650
            FS1(X)= SI *MINK(1.4675*(DC/X)**.343 ),1.)                  HVS02660
            FS2(X)= SI *MINK(0.7761*(DC/X)**.5195),1.)                  HVS02670
            FS3(X)= SI *MINK(0.707*(DC/X)+1.369 ),1.)                   HVS02680
      FSR(X)= SI *MINK(   (ABS(DC/X))**1.02),1.)                        HVS02690
      FD2(X)= SIGN(DMAX,PM)-.05*SIGN(PMAX,PM)/(FS1(X))                  HVS02700
C                                                                       HVS02710
 1000 IF (PRINT) WRITE (6,1010) P,D,PL,DL,NS,NI,PC,DC,SI,K,RULE,A,      HVS02720
     &     PMG,PMN,DMG,DMN,SR1,SR2,SR3,SRM,BACKB,IR,PSEOLD,VEL,        HVS02730
     &     SE,DY,ESE(1),PSE,UPOS,CSE,UNEG,EXCR                         HVS02740
 1010 FORMAT (                                                          HVS02750
     &' S1-    P=',G15.5,'   D=',G15.5,'   PL=',G15.5,'    DL=',G15.5/ HVS02760
     &' S1-   NS=',G15.5,'  NI=',G15.5,'   PC=',G15.5,'    DC=',G15.5/ HVS02770
     &' S1-   SI=',G15.5,'   K=',G15.5,' RULE=',G15.5,'     A=',G15.5/ HVS02780
     &' S1-  PMG=',G15.5,' PMN=',G15.5,'  DMG=',G15.5,'   DMN=',G15.5/ HVS02790
     &' S1-  SR1=',G15.5,' SR2=',G15.5,'  SR3=',G15.5,'   SRM=',G15.5/ HVS02800
     &' S1-BACKB=',G15.5,'  IR=',G15.5,' PSEO=',G15.5,'   VEL=',G15.5/ HVS02810
     &' S1-   SE=',G15.5,'  DY=',G15.5,'  ESE=',G15.5,'   PSE=',G15.5/ HVS02820
     &' S1-  UP1=',G15.5,' UP2=',G15.5,'  UP3=',G15.5,'   CSE=',G15.5/ HVS02830
     &' S1-  UN1=',G15.5,' UN2=',G15.5,'  UN3=',G15.5/                  HVS02840
     &' S1-  EX1=',G15.5,' EX2=',G15.5,'  EX3=',G15.5/                  HVS02850
     &' S1-  EX4=',G15.5,' EX5=',G15.5,'  EX6=',G15.5)                  HVS02860
      IF (IR.GT.0.AND.PRINT) WRITE (6,1020) (I,PR(I),DR(I),FR(I),I=1,IR)HVS02870
 1020 FORMAT                                                            HVS02880
     & (' I=',G15.5,'   PR=',G15.5,'   DR=',G15.5,'   FR=',A )          HVS02890
C                                                                       HVS02900
      AP=ABS(P)                                                         HVS02910
C                                                                       HVS02920
C..... DETERMINE IF ELASTIC                                             HVS02930
      IF (RULE.EQ.0 .AND. AP .LT.PC ) THEN                              HVS02940
            K=SI                                                        HVS02950
            SE=K                                                        HVS02960
            GO TO 9999                                                  HVS02970
      ENDIF                                                             HVS02980
C                                                                       HVS02990
C..... DETERMINE IF WALL IS INACTIVE, IF SO RE_TURN                     HVS03000
```

```
      IF (RULE.LT.0.0) GO TO 9999                                       HYS01600
C                                                                       HYS01610
C..... DETERMINE CURRENT DIRECTION                                      HYS01620
      LRULE=INT(RULE)                                                   HYS01630
      DIR =RDIR                                                         HYS01640
      CALL DIRECT(DIR,DIRL,P,PL,VEL)                                    HYS01650
      RDIR =DIR                                                         HYS01660
      IF (PRINT) WRITE (6,1050) DIR,DIRL                                HYS01670
 1050 FORMAT('   DIR=',I8  ,' DIRL=',I8  )                            HYS01680
C                                                                       HYS01690
C..... DETERMINE MAXIMUN AND MINIMUN VALUES                             HYS01700
      PMG=MAX(PMG, PC,P)                                                HYS01710
      DMG=MAX(DMG, DC,D)                                                HYS01720
      PMM=MIN(PMM,-PC,P)                                                HYS01730
      DMM=MIN(DMM,-DC,D)                                                HYS01740
      DMAX=MAX(DMG,-DMM)                                                HYS01750
      PMAX=MAX(PMG,-PMM)                                                HYS01760
C                                                                       HYS01770
C..... SET MAXIMUN AND MINIMUN VALUES...                                HYS01780
      IF (DIR.EQ.1 .OR. DIR.EQ.2 ) THEN                                 HYS01790
            PM=PMG                                                      HYS01800
            DM=DMG                                                      HYS01810
      ELSE IF (DIR.EQ.3 .OR. DIR.EQ.4) THEN                             HYS01820
            PM=PMM                                                      HYS01830
            DM=DMM                                                      HYS01840
      ENDIF                                                             HYS01850
C                                                                       HYS01860
C..... CALCULATE EQUIVALENT UNLOADING STIFFNESS FOR ENERGY...           HYS01870
       IF (BACKB .OR. RULE.EQ.0.0) THEN                                 HYS01880
            S1=FS1(DMAX)                                                HYS01890
            S2=FS2(DMAX)                                                HYS01900
            S3=FS3(DMAX)                                                HYS01910
            DO=DM-PM*(DMG-DMM)/(PMG-PMM)                                HYS01920
            PA=PM-SIGN(PC,PM)                                           HYS01930
            DA=DM-SIGN(PC,PM)/S1                                        HYS01940
            PB=SIGN  (MIN(ABS(PA),(PC/2)) ),PM)                         HYS01950
            SO2=AMIN((DA-DO),PA,SI)                                     HYS01960
            DB=DA-(PA-PB)/MAX(S2,SO2)                                   HYS01970
            SO3=AMIN((DB-DO),PB,SI)                                     HYS01980
            SE= 0.5 * PM**2 / (                                         HYS01990
     &              (PM+PA)*0.5*(DM-DA)                                HYS02000
     &             +(PA+PB)*0.5*(DA-DB)                                HYS02010
     &             +(    PB)*0.5*(    PB)/MAX(S3,SO3) )                HYS02020
       ENDIF                                                            HYS02030
C                                                                       HYS02040
C..... CHECK TO SEE IF LAST RULE IS STILL IN EFFECT                     HYS02050
      IF (DIR.EQ.OIRL .AND.                                             HYS02060
     &  ((DIR.EQ.1 .AND. P.LT.A) .OR. (DIR.EQ.2 .AND. P.GT.A)  .OR.     HYS02070
     &   (DIR.EQ.3 .AND. P.GT.A) .OR. (DIR.EQ.4 .AND. P.LT.A))) THEN    HYS02080
         RETURN                                                         HYS02090
      ENDIF                                                             HYS02100
C                                                                       HYS02110
C..... CALC EXCURSION RATIO AT EVERY ZERO CROSSING...                   HYS02120
      IF (P**PL.LE.0) CALL DME(O,UPOS,EXCR, DV,DLI,ESE,PSE,PSEOLD,CSE)  HYS02130
C                                                                       HYS02140
C..... ERASE SMALL LOOP FLAGS                                           HYS02150
      IF (IR.GT.0) THEN                                                 HYS02160
        IT=IR                                                           HYS02170
        DO 10 I=1,IT                                                    HYS02180
          IF ((FR(I).EQ.'L' .AND. ( P.LE.PR(I) .OR. D.LE.DR(I) ))  .OR. HYS02190
     &       (FR(I).EQ.'7' .AND. ( P.GE.PR(I) .OR. D.GE.DR(I) ))) THEN  HYS02200
            IR=I-1                                                      HYS02210
            GO TO 15                                                    HYS02220
          ENDIF                                                         HYS02230
   10   CONTINUE                                                        HYS02240
      ENDIF                                                             HYS02250
   15 I=IR+1                                                            HYS02260
C                                                                       HYS02270
      IF (DIR.EQ.1 .OR. DIR.EQ.3 )  THEN                                HYS02280
C..... LOADING RULES ..................................... RULES 1,5-11 HYS02290
        PC2=SIGN(PC/2),PM)                                              HYS02300
C                                                                       HYS02310
C..... LOADING ON BACKBONE                                    RULE 1    HYS02320
   30   IF (BACKB .OR. RULE.EQ.0.0) THEN                                HYS02330
          X=0.                                                          HYS02340
          DO 40 J=2,NS                                                  HYS02350
            DO=OS(J)-ABS(D)                                             HYS02360
            DP=PS(J)-AP                                                 HYS02370
            IF (AP.LT.PS(J) .AND. (DO*DP).GT.0.) THEN                   HYS02380
              S=DP/DO                                                   HYS02390
              IF (S.GT.K .AND. S.LE.SI) THEN                            HYS02400
                K=S                                                     HYS02410
                A=SIGN(PS(J),P)                                         HYS02420
              ENDIF                                                     HYS02430
            ENDIF                                                       HYS02440
   40     CONTINUE                                                      HYS02450
          RULE=1                                                        HYS02460
          BACKB=.TRUE.                                                  HYS02470
          IR=0                                                          HYS02480
C                                                                       HYS02490
C..... LOADING AFTER UNLOADING                          RULES 6-11    HYS02500
        ELSE                                                            HYS02510
          CRACKD=(ABS(DM).GT.DC)                                        HYS02520
          REVRSL=((DIR.EQ.1 .AND. DIRL.EQ.4) .OR. LRULE.EQ.8            HYS02530
     &          .OR. (DIR.EQ.3 .AND. DIRL.EQ.2) .OR. LRULE.EQ.9         HYS02540
     &          .OR. LRULE.EQ.5  .OR. LRULE.EQ.11)                      HYS02550
          IF (CRACKD ) THEN                                             HYS02560
            P2=.95*SIGN(PMAX,PM)                                        HYS02570
            D2=FD2(DMAX)                                                HYS02580
          ELSE                                                          HYS02590
            P2=PM                                                       HYS02600
            D2=DM                                                       HYS02610
          ENDIF                                                         HYS02620
          S=AMIN((D2-D),(P2-P),SI)                                      HYS02630
   20     I=I-1                                                         HYS02640
C                                                                       HYS02650
C..... RELOADING INSIDE SMALL LOOPS                          RULE 11    HYS02660
          IF (I.GT.0 ) THEN                                             HYS02670
            IF (.NOT.(( FR(I).EQ.'7' .AND. DIR.EQ.1 )  .OR.             HYS02680
     &                ( FR(I).EQ.'L' .AND. DIR.EQ.3 )))  GO TO 20     HYS02690
            IF (SR1.LE.0 .OR. SR2.LE.0 .OR. SR3.LE.0 .OR.               HYS02700
     &          SRM.LE.0 ) THEN                                         HYS02710
                I=1                                                     HYS02720
                GO TO 20                                                HYS02730
            ENDIF                                                       HYS02740
            IF (AP.LT.ABS(PC2) .AND. S.GT.SRM) THEN                     HYS02750
                REVRSL=.FALSE.                                          HYS02760
                I=1                                                     HYS02770
                GO TO 20                                                HYS02780
            ENDIF                                                       HYS02790
            IF (ABS(PR(I)).LE. PC/4) THEN                               HYS02800
               K=KMIN(K(D-DR(I)),(P-PR(I)),SI)                          HYS02810
               A=PR(I)                                                  HYS02820
               RULE=11.1+I/1000.                                        HYS02830
            ELSE IF (ABS(PR(I)).LE.3*PC/4) THEN                         HYS02840
               X2=D2-(P2-1.5*PC2)/SR3                                   HYS02850
               X1=X2-PC2/SR2                                            HYS02860
               X=D+(PC2/2-P) /SR2                                       HYS02870
               IF (AP.LT.PC/4 .AND. ((DIR.EQ.1 .AND. X.LT.X1).OR.       HYS02880
     &             (DIR.EQ.3 .AND. X.GT.X1))) THEN                      HYS02890
                  K=KMIN((X1-D),(PC2/2-P),SI)                           HYS02900
                  A=SIGN((PC/4),PM)                                     HYS02910
                  RULE=11.2+I/1000.                                     HYS02920
               ELSE                                                     HYS02930
                  K=KMIN((DR(I)-D),(PR(I)-P),SI)                        HYS02940
                  A=PR(I)                                               HYS02950
                  RULE=11.3+I/1000.                                     HYS02960
               ENDIF                                                    HYS02970
            ELSE                                                        HYS02980
               X2=DR(I)-(PR(I)-1.5*PC2)/SRM                             HYS02990
               X1=X2-PC2/SR2                                            HYS03000
               X=D+(PC2/2-P) /SR2                                       HYS03010
               IF (AP.LT.PC/4 .AND. ((DIR.EQ.1 .AND. X.LT.X1).OR.       HYS03020
     &             (DIR.EQ.3 .AND. X.GT.X1))) THEN                      HYS03030
                  K=KMIN((X1-D),(PC2/2-P),SI)                           HYS03040
                  A=SIGN((PC/4),PM)                                     HYS03050
                  RULE=11.4+I/1000.                                     HYS03060
               ELSE                                                     HYS03070
                  X=D+(1.5*PC2-P) /SR3                                  HYS03080
                  IF (AP.LT. 3*PC/4  .AND.                              HYS03090
     &                ((DIR.EQ.1 .AND. X.LT.X2)                         HYS03100
     &              .OR. (DIR.EQ.3 .AND. X.GT.X2))) THEN                HYS03110
                     K=KMIN((X2-D),(1.5*PC2-P),SI)                      HYS03120
                     A=SIGN((3*PC/4),PM)                                HYS03130
                     RULE=11.5+I/1000.                                  HYS03140
                  ELSE                                                  HYS03150
                     SR=SRM                                             HYS03160
                     K=KMIN((DR(I)-D),(PR(I)-P),SI)                     HYS03170
                     A=PR(I)                                            HYS03180
                     RULE=11.6+I/1000.                                  HYS03190
                  ENDIF                                                 HYS03200
               ENDIF                                                    HYS03210
            ENDIF                                                       HYS03220
C                                                                       HYS03230
C..... INITIAL RELOADING BELOW PC/2  W/O REVERSAL          RULE 6       HYS03240
          ELSE IF (AP.LT.ABS(PC2) .AND. .NOT.REVRSL ) THEN              HYS03250
            K=FS1(DMAX)                                                 HYS03260
            A=PC2                                                       HYS03270
            RULE=6                                                      HYS03280
C                                                                       HYS03290
C..... RELOADING TO P2                                     RULE 7       HYS03300
          ELSE IF (AP.LT.ABS(P2) .AND.                                  HYS03310
     &             (.NOT.REVRSL .OR. AP.GE.PC*0.75)) THEN               HYS03320
                K=S                                                     HYS03330
                A=P2                                                    HYS03340
                RULE=7                                                  HYS03350
C                                                                       HYS03360
C..... RELOADING BELOW PC WITH REVERSAL            RULES 8 & 9          HYS03370
          ELSE IF (AP.LT.PC*0.75 ) THEN                                 HYS03380
                SR=FSR(DMAX)                                            HYS03390
                S1=FS1(DMAX)                                            HYS03400
                S2=FS2(DMAX)                                            HYS03410
                S3=FS3(DMAX)                                            HYS03420
                PA=SIGN(PMAX,PM)-SIGN(PC,PM)                            HYS03430
                DA=SIGN(DMAX,PM)-SIGN(PC,PM)/S1                         HYS03440
                PB=SIGN  (MIN(ABS(PA),(PC/2)) ),PM)                     HYS03450
                DB=DA-(PA-PB)/S2                                        HYS03460
                DOP=DB-PB)/S3                                           HYS03470
                DO=DM-PM*(DMG-DMM)/(PMG-PMM)                            HYS03480
                IF (DIR.EQ.1) DOP=MAX(DOP,DO)                           HYS03490
                IF (DIR.EQ.3) DOP=MIN(DOP,DO)                           HYS03500
              DC2=DOP+PC2/S1                                            HYS03510
              SRM=KMIN((DC2-D2),(PC2-P2),S1)                            HYS03520
              IF (S.GT.SRM) THEN                                        HYS03530
                REVRSL=.FALSE.                                          HYS03540
                GO TO 20                                                HYS03550
              ENDIF                                                     HYS03560
              SRP=KMIN((DC2-D),(PC2-P),S1)                              HYS03570
              SR1=MAX(SRP,MIN(SR,S))                                    HYS03580
              DC2P=D+(PC2-P)/SR1                                        HYS03590
              SR3=KMIN((D2-DC2P),(P2-PC2),SI)                           HYS03600
              SR2=2.00/( 1./SR1 + 1./SR3 )                              HYS03610
              IF (AP.LT.(PC/4) ) THEN                                   HYS03620
                K=SR1                                                   HYS03630
                A=SIGN((PC/4),PM)                                       HYS03640
                RULE=8                                                  HYS03650
              ELSE                                                      HYS03660
                K=SR2                                                   HYS03670
                A=SIGN(PC,PM)*0.75                                      HYS03680
                RULE=9                                                  HYS03690
              ENDIF                                                     HYS03700
C                                                                       HYS03710
C..... LOADING TOWARDS THE BACKBONE CURVE                  RULE 10      HYS03720
          ELSE                                                          HYS03730
            BACKB=.TRUE.                                                HYS03740
            IF (.NOT. CRACKD) GO TO 50                                  HYS03750
            IR=0                                                        HYS03760
            ALPHAP=1.0                                                  HYS03770
            IF (ABS(DM).EQ.ABS(DMAX))  ALPHAP=ALPHA                     HYS03780
            K=KMIN((ALPHAP*SIGN(DMAX,D2)-D2),(SIGN(PMAX,P2)-P2)         HYS03790
     &                    ,SI)                                          HYS03800
            DO 50 J=2,NS                                                HYS03810
              IF ( PS(J).LE.ABS(PMAX) ) GO TO 50                        HYS03820
              JO=J-1                                                    HYS03830
              DO=OS(J)-OS(JO)                                           HYS03840
              DP=PS(J)-PS(JO)                                           HYS03850
              IF ((DP*DO).LE.0.) GO TO 50                               HYS03860
               CALL INTSCT(DM,PM,D,P,K,SIGN(OS(J),P),                   HYS03870
     &                     SIGN(PS(J),P),(DP/DO) )                      HYS03880
              IF (ABS(DX).GE.DS(JO) .AND. ABS(DX).LE.DS(J).AND.         HYS03890
     &            ABS(PX).GT.ABS(PMAX).AND.  PM*PX.GT.0) THEN           HYS03900
                IF (AP.GE.ABS(PX)) GO TO 50                             HYS03910
                K=KMIN((DX-D),(PX-P),SI)                                HYS03920
                A= PM                                                   HYS03930
                RULE=10                                                 HYS03940
                GO TO 100                                               HYS03950
              ENDIF                                                     HYS03960
   50       CONTINUE                                                    HYS03970
            GO TO 50                                                    HYS03980
          ENDIF                                                         HYS03990
        ENDIF                                                           HYS04000
C                                                                       HYS04010
C                                                                       HYS04020
      ELSE                                                              HYS04030
C..... UNLOADING RULES ..........................................       HYS04040
C                                                                       HYS04050
C..... INITILIZE VALUES                                 RULES 2-5     HYS04060
            BACKB=.FALSE.                                               HYS04070
```

```
                S1=FS1(DMAX)                                                    HYSO4080
                S2=FS2(DMAX)                                                    HYSO4090
                S3=FS3(DMAX)                                                    HYSO4100
                DO=DM-PM*(DMG-DMM)/(PMG-PMM)                                    HYSO4110
                PA=PM-SIGN(PC,PM)                                               HYSO4120
                DA=DM-SIGN(PC,PM)/S1                                            HYSO4130
                PB=SIGN( (MIN(ABS(PA),(PC/2)) ),PM)                             HYSO4140
                SO2=KMIN(DA-DO),PA,S1)                                          HYSO4150
                DB=DA-(PA-PB)/MAX(S2,SO2)                                       HYSO4160
                SO5=KMIN((DB-DO),PB,S1)                                         HYSO4170
                SU=0.                                                           HYSO4180
C                                                                               HYSO4190
C.............. UNLOADING INSIDE SMALL POSITIVE LOOPS            RULE 5       HYSO4200
              I=IR+1                                                            HYSO4210
   70         I=I-1                                                           HYSO4220
              IF (I.GT.0) THEN                                                  HYSO4230
                IF (.NOT.((DIR.EQ.2 .AND. FR(I).EQ.'L')    .OR.                 HYSO4240
     &                   (DIR.EQ.4 .AND. FR(I).EQ.'7'>> ) GO TO 70            HYSO4250
C                                                                               HYSO4260
                IF ((DIR.EQ.2 .AND.( P.GT.PA .AND. PR(I).GT.PA) .OR.            HYSO4270
     &               (PA.GE.P .AND.   P.GT.PB .AND. PR(I).GT.PB) .OR.           HYSO4280
     &               (PB.GE.P .AND.   PR(I).GT.O ))) .OR.                       HYSO4290
     &              (DIR.EQ.4 .AND.( P.LT.PA .AND. PR(I).LT.PA) .OR.            HYSO4300
     &               (PA.LE.P .AND.   P.LT.PB .AND. PR(I).LT.PB) .OR.           HYSO4310
     &               (PB.LE.P .AND.   PR(I).LT.O ))) ) THEN                     HYSO4320
                  K=(PR(I)-P)/(DR(I)-D)                                         HYSO4330
                  IF (K.LE.O .OR. K.GT.SI) GO TO 70                             HYSO4340
                  A=PR(I)                                                       HYSO4350
                  RULE=5+I/1000.                                                HYSO4360
                  GO TO 100                                                     HYSO4370
                ELSE                                                            HYSO4380
                  SU=KMIN((DR(I)-D),(PR(I)-P),SI)                               HYSO4390
                ENDIF                                                           HYSO4400
C                                                                               HYSO4410
              ENDIF                                                             HYSO4420
C                                                                               HYSO4430
C.............. CALC DUCTILITY AND EXCURSION RATIO...                          HYSO4440
              IF  (RULE.EQ.1 .OR. RULE.GT.5)  THEN                              HYSO4450
                IF (DIR.EQ.2) THEN                                              HYSO4460
                  DLI= MAX( O , DL )                                            HYSO4470
                  CALL DME(1,UPOS,EXCR, DV,DLI,ESE(2),PSE,PSEOLD,CSE)           HYSO4480
                ELSE IF (DIR.EQ.4) THEN                                         HYSO4490
                  DLI= MIN( O , DL )                                            HYSO4500
                  CALL DME(1,UNEG,EXCR,-DV,DLI,ESE(5),PSE,PSEOLD,CSE)           HYSO4510
                ENDIF                                                           HYSO4520
              ENDIF                                                             HYSO4530
C                                                                               HYSO4540
C.............. UNLOADING ON THE TOP SEGMENT                      RULE 2       HYSO4550
              IF(AP.GT.ABS(PA)) THEN                                            HYSO4560
                K=MAX(S1,SU)                                                    HYSO4570
                A=PA                                                            HYSO4580
                RULE=2                                                          HYSO4590
C.............. UNLOADING ON THE MIDDLE SEGMENT                   RULE 3       HYSO4600
              ELSE IF(AP.GT.ABS(PB)) THEN                                       HYSO4610
                K=MAX(S2,SO2,SU)                                                HYSO4620
                A=PB                                                            HYSO4630
                RULE=5                                                          HYSO4640
C.............. UNLOADING ON THE BOTTOM SEGMENT                   RULE 4       HYSO4650
              ELSE                                                              HYSO4660
                K=MAX(S3,SO3,SU)                                                HYSO4670
                A=O.                                                            HYSO4680
                RULE=4                                                          HYSO4690
              ENDIF                                                             HYSO4700
C                                                                               HYSO4710
      ENDIF                                                                     HYSO4720
C                                                                               HYSO4730
C..... SAVE REVERSAL POINTS FOR SMALL LOOPS.....                               HYSO4740
  100 IRULE=INT(RULE)                                                           HYSO4750
      REVRSL=IRULE.EQ.1.OR.IRULE.EQ.10.OR.LRULE.EQ.1.OR.LRULE.EQ.10            HYSO4760
      IF      ((DIRL.EQ.1 .AND. DIR.EQ.2 .AND. .NOT. REVRSL)                    HYSO4770
     &          .OR. (DIRL.EQ.4 .AND. DIR.EQ.3) ) THEN                        HYSO4780
            IF (IR.LT.NI)  THEN                                                 HYSO4790
                IR=IR+1                                                         HYSO4800
                PR(IR)=P                                                        HYSO4810
                DR(IR)=D                                                        HYSO4820
                FR(IR)='7'                                                      HYSO4830
            ENDIF                                                               HYSO4840
      ELSE IF ((DIRL.EQ.3 .AND. DIR.EQ.4 .AND. .NOT. REVRSL)                    HYSO4850
     &                   .OR. (DIRL.EQ.2 .AND. DIR.EQ.1) ) THEN                HYSO4860
            IF (IR.LT.NI)  THEN                                                 HYSO4870
                IR=IR+1                                                         HYSO4880
                PR(IR)=P                                                        HYSO4890
                DR(IR)=D                                                        HYSO4900
                FR(IR)='L'                                                      HYSO4910
            ENDIF                                                               HYSO4920
      ENDIF                                                                     HYSO4930
      IF ((1.05*SI).GT.K .AND. K.GT.O ) GO TO 9999                              HYSO4940
C                                                                               HYSO4950
      K=O                                                                       HYSO4960
      RULE=-RULE                                                                HYSO4970
 9999 CONTINUE                                                                  HYSO4980
      RETURN                                                                    HYSO4990
C                                                                               HYSO5000
      END                                                                       HYSO5010
C=======================================================================        HYSO5020
C=== TAKEDA HYSTERESIS MODEL ..... HYSTO6 ..............................        HYSO5030
C=======================================================================        HYSO5040
C=======================================================================        HYSO5050
C                                                                               HYSO5060
C   TAKEDA HYSTERISIS MODEL - REFERANCE, S. OTANI, UILU-ENG-74-2029             HYSO5070
C                       TAKEDA & SOZEN, J. ASCE ST. DIV. - DEC. 1970            HYSO5080
C                                                                               HYSO5090
C  PI       CURRENT LOAD                                                        HYSO5100
C  DI       CURRENT DISPL                                                       HYSO5110
C  PS       PREVIOUS LOAD                                                       HYSO5120
C  DS       PREVIOUS DISPL                                                      HYSO5130
C  VI       CURRENT VELOCITY                                                    HYSO5140
C  VL       PREVIOUS VELOCITY                                                   HYSO5150
C  S        CURRENT STIFFNESS                                                   HYSO5160
C  HRLN     CURRENT RULE NUMBER                                                 HYSO5170
C  A        UPPER LOAD LIMIT BEFORE RULE CHANGE                                 HYSO5180
C  B        LOWER LOAD LIMIT BEFORE RULE CHANGE                                 HYSO5190
C  IDRO  ??PREVIOUS DIRECTION                                                   HYSO5200
C  RF                                                                           HYSO5210
C  PC,DC    CRACKING LOAD AND DISPLACEMENT                                      HYSO5220
C  PY,DY    YEILD LOAD AND DISPLACEMENT                                         HYSO5230
C  PU,DU    ULTIMATE LOAD AND DISPLACEMENT                                      HYSO5240
C  OC       ELASTIC STIFFNESS PO/DC                                             HYSO5250
C  CV       POST CRACKING STIFFNESS (PY-PC)/(DY-DC)                             HYSO5260
C  YU       POST YEILD STIFFNESS (PU-PY)/(DU-DY)                                HYSO5270
C  CPY      CRACKING', YEILD STIFFNESS (PY+PC)/(DY+DC)                          HYSO5280
C  CSE                                                                          HYSO5290
C  RNRL     NEXT RULE NUMBER                                                    HYSO5300
C  UM1      PEAK DISPLACEMENT IN THE POSITIVE DIRECTION                         HYSO5310
C  UM2      PEAK LOAD IN THE POSITIVE DIRECTION                                 HYSO5320
C  UM3      PEAK DISPLACEMENT IN THE NEGATIVE DIRECTION                         HYSO5330
C  UM4      PEAK LOAD IN THE NEGATIVE DIRECTION                                 HYSO5340
C  S1       UNLOADING STIFFNESS                                                 HYSO5350
C  XO                                                                           HYSO5360
C  XOUM                                                                         HYSO5370
C  XOV                                                                          HYSO5380
C  UOO                                                                          HYSO5390
C  UO                                                                           HYSO5400
C  X1UM                                                                         HYSO5410
C  U1                                                                           HYSO5420
C  U1O                                                                          HYSO5430
C  X2UO                                                                         HYSO5440
C  U2                                                                           HYSO5450
C  X5U1                                                                         HYSO5460
C  U5                                                                           HYSO5470
C  QV                                                                           HYSO5480
C  MAXFLG  FLAG, IF MAXFLG=0 STIFFNESS=0   BEYOND FAILURE POINT (DU,PU)        HYSO5490
C  ACHNG   LOAD WHERE RULE CHNGS IF LOADING CONTINUES IN CURR DIRECTION        HYSO5500
C  PRINT   PRINT DATA FOR TRACING                                              HYSO5510
C  SE      ELASTIC UNLOADING CURVE                                             HYSO5520
C  ESE     ELASTIC STRAIN ENERGY                                               HYSO5530
C  PSE     PLASTIC STRAIN ENERGY                                               HYSO5540
C  PSEOLD  PLASTIC STRAIN ENERGY AT LAST ZERO CROSSING                         HYSO5550
C  UPOS    POSITIVE DUCTILITY INFO.                                            HYSO5560
C  UNEG    NEGATIVE DUCTILITY INFO.                                            HYSO5570
C  EXCR    EXCRUSION RATIO INFO.                                               HYSO5580
C                                                                              HYSO5590
C  JI      # FOR CURRENT DIRECTION FOR LOADING AND UNLOADING                   HYSO5600
C          # FOR NEW DIRECTION FOR REVERSAL                                     HYSO5610
C          # = 2 FOR POSITIVE LOAD .  # = 1 FOR NEGATIVE LOAD                  HYSO5620
C  IREVSL  SIGN OF LOAD IN NEW DIRECTION                                       HYSO5630
C  ISGN    SIGN OF THE CURRENT LOAD FOR LOADING AND UNLOADING                  HYSO5640
C          SIGN OF THE NEW LOAD FOR REVERSAL                                   HYSO5650
C                                                                              HYSO5660
C                                                                              HYSO5670
      SUBROUTINE HYST06(PI,DI,PS,DS,VI,VS   ,S,HRLN,A,B,IDRO,RF,              HYSO5680
     & PC,DC,PY,DY,PU,DU,OC,CV,YU,CPY,CSE,RNRL,UM1,UM2,UM3,UM4,               HYSO5690
     & S1,XO ,XOUM,XOV,UOO,UO,X1UM,U1,U1O ,X2UO,U2,X5U1,U5,QV,MAXFLG,        HYSO5700
     & ACHNG,PRINT,SE,ESE,PSE,PSEOLD,UPOS,UNEG,EXCR)                          HYSO5710
C     REAL   UM(2,2),IDRO,MAXFLG,UPOS(5),UNEG(5),EXCR(6),ESE(5)               HYSO5720
      LOGICAL PRINT,BTEST                                                      HYSO5730
 8000 IF (PRINT) WRITE (6,8010) PI,DI,PS,DS,VI,VS,S,HRLN,A,B,RF,              HYSO5740
     & PC,DC,PY,DY,PU,DU,OC,CV,YU,CPY,RNRL,UM1,UM2,UM3,UM4,                   HYSO5750
     & S1,XO ,XOUM,XOV,UOO,UO,X1UM,U1,U1O ,X2UO,U2,X5U1,U5,QV,                HYSO5760
     & ACHNG,IDRO,MAXFLG,CSE,                                                 HYSO5770
     & SE,ESE(1),PSE,PSEOLD,UPOS,UNEG,EXCR                                    HYSO5780
 8010 FORMAT(                                                                 HYSO5790
     &' TAKEDA-PI=',G12.5,'   DI=',G12.5,'   PS=',G12.5,'   DS=',G12.5/HYSO5800
     &' TK-     VI=',G12.5,'   VS=',G12.5,'    S=',G12.5,' HRLN=',G12.5/HYSO5810
     &' TK-      A=',G12.5,'    B=',G12.5,'   RF=',G12.5,'   PC=',G12.5/HYSO5820
     &' TK-     DC=',G12.5,'   PY=',G12.5,'   DY=',G12.5,'   PU=',G12.5/HYSO5830
     &' TK-     DU=',G12.5,'   OC=',G12.5,'   CV=',G12.5,'   YU=',G12.5/HYSO5840
     &' TK-    CPY=',G12.5,' RNRL=',G12.5,'  UM1=',G12.5,'  UM2=',G12.5/HYSO5850
     &' TK-    UM3=',G12.5,'  UM4=',G12.5,'   S1=',G12.5,'   XO=',G12.5/HYSO5860
     &' TK-  XOUM=',G12.5,'  XOV=',G12.5,'  UOO=',G12.5,'   UO=',G12.5/HYSO5870
     &' TK-  X1UM=',G12.5,'   U1=',G12.5,'  U1O=',G12.5,' X2UO=',G12.5/HYSO5880
     &' TK-    U2=',G12.5,' X5U1=',G12.5,'   U5=',G12.5,'   QV=',G12.5/HYSO5890
     &' TK- ACHNG=',G12.5,'  IDRO=',G12.5,' MAXF=',G12.5,'  CSE=',G12.5/HYSO5900
     &' TK-     SE=',G12.5,' ESE=',G12.5,'PSE=',G12.5,' PSEO=',G12.5/   HYSO5910
     &' TK-   UP1=',G12.5,'  UP2=',G12.5,'  UP3=',G12.5/                  HYSO5920
     &' TK-   UN1=',G12.5,'  UN2=',G12.5,'  UN3=',G12.5/                  HYSO5930
     &' TK-   EX1=',G12.5,'  EX2=',G12.5,'  EX3=',G12.5/                  HYSO5940
     &' TK-   EX4=',G12.5,'  EX5=',G12.5,'  EX6=',G12.5)                  HYSO5950
C                                                                         HYSO5960
C..... RTN IF CURRENT STATE = PREVIOUS STATE                              HYSO5970
      IF ((PI.EQ.PS .OR. DI.EQ.DS) .AND. RNRL.NE.O ) RETURN               HYSO5980
C                                                                         HYSO5990
C..... GET DIRECTION AND RTN IF CURRENT STATE IS WITHIN LIMITS...         HYSO6000
      CALL IDRECT (IDR,IDRV,IDRVO,PI,PS,VI,VS)                            HYSO6010
      IF (A.GT.B .AND. PI.LE.I .AND. PI.GT.B .AND. IDR.EQ.IDRO) RETURN    HYSO6020
      IF (A.LT.B .AND. PI.GT.A .AND. PI.LT.B .AND. IDR.EQ.IDRO) RETURN    HYSO6030
C                                                                         HYSO6040
C..... CALC EXCURSION RATIO AT EVERY ZERO CROSSING...                     HYSO6050
      IF (PI*PS.LE.O)CALL DME(O,UPOS,EXCR, DV,DI,ESE,PSE,PSEOLD,CSE)      HYSO6060
C                                                                         HYSO6070
C..... INITILIZE DATA FOR FIRST CALL TO SUBROUTINE                        HYSO6080
    1 IF (RNRL.EQ.O) THEN                                                 HYSO6090
          RNRL=1                                                          HYSO6100
          UM1=-DC                                                         HYSO6110
          UM2=-PC                                                         HYSO6120
          UM3= DC                                                         HYSO6130
          UM4= PC                                                         HYSO6140
      ENDIF                                                               HYSO6150
C                                                                         HYSO6160
C..... INITILIZE DATA                                                     HYSO6170
      HRL=RNRL                                                            HYSO6180
      UM(1,1)=UM1                                                         HYSO6190
      UM(1,2)=UM2                                                         HYSO6200
      UM(2,1)=UM3                                                         HYSO6210
      UM(2,2)=UM4                                                         HYSO6220
C                                                                         HYSO6230
      IF (PI.GT.O) THEN                                                   HYSO6240
  901   ISGN =1                                                           HYSO6250
        JI=2                                                              HYSO6260
      ELSE IF (PI.LT.O) THEN                                              HYSO6270
  902   ISGN =-1                                                          HYSO6280
        JI=1                                                              HYSO6290
      ELSE IF (DI.LT.DS) THEN                                             HYSO6300
  905   ISGN =-1                                                          HYSO6310
        JI=1                                                              HYSO6320
      ELSE IF (DI.GT.DS) THEN                                             HYSO6330
  904   ISGN =1                                                           HYSO6340
        JI=2                                                              HYSO6350
      ELSE                                                                HYSO6360
  905   ISGN =1                                                           HYSO6370
        JI=1                                                              HYSO6380
      ENDIF                                                               HYSO6390
C                                                                         HYSO6400
      IF (DI.LE.DS) THEN                                                  HYSO6410
        IREVSL=-1                                                         HYSO6420
      ELSE                                                                HYSO6430
        IREVSL= 1                                                         HYSO6440
      ENDIF                                                               HYSO6450
C                                                                         HYSO6460
      IF (IDR.LT.O) THEN                                                  HYSO6470
       IF (ISGN.GT.O) CALL DME(1,UPOS,EXCR, DV,DI,ESE(2),PSE,PSEOLD,CSE)  HYSO6480
       IF (ISGN.LT.O) CALL DME(1,UNEG,EXCR,-DV,DI,ESE(5),PSE,PSEOLD,CSE)  HYSO6490
      ENDIF                                                               HYSO6500
C                                                                         HYSO6510
      GO TO ( 100, 200, 300, 400, 500, 600, 700, 800, 900,1000,           HYSO6520
     1       1100,1200,1300,1400,1500,1600),HRL                           HYSO6530
C  ===== TAKEDA RULE  1.0 =====   ELASTIC STAGE                           HYSO6540
```

```
  100 IF (IDR) 150,150,110
  110 IF (ABS(PI).GE.PC) GO TO 200
      B=PI
      A=PC*ISGN
      S=OC
           SE=S
      NRL=1
           HRLN=1.11
       GO TO 2000
  150 B=0.
      A=PC*IREVSL
      IF (PI.NE.0) A=PC*SIGN(1.0,PI)
      S=OC
           SE=S
      NRL=1
           HRLN=1.2
       GO TO 2000
C  ===== TAKEDA RULE  2.0 =====  LOADING ON PRIMARY CURVE UP TO YEILD
  200 IF (IDR) 250,250,210
  210 IF (ABS(PI).GE.PY) GO TO 500
      B=PI
      A=PY*ISGN
      S=OY
           S1=(PI+(PC*ISGN))/(DI+OC*ISGN)
           SE=S1
      NRL=2
           HRLN=2.11
       GO TO 2000
  250 B=0.
      A=PI
      S1=(PI+(PC*ISGN))/(DI+OC*ISGN)
      S=S1
           SE=S1
      NRL=5
           HRLN=2.2
       GO TO 2000
C  ===== TAKEDA RULE  3.0 =====  LOADING ON PRIMARY CURVE AFTER YEILD
  300       BOTT=MAX(ABS(DI),ABS(UM1),ABS(UM5) )
            S1=CPY* (DY/BOTT)**0.40
            SE=S1
      IF (IDR) 350,350,310
  310 IF (ABS(PI).GE.PU) GO TO 320
      A=PU*ISGN
      B=PI
      S=YU
      NRL=3
           HRLN=3.11
       GO TO 2000
  320 A=PU*ISGN
      B=PI
      S=0.
      IF (MAXFLG.NE.0) S=YU
      NRL=5
           HRLN=3.12
       GO TO 2000
  330 B=0.
      A=PI
      BOTT=MAX(ABS(DI),ABS(UM1),ABS(UM5) )
      S1=CPY* (DY/BOTT)**0.40
      S=S1
      NRL=4
           HRLN=3.2
       GO TO 2000
C  ===== TAKEDA RULE  4.0 =====  UNLOADING FROM POINT UM  ON THE PRIMARY
C                                CURVE AFTER YEILDING
  400 IF (IDR) 450,440,410
  410 IF (ARS(PI).GE.ABS(UMKJI,2))) GO TO 420
      A=UMKJI,2)
      B=PI
      S=S1
      NRL=4
           HRLN=4.11
       GO TO 2000
  420 A=PU*ISGN
      B=PI
      S=YU
      NRL=5
           HRLN=4.12
       GO TO 2000
  430 A=PI
      B=0.
      S=S1
      NRL=4
           HRLN=4.2
       GO TO 2000
  440 IF (DI.LE.DS) IDRV0=1
      IF (DI.GT.DS) IDRV0=2
      IF (ABS(UMKIDRV0,1)).GT.DC) GO TO 450
      A=PC*IREVSL
      B=0.
      S=S1
      NRL=15
           HRLN=4.51
       GO TO 2000
  450 XO=DI
      XOUM=(0.-UMKIDRV0,2))/(XO-UMKIDRV0.1))
      XOY =(0.-PY*ISGN)/(XO-DY*ISGN)
      IF (PY.GT.ABS(UMKIDRV0,2)) .AND. XOY.GT.XOUM) GO TO 460
      A=UMKIDRV0,2)
      B=0.
      S2=XOUM
      S=S2
      NRL=6
           HRLN=4.32
       GO TO 2000
  460 UMKJI,1)=DY*ISGN
      UMKJI,2)=PY*ISGN
      A=UMKJI,2)
      B=0.
      S2=XOY
      S=S2
      NRL=6
           HRLN=4.33
       GO TO 2000
C  ===== TAKEDA RULE  5.0 =====  UNLOADING FROM POINT UM ON THE PRIMARY
C                                CURVE BEFORE YEILDING
  500 IF (IDR) 550,540,510
  510 IF (ABS(PI).GE.ABSUMKJI,2))) GO TO 200
      A=UMKJI,2)
      B=PI
      S=S1
      NRL=5
           HRLN=5.11
       GO TO 2000
  530 A=PI
      B=0.
      S=S1
      NRL=5
           HRLN=5.2
       GO TO 2000
  540 IF (DI.LE.DS) IDRV=2
      IF (DI.GT.DS) IDRV=1
      IF (ABS(UMKIDRV,1)).GT.DC) GO TO 450
      A=PC*IREVSL
      B=0.
      S=S1
      NRL=14
           HRLN=5.31
       GO TO 2000
C  ===== TAKEDA RULE  6.0 =====  LOADING TOWARDS POINT UM ON THE PRIMAR
C                                CURVE.
  600 IF (IDR) 650,650,610
  610 IF (ABS(PI).GE.ABS(UMKJI,2))) GO TO 210
      A=UMKJI,2)
      B=PI
      XOUM=(PI-UMKJI,2))/(DI-UMKJI,1))
      S=XOUM
      NRL=6
           HRLN=6.11
       GO TO 2000
  650 UOD=DI
      UO=PI
      A=PI
      B=0.
      S=S1
      NRL=7
           HRLN=6.20
       GO TO 2000
C  ===== TAKEDA RULE  7.0 =====  UNLOADING FROM POINT UM AFTER RULE 6
  700 IF (IDR) 750,740,710
  710 IF (ABS(PI).GE.ABS(UO)) GO TO 720
      A=UO
      B=PI
      S=S1
      NRL=7
           HRLN=7.11
       GO TO 2000
  720 A=UMKJI,2)
      B=PI
      XOUM=(0.-UMKJI,2))/(XO-UMKJI,1))
      S=XOUM
      NRL=6
           HRLN=7.12
       GO TO 2000
  730 A=PI
      B=0.
      S=S1
      NRL=7
           HRLN=7.2
       GO TO 2000
  740 A=UMKJI,2)
      B=0.
      X1UM=(PI-UMKJI,2))/(DI-UMKJI,1))
      S=X1UM
      NRL=8
           HRLN=7.5
       GO TO 2000
C  ===== TAKEDA RULE  8.0 =====  LOADING TOWARDS POINT UM ON THE PRIMAR
C                                CURVE
  800 IF (IDR) 850,850,810
  810 IF (ABS(PI).GE.ABS(UMKJI,2))) GO TO 210
      A=UMKJI,2)
      B=PI
      X1UM=(PI-UMKJI,2))/(DI-UMKJI,1))
      S=X1UM
      NRL=8
           HRLN=8.11
       GO TO 2000
  850 U1=PI
      U1D=DI
      A=PI
      B=0.
      S=S1
      NRL=9
           HRLN=8.2
       GO TO 2000
C  ===== TAKEDA RULE  9.0 =====  UNLOADING FROM POINT U1 AFTER RULE 8
  900 IF (IDR) 950,940,910
  910 IF (ABS(PI).GE.ABS(U1       )) GO TO 810
      A=U1
      B=PI
      S=S1
      NRL=9
           HRLN=9.11
       GO TO 2000
  950 A=PI
      B=0.
      S=S1
      NRL=9
           HRLN=9.2
       GO TO 2000
  940 A=UO
      B=0.
      K2UO=(PI-UO)/(DI-UOD)
      S=K2UO
      NRL=10
           HRLN=9.3
       GO TO 2000
C  ===== TAKEDA RULE 10.0 =====  LOADING TOWARDS POINT UO
 1000 IF (IDR) 1050,1050,1010
 1010 IF (ABS(PI).GE.ABS(UO       )) GO TO 610
      A=UO
      B=PI
      K2UO=(PI-UO)/(DI-UOD)
      S=K2UO
      NRL=10
           HRLN=10.11
       GO TO 2000
 1050 U2=PI
      A=PI
      B=0.
      S=S1
      NRL=11
           HRLN=10.2
       GO TO 2000
C  ===== TAKEDA RULE 11.0 =====  UNLOADING FROM POINT U2 AFTER RULE 10
 1100 IF (IDR) 1150,1140,1110
 1110 IF (ABS(PI).GE.ABS(U2       )) GO TO 1010
```

```
      A=U2                                                              HYS04080
      B=PI                                                              HYS04090
      S=S1                                                              HYS04100
      NRL=11                                                            HYS04110
            HRLN=11.11                                                  HYS04120
       GO TO 2000                                                       HYS04130
 1150 A=PI                                                              HYS04140
      B=0.                                                              HYS04150
      S=S1                                                              HYS04160
      NRL=11                                                            HYS04170
            HRLN=11.2                                                   HYS04180
       GO TO 2000                                                       HYS04190
 1160 A=U1                                                              HYS04200
      B=0.                                                              HYS04210
      X5U1=(PI-U1)/(DI-U1D)                                             HYS04220
      S=X5U1                                                            HYS04230
      NRL=12                                                            HYS04240
            HRLN=11.5                                                   HYS04250
       GO TO 2000                                                       HYS04260
C  ===== TAKEDA RULE 12.0 =====   LOADING TOWARDS POINT U1              HYS04270
 1200 IF (IDR) 1250,1250,1210                                           HYS04280
 1210 IF (ABS(PI).GE.ABS(U1       )) GO TO  810                        HYS04290
      A=U1                                                              HYS04300
      B=PI                                                              HYS04310
      X5U1=(PI-U1)/(DI-U1D)                                             HYS04320
      S=X5U1                                                            HYS04330
      NRL=12                                                            HYS04340
            HRLN=12.11                                                  HYS04350
       GO TO 2000                                                       HYS04360
 1250 U5=PI                                                             HYS04370
      A=PI                                                              HYS04380
      B=0.                                                              HYS04390
      S=S1                                                              HYS04400
      NRL=13                                                            HYS04410
            HRLN=12.2                                                   HYS04420
       GO TO 2000                                                       HYS04430
C  ===== TAKEDA RULE 13.0 =====  UNLOADING FROM POINT U5 AFTER RULE 12  HYS04440
 1300 IF (IDR) 1350, 940,1310                                           HYS04450
 1310 IF (ABS(PI).GE.ABS(U5       )) GO TO 1210                        HYS04460
      A=U5                                                              HYS04470
      B=PI                                                              HYS04480
      S=S1                                                              HYS04490
      NRL=15                                                            HYS04500
            HRLN=15.11                                                  HYS04510
       GO TO 2000                                                       HYS04520
 1330 A=PI                                                              HYS04530
      B=0.                                                              HYS04540
      S=S1                                                              HYS04550
      NRL=15                                                            HYS04560
            HRLN=15.2                                                   HYS04570
       GO TO 2000                                                       HYS04580
C  ===== TAKEDA RULE 14.0 =====  LOADING IN THE UNCRACKED DIRECTION AFTEHYS04590
C                                CRACKING IN THE OTHER DIRECTION.       HYS04600
 1400 IF (IDR) 1450,1440,1410                                           HYS04610
 1410 IF (ABS(PI).GE.    PC       ) GO TO  210                         HYS04620
      A=PC*ISGN                                                         HYS04630
      B=0.                                                              HYS04640
      S=S1                                                              HYS04650
      NRL=14                                                            HYS04660
            HRLN=14.11                                                  HYS04670
       GO TO 2000                                                       HYS04680
 1450 A=PI                                                              HYS04690
      B=0.                                                              HYS04700
      S=S1                                                              HYS04710
      NRL=14                                                            HYS04720
            HRLN=14.2                                                   HYS04730
       GO TO 2000                                                       HYS04740
 1440 A=UMKJI,2)                                                        HYS04750
      B=0.                                                              HYS04760
      S=S1                                                              HYS04770
      NRL=5                                                             HYS04780
            HRLN=14.5                                                   HYS04790
       GO TO 2000                                                       HYS04800
C  ===== TAKEDA RULE 15.0 =====  LOADING IN THE UNCRACKED DIRECTION AFTEHYS04810
C                                VEILDING IN THE OTHER DIRECTION.       HYS04820
 1500 IF (IDR) 1550,1540,1510                                           HYS04830
 1510 IF (ABS(PI).GE.    PC       ) GO TO 1520                         HYS04840
      A=PC*ISGN                                                         HYS04850
      B=PI                                                              HYS04860
      S=S1                                                              HYS04870
      NRL=15                                                            HYS04880
            HRLN=15.11                                                  HYS04890
       GO TO 2000                                                       HYS04900
 1520 PQ=PC*ISGN                                                        HYS04910
      DQ=(PQ-PI)/S1+DI                                                  HYS04920
      QV=(PV*ISGN-PQ)/(DV*ISGN-DQ)                                      HYS04930
      A=PV*ISGN                                                         HYS04940
      B=PQ                                                              HYS04950
      S=QV                                                              HYS04960
      NRL=16                                                            HYS04970
            HRLN=15.12                                                  HYS04980
       GO TO 2000                                                       HYS04990
 1530 A=PI                                                              HYS05000
      B=0.                                                              HYS05010
      S=S1                                                              HYS05020
      NRL=15                                                            HYS05030
            HRLN=15.2                                                   HYS05040
       GO TO 2000                                                       HYS05050
 1540 IF (ABS(PI) .GE. ABS(UMKJI,2))) GO TO 1550                        HYS05060
      A=UMKJI,2)                                                        HYS05070
      B=0.                                                              HYS05080
      S=S1                                                              HYS05090
      NRL=4                                                             HYS05100
            HRLN=15.51                                                  HYS05110
       GO TO 2000                                                       HYS05120
 1550 A=PU*IREVSL                                                       HYS05130
      B=0.                                                              HYS05140
      S=VU                                                              HYS05150
      NRL=3                                                             HYS05160
            HRLN=15.52                                                  HYS05170
       GO TO 2000                                                       HYS05180
C  ===== TAKEDA RULE 16.0 =====  LOADING TOWARDS THE VEILD POINT AFTER HYS05190
C                                RULE 15.                               HYS05200
 1600 IF (IDR) 1650,1650,1610                                           HYS05210
 1610 IF (ABS(PI).GE.    PV       ) GO TO 500                          HYS05220
      A=PV*ISGN                                                         HYS05230
      B=PI                                                              HYS05240
      S=QV                                                              HYS05250
      NRL=16                                                            HYS05260
            HRLN=16.11                                                  HYS05270
       GO TO 2000                                                       HYS05280
 1650 UMKJI,1)=DV                                                       HYS05290
      UMKJI,2)=PV                                                       HYS05300
      UO=PI                                                             HYS05310
      UOO=OI                                                            HYS05320
      A=PI                                                              HYS05330
      B=0.                                                              HYS05340
      S=S1                                                              HYS05350
      S2=QV                                                             HYS05360
      XO=DI-PI/S2                                                       HYS05370
      NRL= 7                                                            HYS05380
            HRLN=16.2                                                   HYS05390
       GO TO 2000                                                       HYS05400
C                                                                       HYS05410
C ======> CALCULATE MAXIMUM PREVIOUS DEFORMATION                        HYS05420
 2000     UM1=MINKUMK1,1),DI)                                           HYS05430
          UM2=MINKUMK1,2),PI)                                           HYS05440
          UM3=MAXKUMK2,1),DI)                                           HYS05450
          UM4=MAXKUMK2,2),PI)                                           HYS05460
          RNRL=NRL                                                      HYS05470
          IDRO=IDR                                                      HYS05480
          RF=0.                                                         HYS05490
          IF (IDR.GE.0) ACHNG=A                                         HYS05500
          IF (IDR.LT.0) ACHNG=B                                         HYS05510
       RETURN                                                           HYS05520
       END                                                              HYS05530
C=======================================================================HYS00010
C            HYST07                                                     HYS00010
C                                                                       HYS00030
C       SU_BROU_TINE ELSPLS - ELASTO-PLASTIC BILINEAR HYSTERESIS MODEL  HYS00040
C                                                                       HYS00050
C       FC      = CURRENT LOAD                                          HYS00060
C       PEQ     = MODIFIED CURRENT LOAD                                 HYS00070
C       DC      = CURRENT DISPL.                                        HYS00080
C       FL      = LAST LOAD                                             HYS00090
C       DL      = LAST DISPL.                                           HYS00100
C       PY      = YEILD LOAD                                            HYS00110
C       PYEQ    = MODIFIED YEILD LOAD                                   HYS00120
C       S       = ELASTIC STIFFNESS                                     HYS00130
C       K       = CURRENT STIFFNESS (OUTPUT)                            HYS00140
C       RULE    = CURRENT RULE NUMBER, USED TO TRACE MODEL (OUTPUT)     HYS00150
C       RNRL    = NEXT RULE NUMBER (OUTPUT)                             HYS00160
C       VI      = NOT USED                                              HYS00170
C       VL      = NOT USED                                              HYS00180
C       KIE     = INELASTIC STIFFNESS                                   HYS00190
C                                                                       HYS00200
      SUBROUTINE HYST07(FC,DC,FL,DL,K,RULE,PSEOLD,A,B,                  HYS00210
     &     VI,VL,S,PV,KIE,DV,CSE,PRINT,ESE,PSE,KEV,UPOS,UNEG,EXCR)      HYS00220
      REAL    K,KIE                                                     HYS00230
      LOGICAL PRINT,BTEST                                               HYS00240
      DIMENSION UPOS(5),UNEG(5),EXCR(6),ESE(5)                          HYS00250
      FC=K*(DC-DL)+FL                                                   HYS00260
      IF (PRINT) WRITE (6,1010) FC,DC,FL,DL,K,RULE,VI,CSE,VL,S,PV,KIE,  HYS00270
     &    A,B,KEV,ESE(5),PSE,PSEOLD,UPOS,UNEG,EXCR                      HYS00280
 1010 FORMAT (                                                          HYS00290
     &' ELSPLS-FC=',G13.5,'    DC=',G13.5,'    FL=',G13.5,'  DL=',G13.5/HYS00300
     &' EL-     K=',G13.5,' RULE=',G13.5,'    VI=',G13.5,' CSE=',G13.5/HYS00310
     &' EL-    VL=',G13.5,'     S=',G13.5,'    PY=',G13.5,' KIE=',G13.5/HYS00320
     &' EL-     A=',G13.5,'     B=',G13.5/                             HYS00330
     &' EL-   KEV=',G13.5,'   ESE=',G13.5,'   PSE=',G13.5,' PSO=',G13.5/HYS00340
     &' EL-   UP1=',G13.5,'   UPZ=',G13.5,'   UPS=',G13.5/             HYS00350
     &' EL-   UN1=',G13.5,'   UN2=',G13.5,'   UNS=',G13.5/             HYS00360
     &' EL-   EX1=',G13.5,'   EX2=',G13.5,'   EXS=',G13.5/             HYS00370
     &' EL-   EX4=',G13.5,'   EX5=',G13.5,'   EX6=',G13.5)             HYS00380
C                                                                       HYS00390
C..... CALC EXCURSION RATIO AT EVERY ZERO CROSSING...                   HYS00400
      IF(FC*FL.LE.0)CALL DNE(0,UPOS,EXCR, DV,DL,ESE,PSE,PSEOLD,CSE)     HYS00410
C                                                                       HYS00420
C                                                                       HYS00430
C------------ ELASTIC REGION                                           HYS00440
      IF (RULE.LE.1 .AND. ABS(FC).LT.PY)  THEN                          HYS00450
         K=S                                                            HYS00460
         RULE=1.                                                        HYS00470
         A= PV                                                          HYS00480
         B=-PV                                                          HYS00490
         FC=S*DC                                                        HYS00500
         CALL STRENG (ESE,PSE,PSEOLD,FC,FL,DC,DL,K,S )                 HYS00510
         GO TO 1000                                                     HYS00520
      ENDIF                                                             HYS00530
C                                                                       HYS00540
C------------ INELASTIC REGION                                         HYS00550
      DIR=DC-DL                                                         HYS00560
      IF (DIR.EQ.0) RETURN                                              HYS00570
      PEQ =FC-KIE*DC                                                    HYS00580
      PEQL=FL-KIE*DL                                                    HYS00590
      PYEQ=PY*(1-KIE/S)                                                 HYS00600
      DVEL=VI*VL                                                        HYS00610
      IF (PRINT) WRITE (6,*) 'PEQ,PYEQ,DVEL,VI,VL:'                    HYS00620
      IF (PRINT) WRITE (6,*)  PEQ,PYEQ,DVEL,VI,VL                      HYS00630
C                                                                       HYS00640
C------ LOAD AT OR ABOVE POSITIVE YEILD                                 HYS00650
      IF (RULE.EQ.1) THEN                                               HYS00660
         IF (DIR.GT.0) THEN                                             HYS00670
C           ... POSITIVE YEILD ...                                      HYS00680
            CALL STRENG (ESE,PSE,PSEOLD, PY,FL, DV,DL,S,S )             HYS00690
            FL=PY                                                       HYS00700
            DL=DV                                                       HYS00710
            GO TO 210                                                   HYS00720
         ELSE                                                           HYS00730
C           ... NEGATIVE YEILD ...                                      HYS00740
            CALL STRENG (ESE,PSE,PSEOLD,-PY,FL,-DV,DL,S,S )             HYS00750
            FL=-PY                                                      HYS00760
            DL=-DV                                                      HYS00770
            GO TO 220                                                   HYS00780
         ENDIF                                                          HYS00790
      ELSE                                                              HYS00800
         IF (RULE.EQ.2.10) GO TO 210                                    HYS00810
         IF (RULE.EQ.2.20) GO TO 220                                    HYS00820
         IF (RULE.EQ.2.30) GO TO 230                                    HYS00830
      ENDIF                                                             HYS00840
      WRITE (6,*) 'INVALID RULE IN HYST07-BILINEAR HYSTERESIS MODEL'    HYS00850
C                                                                       HYS00860
C------ LOAD AT OR ABOVE POSITIVE YEILD, ON BACKBONE CURVE              HYS00870
  210 IF (DIR.LE.0) GO TO 250                                           HYS00880
      K=KIE                                                             HYS00890
      RULE=2.10                                                         HYS00900
      FC=PYEQ+DC*KIE                                                    HYS00910
      B= FC                                                             HYS00920
      A= FC*10                                                          HYS00930
      CALL STRENG (ESE,PSE,PSEOLD,FC,FL,DC,DL,K,S )                    HYS00940
      KEV=1                                                             HYS00950
      IF (DVEL.LT.0 ) THEN                                              HYS00960
           K=S                                                          HYS00970
           B= FC                                                        HYS00980
           A= FC                                                        HYS00990
      ENDIF                                                             HYS01000
```

```
      GO TO 1000
C
C------ LOAD AT OR BELOW NEGATIVE YEILD, ON BACKBONE CURVE
  220 IF (DIR.GE.0) GO TO 250
      K=KIE
      RULE=2.20
      FC=-PVEQ+DC*KIE
      B= FC
      A= FC*10
      CALL STRENG (ESE,PSE,PSEOLD,FC,FL,DC,DL,K,S )
      KEY=2
      IF (DVEL.LT.0 ) THEN
          K=S
          B= FC
          A= FC
      ENDIF
      GO TO 1000
C
C------ LOAD IN LINEAR RANGE BETWEEN BACKBONE CURVES
  250 K=S
      RULE=2.3
      FC=S*(DC-DL) + FL
      A= FC + PVEQ - PEQ
      B= FC - PVEQ - PEQ
      IF (FC.GT.A) THEN
         DA=DC*(A-FC)/S
         CALL STRENG (ESE,PSE,PSEOLD, A,FL,DA,DL,K,S )
         DL=DA
         FL=A
         GO TO 210
      ELSE IF (FC.LT.B) THEN
         DB=DC*(B-FC)/S
         CALL STRENG (ESE,PSE,PSEOLD, B,FL,DB,DL,K,S )
         DL=DB
         FL=B
         GO TO 220
      ENDIF
C
C........ CALC D & E'S BASED ON PEAK STRAIN ENERGY
      IF (KEY.EQ.1) CALL DNEC1,UPOS,EXCR, DV,DL,ESE(2),PSE,PSEOLD,CSE)
      IF (KEY.EQ.2) CALL DNEC1,UNEG,EXCR,-DV,DL,ESE(5),PSE,PSEOLD,CSE)
C
C........ CALC STRAIN ENERGY FOR THIS STEP..
      CALL STRENG (ESE,PSE,PSEOLD,FC,FL,DC,DL,K,S )
C
C........ CALC PLACTIC STRAIN ENERGY AT ZERO CROSSING ...
C
      KEY=3
C
C
 1000 IF (PRINT) WRITE (6,1011) FC,K,A,B
 1011 FORMAT
     &(' EL-END-FC=',F8.3,'     K=',F8.3,'     A=',F8.3,'     B=',F8.3)
      IF (PRINT) WRITE (6,1010) FC,DC,FL,DL,K,RULE,VI,VL,S,PV,KIE,
     &     A,B,KEY,ESE,PSE,PSEOLD,UPOS,UNEG,EXCR
      RETURN
      END
C=======================================================================
C
      SUBROUTINE IDRECT (IDR,IDRV,IDRV0,PI,PS,VI,VS)
C
      LOGICAL A,NA,B,NB,C,NC,D,ND,E
      LOGICAL BTEST
C
  1   A = ABS(PI).GE.ABS(PS)
      B = PI*PS .GE.0
      C = PI.EQ.0
      D = VS*VI.GE.0
      NA= .NOT. A
      NB= .NOT. B
      NC= .NOT. C
      ND= .NOT. D
C
      IF ( A .AND.  B .AND. NC .AND.  D ) IDR=1
      IF (          NB .AND. NC .AND. ND ) IDR=1
      IF (NA .AND.  B .AND. NC .AND. ND ) IDR=1
      IF (             C .AND.  D ) IDR=0
      IF (          NB .AND. NC .AND.  D ) IDR=0
      IF ( A .AND.  B .AND. NC .AND. ND ) IDR=-1
      IF (             C .AND. ND ) IDR=-1
      IF (NA .AND.  B .AND. NC .AND.  D ) IDR=-1
          E = PI .GE. PS
          IDRV=1
          IDRV0=2
          IF (E) IDRV=2
          IF (E) IDRV0=1
      RETURN
      END
C=======================================================================
      SUBROUTINE INTOVA(BUG,NDOF,MAXNOD,NNODE,TITLE,
     &                  IDOF,ID,JTFLG,DISP,VEL,ACC,FORCE )
C
      DIMENSION DISP(NDOF),VEL(NDOF),ACC(NDOF),A(6),FORCE(NDOF)
      DIMENSION ID(MAXNOD), JTFLG(MAXNOD), IDOF(MAXNOD,6)
      CHARACTER*80  FORMAT(2)
      CHARACTER*(*) TITLE(2)
      CHARACTER*20 OPTION
      LOGICAL  BUG,HEAD,TEST,FLAG
      LOGICAL BTEST
C
C=============================================
C= INPUT INITIAL DISPL, VELOC AND ACCEL  ===
C=============================================
C
      HEAD=.TRUE.
C--- INPUT: NODE,IGEN,INC,OPTION,DX,DY,DZ,RX,RY,RZ
   10  READ  (5,*) NODE,IGEN,INC,OPTION,(A(I),I=1,6)
       IF (NODE.LE.0) GO TO 500
       IF (HEAD) THEN
           HEAD=.FALSE.
           WRITE (6,5) TITLE(1),TITLE(2)
       ENDIF
       WRITE (6,7) NODE,IGEN,INC,OPTION,(A(I),I=1,6)
    5  FORMAT ('1 STRUCTURE.....: ',A,/
     &       '  SOLUTION.....: ',A,//
     &     '  INITIAL ACCELERATIONS, VELOCITIES DISPLACEMENTS ',
     &     'AND LOADS '/
     &     ' ==========================================',
     &     '==========='//
     &   4X,'NODE',2X,'IGEN',3X,'INC',1X,'COMPONENT',
     &        T44,'TX',10X,'TY',10X,'TZ',
     &        10X,'RX',10X,'RY',10X,'RZ')
    7  FORMAT (2X,3I6,1X,A,1P,6G12.4)
C
C------ DETERMINE IF DISPLACMENT, VELOCITY, ACCELERATION OR LOAD
       IOPT=0
       IF (TEST(OPTION,'DISPL',.FALSE.)) THEN
         IOPT=IBSET(IOPT,1)
       ELSE IF (TEST(OPTION,'VEL',.FALSE.)) THEN
         IOPT=IBSET(IOPT,2)
       ELSE IF (TEST(OPTION,'ACC',.FALSE.)) THEN
         IOPT=IBSET(IOPT,3)
       ELSE IF (TEST(OPTION,'LOA',.FALSE.)) THEN
         IOPT=IBSET(IOPT,4)
       ELSE IF (TEST(OPTION,'FOR',.FALSE.)) THEN
         IOPT=IBSET(IOPT,4)
       ELSE IF (TEST(OPTION,'END',.FALSE.)) THEN
         GO TO 500
       ELSE
         WRITE (6,*) 'INVALID OPTION:',OPTION
         GO TO 10
       ENDIF
C
C------ DETERMINE INTERNAL JOINT NUMBER
   20  JOINT=IQUICK(NODE,ID,NNODE)
       IF (JOINT.LE.0) THEN
         WRITE (6,*) 'NODE #',NODE,
     &   ' WAS NOT FOUND, CHECK INPUT, INITIAL VALUE IS OMITTED '
         GO TO 10
       ENDIF
C
C------ PUT INITIAL VALUES IN PROPER ARRAYS
       DO 50 I=1,6
         IF (A(I).EQ.0) GO TO 50
         II=IDOF(JOINT,I)
         IF (BTEST(JTFLG(JOINT),I+11)) THEN
           WRITE (6,*) 'CONSTRAINED DOF #',I,' COMPONENT IS SKIPED'
         ELSE IF (BTEST(IOPT,1)) THEN
           DISP(II)=A(I)
         ELSE IF (BTEST(IOPT,2)) THEN
           VEL(II)=A(I)
         ELSE IF (BTEST(IOPT,3)) THEN
           ACC(II)=A(I)
         ELSE IF (BTEST(IOPT,4)) THEN
           FORCE(II)=A(I)
         ENDIF
   50  CONTINUE
C
C------ GENERATE VALUES AT OTHER NODES
       IF (IGEN.GT.0) THEN
         IGEN=IGEN-1
         NODE=NODE+INC
         GO TO 20
       ENDIF
C
C------ LOOP TO READ ADDITIONAL VALUES
       GO TO 10
C
  500 CONTINUE
C
C------ PRINT OUT NON ZERO INITIAL VALUES
       FLAG=.TRUE.
       DO 520 I=1,NDOF
         IF (DISP(I).NE.0 .OR. VEL(I).NE.0 .OR. ACC(I).NE.0 .OR.
     &       FORCE(I).NE.0 ) THEN
           IF (FLAG) WRITE (6,510)
           WRITE (6,550) I,FORCE(I),DISP(I),VEL(I),ACC(I)
           FLAG=.FALSE.
         ENDIF
  520  CONTINUE
C
  510 FORMAT(//5X,'INITIAL, NON-ZERO VALUES'/
     &  7X,'DOF',7X,'FORCE',10X,'DISP',11X,'VEL ',11X,'ACC ')
  550 FORMAT (I10,4G15.6)
C
      RETURN
C
      END
C=======================================================================
C - INTERSECTION OF TWO LINES IN POINT SLOPE FORM .....................
C       DEBUG UNIT(6),SUBCHK,SUBTRACE
C       END DEBUG
C     X       = X COORDINATE OF INTERSECTION
C     Y       = Y COORDINATE OF INTERSECTION
C     X1      = X COORDINATE OF LINE 1
C     Y1      = Y COORDINATE OF LINE 1
C     S1      = SLOPE OF LINE 1
C     X2      = X COORDINATE OF LINE 2
C     Y2      = Y COORDINATE OF LINE 2
C     S2      = SLOPE OF LINE 2
C
      SUBROUTINE INTSCT(X,Y,X1,Y1,S1,X2,Y2,S2)
      IMPLICIT REAL(A-H,O-Z)
      IMPLICIT INTEGER(I-N)
      LOGICAL BTEST
      IF (S1.NE.S2) THEN
          B1= Y1 - S1*X1
          B2= Y2 - S2*X2
          X=(B1-B2)/(S2-S1)
          Y=(B1*S2-B2*S1)/(S2-S1)
      ELSE
          WRITE (6,10) X1,Y1,S1,X2,Y2,S2
   10     FORMAT (/' ****** ER','ROR IN INTSCT....',1X,
     &            'S1=S2, NO SOLN,  THUS SET X=Y=0.'//
     &            '   X1=',1P,G15.5,'  Y1=',G15.5,'  S1=',G15.5/
     &            '   X2=',1P,G15.5,'  Y2=',G15.5,'  S2=',G15.5/)
          X=0
          Y=0
          CALL ERRTRA
      ENDIF
      RETURN
      END
C=======================================================================
      INTEGER FUNCTION IQUICK(TARGET,ID,N)
      INTEGER TARGET,ID(N)
      LOGICAL BTEST
C
      IF (ID(1).GT.TARGET .OR. ID(N).LT.TARGET ) GO TO 100
      IQUICK=1
      IF (ID(IQUICK).EQ.TARGET ) RETURN
      IQUICK=N
      IF (ID(IQUICK).EQ.TARGET ) RETURN
      J=(N+1)/2
      IQUICK=J
  10  J= J   /2
      IF (IQUICK.LT.1) IQUICK=1
```

```
      IF (IQUICK.GT.N) IQUICK=N                                         IQU00170
      IF (IDC IQUICK).EQ.TARGET)  RETURN                                IQU00180
      IF (IDC IQUICK).GT.TARGET .AND. J.NE.0 )  THEN                    IQU00190
         IQUICK=IQUICK-J                                                IQU00200
         GO TO 10                                                       IQU00210
      ELSE IF (IDC IQUICK).LT.TARGET .AND. J.NE.0 )  THEN               IQU00220
         IQUICK=IQUICK+J                                                IQU00230
         GO TO 10                                                       IQU00240
      ENDIF                                                             IQU00250
C                                                                       IQU00260
      I=MAX(IQUICK-2,1)                                                 IQU00270
      J=MIN(I+4,N)                                                      IQU00280
      DO 20 IQUICK=I,J,I                                                IQU00290
       IF (IDC IQUICK).EQ.TARGET)  RETURN                               IQU00300
   20 CONTINUE                                                          IQU00310
C                                                                       IQU00320
      DO 30 IQUICK=1,N                                                  IQU00330
       IF (IDC IQUICK).EQ.TARGET)  RETURN                               IQU00340
   30 CONTINUE                                                          IQU00350
C                                                                       IQU00360
  100 WRITE (6,110) TARGET,IQUICK,IDC IQUICK)                           IQU00370
  110 FORMAT (' TARGET:',I8,'   NOT FOUND, LAST IDC',I6,'):',I8)        IQU00380
      IQUICK=0                                                          IQU00390
      RETURN                                                            IQU00400
      END                                                               IQU00410
C=======================================================================IS000010
C                                                                       IS000040
      SUBROUTINE ISORT(ID,INDEX,N)                                      IS000050
      INTEGER ID(N)                                                     IS000060
      INTEGER INDEX(N)                                                  IS000070
      LOGICAL FLAG                                                      IS000080
      LOGICAL BTEST                                                     IS000090
      DO 10 I=1,N                                                       IS000100
       INDEX(I)=I                                                       IS000110
   10 CONTINUE                                                          IS000120
C                                                                       IS000130
      DO 50 I=N,2,-1                                                    IS000140
       FLAG=.TRUE.                                                      IS000150
       DO 20 J=2,I                                                      IS000160
        IF ( IDC INDEX(J)).LT.IDC INDEX(J-1)) ) THEN                    IS000170
           K=INDEX(J)                                                   IS000180
           INDEX(J)=INDEX(J-1)                                          IS000190
           INDEX(J-1)=K                                                 IS000200
           FLAG=.FALSE.                                                 IS000210
        ENDIF                                                           IS000220
   20 CONTINUE                                                          IS000230
      IF (FLAG) RETURN                                                  IS000240
   30 CONTINUE                                                          IS000250
      RETURN                                                            IS000260
      END                                                               IS000270
C=======================================================================JOI00010
      SUBROUTINE JOIDSP(IOPT,MAXNOD,NDOF,NNODE,NLOAD,NCIN,NF,ID,IDOF,   JOI00040
     &     CONST,DISP,TITLE,NAME,STEPID,HEAD,NCOS,COSINE,JTCOS,JTFLG)   JOI00050
      DIMENSION IDOF(MAXNOD,6)  ,CONST(MAXNOD,6),JTFLG(MAXNOD)          JOI00060
      DIMENSION DISP(NDOF,NLOAD)                                        JOI00070
      DIMENSION ID(MAXNOD)                                              JOI00080
      DIMENSION DC(6),GC(6),COSINE(3,3,NCOS),JTCOS(MAXNOD)              JOI00090
      CHARACTER*(*) TITLE(2),STEPID,NAME                                JOI00100
      CHARACTER*15 CDC(6)                                               JOI00110
      CHARACTER*60 EQUAL                                                JOI00120
      LOGICAL HEAD,FLAG,BTEST                                           JOI00130
      INTEGER DOF                                                       JOI00140
      DATA EQUAL                                                        JOI00150
     &  /'============================================================'/JOI00160
C                                                                       JOI00170
      NLEN=MIN(LEN(NAME)+1,60)                                          JOI00180
      NC=1                                                              JOI00190
C                                                                       JOI00200
C==============================                                         JOI00210
C= LOOP FOR EACH LOAD CASE  ===                                         JOI00220
C==============================                                         JOI00230
C                                                                       JOI00240
      DO 300 L=1,NLOAD                                                  JOI00250
        NOSPLS=0                                                        JOI00260
C                                                                       JOI00270
        IF (.NOT.(NCOS.EQ.1 .AND. COSINE(1,1,1).EQ.1 .AND.              JOI00280
     &       COSINE(2,2,1).EQ.1  .AND. COSINE(3,3,1).EQ.1 )             JOI00290
     &       .AND. IOPT.EQ.2 ) GO TO 210                                JOI00300
C                                                                       JOI00310
C================================                                       JOI00320
C= PRINT GLOBAL DISPLACEMENTS ===                                       JOI00330
C================================                                       JOI00340
C NOTE: DISP MAY CONTAIN DISPLACEMENTS, VELOCITIES, ACCELERATIONS OR    JOI00350
C        EIGENVALUES...                                                 JOI00360
C        ALL OF WHICH ARE PRINTED OUT AT EACH NODE....                  JOI00370
C                                                                       JOI00380
        IF (HEAD) WRITE (6,310) TITLE(1),TITLE(2)                       JOI00390
        IF (IOPT.NE.2) WRITE (6,315) NAME,L,STEPID,EQUAL(1:NLEN)        JOI00400
        IF (IOPT.EQ.2) WRITE (6,415) NAME,          EQUAL(1:NLEN)       JOI00410
        DO 100 I=1,NNODE                                                JOI00420
C                                                                       JOI00430
C.............. TRANSFER DISPL TO LOCAL VECTOR ..............          JOI00440
          DO 20 J=1,6                                                   JOI00450
            IJ=IDOF(I,J)                                                JOI00460
   20       DC(J)=DISP(IJ,L)                                            JOI00470
C                                                                       JOI00480
C.............. ADD ROTATIONS TO DISPL FOR CONSTRAINTS                 JOI00490
C.............. IF UNCONSTRAINED, CONST(I,J)=0                         JOI00500
          DC(1) = DC(1) + DC(5)*CONST(I,5) + DC(6)*CONST(I,5)           JOI00510
          DC(2) = DC(2) + DC(4)*CONST(I,1) + DC(6)*CONST(I,6)           JOI00520
          DC(3) = DC(3) + DC(4)*CONST(I,2) + DC(5)*CONST(I,4)           JOI00530
C                                                                       JOI00540
C.............. TRANSFER TO GLOBAL DISPLACEMENTS .....                 JOI00550
          JCOS=JTCOS(I)                                                 JOI00560
          DO 25 J1=1,3                                                  JOI00570
            I2=J1+3                                                     JOI00580
            GC(J1)=0                                                    JOI00590
            GC(I2)=0                                                    JOI00600
            DO 25 J2=1,3                                                JOI00610
              J2=J2+3                                                   JOI00620
              GC(J1)=GC(J1)+COSINE(J1,J1,JCOS)*DC(J1)                   JOI00630
   25         GC(I2)=GC(I2)+COSINE(J1,J1,JCOS)*DC(J2)                   JOI00640
C                                                                       JOI00650
C.............. WRITE DISPL                                            JOI00660
          DO 50 J=1,6                                                   JOI00670
   50       WRITE (CDC(J),9) GC(J)                                      JOI00680
    9     FORMAT (1P,G14.6,' ')                                         JOI00690
          WRITE (6,220) IDC(I),(CDC(K),K=1,6)                           JOI00700
          NOSPLS=NOSPLS+1                                               JOI00710
  100   CONTINUE                                                        JOI00720
C                                                                       JOI00730
      IF (NCOS.EQ.1 .AND. COSINE(1,1,1).EQ.1 .AND. COSINE(2,2,1).EQ.1   JOI00740
     &              .AND. COSINE(3,3,1).EQ.1 ) GO TO 500                JOI00750
C                                                                       JOI00760
C=============================                                          JOI00770
C= PRINT JOINT DISPLACEMENTS ===                                        JOI00780
C=============================                                          JOI00790
C NOTE: DISP MAY CONTAIN DISPLACEMENTS, VELOCITIES, ACCELERATIONS OR    JOI00800
C        EIGENVALUES...                                                 JOI00810
C        ALL OF WHICH ARE PRINTED OUT AT EACH NODE....                  JOI00820
C                                                                       JOI00830
  210   IF (HEAD.AND. NOSPLS.GT.20) WRITE (6,310) TITLE(1),TITLE(2)     JOI00840
        IF (IOPT.NE.2) WRITE (6,316) NAME,L,STEPID,EQUAL(1:NLEN)        JOI00850
        IF (IOPT.EQ.2) WRITE (6,416) NAME,          EQUAL(1:NLEN)       JOI00860
        DO 200 I=1,NNODE                                                JOI00870
C                                                                       JOI00880
C.............. TRANSFER GLOBAL DISPL TO LOCAL VECTOR ..............   JOI00890
          DO 219 J=1,6                                                  JOI00900
            IJ=IDOF(I,J)                                                JOI00910
  219       DC(J)=DISP(IJ,L)                                            JOI00920
C                                                                       JOI00930
C.............. ADD ROTATIONS TO DISPL FOR CONSTRAINTS                 JOI00940
C.............. IF UNCONSTRAINED, CONST(I,J)=0                         JOI00950
          DC(1) = DC(1) + DC(5)*CONST(I,5) + DC(6)*CONST(I,5)           JOI00960
          DC(2) = DC(2) + DC(4)*CONST(I,1) + DC(6)*CONST(I,6)           JOI00970
          DC(3) = DC(3) + DC(4)*CONST(I,2) + DC(5)*CONST(I,4)           JOI00980
C                                                                       JOI00990
C.............. WRITE DISPL                                            JOI01000
          DO 250 J=1,6                                                  JOI01010
            WRITE (CDC(J),9) DC(J)                                      JOI01020
            IJ=IDOF(I,J)                                                JOI01030
  250       IF (IJ.GT.NF) CDC(J)(15:15)='*'                             JOI01040
        WRITE (6,221) IDC(I),(CDC(K),K=1,6),JTCOS(I)                    JOI01050
  200 CONTINUE                                                          JOI01060
      WRITE (6,520)                                                     JOI01070
C                                                                       JOI01080
  500 CONTINUE                                                          JOI01090
C                                                                       JOI01100
      RETURN                                                            JOI01110
C                                                                       JOI01120
   10 FORMAT (1P,G15.7)                                                 JOI01130
  220 FORMAT (5X,I5,6A)                                                 JOI01140
  221 FORMAT (5X,I5,6A,I10)                                             JOI01150
  310 FORMAT ('1 STRUCTURE....: ',A,/                                   JOI01160
     &        ' SOLUTION.....: ',A)                                     JOI01170
  315 FORMAT (//                                                        JOI01180
     &        2X, 'GCS ',A,'.  LOADING #',I5,5X,A, /                   JOI01190
     &        1X,'=====',A//                                           JOI01200
     &  6X,'NODE',7X,'DX',15X,'DY',15X,'DZ',15X,'RX',15X,'RY',15X,'RZ'/)JOI01210
  316 FORMAT (//                                                        JOI01220
     &        2X, 'JCS ',A,'.  LOADING #',I5,5X,A, /                   JOI01230
     &        1X,'=====',A//                                           JOI01240
     &  6X,'NODE',7X,'DX',15X,'DY',15X,'DZ',15X,'RX',15X,'RY',15X,'RZ',  JOI01250
     &  6X,'  COSINE #'/)                                               JOI01260
  415 FORMAT (//                                                        JOI01270
     &        2X, 'GCS ',A/                                            JOI01280
     &        1X,'=====',A/                                            JOI01290
     &     25X,'NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'//   JOI01300
     &  6X,'NODE',7X,'DX',15X,'DY',15X,'DZ',15X,'RX',15X,'RY',15X,'RZ'/)JOI01310
  416 FORMAT (//                                                        JOI01320
     &        2X, 'JCS ',A/                                            JOI01330
     &        1X,'=====',A/                                            JOI01340
     &     25X,'NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'//   JOI01350
     &  6X,'NODE',7X,'DX',15X,'DY',15X,'DZ',15X,'RX',15X,'RY',15X,'RZ',  JOI01360
     &  6X,'  COSINE #'/)                                               JOI01370
  520 FORMAT (4X,'NOTE: * DENOTES A RESTRAINED DEGREE OF FREEDOM')      JOI01380
C                                                                       JOI01390
      END                                                               JOI01400
C=======================================================================JOI00010
      SUBROUTINE JOILOAC&UG,MAXNOD,NDOF,NNODE,NLOAD,NF,ID,IDOF,CONST,   JOI00050
     &     FORCE,DSPFLG,JTFLG,TITLE,HEAD)                               JOI00060
      DIMENSION IDOF(MAXNOD,6),CONST(MAXNOD,6),FORCE(NDOF,NLOAD)        JOI00070
      DIMENSION ID(MAXNOD),JTFLG(MAXNOD)                                JOI00080
      CHARACTER*(*) TITLE(2)                                            JOI00090
      CHARACTER*4  DIR                                                  JOI00100
      CHARACTER*1  CX(6)                                                JOI00110
      CHARACTER*25 SUPT                                                 JOI00120
      CHARACTER*18 DOF                                                  JOI00130
      CHARACTER*14 F(6)                                                 JOI00140
      LOGICAL DSPFLG,FIRST,BUG,HEAD                                     JOI00150
      LOGICAL BTEST                                                     JOI00160
      LOAD=1                                                            JOI00170
      FIRST=.TRUE.                                                      JOI00180
      DSPFLG=.FALSE.                                                    JOI00190
    5 FORMAT ('1 STRUCTURE....: ',A,/                                   JOI00200
     &        ' SOLUTION.....: ',A,/)                                   JOI00210
    6 FORMAT ('    APPLIED JOINT LOADS',/                               JOI00220
     &        '    ====================',/)                            JOI00230
C                                                                       JOI00240
C=======================                                                JOI00250
C= INPUT LOADING DATA ==                                                JOI00260
C=======================                                                JOI00270
   10 READ (5,*) LOAD,JOINT,JTGEN,JTINC,DIR,VALUE                      JOI00280
      IF (INDEX(DIR,'END').NE.0) GO TO 100                              JOI00290
   15 DOF=' '                                                           JOI00300
      IF (FIRST) THEN                                                   JOI00310
          IF (HEAD) WRITE (6,5) TITLE(1),TITLE(2)                       JOI00320
          WRITE (6,6)                                                   JOI00330
          FIRST=.FALSE.                                                 JOI00340
      ENDIF                                                             JOI00350
C                                                                       JOI00360
C------- CHECK FOR A VALID JOINT NUMBER...                              JOI00370
      IF (JOINT.LT.ID(1) .OR. JOINT.GT.ID(NNODE) ) THEN                 JOI00380
         WRITE (6,21) LOAD                                              JOI00390
         GO TO 10                                                       JOI00400
      ENDIF                                                             JOI00410
C                                                                       JOI00420
       J=IQUICK(JOINT,ID,NNODE)                                         JOI00430
       SUPT=' '                                                         JOI00440
C                                                                       JOI00450
C-------------------------------------------------------------------    JOI00460
C------- PUT A LOAD ON A JOINT                                          JOI00470
C NOTES:  - IX, IY, IZ, MX, MY AND Z ARE THE DOF THAT THE LOADS ARE     JOI00480
C           APPLIED TO.                                                 JOI00490
C         - THE MOMENTS RESULTING FROM FX, FY AND FZ ARE DUE TO         JOI00500
C           CONSTRAINTS.  IF A NODE IS NOT CONSTRAINED, THEN THE VALUE  JOI00510
C           OF CONST IS ZERO...                                         JOI00520
C         - LOADS APPLIED TO RESTRAINED DOF ARE INTERPERATED AS         JOI00530
C           SUPPORT DISPLACEMENTS. IE: SUPPORT SETTELMENT OR            JOI00540
C           DISPLACEMENT CONTROL(.....                                  JOI00550
C-------------------------------------------------------------------    JOI00560
      IF (INDEX(DIR,'FX').NE.0) THEN                                    JOI00570
         IX=IDOF(J,1)                                                   JOI00580
         FORCE(IX,LOAD)=FORCE(IX,LOAD)+VALUE                            JOI00590
         IF ( BTEST(JTFLG(J),2) ) THEN                                  JOI00600
           MY=IDOF(J,5)                                                 JOI00610
           FORCE(MY,LOAD)=FORCE(MY,LOAD)+VALUE*CONST(J,3)               JOI00620
           MZ=IDOF(J,6)                                                 JOI00630
```

```
            FORCE(MZ,LOAD)=FORCE(MZ,LOAD)+VALUE*CONST(J,5)
            WRITE (DOF,55) IX,MY,MZ
          ELSE
            WRITE (DOF,55) IX
          ENDIF
          IF (IX.GT.NF) THEN
            DSPFLG=.TRUE.
            SUPT='...JOINT DISPLACEMENT...'
          ENDIF
        ELSE IF (INDEX(DIR,'FY').NE.0) THEN
          IY=IDOF(J,2)
          FORCE(IY,LOAD)=FORCE(IY,LOAD)+VALUE
          IF (BTEST(JTFLG(J),13)) THEN
            MX=IDOF(J,4)
            FORCE(MX,LOAD)=FORCE(MX,LOAD)+VALUE*CONST(J,1)
            MZ=IDOF(J,6)
            FORCE(MZ,LOAD)=FORCE(MZ,LOAD)+VALUE*CONST(J,6)
            WRITE (DOF,55) IY,MX,MZ
          ELSE
            WRITE (DOF,55) IY
          ENDIF
          IF (IY.GT.NF) THEN
            DSPFLG=.TRUE.
            SUPT='...JOINT DISPLACEMENT...'
          ENDIF
        ELSE IF (INDEX(DIR,'FZ').NE.0) THEN
          JZ=IDOF(J,3)
          FORCE(JZ,LOAD)=FORCE(JZ,LOAD)+VALUE
          IF (BTEST(JTFLG(J),14)) THEN
            MX=IDOF(J,4)
            FORCE(MX,LOAD)=FORCE(MX,LOAD)+VALUE*CONST(J,2)
            MY=IDOF(J,5)
            FORCE(MY,LOAD)=FORCE(MY,LOAD)+VALUE*CONST(J,4)
            WRITE (DOF,55) JZ,MX,MY
          ELSE
            WRITE (DOF,55) JZ
          ENDIF
          IF (JZ.GT.NF) THEN
            DSPFLG=.TRUE.
            SUPT='...JOINT DISPLACEMENT...'
          ENDIF
        ELSE IF (INDEX(DIR,'MX').NE.0) THEN
          MX=IDOF(J,4)
          FORCE(MX,LOAD)=FORCE(MX,LOAD)+VALUE
          WRITE (DOF,55) MX
          IF (MX.GT.NF) THEN
            DSPFLG=.TRUE.
            SUPT='...JOINT DISPLACEMENT...'
          ENDIF
        ELSE IF (INDEX(DIR,'MY').NE.0) THEN
          MY=IDOF(J,5)
          FORCE(MY,LOAD)=FORCE(MY,LOAD)+VALUE
          WRITE (DOF,55) MY
          IF (MY.GT.NF) THEN
            DSPFLG=.TRUE.
            SUPT='...JOINT DISPLACEMENT...'
          ENDIF
        ELSE IF (INDEX(DIR,'MZ').NE.0) THEN
          MZ=IDOF(J,6)
          FORCE(MZ,LOAD)=FORCE(MZ,LOAD)+VALUE
          WRITE (DOF,55) MZ
          IF (MZ.GT.NF) THEN
            DSPFLG=.TRUE.
            SUPT='...JOINT DISPLACEMENT...'
          ENDIF
        ELSE
          WRITE (6,20) JOINT,DIR,VALUE
        ENDIF
        WRITE (6,50) LOAD,JOINT,DIR,DOF,VALUE,SUPT
        IF (JTGEN.GT.0) THEN
          JTGEN=JTGEN-1
          JOINT=JOINT+JTINC
          GO TO 15
        ENDIF
        GO TO 10
C
   20 FORMAT (' INVALID JOINT LOAD... ',I6,2X,A,2X,1P,G14.6,
     &       ' --- LOAD IS SKIPPED')
   21 FORMAT (' INVALID LOAD CASE ... ',I6,
     &       ' --- LOAD IS SKIPPED')
   30 FORMAT (' LOAD CASE:',I3,'  JOINT:',I6,'  DIRECTION: ',A,
     &  ' DOF(S)',A,'  MAGNITUDE:',1P,G14.6,2X,A)
   55 FORMAT (I5,',',I5,',',I5,',',I5)
C
C------- RETURN IF NO JOINT LOADS WERE INPUT
  100 IF (FIRST) RETURN
C
      IF (.NOT.BUG) RETURN
C------- PRINT JOINT LOADS ON EACH DOF
      NS=NDOF+5
      DO 120 L=1,NLOAD
        IF (HEAD) WRITE (6,5   ) TITLE(1),TITLE(2)
        WRITE (6,130) L
        DO 120 I=1,NS,5
        I5=MIN(I+4,NDOF)
        DO 110 J=I,I5
          K=J-I+1
          CX(K)=' '
  110     IF (J.GT.NF) CX(K)='*'
          WRITE (6,140) (J,FORCE(J,L),CX(J-I+1),J=I,I5)
  120 CONTINUE
C
  130 FORMAT (' APPLIED JOINT LOADS,   LOADING #',I5,10X,
     &        ' NOTE: * DENOTES SUPPORT DISPLACEMENT'/
     &        ' ========================= '//
     &  5X,5(' DOF',7X,'LOAD    ')/)
  140   FORMAT (5X,5(OP,I5,1P,G14.6,A) )
C
      DO 220 L=1,NLOAD
        IF (HEAD) WRITE (6,5   ) TITLE(1),TITLE(2)
        WRITE (6,200) L
        DO 220 I=1,NNODE
        DO 210 J=1,6
          CX(J)=' '
          IF (BTEST(JTFLG(I),J+11)) THEN
            F(J)=' '
          ELSE
            K=IDOF(I,J)
            WRITE (F(J),240) FORCE(K,L)
            IF (K.GT.NF) CX(J)='*'
          ENDIF
  210     CONTINUE
          WRITE (6,250) ID(I),(F(J),CX(J),J=1,6)
```

```
  220 CONTINUE
C
  200 FORMAT (' APPLIED JOINT LOADS,   LOADING #',I5,10X,
     &        ' NOTE: * DENOTES SUPPORT DISPLACEMENT'/
     &        ' ========================= '//
     &  6X,'NODE',7X,'FX',15X,'FY',15X,'FZ',15X,'MX',15X,'MY',15X,'MZ'/)
  250   FORMAT (5X,I5,12A )
  240   FORMAT (1P,G14.6)
C
      RETURN
      END
C=======================================================================
C  - KMIN - CALCULATES STIFFNESS BASED ON DX AND DV .....................
C     KMIN    =
C     DD      = CHANGE IN DISPLACMENT
C     DP      = CHANGE IN LOAD
C     SI      = MAXIMUM AND DEFULT STIFFNESS
      REAL FUNCTION KMIN(DD,DP,SI)
      LOGICAL BTEST
            IF ((DD*DP).GT. 0.00)  THEN
              KMIN=MIN(DP/DD),SI)
            ELSE
              KMIN=SI
            ENDIF
      RETURN
      END
C=======================================================================
      SUBROUTINE LINACC(IOPT,BUG,DT,T0,TF,T,THETA,NEWKDV,
     &    L1,L2,L3,L4,L5,L6,IMOMAS,IMD,IMDS,IMDKG,NDOF,NFREE,
     &    KGLOAD,KGFORM,
     &    A     ,V     ,D     ,P     ,
     &    DA    ,DV    ,DD    ,DP    ,WORK  ,
     &    MASS  ,MDMASS,DAMP  ,STIFF ,MD    ,
     &    KG    ,MDKG  ,ACC   ,GRAV  ,
     &    S     ,MDS   ,Q     ,R     ,PBAR  ,
     &    FDAMP,FMASS,C0,C1,C2,C3,C4,C5,C6,ALPHA,BETA,C10,C11,ABORT)
      INTEGER FDAMP,FMASS
      REAL MASS,KG
      LOGICAL BUG,NEWKDV,BTEST,ABORT
      DIMENSION A(NDOF),V(NDOF),D(NDOF),DA(NDOF),DV(NDOF),DD(NDOF)
      DIMENSION P(NDOF),DP(NDOF),DUMMY(1),IDUMMY(1),WORK(2*NDOF)
      DIMENSION PBAR(L3:L4),Q(L5:L6),R(L5:L6)
      DIMENSION MDMASS(NDOF+1),MASS(IMOMAS)
      DIMENSION MDKG(NDOF+1)  ,KG(IMDKG)
      DIMENSION MD(NDOF+1)    ,STIFF(IMD)
      DIMENSION MDS(NDOF+1)   ,S(IMDS)     ,DAMP(3)
C
      GO TO (100,200,300) IOPT
C
C=======================
C= INITIALIZE VALUES ===
C=======================
  100 CONTINUE
      ALPHA=DAMP(1)
      BETA =DAMP(2)
      C0=5.*ALPHA/DT + 6./(DT**2)
      C1=1.0/(1+5.*BETA/DT)
      C2=C0*C1
      C3=ALPHA-C2*BETA
      C4=6./DT
      C5=DT/2.
      C6=0.
      C10=5./DT
      C11=6./(DT**2)
      IF (BUG) WRITE (6,101) ALPHA,BETA,C0,C1,C2,C3,C4,C5,C6,C10,C11
  101 FORMAT (1P,
     &' LINACC  ALPHA=',G12.5,' BETA=',G12.5,'  C0=',G12.5,' C1=',G12.5/
     &' LINACC    C2=',G12.5,'   C3=',G12.5,'  C4=',G12.5,' C5=',G12.5/
     &' LINACC    C6=',G12.5,'  C10=',G12.5,' C11=',G12.5)
C
C------- DEVELOP SKYLINE OF DYNAMIC STIFFNESS (S)
C.......  MDS(I)   = ADDRESS OF DYNAMIC STIFF. MAIN DIAGONAL TERM ROW I
C.......  MD(I)    = ADDRESS OF STIFFNESS MAIN DIAGONAL TERM ROW I
C.......  MDMAS(I) = ADDRESS OF MASS MAIN DIAGONAL TERM ROW I
C.......  MDKG(I)  = ADDRESS OF GEOMETRIC STIFF. MAIN DIAGONAL TERM ROW I
C
C.......  LSTIFF = # OF STIFFNESS TERMS IN COLUMN J
C.......  LMASS  = # OF MASS TERMS IN COLUMN J
C.......  LKG    = # OF KG TERMS IN COLUMN J
C.......  L      = # OF S  TERMS IN COLUMN J
C
C
C------ CHECK FOR PROPORTIONAL DAMPING ....
      IF (FDAMP.NE.1) THEN
        WRITE (6,*) '*****************************************'
        WRITE (6,*) '*  PROPORTIONAL DAMPING HAS NOT SPECIFIED  *'
        WRITE (6,*) '*  SOLUTION IS ABORTED...                  *'
        WRITE (6,*) '*****************************************'
        STOP
      ENDIF
      RETURN
C
C=======================================================================
C= SOLVE FOR INCRIMENTAL DISPLACEMENTS, VELOCITIES AND ACCELERATIONS ==
C=======================================================================
  200 CONTINUE
      IF (BUG) THEN
C       CALL ERRTRA
        DO 205 I=L3,L4
  205   WRITE (6,206) I,D(I),V(I),A(I),P(I),DP(I)
  206   FORMAT (' DOF:',I6,'  DISP:',1P,G12.4,'  VEL:',G12.4,
     &        '  ACC:',G12.4,'  LOAD:',   G12.4,'   DP:',G12.4)
      ENDIF
C
C------- FORM DYNAMIC STIFFNESS
C.......  SOME LOCAL VARIABLES
C J     = COLUMN NUMBER FOR MASS AND STIFF
C I     = ROW NUMBER FOR MASS AND STIFF
C J1    = COLUMN NUMBER FOR S
C I1    = ROW NUMBER FOR S
C IJSTIF = ADDRESS OF ELEMENT I,J IN MATRIX STIFF
C IJMASS = ADDRESS OF ELEMENT I,J IN MATRIX MASS
C IJKG   = ADDRESS OF ELEMENT I,J IN GEOMETRIC STIFFNESS MATRIX
C IJS    = ADDRESS OF ELEMENT I1,J1 IN MATRIX S
C LSTIFF = ROW # CONTAINING THE FIRST NONZERO TERM IN COLUMN J OF STIFF
C LMASS  = ROW # CONTAINING THE FIRST NONZERO TERM IN COLUMN J OF MASS
C LKG    = ROW # CONTAINING THE FIRST NONZERO TERM IN COLUMN J OF KG
C LS     = ROW # CONTAINING THE FIRST NONZERO TERM IN COLUMN J OF S
C
C-------- FORM DYNAMIC STIFFNESS IF NEWKDV IS TRUE ----------------------
      IF (NEWKDV) THEN
C
C     DETERMINE THE FACTOR FOR KG
      IF (KGLOAD.EQ.1) THEN
```

```
            CKG=(1+ACC)
      ELSE IF (KGLOAD.EQ.2) THEN
            CKG= 1.00
      ENDIF
C
C---- K(BAR) = STIFFNESS
        DO 225 J=LS,L4
          J1=J-LS+1
          MDJP=MDOF(J+1)
          MDJ =MDOF(J)
          LSTIFF=J-(MDOF(J+1)-MDOF(J)-1)
          MDSJ1P=MDOS(J1+1)
          MDSJ1 =MDOS(J1)
          LS    =J1-(MDOS(J1+1)-MDOS(J1)-1)+L2
          L=MAX(LS,LSTIFF)
          DO 210 I=J,L,-1
            I1=I-LS+1
            IJS=(MDOS(J1)+J1)-I1
            IJSTIF=(MDOF(J)+J)-I
  210       S(IJS)=STIFF(IJSTIF)
C
          IF.(LS.LT.LSTIFF) THEN
            DO 215 I=L-1,LS,-1
              IJS=(MDOS(J1)+J1)-I1
  215         S(IJS)=0
          ENDIF
C
C---- K(BAR) = K(BAR) + C2*MASS
          LMASS=J-(MDMASS(J+1)-MDMASS(J)-1)
          L=MAX(LS,LMASS)
          DO 220 I=J,L,-1
            I1=I-LS+1
            IJS=(MDOS(J1)+J1)-I1
            IJMASS=(MDMASS(J)+J)-I
  220       S(IJS)=MASS(IJMASS)*C2  + S(IJS)
C
          IF (KGFORM.EQ.2) THEN
C
C.          NOTE: THE GEOMETRIC STIFFNESS HAS DERIVED WITH COMPRESSION
C.          BEING POSITIVE, KG HAS POSITIVE TERMS ON THE MAIN DIAGIONAL
C.          REPRESENTING COMPRESSION, THUS KG IS SUBTRACTED FROM S
C.          CKG IS  AN INCREASE FOR VERT ACCEL
C
C---- K(BAR) = K(BAR) - CKG*GEOMETRIC STIFFNESS
            LKG=J-(MDKG(J+1)-MDKG(J)-1)
            L=MAX(LS,LKG)
            DO 222 I=J,L,-1
              I1=I-LS+1
              IJS=(MDOS(J1)+J1)-I1
              IJKG=(MDKG(J)+J)-I
  222         S(IJS)=S(IJS) - KG(IJKG)*CKG
          ENDIF
C
  225   CONTINUE
      ENDIF
C
C------- CALCULATE Q AND R VECTORS
      DO 250 I=LS,L4
        QC(I)= C4*V(I) +  3*A(I)
  250   RC(I)=  3*V(I) + C5*A(I)
C
C------- FORM DYNAMIC LOAD
C
C..... INITILIZE PBAR= OP*THETA
C       THETA=1 FOR LINEAR ACCELERATION
C       THETA=2 WHEN LINACC IS CALLED BY AVERAGE ACCELERATION
      DO 260 I=LS,L4
  260   PBAR(I)=  OP(I)*THETA
C
      IF (FDAMP.EQ.1) THEN
C
C..... PBAR=PBAR +MASS*( QC(I) + C5*RC(I) )
        DO 250 J=LS,L4
          MDJ=MDMASS(J)+J
          M=J-(MDMASS(J+1)-MDMASS(J))+1
          M=MAX(M,LS)
          DO 250 I=M,J
            PBAR(I)=PBAR(I)+ MASS(MDJ-I)*( QC(J)+C5*RC(J) )
  250       CONTINUE
        DO 255 I=LS+1,L4
          MDI=MDMASS(I)+I
          M=I-(MDMASS(I+1)-MDMASS(I))+1
          M=MAX(M,LS)
          DO 255 J=M,I-1
            PBAR(I)=PBAR(I)+ MASS(MDI-J)*( QC(J)+C5*RC(J) )
  255     CONTINUE
      ENDIF
C
      IF (BUG) THEN
        DO 241 I=LS,L4
  241   WRITE (6,242) I,QC(I),RC(I),PBAR(I)
  242   FORMAT (' DOF:',I6,'    Q:',1P,G12.4,'    R:',G12.4,
     &       ' PBAR:',G12.4)
      ENDIF
C
C------- SOLVE FOR INCRIMENTAL DISPLACEMENT OVER TIME INTERVAL DT
C..... NOTE: - WORK IS USED FOR TEMPORARY WORK SPACE
C            - IF NEWKDV=.FALSE. THE DYNAMIC STIFFNESS HAS ALREADY
C              BEEN REDUCED TO UPPER TRIANGULAR FORM
C
      IF (NEWKDV) THEN
          CALL MSOLL(1,BUG,NFREE,IMDS,NFREE,1,1,NFREE,MDS,S,PBAR,WORK,
     &            .FALSE.,DUMMY,IDUMMY,1,
     &            .FALSE.,DUMMY,IDUMMY,1,ABORT)
      ELSE
          CALL MSOLL(2,BUG,NFREE,IMDS,NFREE,1,1,NFREE,MDS,S,PBAR,WORK,
     &            .FALSE.,DUMMY,IDUMMY,1,
     &            .FALSE.,DUMMY,IDUMMY,1,ABORT)
      ENDIF
      IF (ABORT) RETURN
C
C..... NOTE: - PBAR CONTAINS THE INCRIMENTAL DISPLACMENT BETHEEN
C            TIMES T AND T+DT AFTER THE SECOND CALL TO MSOLL
      CALL MSOLL(3,BUG,NFREE,IMDS,NFREE,1,1,NFREE,MDS,S,PBAR,WORK,
     &            .FALSE.,DUMMY,IDUMMY,1,
     &            .FALSE.,DUMMY,IDUMMY,1,ABORT)
      IF (ABORT) RETURN
C
C
C------- DETERMINE ACCELERATION, VELOCITY AND DISPLACEMENT AT TIME DT
C       CA = INCRIMENTAL ACCELERATION BETHEEN TIMES T AND T+DT
C       OV = INCRIMENTAL VELOCTIV BETHEEN TIMES T AND T+DT
C       DD = INCRIMENTAL DISPLACMENT BETHEEN TIMES T AND T+DT
C
      IF (BUG) THEN
        WRITE (6,*) 'INCRIMENTAL RESULTS FROM LINACC....'
        WRITE (6,320)
      ENDIF
      DO 270 I=LS,L4
        DD(I)=C1*( PBAR(I) + BETA*RC(I) )
        OV(I)=C10*DD(I) - RC(I)
        DA(I)=C11*DD(I) - QC(I)
        IF (BUG) WRITE (6,330) I,DD(I),OV(I),DA(I)
  270   CONTINUE
C
      RETURN
C
C=======================================================
C= CALC TOTAL DISPLACEMENTS, VELOCITIES AND ACCELERATIONS ==
C=======================================================
  300 CONTINUE
  320   FORMAT (' DOF',6X,'DD',12X,'DV',12X,'DA',12X,,'DP',
     &                  12X,' D',12X,' V',12X,' A',12X,' P')
  330   FORMAT (1X,I6,1P,8G14.6)
C
      RETURN
      END
C=======================================================
      SUBROUTINE LMATRX(A,MD,IC,IMD,NAME)
      DIMENSION MD(IC+1), A(IMD)
      LOGICAL BTEST
      CHARACTER*(*) NAME
      CHARACTER*20  NUMB,ROW(6)
      LOGICAL PT,DMP
      IJI(I,J)=MD(J)+J-I
C
      CALL GETINT(NAME,'DUMP=',IDUMP,0,.TRUE.,.FALSE.)
      IF (IDUMP.NE.0)  THEN
          DMP=.TRUE.
      ELSE
          DMP=.FALSE.
      ENDIF
          DO 1 I=1,6
    1       ROW(I)=' '
          IR=IC
          WRITE (6,*) ' '
          WRITE (6,*) 'PRINTING MATRIX: ',NAME
          DO 50 ICG=1,IC,6
            IF (ICG.GT.1) WRITE (6,*) ' '
            K=MIN(ICG+5,IC)
            WRITE (6,10) (J,J=ICG,K)
            DO 20 I=1,IR
              L=0
              PT=.FALSE.
              DO 15 J=ICG,K
                IJ=IJI(I,J)
                IF (IJ.GE.MD(J+1) .OR. I.GT.J) THEN
                  NUMB=' '
                ELSE
                  PT=.TRUE.
                  WRITE (NUMB,16) A(IJ)
                  IF (COMP .AND. A(IJ).NE.0)
     &              WRITE (IDUMP,*) I,J,A(IJ),NAME
                ENDIF
                L=L+1
                ROW(L)=NUMB
   15         CONTINUE
   20       IF (PT) WRITE (6,30)  I,(ROW(L),L=1,6)
   50     CONTINUE
          RETURN
   10 FORMAT(T10,6('|',' COLUMN',I5,'  ------- '),'|')
   16 FORMAT(F20.6)
   30 FORMAT(' ROW',I5,6A)
      END
C=======================================================================
      SUBROUTINE MASSIN(INMASS,FMASS,MCOND,BUG,IBUG,TITLE,MAXZ,MAXDZ,
     &    MAXNOD,NCOS,NNODE,IZ,I4,IZID,Z,NZ,DZ,
     &    IZMOMS,IZMASS,IZLM,IZKE,IZMOAT,
     &    NDOF,NMASS,IZCOS,IZIDOF,IZCNST,IZJFLG,IZJCOS,NAME,
     &    NCOND,NFREE,IZMD)
C
      LOGICAL BUG,MCOND
      INTEGER FMASS
      CHARACTER*(*) NAME,TITLE(2)
      DIMENSION Z(MAXZ),NZ(MAXZ),DZ(MAXDZ),MD(7)
      LOGICAL BTEST
      DOUBLE PRECISION DZ
      DATA MD/1,2,4,7,11,16,22/
C
      IF (FMASS.EQ.1) THEN
C-------- INITIALIZE THE DIAGONAL MASS MATRIX
C       MCOND=.TRUE.
        IZMOMS=IZ
        IZ=IZ + NDOF+1
        CALL SKYLINK(1,NDOF,IBUG,TITLE,NZ(IZMOMS),
     &                 6,NZ(IZLM),Z(IZKE),MD,NAME)
C
C-------- INPUT THE MASS DATA
        IZMOAT=IZ
        IZ=IZ + NMASS*12
        IMD=NZ(IZMOMS+NDOF)
        IZMASS=IZ
        CALL MFORM(1,INMASS,NMASS,Z(IZMASS),NZ(IZMOMS),MD,NDOF,IMD,TITLE,
     &             NNODE,MAXNOD,NCOS,BUG,IBUG,
     &             NZ(IZID),Z(IZCOS),NZ(IZIDOF),Z(IZCNST),NZ(IZJFLG),
     &             NZ(IZJCOS),Z(IZMOAT),Z(IZ),NZ(IZ+21),Z(IZ+27),
     &              Z(IZ+ 65),Z(IZ+ 99))
C
C-------- RELEASE UNUSED MASS DATA STORAGE
        IZ=IZ - (INMASS-NMASS)*12
C
C==========================================
C= MODIFY MASS MATRIX FOR CONDENSATION  ===
C==========================================
C THE CONDENSATION PROCESS MAY EXPAND THE MASS MATRIX'S
C SKYLINE.  THE MAXIMUM SKYLINE OF BOTH THE MASS AND
C THE STIFFNESS MATRIX IS USED.
        IF (MCOND .AND. MCOND.GT.0)THEN
          DO 310 I=0,NDOF
            NMT=NZ(IZMOM+I+1)-NZ(IZMOM+I)
            NSTIFF=I+1-NMT+1
  310       NZ(IZMOMS+I)=MIN( NZ(IZMOMS+I),NSTIFF)
        ENDIF
C
C
C-------- DETERMINE THE MASS MATRIX SKYLINE
        CALL SKYLINK(5,NDOF,IBUG,TITLE,NZ(IZMOMS),
```

```
     &                                  6,NZ(IZLM),Z(IZKE),MD,NAME)
C
C--------- GENERATE THE MASS MATRIX
        IZMASS=IZ
        IZ=IZ + NZ(IZMDMS+NDOF)
        DO 400 I=IZMASS,IZ
  400     Z(I)=0.
        IW=IZ
        IZKE=IW
        IZLM=IW+21
        CALL MFORMK2,INMASS,NMASS,Z(IZMASS),NZ(IZMDMS),MD,NDOF,IMD,TITLE,
     &             NNODE,MAXNOD,NCOS,BUG,IBUG,
     &             NZ(IZID),Z(IZCOS),NZ(IZIDOF),Z(IZCNST),NZ(IZJFLG),
     &             NZ(IZJCOS),Z(IZMDAT),Z(IW),NZ(IW+21),Z(IW+27),
     &              Z(IW+ 65),Z(IW+ 99))
C
         IF (BUG) THEN
           IMD=NZ(IZMDMS+NDOF)
           CALL LMATRX(Z(IZMASS),NZ(IZMDMS),NDOF,IMD,NAME)
         ENDIF
       ENDIF
C
       RETURN
       END
C=======================================================================
       SUBROUTINE MATLIB
     & (IOPT   ,LSTTYP ,PRINT ,IREL,,HEAD,NAME,EESE,EPSE.DUCT,EXCR,
     &  P      ,PL     ,D     ,DL    ,V     ,VL     ,LELEM ,LMAT   ,
     &  MAT    ,MATTYP,SE,IELNO,DAMAGE)
       DIMENSION P(12),PL(12),D(12),DL(12),V(12),VL(12)
       LOGICAL KSAME.ERR,FIRST,PRINT,HEAD,BUG2
       LOGICAL BTEST
       DIMENSION SE(100),DUCT(3),EXCR(5)
       CHARACTER*80 NAME
C
       INCLUDE (ZCOMN)
C
C-----CHECK FOR AN INVALID MATERIAL NUMBER ...
       IF (MAT.LE.0 .OR. MAT.GT.NMAT) THEN
          WRITE (6,189) MAT
  189     FORMAT (/5X,20('**'),' ER','ROR, INVALID MATERIAL PROP.#',I5/
     &    5X,'CORRECT INPUT.........')
          CALL ERRTRA
          RETURN
       ENDIF
C
       IC     = NZ(IZMAT+MAT    )
       IOPT   = IOPT
       IF (IOPT.NE.1) THEN
            MATTYP = NZ(IC)
       ELSE
            NZ(IC) = MATTYP
       ENDIF
       IF (LSTTYP.EQ.MATTYP) THEN
          FIRST=.FALSE.
       ELSE
          FIRST=.TRUE.
          IF (HEAD) WRITE (6,505) TITLE(1),TITLE(2)
  505     FORMAT ('1 STRUCTURE....: ',A,/
     &            ' SOLUTION.....: ',A,/)
       ENDIF
C
       DAMAGE=0
       BUG2=BTEST(IBUG,10)
       IF (MATTYP.EQ.1)  THEN
         CALL MAT01
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG2,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &          Z(IC+1),EESE,EPSE,DUCT,EXCR)
       ELSE IF (MATTYP.EQ.2)  THEN
         CALL MAT02
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG2,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &          Z(IC+1),EESE,EPSE,DUCT,EXCR,DAMAGE)
       ELSE IF (MATTYP.EQ.3)  THEN
         CALL MAT03
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG2,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &          Z(IC+1),EESE,EPSE,DUCT,EXCR,DAMAGE)
       ELSE IF (MATTYP.EQ.4)  THEN
         CALL MAT04
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG2,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &          Z(IC+1),EESE,EPSE,DUCT,EXCR,DAMAGE)
       ELSE IF (MATTYP.EQ.6)  THEN
         CALL MAT06
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG2,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &          Z(IC+1),EESE,EPSE,DUCT,EXCR,DAMAGE)
       ELSE IF (MATTYP.EQ.7)  THEN
         CALL MAT07
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG2,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &          Z(IC+1),EESE,EPSE,DUCT,EXCR,DAMAGE)
       ELSE
          WRITE (6,10) IELNO
          ERR=.TRUE.
   10     FORMAT(' ELEMENT ',I6,' IS NOT AVAILABLE' )
       ENDIF
       LSTTYP=MATTYP
C
  110 FORMAT (OP,' ZMATRX(',I6,')=',I15,1P,G20.10)
       MAT    = MAT
       RETURN
       END
C=======================================================================
       SUBROUTINE MAT01
     & (IOPT   ,ISTOP   ,ERR ,FIRST  ,PRINT ,BUG,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &  SMAT,EESE,EPSE,DUCT,EXCR)
C
       IMPLICIT REAL(A-H,O-Z)
       LOGICAL ERR,FIRST,PRINT,BUG
       LOGICAL BTEST
       DIMENSION SE(100),SMAT(14),DUCT(5),EXCR(6)
       CHARACTER*80 TYPE
C
       IF (IOPT.EQ.0) THEN
           LELEM=14
           RETURN
       ENDIF
C
C-----------------------------------------------------------------------
C VARIABLES:
C-----------------------------------------------------------------------
C---------- GLOBAL VARIABLES
C IOPT   = 1, INITIALIZE MATERIAL
C        = 2, GET STIFFNESS
C SMAT   = INTERNAL STORAGE
C SE     = OUTPUT STIFFNESS
C-----------------------------------------------------------------------
C
       GO TO (100,200,300,400),IOPT
C
  100 CONTINUE
C
C------ INPUT DATA
C      E     = SMAT( 1)
C      G     = SMAT( 2)
C      AX    = SMAT( 3)
C      AY    = SMAT( 4)
C      AZ    = SMAT( 5)
C      IX    = SMAT( 6)
C      IY    = SMAT( 7)
C      IZ    = SMAT( 8)
C      KIIZ  = SMAT( 9)
C      KJJZ  = SMAT(10)
C      KIJZ  = SMAT(11)
C      KJJY  = SMAT(12)
C      KIIY  = SMAT(13)
C      KIJY  = SMAT(14)
C
       READ (5,*) TYPE ,( SMAT(I),I=1, 8)
       LELEM = 12
       LMAT= 14
C
       IF (FIRST .OR. BUG) WRITE (6,191)
       WRITE (6,192) MAT   ,(SMAT(K),K=1, 8)
C
       RETURN
C
C========== GET STIFFNESS TERMS ==========
  200 CONTINUE
            E     = SMAT( 1)
            G     = SMAT( 2)
            AX    = SMAT( 3)
            AY    = SMAT( 4)
            AZ    = SMAT( 5)
          RIX   = SMAT( 6)
          RIY   = SMAT( 7)
          RIZ   = SMAT( 8)
          RKIIZ = SMAT( 9)
          RKJJZ = SMAT(10)
          RKIJZ = SMAT(11)
          RKJJY = SMAT(12)
          RKIIY = SMAT(13)
          RKIJY = SMAT(14)
C
       SE(1)=E*AX
       SE(2)=G*AY
       SE(3)=G*AZ
       SE(4)=G*RIX
       SE(5)=E*RIY
       SE(6)=E*RIZ
       SE( 7)=RKIIZ
       SE( 8)=RKJJZ
       SE( 9)=RKIJZ
       SE(10)=RKIIY
       SE(11)=RKJJY
       SE(12)=RKIJY
       SE(13)=EESE
       SE(14)=EPSE
C
  300 SE(15)=0
       SE(14)= (P+PL)/2 * (D-DL)
C
C------ TRANSFER ENERGIES FOR BOTH IOPT=3 AND 4.
  400 EESE=SE(13)
       EPSE=SE(14)
C
C------ TRANSFER D & E FOR BOTH IOPT=3 AND 4.
       DO 410  I=1,3
       DUCT(I)=0
       EXCR(I)=0
  410 EXCR(I+3)=0
C
       RETURN
       END
C=======================================================================
       SUBROUTINE MAT02
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &  SMAT,EESE,EPSE,DUCT,EXCR,DAMAGE)
C
       IMPLICIT REAL(A-H,O-Z)
       LOGICAL ERR,FIRST,PRINT,BUG
       LOGICAL BTEST
       DIMENSION SE(100),SMAT(25),DUCT(3),EXCR(6)
       CHARACTER*80 TYPE
C
C-------------- LENGTH OF ELEMENT DATA REQUIRED...
       LELEM=33
       IF (IOPT.EQ.0)   RETURN
C
C-----------------------------------------------------------------------
C VARIABLES:
C-----------------------------------------------------------------------
C---------- GLOBAL VARIABLES
C IOPT   = 1, INITIALIZE MATERIAL
C        = 2, GET STIFFNESS
C RINPUT = INPUT DATA
C SMAT   = INTERNAL STORAGE
C SE     = OUTPUT STIFFNESS
C-----------------------------------------------------------------------
C            ---- ANLMOD DATA ----
C       SMAT(1)= SC,          COMPRESSION STIFFNESS
C       SMAT(2)= ST1,         INITIAL TENSILE STIFFNESS
C       SMAT(3)= ST2,         POST YEILD TENSILE STIFFNESS
C       SMAT(4)= PY,          YEILD POINT
C       SMAT(5)= ALPHA,       ANLMOD PARAMETER
C       SMAT(6)= BETA,        ANLMOD PARAMETER
```

```
C       SMAT(7)= DVT,           AXLMOD PARAMETER                        MAT00580
C       SMAT(8)= DVC,           AXLMOD PARAMETER                        MAT00590
C       SMAT(9)= CSE,           CONSTANT STRAIN ENERGY                  MAT00600
C       SMAT(10)=DMAX,          ULTIMATE DISPLACEMENT                   MAT00610
C       SMAT(11)=BDAM,          BETA FOR DAMAGE INDEX                   MAT00620
C                                                                       MAT00630
C------------------ INPUT DATA FOR AXLMOD HYSTERESIS MODEL ------------MAT00640
C     THE FOLLOWING DATA IS INPUT ONCE FOR EACH MODEL                   MAT00650
C                                                                       MAT00660
C  SC     = SOMPRESSION STIFFNESS                                       MAT00670
C  ST1    = INITIAL TENSILE STIFFNESS                                   MAT00680
C  ST2    = POST YEILD TENSILE STIFFNESS                                MAT00690
C  PY     = YEILD LOAD                                                  MAT00700
C  ALPHA  = MODEL COIEFICIENTS                                          MAT00710
C  BETA   = MODEL COIEFICIENTS                                          MAT00720
C                                                                       MAT00730
C-----------------------------------------------------------------------MAT00740
C                                                                       MAT00750
C                                                                       MAT00760
      GO TO (100,200,300,400,500),IOPT                                  MAT00770
C                                                                       MAT00780
  100 CONTINUE                                                          MAT00790
C                                                                       MAT00800
      READ (5,*) TYPE ,( SMAT(I),I=1,6),DUCMAX,BDAM                    MAT00810
      LMAT = 11                                                         MAT00820
      SMAT(7)=  SMAT(4)/SMAT(2)                                         MAT00830
      SMAT(8)= -SMAT(4)/SMAT(1)                                         MAT00840
      SMAT(9)=  SMAT(4)**2/(2*SMAT(2))                                  MAT00850
      SMAT(10)=DUCMAX*SMAT(7)                                           MAT00860
      SMAT(11)=BDAM                                                     MAT00870
C                                                                       MAT00880
      IF (FIRST .OR. BUG) WRITE (6,192)                                 MAT00890
      WRITE (6,191) MAT,(SMAT(K),K=1,6),DUCMAX,BDAM                     MAT00900
C                                                                       MAT00910
 191  FORMAT (1X,I5,8G15.6/)                                            MAT00920
 192  FORMAT ('///'     AXLMOD HYSTERESIS MODEL- FOR UNIT LENGTH MEMBER'/MAT00930
     &          '  ==========================================='//MAT00940
     &'  MAT.',6X,'   SC',10X,'  ST1',10X,'  ST2',10X,'   PY',         MAT00950
     &  10X,'ALPHA',10X,' BETA', 6X,'MAX DUCT.',7X,'BETA-DI')           MAT00960
C                                                                       MAT00970
      GO TO 9999                                                        MAT00980
C======================== GET STIFFNESS TERMS ==========================MAT00990
  200 CONTINUE                                                          MAT01000
C       ---- AXLMOD MODEL DATA STORAGE FOR MEMBER ----                  MAT01010
C     SE( 1)= S                                                         MAT01020
C     SE( 2)= HRN                                                       MAT01030
C     SE( 3)= FMAX                                                      MAT01040
C     SE( 4)= DMAX                                                      MAT01050
C     SE( 5)= FK                                                        MAT01060
C     SE( 6)= DK                                                        MAT01070
C     SE( 7)= FP                                                        MAT01080
C     SE( 8)= DP                                                        MAT01090
C     SE( 9)= KR                                                        MAT01100
C     SE(10)= KS                                                        MAT01110
C     SE(11)= KT                                                        MAT01120
C     SE(12)= NRL                                                       MAT01130
C     SE(13)= A                                                         MAT01140
C     SE(14)= S_EQ_UNLOAD - EQUIVALENT UNLOADING STIFFNESS              MAT01150
C     SE(15)= UPOS(1)                                                   MAT01160
C     SE(16)= UPOS(2)                                                   MAT01170
C     SE(17)= UPOS(3)                                                   MAT01180
C     SE(18)= UNEG(1)                                                   MAT01190
C     SE(19)= UNEG(2)                                                   MAT01200
C     SE(20)= UNEG(3)                                                   MAT01210
C     SE(21)= EXCR(1)                                                   MAT01220
C     SE(22)= EXCR(2)                                                   MAT01230
C     SE(23)= EXCR(3)                                                   MAT01240
C     SE(24)= EXCR(4)                                                   MAT01250
C     SE(25)= EXCR(5)                                                   MAT01260
C     SE(26)= EXCR(6)                                                   MAT01270
C     SE(27)= ESE(1)                                                    MAT01280
C     SE(28)= ESE(2)                                                    MAT01290
C     SE(29)= ESE(3)                                                    MAT01300
C     SE(30)= PSE                                                       MAT01310
C     SE(31)= PSE_OLD                                                   MAT01320
C     SE(32)= POS_DMAX                                                  MAT01330
C     SE(33)= POS_PMAX                                                  MAT01340
C                                                                       MAT01350
      DO 210 I=1,LELEM                                                  MAT01360
  210   SE(I)=0                                                         MAT01370
      CALL HYST02(P ,D ,PL,DL,V ,VL,SE(1),SE(2),                        MAT01380
     & SMAT(1),SMAT(2),SMAT(3),SMAT(4),SMAT(5),SMAT(6),SMAT(7),SMAT(8),MAT01390
     & SMAT(9),                                                         MAT01400
     & SE(5),SE(6),SE(5),SE(6),SE(7),SE(8),SE(9),SE(10),SE(11),SE(12), MAT01410
     & SE(15),BUG,  SE(14),SE(27),SE(30),SE(31),SE(15),SE(18),SE(21),   MAT01420
     & IDR)                                                             MAT01430
      IF (BUG)  WRITE (6,*) 'STIFF=',STIFF,'   A=',A                    MAT01440
      GO TO 9999                                                        MAT01450
C                                                                       MAT01460
C======================== GET STIFFNESS TERMS ==========================MAT01470
  300 CONTINUE                                                          MAT01480
C     SE( 1)= A                                                         MAT01490
C     SE(15)= A                                                         MAT01500
      ICVC=0                                                            MAT01510
      IF (BUG) THEN                                                     MAT01520
        WRITE (6,*) '------------ IN MAT02 ---- IELNO:',IELNO,           MAT01530
     &              '-------------------------------------'             MAT01540
        WRITE (6,*) 'A:',SE(15)                                         MAT01550
        WRITE (6,*) '  P,D=',P,D  ,'  V=',V                             MAT01560
        WRITE (6,*) 'PL,DL=',PL,DL,' VL=',VL                            MAT01570
      ENDIF                                                             MAT01580
      IDR=999                                                           MAT01590
  310 STIFF=SE(1)                                                       MAT01600
      P=PL+(D-DL)*STIFF                                                 MAT01610
      A=SE(15)                                                          MAT01620
      IF (A.EQ.0 .AND. SE(2).EQ.0) A=P                                  MAT01630
      ICVC=ICVC+1                                                       MAT01640
      IF (BUG) THEN                                                     MAT01650
       WRITE (6,*) 'STIFF=',STIFF,'   A=',A,'   IDR=',IDR               MAT01660
       WRITE (6,*) '    P=',P    ,'ICVC=',ICVC                         MAT01670
      ENDIF                                                             MAT01680
C                                                                       MAT01690
C----- IS P WITHIN LIMITS ?                                             MAT01700
      IF (ICVC.GT.5) THEN                                               MAT01710
            WRITE (6,*) 'MORE THAN 5 CYCLES IN MAT02.  IELNO:',IELNO    MAT01720
     ELSE IF ((PL.LE.P .AND. P.LT.A) .OR. (PL.GE.P .AND. P.GT.A) .OR.   MAT01730
     &       (DL.EQ.D)) THEN                                            MAT01740
            CALL STRENG (SE(27),SE(30),SE(31),P,PL,D,DL,SE(1),SE(14) )  MAT01750
            IF (BUG) WRITE (6,*) 'C------ IS P WITH IN LIMITS ? '       MAT01760
            CALL HYST02(P ,D ,PL,DL,V ,VL,SE(1),SE(2),                  MAT01770
     &      SMAT(1),SMAT(2),SMAT(3),SMAT(4),SMAT(5),SMAT(6),            MAT01780
     &      SMAT(7),SMAT(8),SMAT(9),SE(5),SE(4),SE(5),SE(6),SE(7),SE(8),MAT01790
     &      SE(9),SE(10),SE(11),SE(12),SE(13),BUG,                      MAT01800
     &      SE(14),SE(27),SE(30),SE(31),SE(15),SE(18),SE(21),IDR )      MAT01810
            V=VL                                                        MAT01820
C------ IS P GREATER THAN LIMITS ?                                      MAT01830
       ELSE IF (((PL.LE.A .AND. A.LE.P) .OR. (PL.GE.A .AND. A.GE.P)).AND.MAT01840
     &      (STIFF.NE.0.00)) THEN                                       MAT01850
            DI=DL+(A-PL)/STIFF                                          MAT01860
            PI=A                                                        MAT01870
            CALL STRENG (SE(27),SE(30),SE(31),PI,PL,DI,DL,SE(1),SE(14)) MAT01880
            DL=DI                                                       MAT01890
            PL=PI                                                       MAT01900
            IF (ABS(P-PL).GT. 0.005*ABS(P+PL)) THEN                     MAT01910
                PI=PL+(P-PL)*.01                                        MAT01920
                DI=DL+(D-DL)*.01                                        MAT01930
            ELSE                                                        MAT01940
                PI=P                                                    MAT01950
                DI=D                                                    MAT01960
            ENDIF                                                       MAT01970
            IF (BUG) THEN                                               MAT01980
              WRITE (6,*) 'C------ IS P GREATER THAN LIMITS ? '         MAT01990
              WRITE (6,*) 'PL,DL=',PL,DL                                MAT02000
              WRITE (6,*) 'PI,DI=',PI,DI                                MAT02010
            ENDIF                                                       MAT02020
            CALL HYST02(PI,DI,PL,DL,1.,1.,SE(1),SE(2),                  MAT02030
     &      SMAT(1),SMAT(2),SMAT(3),SMAT(4),SMAT(5),SMAT(6),            MAT02040
     &      SMAT(7),SMAT(8),SMAT(9),SE(5),SE(4),SE(5),SE(6),SE(7),SE(8),MAT02050
     &      SE(9),SE(10),SE(11),SE(12),SE(13),BUG,                      MAT02060
     &      SE(14),SE(27),SE(30),SE(31),SE(15),SE(18),SE(21),IDR )      MAT02070
            V=VL                                                        MAT02080
            GO TO 310                                                   MAT02090
      ENDIF                                                             MAT02100
C                                                                       MAT02110
C------- SAVE MAXIMUM PEAK POSITIVE DISPL                               MAT02120
      IF (D.GT.SE(32)) THEN                                             MAT02130
          SE(32)=D                                                      MAT02140
          SE(33)=P                                                      MAT02150
      ENDIF                                                             MAT02160
C                                                                       MAT02170
      IF (BUG)  WRITE (6,*) 'STIFF=',STIFF,'   A=',SE(15)               MAT02180
C                                                                       MAT02190
C------ TRANSFER ENERGIES FOR BOTH IOPT=3 AND 4.                        MAT02200
  400 EESE=SE(27)                                                       MAT02210
      EPSE=SE(30)                                                       MAT02220
C                                                                       MAT02230
C------ TRANSFER DUCTILITIES AND EXCRUSION RATIOS FOR BOTH IOPT=3 AND 4.MAT02240
      DO 410 I=1,5                                                      MAT02250
      DUCT(I)=MAX(SE(14+I),SE(17+I))                                    MAT02260
      EXCR(I)=SE(20+I)                                                  MAT02270
  410 EXCR(I+5)=SE(20+I+5)                                              MAT02280
C                                                                       MAT02290
      RETURN                                                            MAT02300
C                                                                       MAT02310
C==================== DAMAGE INDEX ====================================MAT02320
  500 CONTINUE                                                          MAT02330
      EESE=SE(27)                                                       MAT02340
      EPSE=SE(30)                                                       MAT02350
      BDAM=SMAT(11)                                                     MAT02360
      D =SE(32)                                                         MAT02370
      P =SE(33)                                                         MAT02380
      DAMAGE=ABS(D)/SMAT(10) + BDAM /( SMAT(4)*SMAT(10)) * EPSE         MAT02390
C                                                                       MAT02400
 9999 CONTINUE                                                          MAT02410
C                                                                       MAT02420
      RETURN                                                            MAT02430
      END                                                               MAT02440
C=======================================================================MAT00010
      SUBROUTINE MAT05                                                  MAT00050
     & (IOPT ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG.                         MAT00060
     &  MAT  ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,                         MAT00070
     &  SMAT,EESE,EPSE,DUCT,EXCR,DAMAGE)                               MAT00080
C                                                                       MAT00090
      IMPLICIT REAL(A-H,O-Z)                                            MAT00100
      LOGICAL ERR,FIRST,PRINT,BUG                                       MAT00110
      LOGICAL BTEST                                                     MAT00120
      DIMENSION SE(100),SMAT(50),DUCT(5),EXCR(6)                        MAT00150
      CHARACTER*80 TYPE                                                 MAT00160
C                                                                       MAT00150
      IF (IOPT.NE.1) LELEM=54+3*SMAT(2)+2                               MAT00160
      IF (IOPT.EQ.0)  RETURN                                            MAT00170
C                                                                       MAT00180
C-----------------------------------------------------------------------MAT00190
C VARIABLES:                                                            MAT00200
C-----------------------------------------------------------------------MAT00210
C---------- GLOBAL VARIABLES                                           MAT00220
C IOPT   = 1, INITIALIZE MATERIAL                                       MAT00250
C        = 2, GET STIFFNESS                                             MAT00240
C SMAT   = INTERNAL STORAGE                                             MAT00250
C SE     = OUTPUT STIFFNESS                                             MAT00260
C-----------------------------------------------------------------------MAT00270
C         ---- BEND1 DATA ----                                          MAT00280
C       SMAT(1)= NSEGS,         NUMBER OF BACKBONE SEGMENTS             MAT00290
C       SMAT(2)= NI,            NUMBER OF SMALL AMPLITUDE LOOPS         MAT00300
C       SMAT(5)= CSE,           CONSTANT STRAIN ENERGY...               MAT00310
C       SMAT(4)= DY,            YEILD POINT FOR DUCTILITIES             MAT00320
C       SMAT(5          )= BDAM  BETA FOR DAMAGE INDEX                  MAT00330
C       SMAT(5+I        )= PP,   BACKBONE MOMENT FOR POINT I            MAT00340
C       SMAT(5+I+NSEGS)= OP.   BACKBONE ROTATION FOR POINT I            MAT00350
C                                                                       MAT00360
      GO TO (100,200,300,400,500),IOPT                                  MAT00370
C                                                                       MAT00380
C                                                                       MAT00390
  100 CONTINUE                                                          MAT00400
C------------------ INPUT DATA FOR BEND1  HYSTERESIS MODEL ------------MAT00410
C                                                                       MAT00420
C  NSEG   = NUMBER OF BACKBONE SEGEMENTS                                MAT00430
C  NI     = NUMBER OF SMALL LOOP POINTS                                 MAT00440
```

```
C  EI      = ELASTIC STIFFNESS                                          CMAT00450
C  G       = MOMENT GRADIENT                                            CMAT00460
C  PP      = BACKBONE LOAD                                              CMAT00470
C  DP      = BACKBONE DISPLACEMENT                                      CMAT00480
C                                                                       CMAT00490
C-----------------------------------------------------------------------CMAT00500
      READ (5,*) TYPE ,NSEG,NI,DV,BDAM                                  MAT00510
      READ(5,*) ( SMAT(5+I      ),I=1,NSEG)                             MAT00520
      READ(5,*) ( SMAT(5+I+NSEG),I=1,NSEG)                              MAT00530
C                                                                       MAT00540
      LELEM = 34+3*NI +2                                                MAT00550
      LMAT = 5+2*NSEG                                                   MAT00560
      SMAT(1)=NSEG                                                      MAT00570
      SMAT(2)=NI                                                        MAT00580
      SMAT(4)=DV                                                        MAT00600
      SMAT(5)=BDAM                                                      MAT00610
C------------ CALC CONSTANT STRAIN ENERGY....                           MAT00620
      DL=0                                                              MAT00630
      PL=0                                                              MAT00640
      DO 10 I=1,NSEG                                                    MAT00650
            P=SMAT(5+I)                                                 MAT00660
            D=SMAT(5+I+NSEG)                                            MAT00670
            IF (D.LE.DV) THEN                                           MAT00680
              CSE=CSE + 0.5*(P+PL)*(D-DL)                               MAT00690
              PL=P                                                      MAT00700
              DL=D                                                      MAT00710
            ELSE IF (D.EQ.DL) THEN                                      MAT00720
              GO TO 20                                                  MAT00730
            ELSE                                                        MAT00750
              PV=PL+ (DV-DL)*(P-PL)/(D-DL)                              MAT00750
              CSE=CSE + 0.5*(PV+PL)*(DV-DL)                             MAT00760
              GO TO 20                                                  MAT00770
            ENDIF                                                       MAT00780
   10 CONTINUE                                                          MAT00790
   20 SMAT(3)=CSE                                                       MAT00800
C                                                                       MAT00810
      IF (FIRST .OR. BUG) WRITE (6,55)                                  MAT00820
      WRITE (6,57) MAT,NI,DV,BDAM,(SMAT(5+K),SMAT(5+K+NSEG),K=1,NSEG)   MAT00830
      WRITE (6,*) ' '                                                   MAT00840
C                                                                       MAT00850
   55   FORMAT(/*      BEND1 HYSTERESIS MODEL DATA - UNIT LENGTH MEMBER' MAT00860
     &  /       '      ==========================================',     MAT00870
     &         /*       BACKBONE CURVE POINTS - MATL  NI ',5X,'DV',9X,  MAT00880
     &    ' BETA          ',5X,' MOMENT',8X,'ROTATION')                  MAT00890
   57   FORMAT      ( (28X,2I4,2X,G15.6,5G15.6)/(68X,2G15.6))           MAT00900
C                                                                       MAT00910
      RETURN                                                            MAT00920
C========================== GET STIFFNESS TERMS ========================MAT00930
   200 CONTINUE                                                         MAT00940
C           ---- BEND1 MODEL DATA STORAGE FOR MEMBER ----               MAT00950
C            SE( 1)= K                                                  MAT00960
C            SE( 2)= RULE                                               MAT00970
C            SE( 5)= DIR                                                MAT00980
C            SE( 4)= A                                                  MAT00990
C            SE( 5)= PMQ                                                MAT01000
C            SE( 6)= PMM                                                MAT01010
C            SE( 7)= DMQ                                                MAT01020
C            SE( 8)= DMM                                                MAT01030
C            SE( 9)= BACKB                                              MAT01040
C            SE(10)= ALPHA1                                             MAT01050
C            SE(11)= ALPHA2                                             MAT01060
C            SE(12)= IR                                                 MAT01070
C            SE(13)= SR1                                                MAT01080
C            SE(14)= SR2                                                MAT01090
C            SE(15)= SR3                                                MAT01100
C            SE(16)= SRM                                                MAT01110
C            SE(17)= E_EQ_UNLOAD                                        MAT01120
C            SE(18)= UPOS(1)                                            MAT01130
C            SE(19)= UPOS(2)                                            MAT01140
C            SE(20)= UPOS(3)                                            MAT01150
C            SE(21)= UNEG(1)                                            MAT01160
C            SE(22)= UNEG(2)                                            MAT01170
C            SE(23)= UNEG(3)                                            MAT01180
C            SE(24)= EXCR(1)                                            MAT01190
C            SE(25)= EXCR(2)                                            MAT01200
C            SE(26)= EXCR(3)                                            MAT01210
C            SE(27)= EXCR(4)                                            MAT01220
C            SE(28)= EXCR(5)                                            MAT01230
C            SE(29)= EXCR(6)                                            MAT01240
C            SE(30)= EESE(1)                                            MAT01250
C            SE(31)= EESE(2)                                            MAT01260
C            SE(32)= EESE(3)                                            MAT01270
C            SE(33)= EPSE                                               MAT01280
C            SE(34)= PSE_OLD                                            MAT01290
C            SE(34+1     )=PR( 1)                                       MAT01300
C            SE(34+    NI)=PR(NI)                                       MAT01310
C            SE(34+1+  NI)=DR( 1)                                       MAT01320
C            SE(34+  2*NI)=DR(NI)                                       MAT01330
C            SE(34+1+2*NI)=FR( 1)                                       MAT01340
C            SE(34+  3*NI)=FR(NI)                                       MAT01350
C            SE(35+  3*NI)=P_MAX                                        MAT01360
C            SE(36+  3*NI)=D_MAX                                        MAT01370
C                                                                       MAT01380
      NSEG=SMAT(1)                                                      MAT01390
      NI  =SMAT(2)                                                      MAT01400
      J1=  NSEG+6                                                       MAT01410
      J5=  NI+35                                                        MAT01420
      J6=2*NI+35                                                        MAT01430
      J5=3*NI+35-1 +2                                                   MAT01440
      SI=SMAT(6)/SMAT(6+NSEG)                                           MAT01450
      DO 210 I=1,J5                                                     MAT01460
  210  SE(I)=0                                                          MAT01470
      CALL HYSTOS(P,D,PL,DL,V ,NSEG,NI,                                 MAT01480
     & SMAT( 6),SMAT(J1),SMAT( 6),SMAT(J1),SI,SMAT(5),                  MAT01490
     & SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),SE( 7),SE( 8),SE( 9), MAT01500
     & SE(10),SE(11),SE(12),SE(13),SE(14),SE(15),SE(16),SE(35),SE(J5), MAT01510
     & SE(J6),BUG,                                                      MAT01520
     & SE(17),SMAT(4),SE(30) ,SE(33),SE(34),SE(18),SE(21),SE(24))      MAT01530
      RETURN                                                            MAT01540
C                                                                       MAT01550
C========================== GET STIFFNESS TERMS ========================MAT01560
   300 CONTINUE                                                         MAT01570
C           SE( 4)= A                                                   MAT01580
      NSEG=SMAT(1)                                                      MAT01590
      NI  =SMAT(2)                                                      MAT01600
      J1=  NSEG+6                                                       MAT01610
      J5=  NI+35                                                        MAT01620
      J6=2*NI+35                                                        MAT01630
      SI=SMAT(6)/SMAT(6+NSEG)                                           MAT01640
C     SET VELOCITY FLAG TO 1 TO REMOVE INFL OF HIGH FREQ VELOC          MAT01650
C     DVEL=V*VL                                                         MAT01660
      DVEL = 1                                                          MAT01670
      ICVC=0                                                            MAT01680
      IF (BUG) THEN                                                     MAT01690
        WRITE (6,*) '------------ IN MATOS ------------------------'     MAT01700
        WRITE (6,*) 'P,D',P,D                                           MAT01710
        WRITE (6,*) 'PL,DL=',PL,DL                                      MAT01720
      ENDIF                                                             MAT01730
  310 STIFF=SE(1)                                                       MAT01740
      P=PL+(D-DL)*STIFF                                                 MAT01750
      A=SE(4)                                                           MAT01760
      IF (A.EQ.0 .AND. SE(2).EQ.0) THEN                                 MAT01770
           A=SIGN(SMAT(6),P)                                            MAT01780
           IF (P.LT.PL .AND. PL.GT.0 .AND. P.GT.0) A=0                  MAT01790
           IF (P.GT.PL .AND. PL.LT.0 .AND. P.LT.0) A=0                  MAT01800
      ENDIF                                                             MAT01810
      ICVC=ICVC+1                                                       MAT01820
      IF (BUG) THEN                                                     MAT01830
       WRITE (6,*) 'STIFF=',STIFF,'   A=',A                             MAT01840
       WRITE (6,*) '    P=',P     ,'ICVC=',ICVC                        MAT01850
      ENDIF                                                             MAT01860
C                                                                       MAT01870
C----- IS P WITHIN LIMITS ?                                             MAT01880
      IF (ICVC.GT.10) THEN                                              MAT01890
           WRITE (6,*) 'MORE THAN 10 CYCLES IN MATOS.  IELNO:',IELNO    MAT01900
      ELSE IF ((PL.LE.P .AND. P.LT.A) .OR. (PL.GE.P .AND. P.GT.A) .OR.  MAT01910
     &     (DL.EQ.D)) THEN                                              MAT01920
           IF (BUG) WRITE (6,*) 'C----- IS P WITH IN LIMITS ? '         MAT01930
           CALL STRENG (SE(30),SE(33),SE(34),P,PL,D,DL,SE(1),SE(17) )   MAT01940
           CALL HYSTOS(P,D,PL,DL,DVEL ,NSEG,NI,                         MAT01950
     &     SMAT( 6),SMAT(J1),SMAT( 6),SMAT(J1),SI,SMAT(5),              MAT01960
     &     SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),                   MAT01970
     &     SE( 7),SE( 8),SE( 9),                                        MAT01980
     &     SE(10),SE(11),SE(12),SE(13),SE(14),SE(15),                   MAT01990
     &     SE(16),SE(35),SE(J5),                                        MAT02000
     &     SE(J6),BUG,                                                  MAT02010
     &     SE(17),SMAT(4),SE(30) ,SE(33),SE(34),SE(18),SE(21),SE(24))   MAT02020
           DVEL=1                                                       MAT02030
C----- IS P GREATER THAN LIMITS ?                                       MAT02040
      ELSE IF (((PL.LE.A .AND. A.LE.P) .OR. (PL.GE.A .AND. A.GE.P)).AND.MAT02050
     &     (STIFF.NE.0.00)) THEN                                        MAT02060
           DI=DL+(A-PL)/STIFF                                           MAT02070
           PI=A                                                         MAT02080
           CALL STRENG (SE(30),SE(33),SE(34),PI,PL,DI,DL,SE(1),SE(17) ) MAT02090
           DL=DI                                                        MAT02100
           PL=PI                                                        MAT02110
           IF (ABS(P-PL).GT. 0.0005*ABS(P+PL)) THEN                     MAT02120
                PI=PL+(P-PL)*.001                                       MAT02130
                DI=DL+(D-DL)*.001                                       MAT02140
           ELSE                                                         MAT02150
                PI=P                                                    MAT02200
                DI=D                                                    MAT02210
           ENDIF                                                        MAT02220
           IF (BUG) THEN                                                MAT02230
             WRITE (6,*) 'C----- IS P GREATER THAN LIMITS ? '           MAT02240
             WRITE (6,*) 'PL,DL=',PL,DL                                 MAT02250
             WRITE (6,*) 'PI,DI=',PI,DI                                 MAT02260
           ENDIF                                                        MAT02270
           CALL HYSTOS(PI,DI,PL,DL,1.00 ,NSEG,NI,                       MAT02280
     &     SMAT( 6),SMAT(J1),SMAT( 6),SMAT(J1),SI,SMAT(5),              MAT02290
     &     SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),                   MAT02300
     &     SE( 7),SE( 8),SE( 9),                                        MAT02310
     &     SE(10),SE(11),SE(12),SE(13),SE(14),SE(15),                   MAT02320
     &     SE(16),SE(35),SE(J5),                                        MAT02330
     &     SE(J6),BUG,                                                  MAT02340
     &     SE(17),SMAT(4),SE(30) ,SE(33),SE(34),SE(18),SE(21),SE(24))   MAT02350
           GO TO 310                                                    MAT02360
C----- IS P LESS THAN LIMITS ?                                          MAT02370
      ELSE IF ((P.LT.PL .AND. PL.LE.A) .OR. (P.GT.PL .AND. PL.GE.A))THENMAT02380
           IF (ABS(P-PL).GT. 0.0005*ABS(P+PL)) THEN                     MAT02410
                PI=PL+(P-PL)*.001                                       MAT02420
                DI=DL+(D-DL)*.001                                       MAT02430
           ELSE                                                         MAT02440
                PI=P                                                    MAT02450
                DI=D                                                    MAT02460
           ENDIF                                                        MAT02470
           IF (BUG) THEN                                                MAT02480
             WRITE (6,*) 'C----- IS P LESS THAN LIMITS ? '              MAT02490
             WRITE (6,*) 'PI,DI=',PI,DI                                 MAT02500
           ENDIF                                                        MAT02510
           CALL HYSTOS(PI,DI,PL,DL,1.00 ,NSEG,NI,                       MAT02520
     &     SMAT( 6),SMAT(J1),SMAT( 6),SMAT(J1),SI,SMAT(5),              MAT02530
     &     SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),                   MAT02540
     &     SE( 7),SE( 8),SE( 9),                                        MAT02550
     &     SE(10),SE(11),SE(12),SE(13),SE(14),SE(15),                   MAT02560
     &     SE(16),SE(35),SE(J5),                                        MAT02570
     &     SE(J6),BUG,                                                  MAT02580
     &     SE(17),SMAT(4),SE(30) ,SE(33),SE(34),SE(18),SE(21),SE(24))   MAT02590
           GO TO 310                                                    MAT02600
      ENDIF                                                             MAT02610
C                                                                       MAT02620
C-------- SAVE MAXIMUM DISPL AND IT'S LOAD                              MAT02630
      IF (ABS(D).GT.ABS(SE(36+3*NI))) THEN                              MAT02640
           SE(35+  3*NI)=P                                              MAT02650
           SE(36+  3*NI)=D                                              MAT02660
      ENDIF                                                             MAT02670
C                                                                       MAT02680
      IF (BUG) WRITE (6,*) 'STIFF=',SE(1),'   A=',SE(4)                 MAT02690
C                                                                       MAT02700
C------- TRANSFER ENERGIES FOR BOTH IOPT=5 AND 4.                       MAT02710
      DO 400 I=1,30                                                     MAT02720
  400 EESE=SE(30)                                                       MAT02730
      EPSE=SE(33)                                                       MAT02740
C                                                                       MAT02750
C------- TRANSFER DUCTILITIES AND EXCURSION RATIOS FOR BOTH IOPT=5 AND 4.MAT02760
      DO 410 I=1,3                                                      MAT02770
      DUCT(I)=MAX(SE(17+I),SE(20+I))                                    MAT02780
      EXCR(I+3)=SE(25+I+3)                                              MAT02790
  410 EXCR(I)=SE(25+I)                                                  MAT02800
C                                                                       MAT02810
      RETURN                                                            MAT02820
C                                                                       MAT02830
C========================== DAMAGE INDEX ===============================MAT02840
  500 EESE =SE(30)                                                      MAT02850
      EPSE =SE(33)                                                      MAT02860
      NSEGS=SMAT(1)                                                     MAT02870
      NI  =SMAT(2)                                                      MAT02880
      DMAX =SMAT(5+2*NSEGS)                                             MAT02890
      DV   =SMAT(4)                                                     MAT02900
      BDAM =SMAT(5)                                                     MAT02910
      P    =SE(35+  3*NI)                                               MAT02920
      D    =SE(36+  3*NI)                                               MAT02930
C                                                                       MAT02940
      DI   =SMAT(6+NSEGS)                                               MAT02950
      IF (DV.LE.DI) THEN                                                MAT02960
           PV=SMAT(6)*DV/DI                                             MAT02970
      ELSE                                                              MAT02980
           DO 510 I=2,NSEGS                                             MAT02990
             PI=SMAT(4+I)                                               MAT03000
```

```
            P2=SMAT(5+I)                                                MAT05000
            D1=SMAT(4+I+NSEGS)                                          MAT05010
            D2=SMAT(5+I+NSEGS)                                          MAT05020
            IF (DV.GE.D1 .AND. DV.LE.D2)  THEN                          MAT05030
              PV=P1+(DV-D1)*(P2-P1)/(D2-D1)                             MAT05040
              GO TO 520                                                 MAT05050
            ENDIF                                                       MAT05060
  510       CONTINUE                                                    MAT05070
      ENDIF                                                             MAT05080
  520 DAMAGE=ABS(D)/DMAX + BDAM / (PV*DMAX) * EPSE                      MAT05090
C                                                                       MAT05100
      RETURN                                                            MAT05110
      END                                                               MAT05120
C=======================================================================MAT00040
      SUBROUTINE MAT04                                                  MAT00050
     & (IOPT   ,IELNO   ,ERR  ,FIRST  ,PRINT ,BUG,                     MAT00060
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,                       MAT00070
     &  SMAT,EESE,EPSE,DUCT,EXCR,DAMAGE)                               MAT00080
C                                                                       MAT00090
      IMPLICIT REAL(A-H,O-Z)                                            MAT00100
      LOGICAL ERR,FIRST,PRINT,BUG                                       MAT00110
      LOGICAL BTEST                                                     MAT00120
      DIMENSION SE(100),SMAT(20),DUCT(5),EXCR(6)                        MAT00130
      CHARACTER*80 TYPE                                                 MAT00140
C                                                                       MAT00150
      IF (IOPT.NE.1) LELEM=52+3*SMAT(2)  +2                             MAT00160
      IF (IOPT.EQ.0) RETURN                                             MAT00170
C                                                                       MAT00180
C-----------------------------------------------------------------------MAT00190
C VARIABLES:                                                            MAT00200
C-----------------------------------------------------------------------MAT00210
C---------- GLOBAL VARIABLES                                           MAT00220
C IOPT   = 1, INITIALIZE MATERIAL                                       MAT00230
C        = 2, GET STIFFNESS                                             MAT00240
C RINPUT = INPUT DATA                                                   MAT00250
C SMAT   = INTERNAL STORAGE                                             MAT00260
C SE     = OUTPUT STIFFNESS                                             MAT00270
C-----------------------------------------------------------------------MAT00280
C          ---- SHEAR1 DATA ----                                        MAT00290
C            SMAT(1)= NSEGS,          NUMBER OF BACKBONE SEGMENTS       MAT00300
C            SMAT(2)= NI,             NUMBER OF SMALL AMPLITUDE LOOPS   MAT00310
C            SMAT(3)= CSE,            CONSTAINT STRAIN ENERGY...        MAT00320
C            SMAT(4)= DV,             VEILD DISPLACEMENT                MAT00330
C            SMAT(4+I)= PP,           BACKBONE SHEAR FOR POINT I        MAT00340
C            SMAT(4+I+NSEG)= DP,      BACKBONE DISPLACEMENT FOR POINT I MAT00350
C            SMAT(5+2*NSEG)= BDAM     BETA FOR DAMAGE INDEX             MAT00360
C                                                                       MAT00370
      GO TO (100,200,300,400,500),IOPT                                  MAT00380
C                                                                       MAT00390
  100 CONTINUE                                                          MAT00400
C                                                                       MAT00410
C------------------ INPUT DATA FOR SHEAR1 HYSTERESIS MODEL ------------CMAT00420
C                                                                       CMAT00430
C  NSEG    = NUMBER OF BACKBONE SEGEMENTS                               CMAT00440
C  NI      = NUMBER OF SMALL LOOP POINTS                                CMAT00450
C  PP      = BACKBONE LOAD                                              CMAT00460
C  DP      = BACKBONE DISPLACEMENT                                      CMAT00470
C-----------------------------------------------------------------------CMAT00480
      READ (5,*) TYPE ,NSEG,NI,DV,BDAM                                  MAT00490
      READ (5,*) ( SMAT(4+I    ),I=1,NSEG)                              MAT00500
      READ (5,*) ( SMAT(4+I+NSEG),I=1,NSEG)                             MAT00510
      LELEM = 52+3*NI +2                                                MAT00520
      LMAT = 5+2*NSEG                                                   MAT00530
      SMAT(1)=NSEG                                                      MAT00540
      SMAT(2)=NI                                                        MAT00550
      SMAT(4)=DV                                                        MAT00560
      SMAT(5+2*NSEG)=BDAM                                               MAT00570
C-------------- CALC CONSTANT STRAIN ENERGY....                        MAT00580
      DL=0                                                              MAT00590
      PL=0                                                              MAT00600
      DO 10 I=1,NSEG                                                    MAT00610
            P=SMAT(4+I)                                                 MAT00620
            D=SMAT(4+I+NSEG)                                            MAT00630
            IF (D.LE.DV) THEN                                           MAT00640
              CSE=CSE + 0.5*(P+PL)*(D-DL)                               MAT00650
              PL=P                                                      MAT00660
              DL=D                                                      MAT00670
            ELSE IF (D.EQ.DL) THEN                                      MAT00680
              GO TO 20                                                  MAT00690
            ELSE                                                        MAT00700
              PV=PL+ (DV-DL)*(P-PL)/(D-DL)                              MAT00710
              CSE=CSE + 0.5*(PV+PL)*(DV-DL)                             MAT00720
              GO TO 20                                                  MAT00730
            ENDIF                                                       MAT00740
   10 CONTINUE                                                          MAT00750
   20 SMAT(3)=CSE                                                       MAT00760
C                                                                       MAT00770
C                                                                       MAT00780
      IF (FIRST .OR. BUG) WRITE (6,65)                                  MAT00790
      WRITE (6,67) MAT,NI,DV,BDAM,(SMAT(4+K),SMAT(4+K+NSEG),K=1,NSEG)   MAT00800
      WRITE (6,*)  '                                                    MAT00810
   65 FORMAT('///'     SHEAR1 HYSTERESIS MODEL DATA - UNIT LENGTH MEMBER'MAT00820
     &  /       '     ==========================================',      MAT00830
     &         /'     BACKBONE CURVE POINTS - MAT.  NI         DV     ', MAT00840
     &  5X,'BETA',6X,    3X,' STRESS',8X,' STRAIN')                     MAT00850
   67 FORMAT     ( (28X,2I4,2X,4G15.6)/(68X,2G15.6))                    MAT00860
C                                                                       MAT00870
      RETURN                                                            MAT00880
C======================== GET STIFFNESS TERMS ==========================MAT00890
  200 CONTINUE                                                          MAT00900
C          ---- SHEAR1 MODEL DATA STORAGE FOR MEMBER ----               MAT00910
C       SE( 1)= K                                                       MAT00920
C       SE( 2)= RULE                                                    MAT00930
C       SE( 3)= DIR                                                     MAT00940
C       SE( 4)= A                                                       MAT00950
C       SE( 5)= PMG                                                     MAT00960
C       SE( 6)= PMH                                                     MAT00970
C       SE( 7)= DMG                                                     MAT00980
C       SE( 8)= DMH                                                     MAT00990
C       SE( 9)= BACKB                                                   MAT01000
C       SE(10)= SR1                                                     MAT01010
C       SE(11)= SR2                                                     MAT01020
C       SE(12)= SR3                                                     MAT01030
C       SE(13)= SRM                                                     MAT01040
C       SE(14)= IR                                                      MAT01050
C       SE(15)= S_EQ_UNLOAD                                             MAT01060
C       SE(16)= UPOS(1)                                                 MAT01070
C       SE(17)= UPOS(2)                                                 MAT01080
C       SE(18)= UPOS(3)                                                 MAT01090
C       SE(19)= UNEG(1)                                                 MAT01100
C       SE(20)= UNEG(2)                                                 MAT01110
C       SE(21)= UNEG(3)                                                 MAT01120
C       SE(22)= EXCR(1)                                                 MAT01130
C       SE(23)= EXCR(2)                                                 MAT01140
C       SE(24)= EXCR(3)                                                 MAT01150
C       SE(25)= EXCR(4)                                                 MAT01160
C       SE(26)= EXCR(5)                                                 MAT01170
C       SE(27)= EXCR(6)                                                 MAT01180
C       SE(28)= ESE                                                     MAT01190
C       SE(29)= ESE                                                     MAT01200
C       SE(30)= ESE                                                     MAT01210
C       SE(31)= PSE                                                     MAT01220
C       SE(32)= PSE_OLD                                                 MAT01230
C       SE(52+1     )=PR( 1)                                            MAT01240
C       SE(52+   NI)=PR(NI)                                             MAT01250
C       SE(52+1+ NI)=DR( 1)                                             MAT01260
C       SE(52+ 2*NI)=DR(NI)                                             MAT01270
C       SE(52+1+2*NI)=FR( 1)                                            MAT01280
C       SE(52+ 3*NI)=FR(NI)                                             MAT01290
C       SE(53+ 3*NI)=P_MAX                                              MAT01300
C       SE(54+ 3*NI)=D_MAX                                              MAT01310
C                                                                       MAT01320
      NSEG=SMAT(1)                                                      MAT01330
      NI  =SMAT(2)                                                      MAT01340
      SI  = SMAT(5)/SMAT(5+NSEG)                                        MAT01350
      J1=NSEG+5                                                         MAT01360
      J5=NI+55                                                          MAT01370
      J4=2*NI+33                                                        MAT01380
      J5=5*NI+55-1 +2                                                   MAT01390
      DO 210 I=1,J5                                                     MAT01400
  210 SE(I)=0                                                           MAT01410
          CALL MYSTO4( P, D,PL,DL, V  ,NSEG,NI,BUG,                     MAT01420
     &          SMAT( 5),SMAT(J1),SMAT( 5),SMAT(J1),SI,SMAT(5),SMAT(4), MAT01430
     &          SE( 1)  ,SE( 2)  ,SE( 5)  ,SE( 4)  ,SE( 5) ,SE( 6) ,   MAT01440
     &          SE( 7)  ,SE( 8)  ,SE( 9)  ,SE(10)  ,SE(11) ,SE(12) ,   MAT01450
     &          SE(13)  ,SE(14)  ,SE(15)  ,SE(16)  ,SE(19) ,SE(22) ,   MAT01460
     &          SE(28)  ,SE(31)  ,SE(32)  ,SE(33)  ,SE(J5) ,SE(J4) )   MAT01470
      RETURN                                                            MAT01480
C                                                                       MAT01490
C======================== GET STIFFNESS TERMS ==========================MAT01500
  300 CONTINUE                                                          MAT01510
      NSEG=SMAT(1)                                                      MAT01520
      NI  =SMAT(2)                                                      MAT01530
      SI  = SMAT(5)/SMAT(5+NSEG)                                        MAT01540
      J1=NSEG+5                                                         MAT01550
      J5=NI+55                                                          MAT01560
      J4=2*NI+33                                                        MAT01570
      DVEL=V*VL                                                         MAT01580
C     SET VELOCITY FLAG TO 1 TO REMOVE INFL OF HIGH FREQ VELOC          MAT01590
      DVEL = 1                                                          MAT01600
      ICYC=0                                                            MAT01610
      IF (BUG) THEN                                                     MAT01620
        WRITE (6,*) '------------ IN MAT04 -------------------------'    MAT01630
        WRITE (6,*) 'P,D',P,D                                           MAT01640
        WRITE (6,*) 'PL,DL=',PL,DL                                      MAT01650
      ENDIF                                                             MAT01660
  310 STIFF=SE(1)                                                       MAT01670
      P=PL+(D-DL)*STIFF                                                 MAT01680
      A=SE(4)                                                           MAT01690
      IF (A.EQ.0 .AND. SE(2).EQ.0) THEN                                 MAT01700
          A=SIGN(SMAT(5),P)                                             MAT01710
          IF (P.LT.PL .AND. PL.GT.0 .AND. P.GT.0) A=0                   MAT01720
          IF (P.GT.PL .AND. PL.LT.0 .AND. P.LT.0) A=0                   MAT01730
      ENDIF                                                             MAT01740
      ICYC=ICYC+1                                                       MAT01750
      IF (BUG) THEN                                                     MAT01760
       WRITE (6,*) 'STIFF=',STIFF,'   A=',A                             MAT01770
       WRITE (6,*) '    P=',P     ,'ICYC=',ICYC                        MAT01780
      ENDIF                                                             MAT01790
C                                                                       MAT01800
C----- IS P WITHIN LIMITS ?                                             MAT01810
      IF (ICYC.GT.10) THEN                                              MAT01820
            WRITE (6,*) 'MORE THAN 10 CYCLES IN MAT04,  IELNO=',IELNO   MAT01830
      ELSE IF ((PL.LE.P .AND. P.LT.A) .OR. (PL.GE.P .AND. P.GT.A) .OR.  MAT01840
     &     (DL.EQ.D)) THEN                                              MAT01850
          IF (BUG) WRITE (6,*) 'C------ IS P WITH IN LIMITS ? '         MAT01860
          CALL STRENG (SE(28),SE(51),SE(52),P,PL,D,DL,SE(1),SE(15) )    MAT01870
          CALL MYSTO4( P, D,PL,DL,DVEL,NSEG,NI,BUG,                     MAT01880
     &          SMAT( 5),SMAT(J1),SMAT( 5),SMAT(J1),SI,SMAT(5),SMAT(4), MAT01890
     &          SE( 1)  ,SE( 2)  ,SE( 5)  ,SE( 4)  ,SE( 5) ,SE( 6) ,   MAT01900
     &          SE( 7)  ,SE( 8)  ,SE( 9)  ,SE(10)  ,SE(11) ,SE(12) ,   MAT01910
     &          SE(13)  ,SE(14)  ,SE(15)  ,SE(16)  ,SE(19) ,SE(22) ,   MAT01920
     &          SE(28)  ,SE(51)  ,SE(52)  ,SE(53)  ,SE(J5) ,SE(J4) )   MAT01930
C----- IS P GREATER THAN LIMITS ?                                       MAT01940
      ELSE IF (((PL.LE.A .AND. A.LE.P) .OR. (PL.GE.A .AND. A.GE.P)).AND.MAT01950
     &     (STIFF.NE.0.00)) THEN                                        MAT01960
          DL=DL+(A-PL)/STIFF                                            MAT01970
          P1=A                                                          MAT01980
          CALL STRENG (SE(28),SE(51),SE(52),P1,PL,D1,DL,SE(1),SE(15) )  MAT01990
          DL=D1                                                         MAT02000
          PL=P1                                                         MAT02010
          IF (ABS(P-PL).GT. 0.0005*ABS(P+PL)) THEN                      MAT02020
              PI=PL+(P-PL)*.001                                         MAT02030
              DI=DL+(D-DL)*.001                                         MAT02040
          ELSE                                                          MAT02050
              PI=P                                                      MAT02060
              DI=D                                                      MAT02070
          ENDIF                                                         MAT02080
          IF (BUG) THEN                                                 MAT02090
            WRITE (6,*) 'C----- IS P GREATER THAN LIMITS ? '            MAT02100
            WRITE (6,*) 'PL,DL=',PL,DL                                  MAT02110
            WRITE (6,*) 'PI,DI=',PI,DI                                  MAT02120
          ENDIF                                                         MAT02130
          CALL MYSTO4(PI,DI,PL,DL,1.00,NSEG,NI,BUG,                     MAT02140
     &          SMAT( 5),SMAT(J1),SMAT( 5),SMAT(J1),SI,SMAT(5),SMAT(4), MAT02150
     &          SE( 1)  ,SE( 2)  ,SE( 5)  ,SE( 4)  ,SE( 5) ,SE( 6) ,   MAT02160
     &          SE( 7)  ,SE( 8)  ,SE( 9)  ,SE(10)  ,SE(11) ,SE(12) ,   MAT02170
     &          SE(13)  ,SE(14)  ,SE(15)  ,SE(16)  ,SE(19) ,SE(22) ,   MAT02180
     &          SE(28)  ,SE(51)  ,SE(52)  ,SE(53)  ,SE(J5) ,SE(J4) )   MAT02190
          GO TO 310                                                     MAT02200
C----- IS P LESS THAN LIMITS ?                                          MAT02210
      ELSE IF ((P.LT.PL .AND. PL.LE.A) .OR. (P.GT.PL .AND. PL.GE.A))THENMAT02220
            IF (ABS(P-PL).GT. 0.0005*ABS(P+PL)) THEN                    MAT02230
                PI=PL+(P-PL)*.001                                       MAT02240
                DI=DL+(D-DL)*.001                                       MAT02250
            ELSE                                                        MAT02260
                PI=P                                                    MAT02270
                DI=D                                                    MAT02280
            ENDIF                                                       MAT02290
            IF (BUG) THEN                                               MAT02300
              WRITE (6,*) 'C------ IS P LESS THAN LIMITS ? '            MAT02310
              WRITE (6,*) 'PI,DI=',PI,DI                                MAT02320
            ENDIF                                                       MAT02330
            CALL MYSTO4(PI,DI,PL,DL,1.00,NSEG,NI,BUG,                   MAT02340
     &          SMAT( 5),SMAT(J1),SMAT( 5),SMAT(J1),SI,SMAT(5),SMAT(4), MAT02350
     &          SE( 1)  ,SE( 2)  ,SE( 5)  ,SE( 4)  ,SE( 5) ,SE( 6) ,   MAT02360
     &          SE( 7)  ,SE( 8)  ,SE( 9)  ,SE(10)  ,SE(11) ,SE(12) ,   MAT02370
     &          SE(13)  ,SE(14)  ,SE(15)  ,SE(16)  ,SE(19) ,SE(22) ,   MAT02380
     &          SE(28)  ,SE(51)  ,SE(52)  ,SE(53)  ,SE(J5) ,SE(J4) )   MAT02390
          GO TO 310                                                     MAT02400
      ENDIF                                                             MAT02410
      IF (BUG) THEN                                                     MAT02420
        WRITE (6,*) 'P,D,PL,DL,A,STIFF',P,D,PL,DL,A,STIFF               MAT02430
        WRITE (6,*) 'ICYC=',ICYC                                        MAT02440
      ENDIF                                                             MAT02450
```

```
     &          SE(28) ,SE(31) ,SE(32) ,SE(33) ,SE(J5) ,SE(J4) )
          GO TO 510
      ENDIF
C
      IF (BUG) WRITE (6,*) 'STIFF=',STIFF,'   A=',SE(4)
C
C------ SAVE MAXIMUN DISPL AND LOAD
      IF (ABS(D).GT.ABS(SE(34+5*NI))) THEN
            SE(33+ 5*NI)=P
            SE(34+ 5*NI)=D
      ENDIF
C
C------ TRANSFER ENERGIES FOR BOTH IOPT=3 AND 4.
  400 EESE=SE(28)
      EPSE=SE(31)
C
C------ TRANSFER DUCTILITIES AND EXCRUSION RATIOS FOR BOTH IOPT=5 AND 4.
      DO 410 I=1,5
      DUCT(I)=MAX(SE(15+I),SE(18+I))
      EXCR(I)=SE(21+I)
  410 EXCR(I+5)=SE(21+I+3)
      RETURN
C
C=========================== DAMAGE INDEX ============================
  500 EESE =SE(28)
      EPSE =SE(31)
      NSEGS=SMAT(1)
      NI   =SMAT(2)
      DMAX =SMAT(4+2*NSEGS)
      DV   =SMAT(4)
      BDAM =SMAT(5+2*NSEG)
      P=SE(33+  3*NI)
      D=SE(34+  5*NI)
      IF (DV.LE.SMAT(5+NSEGS)) THEN
            PY=SMAT(5)*DV/SMAT(5+NSEGS)
      ELSE
            DO 510 I=2,NSEGS
              P1=SMAT(5+I)
              P2=SMAT(4+I)
              D1=SMAT(5+I+NSEGS)
              D2=SMAT(4+I+NSEGS)
              IF (DV.GE.D1 .AND. DV.LE.D2)  THEN
                PY=P1+(DV-D1)*(P2-P1)/(D2-D1)
                GO TO 520
              ENDIF
  510       CONTINUE
      ENDIF
  520 DAMAGE=ABS(D)/DMAX + BDAM / (PY*DMAX) * EPSE
C
      RETURN
      END
C=======================================================================
      SUBROUTINE MAT06
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG.
     &  MAT    ,SE    ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &  SMAT,EESE,EPSE,DUCT,EXCR,DAMAGE)
C
      IMPLICIT REAL(A-H,O-Z)
      LOGICAL ERR,FIRST,PRINT,BUG
      LOGICAL BTEST
      DIMENSION SE(100),SMAT(15),DUCT(5),EXCR(8)
C
      LELEM=7
      IF (IOPT.EQ.0) RETURN
C
C-----------------------------------------------------------------------
C VARIABLES:
C-----------------------------------------------------------------------
C---------- GLOBAL VARIABLES
C IOPT   = 1. INITIALIZE MATERIAL
C        = 2. GET STIFFNESS
C NINPUT = INPUT DATA
C SMAT   = INTERNAL STORAGE
C SE     = OUTPUT STIFFNESS
C-----------------------------------------------------------------------
C          ---- TAKEDA DATA ----
C       SMAT( 1)= EI,           ELASTIC STIFFNESS
C       SMAT( 2)= PC,           CRACKING LOAD
C       SMAT( 3)= DC,           CRACKING DISPLACEMENT
C       SMAT( 4)= PV,           YEILD LOAD
C       SMAT( 5)= DV,           YEILD DISPLACEMENT
C       SMAT( 6)= PU,           ULTIMATE LOAD
C       SMAT( 7)= DU,           ULTIMATE DISPLACEMENT
C       SMAT( 8)= OC,           INITIAL STIFFNESS
C       SMAT( 9)= CV,           CRACKED STIFFNESS
C       SMAT(10)= VU,           ULTIMATE STIFFNESS
C       SMAT(11)= C*V,          RELOADING STIFFNESS
C       SMAT(12)= CSE           CONSTANT STRAIN ENERGY AT YEILD
C       SMAT(13)= BDAM          BETA FOR DAMAGE INDEX
C
C----------------- INPUT DATA FOR TAKEDA HYSTERESIS MODEL ----------CMAT00450
C     THE FOLLOWING DATA IS INPUT ONCE FOR EACH MODEL               CMAT00460
C                                                                   CMAT00470
C  EI      = MOMENT OF INERTIA                                      CMAT00480
C  G       = MOMENT GRADIENT                                        CMAT00490
C  PC,DC   = CRACKING LOAD AND DISPLACEMENT                         CMAT00500
C  PY,DY   = YEILD LOAD AND DISPLACEMENT                            CMAT00510
C  PU,DU   = ULTIMATE LOAD AND DISPLACEMENT                         CMAT00520
C                                                                   CMAT00530
C-----------------------------------------------------------------------
      GO TO (100,200,300,400,500),IOPT
C
  100 CONTINUE
      READ (5,*) TYPE ,                 (SMAT(I),I=2,7),BDAM
C
      LMAT = 15
      SMAT( 8)= SMAT(2)          / SMAT(3)
      SMAT( 9)=(SMAT(4)-SMAT(2))/(SMAT(5)-SMAT(3))
      SMAT(10)=(SMAT(6)-SMAT(4))/(SMAT(7)-SMAT(5))
      SMAT(11)=(SMAT(2)+SMAT(4))/(SMAT(3)+SMAT(5))
      SMAT( 1)=1
      SMAT(12)=SMAT(2)*SMAT(3)/2 +.5*(SMAT(2)+SMAT(4))*(SMAT(5)-SMAT(3))
      SMAT(15)=BDAM
C
      IF (FIRST .OR. BUG) WRITE (6,5)
      WRITE (6,6) MAT,BDAM,             SMAT(2),SMAT(4),SMAT(6),
     &                                  SMAT(3),SMAT(5),SMAT(7)
    5 FORMAT(///'     TAKEDA HYSTERESIS MODEL- FOR UNIT LENGTH MEMBER'/
     &       '  ================================================='//
     &    ' MAT.    BETA                ',
     &    7X,'CRACK',10X,'YEILD', 7X,'ULTIMATE')
    6 FORMAT (/1X,I5,G15.6,1X, 15X ,2X,3G15.6,/39X,3G15.6/)
C
      RETURN
C========================== GET STIFFNESS TERMS ========================
  200 CONTINUE
C          ---- TAKEDA MODEL DATA STORAGE FOR MEMBER ----
C       SE( 1)= S
C       SE( 2)= HRN
C       SE( 3)= A
C       SE( 4)= B
C       SE( 5)= IDRO
C       SE( 6)= RF
C       SE( 7)= RNRL
C       SE( 8)= UM1
C       SE( 9)= UM2
C       SE(10)= UM3
C       SE(11)= UM4
C       SE(12)= S1
C       SE(13)= XO
C       SE(14)= XOUM
C       SE(15)= XOV
C       SE(16)= UOD
C       SE(17)= UO
C       SE(18)= K1UM
C       SE(19)= U1
C       SE(20)= U1D
C       SE(21)= X2UO
C       SE(22)= U2
C       SE(23)= K5U1
C       SE(24)= U3
C       SE(25)= QY
C       SE(26)= MAXFLG
C       SE(27)= ACHNG
C       SE(28)= S_EQ_UNLOAD
C       SE(29)= EESE
C       SE(30)= EESE
C       SE(31)= EESE
C       SE(32)= EPSE
C       SE(33)= PSE_OLD
C       SE(34)= UPOS(1)
C       SE(35)= UPOS(2)
C       SE(36)= UPOS(3)
C       SE(37)= UNEG(1)
C       SE(38)= UNEG(2)
C       SE(39)= UNEG(3)
C       SE(40)= EXCR(1)
C       SE(41)= EXCR(2)
C       SE(42)= EXCR(3)
C       SE(43)= EXCR(4)
C       SE(44)= EXCR(5)
C       SE(45)= EXCR(6)
C       SE(46)= P_MAX
C       SE(47)= D_MAX
C
      EI=SMAT(1)
      GRAD=SMAT(12)
      DO 210 I=1,47
  210   SE(I)=0
      CALL HYST06(P,D,PL,DL,V ,VL,
     & SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),
     & SMAT( 2),SMAT( 5),SMAT( 9),SMAT( 5),SMAT( 6),SMAT( 7),SMAT( 8),
     & SMAT( 9),SMAT(10),SMAT(11),SMAT(12),
     & SE( 7),SE( 8),SE( 9),SE(10),SE(11),SE(12),SE(13),SE(14),SE(15),
     & SE(16),SE(17),SE(18),SE(19),SE(20),SE(21),SE(22),SE(23),SE(24),
     & SE(25),SE(26),SE(27),BUG,
     & SE(28),SE(29),SE(52),SE(53),SE(54),SE(57),SE(40))
C
      RETURN
C========================== GET STIFFNESS TERMS ========================
  500 CONTINUE
      ICVC=0
      IF (BUG) THEN
        WRITE (6,*) '------------ IN MAT06 ------------------------'
        WRITE (6,*) '  P,D=',P,D   ,'  V=',V
        WRITE (6,*) 'PL,DL=',PL,DL,' VL=',VL
      ENDIF
  510 STIFF=SE(1)
      P=PL+(D-DL)*STIFF
      A=SE(27)
      IF (A.EQ.0 .AND. SE(7).EQ.0) A=SMAT(2)
      ICVC=ICVC+1
      IF (BUG) THEN
       WRITE (6,*) 'STIFF=',STIFF,'   A=',A
       WRITE (6,*) '     P=',P     ,'ICVC=',ICVC
      ENDIF
C
C----- IS P WITHIN LIMITS ?
      IF (ICVC.GT.5) THEN
            WRITE (6,*) 'MORE THAN 5 CYCLES IN MAT06,  IELNO:',IELNO
      ELSE IF ((PL.LE.P .AND. P.LE.A) .OR. (PL.GE.P .AND. P.GE.A) .OR.
     &     (DL.EQ.D)) THEN
            IF (BUG) WRITE (6,*) 'C----- IS P WITH IN LIMITS ? '
            CALL STRENG(SE(29),SE(52),SE(53),P,PL,D,DL,SE(1),SE(28))
            CALL HYST06( P,D ,PL,DL,V ,VL,
     &       SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),
     &       SMAT( 2),SMAT( 5),SMAT( 9),SMAT( 5),SMAT( 6),
     &       SMAT( 7),SMAT( 8),SMAT( 9),SMAT(10),SMAT(11),SMAT(12),
     &       SE( 7),SE( 8),SE( 9),SE(10),SE(11),SE(12),SE(13),
     &       SE(14),SE(15),SE(16),SE(17),SE(18),SE(19),SE(20),
     &       SE(21),SE(22),SE(23),SE(24),SE(25),SE(26),SE(27),BUG,
     &       SE(28),SE(29),SE(52),SE(53),SE(54),SE(57),SE(40))
C----- IS P GREATER THAN LIMITS ?
      ELSE IF (((PL.LE.A .AND. A.LT.P) .OR. (PL.GE.A .AND. A.GE.P)).AND.
     &     (STIFF.NE.0.00)) THEN
            DL=DL+(A-PL)/STIFF
            PL=A
            CALL STRENG(SE(29),SE(32),SE(33),P1,PL,D1,DL,SE(1),SE(28))
            IF (BUG) THEN
              WRITE (6,*) 'C----- IS P GREATER THAN LIMITS ? '
              WRITE (6,*) 'PL,DL=',PL,DL
              WRITE (6,*) 'P1,D1=',P1,D1
            ENDIF
            CALL HYST06(P1,D1,PL,DL,1.,1.,
     &       SE( 1),SE( 2),SE( 3),SE( 4),SE( 5),SE( 6),
     &       SMAT( 2),SMAT( 5),SMAT( 9),SMAT( 5),SMAT( 6),
     &       SMAT( 7),SMAT( 8),SMAT( 9),SMAT(10),SMAT(11),SMAT(12),
     &       SE( 7),SE( 8),SE( 9),SE(10),SE(11),SE(12),SE(13),
     &       SE(14),SE(15),SE(16),SE(17),SE(18),SE(19),SE(20),
     &       SE(21),SE(22),SE(23),SE(24),SE(25),SE(26),SE(27),BUG,
     &       SE(28),SE(29),SE(52),SE(53),SE(54),SE(57),SE(40))
            DL=D1
            PL=P1
C
            GO TO 510
C----- IS P LESS THAN LIMITS ?
```

```
      ELSE IF ((P.LT.PL .AND. PL.LE.A) .OR. (P.GT.PL .AND. PL.GE.A))THEN
           IF (ABS(P-PL).GT. 0.0005*ABS(P+PL)) THEN
               PI=PL+(P-PL)*.001
               DI=DL+(D-DL)*.001
           ELSE
               PI=P
               DI=D
           ENDIF
           IF (BUG) THEN
             WRITE (6,*) 'C----- IS P LESS THAN LIMITS ? '
             WRITE (6,*) 'PI,DI=',PI,DI
           ENDIF
           CALL HVST06(PI,DI,PL,DL,1.,1.,
     &     SE( 1),SE( 2),SE( 5),SE( 4),SE( 5),SE( 6),
     &     SMAT( 2),SMAT( 3),SMAT( 4),SMAT( 5),SMAT( 6),
     &     SMAT( 7),SMAT( 8),SMAT( 9),SMAT(10),SMAT(11),SMAT(12),
     &     SE( 7),SE( 8),SE( 9),SE(10),SE(11),SE(12),SE(13),
     &     SE(14),SE(15),SE(16),SE(17),SE(18),SE(19),SE(20),
     &     SE(21),SE(22),SE(25),SE(24),SE(25),SE(26),SE(27),BUG,
     &     SE(28),SE(29),SE(32),SE(33),SE(34),SE(37),SE(40))
           GO TO 510
      ENDIF
C
      IF (BUG)  WRITE (6,*) 'STIFF=',STIFF,' A,B=',SE(3),SE(4),SE(27)
C
C------ MAXIMUM DISPLACEMENT
      IF (ABS(D).GT.ABS(SE(47))) THEN
          SE(46)= P
          SE(47)= D
      ENDIF
C
C------ TRANSFER ENERGIES FOR BOTH IOPT=5 AND 4.
  400 EESE=SE(29)
      EPSE=SE(32)
C
C------ TRANSFER DUCTILITIES AND EXCRUSION RATIOS FOR BOTH IOPT=5 AND 4.
      DO 410 I=1,3
      DUCT(I)=MAX(SE(33+I),SE(36+I))
      EXCR(I)=SE(39+I)
  410 EXCR(I+3)=SE(42+I)
      RETURN
C
C========== DAMAGE INDEX ==========
  500 EESE=SE(29)
      EPSE=SE(32)
      PV=  SMAT( 4)
      PU=  SMAT( 6)
      OU=  SMAT( 7)
      P =  SE(46)
      D =  SE(47)
      BDAM=SMAT(13)
      DAMAGE=ABS(D)/DU + BDAM/(PV*OU) * EPSE
C
      RETURN
      END
C=======================================================================
      SUBROUTINE MAT07
     & (IOPT   ,IELNO   ,ERR ,FIRST  ,PRINT ,BUG,
     &  MAT    ,SE   ,P,PL,D,DL,V,VL,LELEM,LMAT,
     &  SMAT,EESE,EPSE,DUCT,EXCR,DAMAGE)
C
      IMPLICIT REAL(A-H,O-Z)
      LOGICAL ERR,FIRST,PRINT,BUG
      LOGICAL BTEST
      DIMENSION SE(100),SMAT(7),DUCT(3),EXCR(6)
      CHARACTER*80 TYPE
C
C------- RESERVE STORAGE FOR ELEMENT
      LELEM=24
      IF (IOPT.EQ.0) RETURN
C
C-----------------------------------------------------------------------
C VARIABLES:
C-----------------------------------------------------------------------
C---------- GLOBAL VARIABLES
C IOPT   = 1, INITIALIZE MATERIAL
C          2, GET STIFFNESS
C RINPUT = INPUT DATA
C SMAT   = INTERNAL STORAGE
C SE     = OUTPUT STIFFNESS
C-----------------------------------------------------------------------
      GO TO (100,200,300,400,500),IOPT
C
  100 CONTINUE
C
C----- INTERPERTATE INPUT DATA
C     KE    = SMAT( 1)
C     PV    = SMAT( 2)
C     KIE   = SMAT( 3)
C     OV    = SMAT( 4)
C     CSE   = SMAT( 5)
C     DMAX  = SMAT( 6)
C     BDAM  = SMAT( 7)
C
      READ (5,*) TYPE ,(SMAT(I),I=1,3),DUCMAX,BDAM
C
      LMAT = 7
      SMAT(4)=SMAT(2)/SMAT(1)
      SMAT(5)=SMAT(4)*SMAT(2)/2
      SMAT(6)=SMAT(4)*DUCMAX
      SMAT(7)=BDAM
C
      IF (FIRST .OR. BUG) WRITE (6,191)
      WRITE (6,192) MAT,(SMAT(K),K=1,3),DUCMAX,BDAM
C
  191 FORMAT (///'     BILINEAR HYSTERESIS MODEL PARAMETERS'/
     &        '     ====================================='//
     &  ' MAT.',6X,'  KE',10X,'  PV',10X,'  KIE', 7X,'MAX DUCT',
     &        10X,' BETA')
  192 FORMAT (1X,I5,7G15.6/)
C
      RETURN
C
C========== INITILIZE STIFFNESS ==========
  200 CONTINUE
C         ---- ELASTO-PLASTIC MODEL DATA STORAGE FOR MEMBER ----
C     SE( 1)= K
C     SE( 2)= RULE
C     SE( 3)= EPSE_OLD
C     SE( 4)= A, UPPER LIMIT BEFORE RULE CHANGE
C     SE( 5)= B, LOWER LIMIT BEFORE RULE CHANGE
C     SE( 6)=EESE(1)
C     SE( 7)=EESE(2)
C     SE( 8)=EESE(3)
C     SE( 9)=EPSE
C     SE(10)=KEY
C     SE(11)=UP0SC(1)
C     SE(12)=UP0SC(2)
C     SE(13)=UP0SC(3)
C     SE(14)=UNEG(1)
C     SE(15)=UNEG(2)
C     SE(16)=UNEG(3)
C     SE(17)=EXCR(1)
C     SE(18)=EXCR(2)
C     SE(19)=EXCR(3)
C     SE(20)=EXCR(4)
C     SE(21)=EXCR(5)
C     SE(22)=EXCR(6)
C     SE(23)=P_MAX
C     SE(24)=D_MAX
C
      INITILIZE RULE #1 - ELASTIC BEHAVIOR
C
      DO 210 I=1,22
  210   SE(I)=0
C
      CALL HVST07(0.,0.,0.,0.,SE(1),SE(2),SE(3),SE(4),SE(5),V,VL,
     &    SMAT(1),SMAT(2),SMAT(3),SMAT(4),SMAT(5),BUG,
     &    SE(6),SE(9),SE(10),SE(11),SE(14),SE(17))
C
      RETURN
C
C============ GET STIFFNESS TERMS AND ACTUAL FORCE ============
  300 CONTINUE
C
C--------- CHECK TO SEE IF DD = 0 ?
      P=P
      D=0
      IF (D.EQ.DL)  RETURN
C
C--------- CALL HVST07 TO CALC NEW STIFFNESS AND GET LOAD P
      CALL HVST07(P,D,PL,DL,SE(1),SE(2),SE(3),SE(4),SE(5),V,VL,
     &    SMAT(1),SMAT(2),SMAT(3),SMAT(4),SMAT(5),BUG,
     &    SE(6),SE(9),SE(10),SE(11),SE(14),SE(17))
C
      SEL=SE(1)
C
C------- TRANSFER ENERGIES FOR BOTH IOPT=5 AND 4.
      IF (ABS(D).GT.ABS(SE(24)))THEN
          SE(23)=P
          SE(24)=D
      ENDIF
C
C------- TRANSFER ENERGIES FOR BOTH IOPT=5 AND 4.
  400 EESE=SE(6)
      EPSE=SE(9)
C
C------- TRANSFER DUCTILITIES AND EXCRUSION RATIOS FOR BOTH IOPT=5 AND 4.
      DO 410 I=1,3
      DUCT(I)=MAX(SE(10+I),SE(13+I))
      EXCR(I+3)=SE(16+I+3)
  410 EXCR(I)=SE(16+I)
      RETURN
C
C========== DAMAGE INDEX ==========
  500 EESE=SE(6)
      EPSE=SE(9)
      PV    = SMAT( 2)
      OV    = SMAT( 4)
      DMAX  = SMAT( 6)
      BDAM  = SMAT( 7)
      P=   SE(23)
      D=   SE(24)
      DAMAGE=ABS(D)/DMAX + BDAM / (PV*DMAX) * EPSE
C
      RETURN
      END
C=======================================================================
      SUBROUTINE MFORM(IOPT,INMASS,NMASS,MASS,MDM,ND,NDOF,IMD,TITLE,
     &             NNODE,MAXNOD,NCOS,BUG,IBUG,
     &             ID,COSINE,IDOF,CONST,JTFLG,
     &             JTCOS,DATA,SMASS,LM,A,S,SAT)
      DIMENSION ID(MAXNOD),     JTFLG(MAXNOD),   JTCOS(MAXNOD),
     &          IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CONST(MAXNOD,6)
      DIMENSION A(6,6),S(6,6),SAT(6,6),MDM(NDOF+1),MD(7)
      REAL      MASS(IMD)
      LOGICAL BTEST
      DIMENSION DATA(12,INMASS) ,SMASS(21) ,LM(6)
      CHARACTER*150 ELNO
      CHARACTER*(*) TITLE(2)
      LOGICAL BUG
C
C-------- PRINT HEADER AND INPUT MASS DATA
      IF (IOPT.EQ.1) THEN
       WRITE (6,5) TITLE(1),TITLE(2)
    5  FORMAT ('1 STRUCTURE.....: ',A,/
     &        '  SOLUTION.....: ',A,//
     &        '  LUMPED NODE MASSES '/
     &        '  ==================='//
     &   4X,'NODE',5X,'MX',10X,'MY',10X,'MZ',
     &       9X,'RXX', 9X,'RYY', 9X,'RZZ',
     &       9X,'RXY', 9X,'RXZ', 9X,'RYZ',5X,'  IGEN   INC')
C
      NMASS=1
C  INPUT:  NODE, PX,PY,PZ, RX,RY,RZ, RXY,RXZ,RYZ, IGEN, INC
    6 READ  (5,*) (DATA(I,NMASS),I=1,12)
      NODE = DATA( 1,NMASS)
      IGEN = DATA(11,NMASS)
      INC  = DATA(12,NMASS)
      IF (NODE.GT.0) THEN
          WRITE (6,7) NODE,(DATA(I,NMASS),I=2,10),IGEN,INC
          IF (NMASS.LT.INMASS) THEN
              NMASS=NMASS+1
              GO TO 6
          ENDIF
      ELSE
          NMASS=NMASS-1
      ENDIF
    7 FORMAT (2X,I6,1P,9G12.4,2I6)
C     ELSE IF (IOPT.EQ.2) THEN
C     ENDIF
C
      DO 100 IMASS=1,NMASS
        NODE=DATA( 1,IMASS)
        PX  =DATA( 2,IMASS)
        PY  =DATA( 3,IMASS)
        PZ  =DATA( 4,IMASS)
```

```
      RX  =DATA( 5,IMASS)
      RY  =DATA( 6,IMASS)
      RZ  =DATA( 7,IMASS)
      RXY =DATA( 8,IMASS)
      RXZ =DATA( 9,IMASS)
      RYZ =DATA(10,IMASS)
      IGEN=DATA(11,IMASS)
      INC =DATA(12,IMASS)
C
    9 JOINT=IQUICK(NODE,ID,NNODE)
C
      IF (JOINT.LE.0) THEN
        WRITE (6,*) 'NODE #',NODE,
     &  ' WAS NOT FOUND, CHECK INPUT, MASS IS OMITTED '
        GO TO 90
      ENDIF
C
      ICOS  =JTCOS(JOINT)
C
C-----> SET UP NODE MASS MATRIX IN XYZ COORDINATE SYSTEM
C------ SET UP TRANSFORMATION MATRIX, A = (CONST)
      DO 10 I=1,6
      DO 10 J=1,6
   10   A(I,J)=0
      DO 20 I=1,6
   20   A(I,I)=1
        A(4,2)=CONST(JOINT,1)
        A(4,3)=CONST(JOINT,2)
        A(5,1)=CONST(JOINT,3)
        A(5,3)=CONST(JOINT,4)
        A(6,1)=CONST(JOINT,5)
        A(6,2)=CONST(JOINT,6)
C
C------ SET UP NODE MASS MATRIX
      DO 30 I=1,6
      DO 30 J=I,6
   30   SC(I,J)=0
C
        S(1,1)= PX
        S(2,2)= PY
        S(3,3)= PZ
        S(4,4)= RX
        S(5,5)= RY
        S(6,6)= RZ
        S(4,5)=-RXY
        S(4,6)=-RXZ
        S(5,6)=-RYZ
C
      DO 40 I=1,6
      DO 40 J=I+1,6
   40   SC(J,I)=SC(I,J)
C
C------ TRANSFORM MASS MATRIX TO ACCOUNT FOR CONSTRAINTS
      DO 50 I=1,6
      DO 50 J=1,6
        SAT(I,J)=0
        DO 50 K=1,6
   50     SAT(I,J)=SAT(I,J) + SC(I,K)*A(J,K)
C
      DO 60 I=1,6
      DO 60 J=1,6
        SC(I,J)=0
        DO 60 K=1,6
   60     SC(I,J)=SC(I,J) + A(I,K)*SAT(K,J)
C
C..... TRANSFER MASS MATRIX TO HALF STORAGE MODE
        IJ=0
        DO 70 J=1,6
        DO 70 I=J,1,-1
          IJ=IJ+1
   70     SMASS(IJ)=SC(I,J)
C..... INITIALIZE THE DIAGONAL MASS MATRIX
        DO 80  I=1,6
   80     LM(I)=IDOF(JOINT,I)
C
C
      IF (IOPT.EQ.1) THEN
C
C..... CALL SKYLIN TO RESERVE SPACE FOR THIS NODE'S MASS
          IF (BUG) THEN
            WRITE (ELNO,915)  NODE,(LM(I),I=1,6)
            CALL LMATRX(SMASS,MD,6,21,ELNO)
          ENDIF
        CALL SKYLINC2,NDOF,IBUG,TITLE,MDM,6,
     &         LM,SMASS,MD,'GLOBAL MASS')
C
      ELSE IF (IOPT.EQ.2) THEN
          IF (BUG) THEN
            WRITE (ELNO,915)  NODE,(LM(I),I=1,6)
  915       FORMAT ('MASS AT NODE #',I6,' LM:',6I6)
            CALL LMATRX(SMASS,MD,6,21,ELNO)
          ENDIF
C
          CALL FORML(MASS,MDM,NDOF,IMD,SMASS,MD,6,21,LM,1.0 )
C
      ENDIF
   90 CONTINUE
C
      IF (IGEN.GT.0) THEN
         IGEN=IGEN-1
         NODE=NODE+INC
         GO TO 9
      ENDIF
C
  100 CONTINUE
C
      RETURN
      END
C=======================================================================
C  MATRIX SOLUTION OF EQUATION AX=P, WHERE THE UPPER TRIANGULAR MATRIX
C    OF THE SYMMETRIC MATRIX A IS STORED BY SKYLINES. P IS STORED AS
C    A FULL MATRIX.  THE MATRIX P IS REPLACED BY X AFTER BACK
C    SUBSTITUTION IS COMPLEATE.  BOTH A AND P ARE ALTERED.....
C  GUYAN OPTION: IF GUYAN IS .TRUE. MASS MATRIX IS ALSO CONDENSED
C=======================================================================
      SUBROUTINE MSOLL(IOPT,PRINT,N,IMD,LROW,NLC,L1,L2,MD,A,P,FACTOR,
     & GUYAN ,MASS,MOMASS,IMOMAS,
     & KGCOND,KG  ,MDKG  ,IMDKG ,ABORT)
      DIMENSION A(IMD),FACTOR(N),P(LROW,NLC),MD(N+1)
      DIMENSION MOMASS(N+1),MDKG(N+1)
      LOGICAL BTEST,ABORT
      REAL    MASS(IMOMAS),KG(IMDKG)
      LOGICAL GUYAN,PRINT,KGCOND
C
      IJI(I,J)=MD(J)+J-I
      IJM(I,J)=MOMASS(J)+J-I
      IJK(I,J)=MDKG(J)+J-I
C-----------------------------------------------------------------------
C  MATRIX STORAGE SCHEME:
C -THE MATRIX A IS BANDED AND SYMMETRIC. THUS ONLY THE SKYLINE ABOVE
C  AND THE MAIN DIAGONAL NEEDS TO BE STORED.
C  THE MATRIX A IS STORED IN A LINEAR ARRAY BY COLUMNS.  THE VALUE ON
C  THE MAIN DIAGONAL IS STORED IN THE LINEAR ARRAY FIRST.
C  ADDRESSES OF THE LINEAR ARRAY ELEMENTS CONTAINING THE MAIN DIAGONAL
C  TERMS ARE STORED IN MD.
C  EXAMPLE: LET 8 BE THE 5X5 FULL SYMETRIC MATRIX BELOW...
C     | 1 | 3 |   |   | 12 |
C     |   | 2 | 5 |   | 11 |  NOTE: BC1,3),BC1,4) AND BC2,4) ARE ZERO.
C  B= |   |   | 4 | 7 | 10 |
C     |  SYMM.|   | 6 | 9 |  MD= 1, 2, 4, 6, 8, 15
C     |       |   |   | 8 |
C  THUS, BC5,5)=AC MD(5) )= AC4)
C        BCI,J)=AC MD(J)+J-I ) IF J>I AND (MD(J)+J-I)<(MD(J+1)
C              =0             IF J>I AND (MD(J)+J-I)>MD(J+1)
C -MATRIX P IS NOT SYMMETRIC, AND IS STORED AS A FULL MATRIX
C-----------------------------------------------------------------------
C  VARIABLE TABLE:
C  IOPT  = OPTION NUMBER
C  N     = NUMBER OF DEGREES OF FREEDOM OF A MATRIX
C  NLC   = NUMBER OF LOAD CASES FOR THE P MATRIX
C  L1    = FIRST ROW TO BE WORKED WITH
C  L2    = LAST ROW TO BE WORKED WITH
C  MD    = ADDRESS OF THE MAIN DIAGONAL TERMS OF A
C  P     = LAOD MATRIX ON INPUT, SOLN (X) OF AX=P ON COMPLEATION OF
C          IOPT=4
C  FACTOR= TEMPORARY STORAGE MATRIX
C  GUYAN = GUYAN REDUCTION FLAG FOR MASS MATRIX
C  MASS  = MASS MATRIX
C  MOMASS= MASS MATRIX ADDRESSES OF MAIN DIAGONAL ELEMENTS
C  KGCOND= FLAG FOR CONDENSING OUT KG
C  KG    = GEOMETRIC STIFFNESS MATRIX
C  MDKG  = KG MATRIX ADDRESSES OF MAIN DIAGONAL ELEMENTS
C  L     = ROW NUMBER FOR ELIMINATION
C  I     = ROW NUMBER
C  J     = COLUMN NUMBER
C-----------------------------------------------------------------------
C
      L1=MAX(L1,1)
      L2=MIN(L2,N)
C
      GO TO (10,110,210,310),IOPT
C
   10 CONTINUE
C--------- IOPT=1, REDUCE A AND P ------------------------------------
C         GAUSSIAN ELIMINIATION IS USED TO REDUCE THE A AND P MATRICIES
C         FROM ROW L1 TO ROW L2.  IF L1 IS NOT 1, IT IS ASSUMED THAT
C         A AND P ARE ALLREADY REDUCED FROM 1 TO L1...
C---------------------------------------------------------------------
      IF (PRINT) THEN
        WRITE (6,500) 'IOPT=1, REDUCE STIFFNESS AND LOADS',L1,L2,N
        CALL LMATRX(A,MD,N,IMD ,'INPUT STIFFNESS')
        CALL MMATRX(P,N  ,NLC,'INPUT LOAD')
        IF (GUYAN ) CALL LMATRX(MASS,MOMASS,N,IMOMAS,'INPUT MASS')
        IF (KGCOND) CALL LMATRX(KG  ,MDKG  ,N,IMDKG ,'INPUT KG  ')
      ENDIF
C
      DO 100 L=L1,L2
        IF (L.EQ.N) GO TO 100
C
C------- TEST FOR ZERO PIVOT.................
        IF (A(MD(L)).LE.0) THEN
          WRITE (6,2000) L,MD(L),A(MD(L))
          ABORT=.TRUE.
          RETURN
        ENDIF
C
C------- CALCULATE THE FACTORS EACH ROW IS MULTIPLIED BY
        DO 50 I=(L+1),N
          LI=IJI(L,I)
          IF ( LI.LT.MD(I+1) ) THEN
              FACTOR(I)=A(LI) /A( MD(L) )
          ELSE
              FACTOR(I)=0
          ENDIF
   50     CONTINUE
C
C------- REDUCE A BY COLUMNS (J) EACH ROW (I)...
        DO 40 I=(L+1),N
          LJ=IJI(L,J)
          IF (LJ.GE.MD(J+1)) GO TO 40
          DO 35 I=(L+1),J
            IJ=IJI(I,J)
            A(IJ)=A(IJ)-FACTOR(I)*A(LJ)
   35     CONTINUE
   40   CONTINUE
C
C------- REDUCE P BY COLUMNS (J) EACH ROW (I)...
        DO 50 J=1,NLC
          IF (P(L,J).EQ.0) GO TO 50
          DO 45 I=(L+1),N
   45       P(I,J)=P(I,J)-FACTOR(I)*P(L,J)
   50   CONTINUE
C
C------- REDUCE MASS MATRIX BY GUYAN REDUCTION
      IF (GUYAN) THEN
C
C..... M22=M22 - K21/K11 * M11 * 1/K11 * K12
        RM=MASS(MOMASS(L))
        IF (RM.NE.0) THEN
            DO 60 J=L+1,N
                IF (FACTOR(J).EQ.0) GO TO 60
                II=MAX(L+1,J-(MOMASS(J+1)-MOMASS(J)) )
                DO 55 I=II,J
                    I=IJM(I,J)
   55               MASS(IJ)=MASS(IJ)-FACTOR(I)*RM*FACTOR(J)
   60       CONTINUE
        ENDIF
C
C..... M22=M22 - K21/K11 * M12
        DO 66 J=L+1,N
            LJ=IJM(L,J)
            IF (LJ.GE.MOMASS(J+1)) GO TO 66
            IF (MASS(LJ).EQ.0) GO TO 66
            DO 65 I=L+1,J
                IJ=IJM(I,J)
                IF (IJ.LT.MOMASS(J+1))
     &          MASS(IJ)=MASS(IJ)-FACTOR(I)*MASS(LJ)
   65       CONTINUE
```

```
   66    CONTINUE                                                          MS001540
C                                                                          MS001550
C......... M22=M22 - M21 * 1/K11 * K12                                     MS001560
         DO 70 J=L+1,N                                                     MS001570
              IF (FACTOR(J).EQ.0) GO TO 70                                 MS001580
              DO 69 I=L+1,J                                                MS001590
                   LI=IJK(L,I)                                             MS001600
                   IF (LI.GE.MDMASS(I+1)) GO TO 69                         MS001610
                   IJ=IJK(I,J)                                             MS001620
                   IF (IJ.LT.MDMASS(J+1))                                  MS001630
     &             MASS(IJ)=MASS(IJ)-FACTOR(J)*MASS(LI)                    MS001640
   69              CONTINUE                                                MS001650
   70    CONTINUE                                                          MS001660
      ENDIF                                                                MS001670
C                                                                          MS001680
C------- REDUCE GEOMETRIC STIFFNESS MATRIX                                 MS001690
      IF (KGCOND) THEN                                                     MS001700
C                                                                          MS001710
C......... KG22=KG22 + K21/K11 * KG11 * 1/K11 * K12                       MS001720
         RKG=KG(MDKG(L))                                                   MS001730
         IF (RKG.NE.0) THEN                                                MS001740
              DO 80 J=L+1,N                                                MS001750
                   IF (FACTOR(J).EQ.0) GO TO 80                            MS001760
                   IJ=MAX(L+1,J-(MDKG(J+1)-MDKG(J)) )                      MS001770
                   DO 75 I=IJ,J                                            MS001780
                        IJ=IJK(I,J)                                        MS001790
   75                   KG(IJ)=KG(IJ)+FACTOR(I)*RKG*FACTOR(J)              MS001800
   80         CONTINUE                                                     MS001810
         ENDIF                                                             MS001820
C                                                                          MS001830
C......... KG22=KG22 - K21/K11 * KG12                                      MS001840
         DO 86 J=L+1,N                                                     MS001850
              LJ=IJK(L,J)                                                  MS001860
              IF (LJ.GE.MDKG(J+1)) GO TO 86                                MS001870
              IF (KG(LJ).EQ.0) GO TO 86                                    MS001880
              DO 85 I=L+1,J                                                MS001890
                   IJ=IJK(I,J)                                             MS001900
                   IF (IJ.LT.MDKG(J+1))                                    MS001910
     &             KG(IJ)=KG(IJ)-FACTOR(I)*KG(LJ)                          MS001920
   85              CONTINUE                                                MS001930
   86    CONTINUE                                                          MS001940
C                                                                          MS001950
C......... KG22=KG22 - KG21 * 1/K11 * K12                                  MS001960
         DO 90 J=L+1,N                                                     MS001970
              IF (FACTOR(J).EQ.0) GO TO 90                                 MS001980
              DO 89 I=L+1,J                                                MS001990
                   LI=IJK(L,I)                                             MS002000
                   IF (LI.GE.MDKG(I+1)) GO TO 89                           MS002010
                   IJ=IJK(I,J)                                             MS002020
                   IF (IJ.LT.MDKG(J+1))                                    MS002030
     &             KG(IJ)=KG(IJ)-FACTOR(J)*KG(LI)                          MS002040
   89              CONTINUE                                                MS002050
   90    CONTINUE                                                          MS002060
C                                                                          MS002070
      ENDIF                                                                MS002080
C                                                                          MS002090
C                                                                          MS002100
  100 CONTINUE                                                             MS002110
      IF (PRINT) THEN                                                      MS002120
        CALL LMATRX(A,MD,N,IMD,'REDUCED STIFFNESS')                        MS002130
        CALL MMATRX(P,N ,NLC,'REDUCED LOAD')                               MS002140
        IF (GUYAN)  CALL LMATRX(MASS,MDMASS,N,IDMAS,'REDUCED MASS')        MS002150
        IF (KGCOND) CALL LMATRX(KG  ,MDKG  ,N,IMDKG ,'REDUCED KG')        MS002160
      ENDIF                                                                MS002170
      RETURN                                                               MS002180
C                                                                          MS002190
  110 CONTINUE                                                             MS002200
C--------- IOPT=2, REDUCE P ONLY ------------------------------------------MS002210
C          GAUSSIAN ELIMINIATION IS USED TO REDUCE THE P MATRIX            MS002220
C          FROM ROW L1 TO ROW L2.  IF L1 IS NOT 1, IT IS ASSUMED THAT      MS002230
C          A AND P ARE ALLREADY REDUCED FROM 1 TO L1...                    MS002240
C--------------------------------------------------------------------------MS002250
      IF (PRINT) THEN                                                      MS002260
        WRITE (6,500) 'IOPT=2, REDUCE LOADS ONLY',L1,L2,N                  MS002270
        CALL LMATRX(A,MD,N,MD(N+1),'REDUCED STIFFNESS')                    MS002280
        CALL MMATRX(P,N ,NLC,'NEW LOAD MATRIX ')                           MS002290
      ENDIF                                                                MS002300
C                                                                          MS002510
C------- LOOP FOR EACH ROW (L) TO REDUCE....                               MS002520
      DO 160 L=L1,L2                                                       MS002330
C                                                                          MS002340
      IF (A(MD(L)).LE.0) THEN                                              MS002350
        WRITE (6,2000) L,MD(L),A(MD(L))                                    MS002360
        ABORT=.TRUE.                                                       MS002370
        RETURN                                                             MS002380
      ENDIF                                                                MS002390
C                                                                          MS002400
C     LOOP FOR EACH COLUMN (J)                                             MS002410
      DO 150 J=1,NLC                                                       MS002420
        IF (P(L,J).EQ.0) GO TO 150                                         MS002430
        DO 120 I=(L+1),N                                                   MS002440
          LI=IJK(L,I)                                                      MS002450
          IF (LI.LT.MD(I+1)) P(I,J)=P(I,J)-A(LI)*P(L,J)/A(MD(L))           MS002460
  120     CONTINUE                                                         MS002470
  150 CONTINUE                                                             MS002480
  160 CONTINUE                                                             MS002490
      RETURN                                                               MS002500
C                                                                          MS002520
  210 CONTINUE                                                             MS002530
C--------- IOPT=3, BACK SUBSTUTION ----------------------------------------MS002540
C          BACK SUBSTUTION IS USED TO SOLVE FOR X IN SX=V, WHERE S IS      MS002550
C          THE REDUCE UPPER TRIANGULAR FACTOR OF THE A MATRIX, AND V       MS002560
C          IS THE REDUCED FORRM OF THE P MATRIX.  THE RESULTS (X) ARE      MS002570
C          STORED IN P.                                                    MS002580
C          BACK SUBSTUTION ONLY TAKES PLACE BETWEEN ROWS L1 AND L2.        MS002590
C--------------------------------------------------------------------------MS002600
      IF (PRINT) THEN                                                      MS002610
        WRITE (6,500) 'IOPT=3, BACK SUBSTUTION',L1,L2,N                    MS002620
        CALL LMATRX(A,MD,N,MD(N+1),'REDUCED STIFFNESS')                    MS002630
        CALL MMATRX(P,N ,NLC,'REDUCED LOAD')                               MS002640
      ENDIF                                                                MS002650
C                                                                          MS002660
C------ LOOP FOR EACH ROW L                                               MS002670
C                                                                          MS002680
C     CALCULAGE X FOR EACH LOAD CASE, STORE IN P                           MS002770
      DO 250 J=L2,L1,-1                                                    MS002780
      IF (A(MD(J)).LE.0) THEN                                              MS002790
        WRITE (6,2000) J,MD(J),A(MD(J))                                    MS002800
        CALL ERRTRA                                                        MS002810
        ABORT=.TRUE.                                                       MS002820
        RETURN                                                             MS002830
      ENDIF                                                                MS002840
      DO 215 LC=1,NLC                                                      MS002850
  215   P(J,LC)=P(J,LC)/A(MD(J))                                           MS002860
                                                                           MS002870
        IF (J.GT.1) THEN                                                   MS002880
          IJ=MD(J-1)                                                       MS002890
          NI=L-(MD(J-1)-MD(J)) +1                                          MS002900
          DO 220 I=NI,J-1,+1                                               MS002910
            IJ=IJ-1                                                        MS002920
            DO 220 LC=1,NLC                                                MS002930
              P(I,LC)=P(I,LC)-P(J,LC)*A(IJ)                                MS002940
  220   CONTINUE                                                           MS002990
        ENDIF                                                              MS003010
  250 CONTINUE                                                             MS003020
C                                                                          MS003030
C                                                                          MS003040
      IF (PRINT) CALL MMATRX(P,N ,NLC,'DISPLACEMENT')                      MS003070
C                                                                          MS005150
      RETURN                                                               MS003160
C                                                                          MS005170
  310 CONTINUE                                                             MS003180
C--------- IOPT=4, REDUCED MULTIPLICATION ---------------------------------MS003190
C          SOLVE FOR V IN S*D=P WHERE S HAS BEEN REDUCED BY GAUSSIAN       MS005200
C          ELIMINATION.                                                    MS005210
C          THE RESULTING LOAD IS STORED IN P.                              MS005220
C          THE DISPLACEMENTS ARE INPUT IN D.                               MS005230
C          MULTIPLACATION ONLY TAKES PLACE BETWEEN ROWS L1 AND L2.         MS005240
C--------------------------------------------------------------------------MS005250
      IF (PRINT) THEN                                                      MS005260
        WRITE (6,500) 'IOPT=4, REDUCED MULTIPLACATION',L1,L2,N             MS003270
        CALL LMATRX(A,MD,N,MD(N+1),'REDUCED STIFFNESS')                    MS003280
        CALL MMATRX(FACTOR,N ,1,'DISPLACEMENT')                            MS005290
      ENDIF                                                                MS005300
C                                                                          MS005310
C------- ONLY ONE LOAD CASE....                                            MS003320
      LC=1                                                                 MS005330
C       ---- ZERO P ----                                                   MS003340
        DO 520 J=L1,L2                                                     MS005350
          IF (A(MD(J)).LE.0) THEN                                          MS005360
            WRITE (6,2000) J,MD(J),A(MD(J))                                MS005370
            ABORT=.TRUE.                                                   MS005380
            RETURN                                                         MS005390
          ENDIF                                                            MS005400
  520     P(J,LC)=0                                                        MS005410
C                                                                          MS005420
C---------- MULTIPLY S_RED * DISP = P                                      MS005430
      DO 540 J=L1,N                                                        MS005440
        IC1=J-(MD(J-1)-MD(J)) +1                                           MS005450
        IC2=MAX(IC1,L1)                                                    MS005460
        DO 540 I=J,IC2,-1                                                  MS005470
          IJ=IJK(I,J)                                                      MS005480
  540     P(I,LC)=P(I,LC)+A(IJ)*FACTOR(J)                                  MS005490
      IF (PRINT) CALL MMATRX(P,N ,1,'REDUCED LOAD')                        MS005500
C                                                                          MS005510
C------- UNREDUCE THE LOAD MATRIX ...                                      MS005520
      DO 560 L=L2,L1,-1                                                    MS005530
C                                                                          MS005540
C       LOOP FOR EACH COLUMN (J)                                           MS005550
        DO 550 J=1,NLC                                                     MS005560
          IF (P(L,LC).EQ.0) GO TO 550                                      MS005570
          DO 545 I=(L+1),N                                                 MS005580
            LI=IJK(L,I)                                                    MS005590
            IF (LI.LT.MD(I+1)) P(I,LC)=P(I,LC)+A(LI)*P(L,LC)/A(MD(L))      MS005600
  545       CONTINUE                                                       MS005610
  550   CONTINUE                                                           MS005620
  560   CONTINUE                                                           MS005630
C                                                                          MS005640
  400   CONTINUE                                                           MS005650
C                                                                          MS005660
      IF (PRINT) CALL MMATRX(P,N ,1,'LOAD')                                MS005670
C                                                                          MS005680
      RETURN                                                               MS005690
  500 FORMAT (/2X,A,2X,'L1:',I5,'   L2:',I5,'   N:',I5)                    MS005700
 2000     FORMAT (///1X,50('*')/                                           MS005710
     &            ' *  ER','ROR IN SUBROUTINE MSOLL',          T51,'*'/   MS003720
     &            ' *  PIVOT IS LE 0 ------- ROW:',I5,          T51,'*'/   MS005730
     &            ' *                STORAGE LOC:',I5,          T51,'*'/   MS005740
     &            ' *                        PIVOT:',1P,G15.6,  T51,'*'/   MS005750
     &            ' *  SOLUTION IS ABORTED ',                   T51,'*'/   MS005760
     &           1X,50('*')/)                                              MS005770
      END                                                                  MS005780
C--------------------------------------------------------------------------MS005790
      SUBROUTINE MODDOF(NDOF,NCOND,NFREE,NREST,MAXNOD,NMODE,NCOS,          N0000010
     &    SCALE,NSUPT,NCONST,                                              N0000040
     &    ID,IDOF,INDEX,COORD,COSINE,CONST,                                N0000050
     &    JTFLG,JTCOS,BUG,MSTRJT)                                          N0000060
C                                                                          N0000070
      DIMENSION COORD(MAXNOD,3),ID(MAXNOD),INDEX(MAXNOD),IFIX(6),JFIX(6),  N0000080
     &          IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CONST(MAXNOD,6),          N0000090
     &          JTFLG(MAXNOD),JTCOS(MAXNOD),MSTRJT(MAXNOD,6)               N0000100
      DIMENSION VX(3),VY(3),VZ(3)                                          N0000110
      LOGICAL FLAG,PT,BUG,TEST,BTEST                                       N0000120
      CHARACTER*15 TYPE                                                    N0000130
      CHARACTER*2  CX(6)                                                   N0000140
C--------------------------------------------------------------------------N0000150
C VARIABLE TABLE...                                                        N0000160
C COORD  = JOINT COORDINATES  X,Y,Z                                        N0000170
C ID     = JOINT NUMBER                                                    N0000180
C IDOF   = GLOBAL DEGREES OF FREEDOM ASSOC. WITH JOINT FX,FY,FZ,MX,MY,MZ   N0000190
C JTCOS  = ROW IN COSINE MATRIX ASSOC WITH JOINT                           N0000200
C COSINE = COSINE MATRIX                                                   N0000210
C CONST  = GLOBAL CONSTRAINT MATRIX FOR EACH NODE                          N0000220
C JTFLG  = CONTAINS JOINT FLAG'S--- MUST BE 2+ BIT'S LONG.....             N0000230
C          BIT'S  0 TO  5 RESTRAIN   OOF FX,FY,FZ,MX,MY,MZ                 N0000240
C          BIT'S  6 TO 11 CONDENSE   OOF FX,FY,FZ,MX,MY,MZ                 N0000250
C          BIT'S 12 TO 17 CONSTRAIN  OOF FX,FY,FZ,MX,MY,MZ                 N0000260
C          BIT'S 18 TO 23 ELIMINATE  OOF FX,FY,FZ,MX,MY,MZ                 N0000270
C MSTRJT = INTERNAL MASTER JT # OF CONSTRAINED JOINTS.....                 N0000280
C--------------------------------------------------------------------------N0000290
C                                                                          N0000300
C                                                                          N0000310
C-------- INPUT NODES                                                      N0000320
      IF (BUG) WRITE (6,1000) 'INPUT NODES'                                N0000330
      I=0                                                                  N0000340
   20 I=I+1                                                                N0000350
      READ (5, *) ID(I),(COORD(I,J),J=1,3),JTCOS(I),IGEN                   N0000360
      IF (ID(I).LE.0) THEN                                                 N0000370
        NNODE=I-1                                                          N0000380
        GO TO 26                                                           N0000390
      ENDIF                                                                N0000400
      IF (BUG) WRITE(6,30) I,ID(I),(COORD(I,J),J=1,3),JTCOS(I)             N0000410
      IF (IGEN.GT.0) THEN                                                  N0000420
          READ (5, *) IDINC,COORD1,COORD2,COORD3                           N0000430
          DO 25 K=1,IGEN                                                   N0000440
              I=I+1                                                        N0000450
              ID(I)=ID(I-1)+IDINC                                          N0000460
              IF (ID(I).LE.0) THEN                                         N0000470
                NNODE=I-1                                                  N0000480
```

```
                GO TO 26                                                N0000490
              ENDIF                                                     N0000500
              COORDX(I,1)=COORDX(I-1,1)+COORD1                          N0000510
              COORDX(I,2)=COORDX(I-1,2)+COORD2                          N0000520
              COORDX(I,3)=COORDX(I-1,3)+COORD3                          N0000530
              JTCOSC(I)=JTCOSC(I-1)                                     N0000540
   25         IF (BUG) WRITE(6,30) I,IDX(I),(COORDX(I,J),J=1,3),JTCOSC(I)N0000550
     ENDIF                                                              N0000560
     IF (I.LT.NNODE) GO TO 20                                           N0000570
   26 CONTINUE                                                          N0000580
C                                                                       N0000590
C------- SORT NODES                                                     N0000600
     CALL ISORT(ID,INDEX,NNODE)                                         N0000610
C                                                                       N0000620
C       MOVE NODE INFO. TO TEMPORARY STORGAE                            N0000630
     DO 40 I=1,NNODE                                                    N0000640
      J=INDEX(I)                                                        N0000650
      IDOF(I,1)=IDX(J)                                                  N0000660
      IDOF(I,2)=JTCOSC(J)                                               N0000670
C     IDOF(I,2)=JTCOSC(I)       OLD CODE                                N0000680
      DO 40 K=1,3                                                       N0000690
   40   CONST(I,K)=COORDX(J,K)*SCALE                                     N0000700
C                                                                       N0000710
C       MOVE NODES INFO BACK TO MAIN STORAGE, DELETE DUPLICATE INFO.    N0000720
     NNOLD=NNODE                                                        N0000730
     I=0                                                                N0000740
     J=0                                                                N0000750
   45 I=I+1                                                             N0000760
     J=J+1                                                              N0000770
      IDX(I)=IDOF(J,1)                                                  N0000780
      IF (I.GE.2) THEN                                                  N0000790
        IF (IDX(I).EQ.IDX(I-1) ) THEN                                   N0000800
          I=I-1                                                         N0000810
          NNODE=NNODE-1                                                 N0000820
        ENDIF                                                           N0000830
      ENDIF                                                             N0000840
      JTCOSC(I)=IDOF(J,2)                                               N0000850
      DO 44 K=1,3                                                       N0000860
   44   COORDX(I,K)=CONST(J,K)                                           N0000870
     IF (I.LT.NNODE) GO TO 45                                           N0000880
C                                                                       N0000890
   50 FORMAT(' NODE(',I3,'):',I4,'  X:',F10.4,'  Y:',F10.4,'  Z:',F10.4,N0000900
     &      :,'  COS#:',I3)                                            N0000910
C                                                                       N0000920
C------- GLOBAL JOINT COSINE MATRIX                                     N0000930
     IF (BUG)  WRITE (6,1000) 'DIRECTION COSINES ... '                  N0000940
     DO 55 ICOS=1,NCOS                                                  N0000950
        READ (5,*) VX,VY                                                N0000960
        CALL CROSS(VZ,VX,VY,0)                                          N0000970
        CALL CROSS(VY,VZ,VX,1)                                          N0000980
        DO 50 J=1,3                                                     N0000990
           COSINE(1,J,ICOS)=VX(J)                                       N0001000
           COSINE(2,J,ICOS)=VY(J)                                       N0001010
   50      COSINE(3,J,ICOS)=VZ(J)                                       N0001020
        IF (BUG) WRITE (6,56) ICOS,' VX:',VX,' VY:',VY,' VZ:',VZ        N0001030
   55 CONTINUE                                                          N0001040
   56 FORMAT(' COS(',I3,') ',                                           N0001050
     &     3(A,SS,F9.5,' I',SP,F9.5,' J',F9.5,' K  '),SS)               N0001060
C                                                                       N0001070
C------- INPUT RESTRAINTS                                               N0001080
C       ZERO IDOF MATRIX WHICH WILL HOLD RESTRAINT, CONSTRAINT AND      N0001090
C       ULTIMATLY DOF INFORMATION                                       N0001100
     DO 59 I=1,NNODE                                                    N0001110
   59   JTFLG(I)=0                                                      N0001120
     I=0                                                                N0001130
     FLAG=.TRUE.                                                        N0001140
   60 I=I+1                                                             N0001150
     IF (I.LE.NSUPT) THEN                                               N0001160
        IF (FLAG.AND.BUG) WRITE (6,1000) ' NODE RESTRAINTS '            N0001170
        FLAG=.FALSE.                                                    N0001180
        READ (5, *) NODE,IFM,IGEN,NODINC                                N0001190
   65   IF (NODE.EQ.0) THEN                                             N0001200
           DO 70 J=1,NNODE                                              N0001210
             DO 68 II=1,6                                               N0001220
               IF (IFMC(II).NE.0) JTFLG(J)=IBSET(JTFLG(J),II-1)         N0001230
               IF (IFMC(II).EQ.2) JTFLG(J)=IBSET(JTFLG(J),II+17)        N0001240
   68          CONTINUE                                                 N0001250
             IF (BUG) WRITE (6,80) IDX(J),IFM                           N0001260
   70      CONTINUE                                                     N0001270
        ELSE                                                            N0001280
           J =IQUICK(NODE,ID,NNODE)                                     N0001290
           IF (J.EQ.0) THEN                                             N0001300
             IF (BUG) WRITE (6, 75) NODE                                N0001310
           ELSE                                                         N0001320
             DO 71 II=1,6                                               N0001330
               IF (IFMC(II).NE.0) JTFLG(J)=IBSET(JTFLG(J),II-1)         N0001340
               IF (IFMC(II).EQ.2) JTFLG(J)=IBSET(JTFLG(J),II+17)        N0001350
   71          CONTINUE                                                 N0001360
             IF (BUG) WRITE (6,80) IDX(J),IFM                           N0001370
           ENDIF                                                        N0001380
           IF (IGEN.GT.0) THEN                                          N0001390
             NODE=NODE+NODINC                                           N0001400
             IGEN=IGEN-1                                                N0001410
             GO TO 65                                                   N0001420
           ENDIF                                                        N0001430
        ENDIF                                                           N0001440
        GO TO 60                                                        N0001450
     ENDIF                                                              N0001460
   75 FORMAT (' NODE#',I5,' NOT FOUND, RESTRAINT IGNORED...')           N0001470
   80 FORMAT (' NODE(',I4,') FX:',I2,'  FY:',I2,'  FZ:',I2,             N0001480
     &                    ' MX:',I2,'  MY:',I2,'  MZ:',I2)             N0001490
C                                                                       N0001500
C------- INPUT NODE DOF'S TO BE CONDENSED OUT...                        N0001510
     I=0                                                                N0001520
     FLAG=.TRUE.                                                        N0001530
   90 I=I+1                                                             N0001540
     IF (I.LE.NCOND) THEN                                               N0001550
        IF (FLAG.AND.BUG)                                               N0001560
     &     WRITE (6,1000) ' NODE DOF''S TO BE CONDENSED OUT'            N0001570
        FLAG=.FALSE.                                                    N0001580
        READ (5, *) NODE,IFM, IGEN,NODINC                               N0001590
   95   IF (NODE.EQ.0) THEN                                             N0001600
           DO 100 J=1,NNODE                                             N0001610
             PT=.FALSE.                                                 N0001620
             DO 99 II=1,6                                               N0001630
               IF (IFMC(II).NE.0.AND. .NOT.BTEST(JTFLG(J),II-1))THEN    N0001640
                 JTFLG(J)=IBSET(JTFLG(J),II+5)                          N0001650
                 JFMC(II)=IFMC(II)                                      N0001660
                 PT=.TRUE.                                              N0001670
               ELSE                                                     N0001680
                 JFMC(II)=0                                             N0001690
               ENDIF                                                    N0001700
   99          CONTINUE                                                 N0001710
  100          IF (PT.AND.BUG) WRITE (6,80) IDX(J),JFM                  N0001720
           ELSE                                                         N0001730
             J =IQUICK(NODE,ID,NNODE)                                   N0001740
             IF (J.EQ.0) THEN                                           N0001750
               IF (BUG) WRITE (6,110) NODE                              N0001760
             ELSE                                                       N0001770
               DO 101 II=1,6                                            N0001780
                 IF (IFMC(II).NE.0.AND. .NOT.BTEST(JTFLG(J),II-1))THEN  N0001790
                   JTFLG(J)=IBSET(JTFLG(J),II+5)                        N0001800
                   JFMC(II)=IFMC(II)                                    N0001810
                 ELSE                                                   N0001820
                   JFMC(II)=0                                           N0001830
                 ENDIF                                                  N0001840
  101            CONTINUE                                               N0001850
               IF (PT.AND.BUG) WRITE (6,80) IDX(J),JFM                  N0001860
             ENDIF                                                      N0001870
             IF (IGEN.GT.0) THEN                                        N0001880
               NODE=NODE+NODINC                                         N0001890
               IGEN=IGEN-1                                              N0001900
               GO TO 95                                                 N0001910
             ENDIF                                                      N0001920
           ENDIF                                                        N0001930
           GO TO 90                                                     N0001940
     ENDIF                                                              N0001950
  110    FORMAT (' NODE#',I5,' NOT FOUND, DOF ARE NOT CONDENSED...')     N0001960
C                                                                       N0001970
C                                                                       N0001980
C------- INPUT CONSTRAINTS...                                           N0001990
     I=0                                                                N0002000
     FLAG=.TRUE.                                                        N0002010
  400 I=I+1                                                             N0002020
     IF (I.LE.NCONST) THEN                                              N0002030
        IF (FLAG) WRITE (6,1000) ' NODE CONSTRAINTS '                   N0002040
        FLAG=.FALSE.                                                    N0002050
        READ (5, *) ITYPE,MASTER,NODE,IGEN,NODINC                       N0002060
  410   J =IQUICK(NODE,ID,NNODE)                                        N0002070
        IF (J.EQ.0) THEN                                                N0002080
          WRITE (6,420) NODE                                            N0002090
        ELSE                                                            N0002100
          IF (ITYPE.EQ.0) THEN                                          N0002110
            TYPE='SD-RIGID BODY'                                        N0002120
            DO 415 K=1,6                                                N0002130
              IF (.NOT.BTEST(JTFLG(J),K-1)) THEN                        N0002140
                JTFLG(J)=IBSET(JTFLG(J),K+11)                           N0002150
                MSTRJT(J,K)=IQUICK(MASTER,ID,NNODE)                     N0002160
              ELSE                                                      N0002170
                WRITE(6,450) NODE,K                                     N0002180
              ENDIF                                                     N0002190
  415       CONTINUE                                                    N0002200
          ELSE IF (ITYPE.EQ.1) THEN                                     N0002210
            TYPE='KY-PLANE'                                             N0002220
            K=1                                                         N0002230
              IF (.NOT.BTEST(JTFLG(J),K-1)) THEN                        N0002240
                JTFLG(J)=IBSET(JTFLG(J),K+11)                           N0002250
                MSTRJT(J,K)=IQUICK(MASTER,ID,NNODE)                     N0002260
              ELSE                                                      N0002270
                WRITE(6,450) NODE,K                                     N0002280
              ENDIF                                                     N0002290
            K=2                                                         N0002300
              IF (.NOT.BTEST(JTFLG(J),K-1)) THEN                        N0002310
                JTFLG(J)=IBSET(JTFLG(J),K+11)                           N0002320
                MSTRJT(J,K)=IQUICK(MASTER,ID,NNODE)                     N0002330
              ELSE                                                      N0002340
                WRITE(6,450) NODE,K                                     N0002350
              ENDIF                                                     N0002360
            K=6                                                         N0002370
              IF (.NOT.BTEST(JTFLG(J),K-1)) THEN                        N0002380
                JTFLG(J)=IBSET(JTFLG(J),K+11)                           N0002390
                MSTRJT(J,K)=IQUICK(MASTER,ID,NNODE)                     N0002400
              ELSE                                                      N0002410
                WRITE(6,450) NODE,K                                     N0002420
              ENDIF                                                     N0002430
          ELSE                                                          N0002440
            WRITE (6,440) ITYPE,MASTER,NODE                             N0002450
          ENDIF                                                         N0002460
          WRITE (6,450) TYPE,MASTER,NODE                                N0002470
          ENDIF                                                         N0002480
          IF (IGEN.GT.0) THEN                                           N0002490
            NODE=NODE+NODINC                                            N0002500
            IGEN=IGEN-1                                                 N0002510
            GO TO 410                                                   N0002520
          ENDIF                                                         N0002530
        GO TO 400                                                       N0002540
     ENDIF                                                              N0002550
  420 FORMAT (' NODE:',I6,' IS NOT FOUND, CONSTRAINT IS IGNORED')       N0002560
  430 FORMAT (' NODE:',I6,' IS RESTRAINED, CONSTRAINT',I2,              N0002570
     &  ' IS IGNORED...')                                               N0002580
  440 FORMAT (' INVALID CONSTRAINT TYPE',I5,                            N0002590
     &        ' MASTER:',I6,' SLAVE:',I6)                               N0002600
  450 FORMAT (5X,A,'CONSTRAINT,'                                        N0002610
     &        ' MASTER:',I6,' SLAVE:',I6)                               N0002620
C                                                                       N0002630
     IF (BUG) WRITE (6,1000) 'JOINT RESTRAINT CODES...'                 N0002640
     IF (BUG) CALL PBIT(NNODE,JTFLG,ID)                                 N0002650
C                                                                       N0002660
C=======================================                                N0002670
C=== DETERMINE THE DEGREES OF FREEDOM...  ===                           N0002680
C=======================================                                N0002690
C.....ZERO CONSTRAINT MATRIX WHICH WILL HOLD THE MOMENT                 N0002700
C.....TRANSFORMATION FOR CONSTRAINED DOF                                N0002710
     DO 499 I=1,NNODE                                                   N0002720
       DO 499 J=1,6                                                     N0002730
  499    CONST(I,J)=0                                                   N0002740
     NDOF=0                                                             N0002750
C                                                                       N0002760
C.....DEGREES OF FREEDOM TO BE CONDENSED OUT                            N0002770
     DO 500 I=1,NNODE                                                   N0002780
       DO 500 J=1,6                                                     N0002790
         JT=JTFLG(I)                                                    N0002800
         IF (BTEST(JT,J+5) .AND. .NOT.BTEST(JT,J+11)) THEN              N0002810
           NDOF=NDOF+1                                                  N0002820
           IDOF(I,J)=NDOF                                               N0002830
         ENDIF                                                          N0002840
  500 CONTINUE                                                          N0002850
     NCOND=NDOF                                                         N0002860
C                                                                       N0002870
C.....UNCONSTRAINED AND UNRESTRAINED DEGREES OF FREEDOM                 N0002880
     DO 510 I=1,NNODE                                                   N0002890
       DO 510 J=1,6                                                     N0002900
         NODE=IDOF(I,J)                                                 N0002910
         IF (.NOT.(BTEST(JTFLG(I),J-1) .OR. BTEST(JTFLG(I),J+5) .OR.    N0002920
     &       BTEST(JTFLG(I),J+11)) ) THEN                               N0002930
C          IF (IDOF(I,J).EQ. 0) THEN                                    N0002940
             NDOF=NDOF+1                                                N0002950
             IDOF(I,J)=NDOF                                             N0002960
```

```
        ENDIF                                                           N0002990
  510 CONTINUE                                                          N0003000
      NFREE=NDOF-NCOND                                                  N0003010
C                                                                       N0003020
C.......RESTRAINED DEGREES OF FREEDOM                                   N0003030
      DO 530 I=1,NNODE                                                  N0003040
       DO 530 J=1,6                                                     N0003050
         IF (BTEST(JTFLG(I),J-1) .AND. .NOT.BTEST(JTFLG(I),J+17)        N0003060
     &                           .AND. .NOT.BTEST(JTFLG(I),J+11)) THEN  N0003070
            NDOF=NDOF+1                                                 N0003080
            IDOF(I,J)=NDOF                                              N0003090
         ENDIF                                                          N0003100
  530 CONTINUE                                                          N0003110
C                                                                       N0003120
C.......RESTRAINED DEGREES OF FREEDOM                                   N0003130
      DO 535 I=1,NNODE                                                  N0003140
       DO 535 J=1,6                                                     N0003150
         IF (BTEST(JTFLG(I),J+17) .AND. .NOT.BTEST(JTFLG(I),J+11)) THEN N0003160
            NDOF=NDOF+1                                                 N0003170
            IDOF(I,J)=NDOF                                              N0003180
         ENDIF                                                          N0003190
  535 CONTINUE                                                          N0003200
      NREST=NDOF-NFREE-NCOND                                            N0003210
C                                                                       N0003220
C.......CONSTRAINED DEGREES OF FREEDOM                                  N0003230
C         I = NODE ID OF SLAVE NODE                                     N0003240
C         II= NODE ID OF MASTER NODE                                    N0003250
      DO 520 I=1,NNODE                                                  N0003260
       DO 520 J=1,6                                                     N0003270
C                                                                       N0003280
C         ... CHECK DOF FOR CONSTRAINT ...                              N0003290
        IF (BTEST(JTFLG(I),J+11)) THEN                                  N0003300
C                                                                       N0003310
C             ... DETERMINE MASTER NODE AND DOF # ...                   N0003320
            II=I                                                        N0003330
  515       NODE=IDOF(II)                                               N0003340
            II=MSTRJT(II,J)                                             N0003350
            IF (II.EQ.0) THEN                                           N0003360
               WRITE (6,525) NODE,IDOF(I,J                              N0003370
               IDOF(I,J)=0                                              N0003380
               GO TO 520                                                N0003390
            ENDIF                                                       N0003400
            IF (BTEST(JTFLG(II),J+11)) GO TO 515                        N0003410
            IDOF(I,J)=IDOF(II,J)                                        N0003420
C                                                                       N0003430
C             ...CHECK FOR DIFFERENT JOINT COORDINATE SYSTEMS...        N0003440
            IF (JTCOS(I).NE.JTCOS(II)) THEN                             N0003450
               WRITE (6,411) IDOF(I),IDOF(II)                           N0003460
  411          FORMAT('  CONFLICTING COSINE ID NUMBERS, MASTER:',      N0003470
     &          I6,'  SLAVE:',I6,'  CORRECT INPUT & RERUN...')         N0003480
            ENDIF                                                       N0003490
C                                                                       N0003500
C             ...CALCULATE THE CONSTRAINT MATRIX...                     N0003510
            IF (J.GE.4) THEN                                            N0003520
              XG=COORD(I,1)-COORD(II,1)                                 N0003530
              YG=COORD(I,2)-COORD(II,2)                                 N0003540
              ZG=COORD(I,3)-COORD(II,3)                                 N0003550
              N  =JTCOS(I)                                              N0003560
              XMS=COSINE(1,1,N)*XG+COSINE(1,2,N)*YG+COSINE(1,3,N)*ZG    N0003570
              YMS=COSINE(2,1,N)*XG+COSINE(2,2,N)*YG+COSINE(2,3,N)*ZG    N0003580
              ZMS=COSINE(3,1,N)*XG+COSINE(3,2,N)*YG+COSINE(3,3,N)*ZG    N0003590
            ENDIF                                                       N0003600
            IF (J.EQ.4) THEN                                            N0003610
                CONST(I,1)=-ZMS                                         N0003620
                CONST(I,2)= YMS                                         N0003630
            ELSE IF (J.EQ.5) THEN                                       N0003640
                CONST(I,3)= ZMS                                         N0003650
                CONST(I,4)=-XMS                                         N0003660
            ELSE IF (J.EQ.6) THEN                                       N0003670
                CONST(I,5)=-YMS                                         N0003680
                CONST(I,6)= XMS                                         N0003690
            ENDIF                                                       N0003700
        ENDIF                                                           N0003710
  520 CONTINUE                                                          N0003720
  525 FORMAT (' NODE=',I6,' NOT FOUND WHILE ASSIGNING CONSTRAINED',     N0003730
     & ' DOF TO NODE=',I6,' DOF=',I2,' ---DOF IS DELETED ')             N0003740
C                                                                       N0003750
C       PRINT OUT ORDERED NODE INFO, AND DEGREES OF FREEDOM             N0003760
      WRITE (6,1000) 'NODE COORDINATES AND DEGREES OF FREEDOM'          N0003770
      WRITE (6,600) NDOF,NCOND,NFREE,NREST                              N0003780
  600 FORMAT                                                            N0003790
     & (4X,'TOTAL NUMBER OF DEGREES OF FREEDOM.............',I6,/       N0003800
     &  4X,'NUMBER OF DEGREES OF FREEDOM CONDENSED OUT.....',I6,/       N0003810
     &  4X,'NUMBER OF FREE DEGREES OF FREEDOM..............',I6,/       N0003820
     &  4X,'NUMBER OF RESTRAINED DEGREES OF FREEDOM........',I6,//      N0003830
     &  4X,'  NODE COS#',4X,'X-COORD',8X,'Y-COORD',8X,'Z-COORD',4X,     N0003840
     &  4X,'FX',6X,'FY',6X,'FZ',6X,'MX',6X,'MY',6X,'MZ')                N0003850
       DO 610 I=1,NNODE                                                 N0003860
         DO 605 J=1,6                                                   N0003870
            IF (BTEST(JTFLG(I),J-1)) THEN                               N0003880
              CX(J)='-R'                                                N0003890
            ELSE IF (BTEST(JTFLG(I),(J+11))) THEN                       N0003900
              CX(J)='-C'                                                N0003910
            ELSE                                                        N0003920
              CX(J)='  '                                                N0003930
            ENDIF                                                       N0003940
  605    CONTINUE                                                       N0003950
  610  WRITE(6,620) IDOF(I),JTCOS(I),(COORD(I,J),J=1,3),                N0003960
     & (IDOF(I,K),CX(K),K=1,6)                                          N0003970
  620 FORMAT(4X,I6,1P,I5,3G15.5,0P,6(I6,A2) )                           N0003980
      WRITE(6,625)                                                      N0003990
  625 FORMAT(10X,'NOTE: R - RESTRAINED DEGREE OF FREEDOM'/              N0004000
     &       10X,'      C - CONSTRAINED DEGREE OF FREEDOM')            N0004010
C                                                                       N0004020
C--------- PRINT OUT CONSTRAINT MATRIX                                  N0004030
      IF (BUG) THEN                                                     N0004040
       FLAG=.TRUE.                                                      N0004050
       DO 540 I=1,NNODE                                                 N0004060
         TEST=.FALSE.                                                   N0004070
         DO 536 J=1,6                                                   N0004080
  536    TEST = TEST .OR. BTEST(JTFLG(I),(J+11))                        N0004090
         IF (TEST) THEN                                                 N0004100
            IF (FLAG) WRITE (6,1000) 'NODE CONSTRAINT MATRIX '          N0004110
            FLAG=.FALSE.                                                N0004120
            WRITE(6,550) IDOF(I),(CONST(I,J),J=1,6)                     N0004150
         ENDIF                                                          N0004140
  540  CONTINUE                                                         N0004150
  550  FORMAT(4X,I6,1P,                                                 N0004160
     &    '  T12:',G12.5,'  T13:',G12.5,                                N0004170
     &    '  T21:',G12.5,'  T23:',G12.5,                                N0004180
     &    '  T31:',G12.5,'  T32:',G12.5)                                N0004190
       ENDIF                                                            N0004200
C                                                                       N0004210
C--------- PRINT OUT DIRECTION COSINES                                  N0004220
      WRITE (6,1000) 'DIRECTION COSINES ... '                           N0004230
      DO 560 ICOS=1,NCOS                                                N0004240
         WRITE (6,56)  ICOS,' VX:',(COSINE(1,J,ICOS),J=1,3),            N0004250
     &                 ' VY:',(COSINE(2,J,ICOS),J=1,3),                 N0004260
     &                 ' VZ:',(COSINE(3,J,ICOS),J=1,3)                  N0004270
  560 CONTINUE                                                          N0004280
C                                                                       N0004290
      RETURN                                                            N0004300
 1000 FORMAT (//,5X,A/)                                                 N0004310
      END                                                               N0004320
C=======================================================================PBI00010
      SUBROUTINE PBIT(NNODE,JTFLG,ID)                                   PBI00040
      DIMENSION JTFLG(NNODE),ID(NNODE)                                  PBI00050
      LOGICAL   FLAG(0:51)                                              PBI00060
      LOGICAL BTEST                                                     PBI00070
      WRITE (6,40)                                                      PBI00080
   40 FORMAT (14X,' RESTRAINT',',  CONDENSE',',CONSTRAIN',',ELIMINATE',/ PBI00090
     &       14X,4('4X,"FFFMMM"),/14X,4(4X,'XYZXYZ'))                   PBI00100
      DO 20 I=1,NNODE                                                   PBI00110
        DO 10 J=0,23                                                    PBI00120
   10   FLAG(J)=BTEST(JTFLG(I),J)                                       PBI00130
   20   WRITE (6,30) ID(I),(FLAG(K),K=0,23),JTFLG(I)                    PBI00140
   30 FORMAT (' NODE(',I4,') ',4(4X,6L1),I10)                           PBI00150
      RETURN                                                            PBI00160
      END                                                               PBI00170
C=======================================================================REA00010
      SUBROUTINE REACTN(IOPT,PRINT,NNODE,L1,L2,L3,L4,FORCE,DISPL,       REA00050
     &    MD,A,KM,NDOF,NLOAD,FACTOR,                                    REA00060
     &    MAXNOD,IDOF,COORD,ID,TITLE,STEPID,HEAD,                       REA00070
     &    NCOS,COSINE,JTCOS,JTFLG,SUM)                                  REA00080
      DIMENSION A(KM),MD(NDOF+1)                                        REA00090
      DIMENSION FORCE(NDOF,NLOAD),DISPL(NDOF,NLOAD)                     REA00100
      DIMENSION IDOF(MAXNOD,6)                                          REA00110
      DIMENSION ID(MAXNOD),COORD(MAXNOD,3)                              REA00120
      DIMENSION F(6),G(6),SUM(6),COSINE(3,3,NCOS),JTCOS(MAXNOD)         REA00130
      DIMENSION JTFLG(MAXNOD)                                           REA00140
      CHARACTER*4 TITLE(2),STEPID                                       REA00150
      CHARACTER*15 CD(6)                                                REA00160
      LOGICAL PRINT,HEAD,FLAG,BTEST                                     REA00170
C                                                                       REA00180
      IF (IOPT.EQ.1) THEN                                               REA00190
C                                                                       REA00200
C=============================================                          REA00210
C= MODIFY LOADS FOR RESTRAINT DISPLACEMENTS ===                         REA00220
C=============================================                          REA00230
C CONSIDER THE PARTITIONED STIFFNESS WHERE X2 ARE THE RESTRAINT DISPL.  REA00240
C                                                                       REA00250
C   |P1|   | K11 | K12 | |X1|                                          REA00260
C   |--| = |-----+-----| |--|                                          REA00270
C   |P2|   | K21 | K22 | |X2|                                          REA00280
C                                                                       REA00290
C   THE MODIFIED FREE LOAD IS (P1-K12*X2) = K11*X1                      REA00300
C-------------------------------------------------------C              REA00310
C     DO 10 I=L1,L2                                     C  FULL STORAGE REA00320
C     DO 10 J=1,NLOAD                                   C  EQUIVALENT   REA00330
C     DO 10 K=L3,L4                                     C               REA00340
C  10   FORCE(I,J)=FORCE(I,J)-A(I,K)*DISPL(K,J)         C               REA00350
C-------------------------------------------------------C              REA00360
      DO 20 K=L3,L4                                                     REA00370
      MDK=MD(K+1)                                                       REA00380
        DO 20 J=1,NLOAD                                                 REA00390
        DO 10 I=L2,L1,-1                                                REA00400
          IK=MDK(K)+K-I                                                 REA00410
          IF (IK.GE.MDK) GO TO 20                                       REA00420
          FIJ=FORCE(I,J)                                                REA00430
          AIK=A(IK)                                                     REA00440
          DKJ=DISPL(K,J)                                                REA00450
   10     FORCE(I,J)=FORCE(I,J)-A(IK)*DISPL(K,J)                        REA00460
   20   CONTINUE                                                        REA00470
        IF (PRINT)                                                      REA00480
     &  CALL MMATRX(FORCE,NDOF,NLOAD,'MODIFIED LOADS')                  REA00490
C                                                                       REA00500
      ELSE IF (IOPT.EQ.2) THEN                                          REA00510
C                                                                       REA00520
C=======================                                                REA00530
C= SOLVE FOR REACTIONS ===                                              REA00540
C=======================                                                REA00550
C CONSIDER THE PARTITIONED STIFFNESS WHERE X2 ARE THE RESTRAINT DISPL.  REA00560
C                                                                       REA00570
C   |P1|   | K11 | K12 | |X1|                                          REA00580
C   |--| = |-----+-----| |--| * FACTOR                                  REA00590
C   |P2|   | K21 | K22 | |X2|                                          REA00600
C                                                                       REA00610
C   THE REACTIONS ARE  P2 = P2 + ( K21*X1 + K22*X2 ) * FACTOR           REA00620
C   NOTE: BOTH P1 AND P2 CONTAIN FIXED END FORCES FROM MEMBER LOADS...  REA00630
C         THUS, K21*X1 + K22*X2 IS ADDED TO P2 TO GET THE REACTIONS.    REA00640
C-------------------------------------------------------C              REA00650
C     DO 120 I=L3,L4                                    C    FULL STORAGE REA00660
C     DO 120 J=1,NLOAD                                  C    MODE EQUIVALENT REA00670
CC     FORCE(I,J)=0    * P2 CONTAINS FEF!!!             C              REA00680
C      DO 110 K=L1,L2                                   C              REA00690
C 110   FORCE(I,J)=FORCE(I,J)+A(I,K)*DISPL(K,J)        C              REA00700
C      DO 120 K=L3,L4                                   C              REA00710
C 120   FORCE(I,J)=FORCE(I,J)+A(I,K)*DISPL(K,J)        C              REA00720
C-------------------------------------------------------C              REA00730
C     DO 100 J=1,NLOAD                                                  REA00740
C     DO 100 I=L3,L4                                                    REA00750
C 100  FORCE(I,J)=0                                                     REA00760
C                                                                       REA00770
      DO 120 I=L3,L4                                                    REA00780
      DO 120 J=1,NLOAD                                                  REA00790
      DO 110 K=L2,L1,-1                                                 REA00800
        KI=MD(I)+I-K                                                    REA00810
        IF (KI.GE.MD(I+1)) GO TO 120                                    REA00820
        AKI=A(KI)                                                       REA00830
  110   FORCE(I,J)=FORCE(I,J)+A(KI)*DISPL(K,J)*FACTOR                   REA00840
  120 CONTINUE                                                          REA00850
C                                                                       REA00860
      DO 140 I=L3,L4                                                    REA00870
      DO 140 K=L3,L4                                                    REA00880
        IF (I.LE.K) THEN                                                REA00890
          IK=MD(K)+K-I                                                  REA00900
          MDK=MD(K+1)                                                   REA00910
        ELSE                                                            REA00920
          IK=MD(I)+I-K                                                  REA00930
          MDK=MD(I+1)                                                   REA00940
        ENDIF                                                           REA00950
        IF (IK.GE.MDK)  GO TO 140                                       REA00960
        AIK=A(IK)                                                       REA00970
        DO 130 J=1,NLOAD                                                REA00980
  130     FORCE(I,J)=FORCE(I,J)+A(IK)*DISPL(K,J)*FACTOR                 REA00990
  140 CONTINUE                                                          REA01000
C                                                                       REA01010
        IF (PRINT)                                                      REA01020
```

```
     &  CALL MMATRX(FORCE,NDOF,NLOAD,'FINAL LOADS AND REACTIONS')
C
      ELSE IF (IOPT.EQ.5 .OR. IOPT.EQ.6) THEN
         NF=L2
C=============================
C= LOOP FOR EACH LOAD CASE  ===
C=============================
      DO 300 L=1,NLOAD
C
C=============================
C= PRINT GLOBAL REACTIONS    ===
C=============================
C
      IF (PRINT) THEN
          IF (HEAD) WRITE (6,320) TITLE(1),TITLE(2)
            NREACT=0
            IF (IOPT.NE.4) WRITE (6,321) L,STEPID
            IF (IOPT.EQ.4) WRITE (6,421)
      ENDIF
        IF (IOPT.EQ.5) THEN
          DO 190 I=1,6
 190        SUM(I)=0
        ENDIF
C
      IF ( .NOT.(NCOS.EQ.1 .AND. COSINE(1,1,1).EQ.1 .AND.
     &           COSINE(2,2,1).EQ.1 .AND. COSINE(3,3,1).EQ.1)
     &   .AND. IOPT.EQ.4  ) GO TO 279

C
        DO 200 I=1,NNODE
C
C.............. GET REACTNS IN JOINT COORDINATE SYSTEM
          FLAG=.FALSE.
          DO 209 J=1,6
            IJ=IDOF(I,J)
            IF (IJ.LE.NF .OR. BTEST(JTFLG(I),J+11)) THEN
               F(J)=0
            ELSE
               F(J)=FORCE(IJ,L)
               FLAG=.TRUE.
            ENDIF
 209      CONTINUE
          IF (.NOT.FLAG) GO TO 200
C
C.............. TRANSFER REACTNS TO GLOBAL COORDINATE SYSTEM
          DO 229 I1=1,5
            I2=I1+5
            G(I1)=0
            G(I2)=0
            DO 219 J1=1,3
              J2=J1+3
              G(I1)=G(I1) + COSINE(J1,I1,JTCOS(I))*F(J1)
 219          G(I2)=G(I2) + COSINE(J1,I1,JTCOS(I))*F(J2)
            IF (PRINT) THEN
                WRITE (CDC(I1),210) G(I1)
                WRITE (CDC(I2),210) G(I2)
            ENDIF
 229      CONTINUE
C
C.............. SUM REACTIONS ..........
        IF (IOPT.EQ.5) THEN
          DO 240 I1=1,6
 240        SUM(I1)=SUM(I1) + G(I1)
          SUM(4)= SUM(4) - G(2)*COORD(I,3) + G(3)*COORD(I,2)
          SUM(5)= SUM(5) + G(1)*COORD(I,3) - G(3)*COORD(I,1)
          SUM(6)= SUM(6) - G(1)*COORD(I,2) + G(2)*COORD(I,1)
        ENDIF
C
C.............. PRINT GLOBAL REACTIONS .........
          NREACT=NREACT+1
          IF (PRINT) WRITE (6,220) ID(I),(CDC(K),K=1,6)
 200    CONTINUE
        IF (IOPT.EQ.5 .AND. PRINT ) WRITE (6,250) SUM
 279    IF (IOPT.EQ.4 .AND. PRINT ) WRITE (6,251) SUM
C
C=================================================
C= PRINT LOCAL JOINT COORDINATE SYSTEMS REACTIONS  ===
C=================================================
C
      IF (.NOT. PRINT ) GO TO 300
      IF (NCOS.EQ.1 .AND. COSINE(1,1,1).EQ.1 .AND. COSINE(2,2,1).EQ.1
     &                .AND. COSINE(3,3,1).EQ.1 ) GO TO 300

      IF (HEAD.AND. NREACT.GT.20) WRITE (6,320) TITLE(1),TITLE(2)
        IF (IOPT.NE.4) WRITE (6,322) L,STEPID
        IF (IOPT.EQ.4) WRITE (6,422)
        DO 290 I=1,NNODE
C
C.............. GET REACTNS IN JOINT COORDINATE SYSTEM
          FLAG=.FALSE.
          DO 280 J=1,6
            IJ=IDOF(I,J)
            IF (IJ.LE.NF .OR. BTEST(JTFLG(I),J+11)) THEN
               CDC(J)=' '
            ELSE
               WRITE (CDC(J),210) FORCE(IJ,L)
               FLAG=.TRUE.
            ENDIF
 280      CONTINUE
          IF (.NOT.FLAG) GO TO 290
C
C.............. PRINT REACTNS IN LOCAL COORDINATE SYSTEM .....
          WRITE (6,221) ID(I),(CDC(K),K=1,6),JTCOS(I)
 290    CONTINUE
 300  CONTINUE
C
 210  FORMAT (1P,G15.7)
 220  FORMAT (5X,I5,6A)
 221  FORMAT (5X,I5,6A,I10)
 250  FORMAT (/'          ',90('-')./
     &        ' SUMMATION',6G15.7)
 251  FORMAT (' MAX OF ALL',89('-')./
     &        ' GCS SUMM.',6G15.7 )
 320  FORMAT ('1 STRUCTURE..... : ',A,/
     &        '  SOLUTION..... : ',A)
 321  FORMAT (///'  GCS RESTRAINT REACTIONS,   LOADING #',I5,5X,A,/
     &        ' ==========================='//
     &  6X,'NODE',7X,'FX',13X,'FY',13X,'FZ',13X,'MX',13X,'MY',13X,'MZ'/
 322  FORMAT (///'  JCS RESTRAINT REACTIONS,   LOADING #',I5,5X,A,/
     &        ' ==========================='//
     &  6X,'NODE',7X,'FX',13X,'FY',13X,'FZ',13X,'MX',13X,'MY',13X,'MZ',
     &  6X,'  COSINE #'/)
 421  FORMAT (///'  MAXIMUM GCS RESTRAINT REACTIONS',
     &        5X,'NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'//
     &        ' ==========================='//
     &  6X,'NODE',7X,'FX',13X,'FY',13X,'FZ',13X,'MX',13X,'MY',13X,'MZ'/)
 422  FORMAT (///'  MAXIMUM JCS RESTRAINT REACTIONS ',
     &        5X,'NOTE: MAXIMUM VALUES MAY NOT OCCUR SIMULTANEOUSLY'//
     &        ' ==========================='//
     &  6X,'NODE',7X,'FX',13X,'FY',13X,'FZ',13X,'MX',13X,'MY',13X,'MZ',
     &  6X,'  COSINE #'/)
C
      ELSE
         WRITE (6,*)'INVALID OPTION IN SUBROUTINE  REA_CTN, IOPT=',IOPT
      ENDIF
      RETURN
      END
C=======================================================================
      SUBROUTINE ROTXYZ(VX,VY,COSINE,CT,EX,EY,EZ,CTXYZ,GLOXYZ,NROW)
      LOGICAL BTEST
      DIMENSION VX(3),VY(3),VZ(3),COSINE(3,3),CT(3,3),
     &    XYZ(3,3),CTXYZ(3,3),CE(3,3),GLOXYZ(NROW,6)
C-----------------------------------------------------------------------
C
C     THIS SUBROUTINE PREFORMS SEVERAL FUNCTIONS:
C     1) CALCULATE LOCAL COORDINATE SYS COSINE MATRIX.
C     2) CALCULATE ROTATION MATRIX BETWEEN LOCAL AND JOINT COORD (CT)
C     3) CALCULATE TRANSFORMATION FOR GLOBAL CONSTRAINTS AND MEMBER
C        END DEPTH- STORE IN (CTXYZ)
C
C--- INPUT VARIABLES-----------------------------------------------------
C     COSINE = INPUT JOINT COORDINATE DIRECTION COSINE
C     VX     = INPUT X-DIRECTION OF ELEMENT COORDINATE SYS
C     VY     = INPUT VECTOR IN THE ELEMENT COORDINATE SYS  XY PLANE.
C     GLOXYZ = GLOBAL CONSTRAINT TRANSFORMATION COIEFFICIENTS
C     EX     = MEMBER END ECCENTRICITY IN THE LOCAL X AXIS.
C     EY     = MEMBER END ECCENTRICITY IN THE LOCAL Y AXIS.
C     EZ     = MEMBER END ECCENTRICITY IN THE LOCAL Z AXIS.
C
C--- OUTPUT VARIABLES----------------------------------------------------
C     VX     = UNIT VECTOR, X-DIRECTION OF ELEMENT COORDINATE SYS
C     VY     = UNIT VECTOR, Y-DIRECTION OF ELEMENT COORDINATE SYS
C     VZ     = UNIT VECTOR, Z-DIRECTION OF ELEMENT COORDINATE SYS
C     CT     = ROTATED COORDINATE TRANSFORMATION
C     CTXYZ  = CONSTRAINT AND MEMBER END DEPTH TRANSFORMATION
C
C    | FX |                          | FX |
C    | FY |                          | FY |
C    | FZ |          |   CT   | 0 |  | FZ |
C    | ---|    =     |--------+---|  |----|
C    | MX |          | CTXYZ  | CT |  | MX |
C    | MZ |          |                | MY |
C    | MZ |          |                | MZ |
C          GLOBAL                          ELEMENT COORD.
C
C-----------------------------------------------------------------------
C
      CROSSI(A1,B1,C1,A2,B2,C2)=B1*C2-B2*C1
      CROSSJ(A1,B1,C1,A2,B2,C2)=C1*A2-C2*A1
      CROSSK(A1,B1,C1,A2,B2,C2)=A1*B2-A2*B1
      VALUE (A1,A2,A3)=SQRT(A1**2+A2**2+A3**2)
      DOT   (A1,B1,C1,A2,B2,C2)=A1*A2+B1*B2+C1*C2
C
C ------- CHECK ANGLE BETWEEN VX AND VY
      C1=VCOS(VX,VY)
      IF (ABS(C1).GT.999) WRITE (6,*) 'VY IS PARAELL TO VX, REINPUT VY'
C
C ------- CHECK VY, IF !VY!.EQ.0 , SET VY IN GLOBAL YZ PLANE...
      AVY=VALUE(VY(1),VY(2),VY(3))
      IF (AVY.EQ.0) THEN
          VZ(1)=1.0
          VZ(2)= .0
          VZ(3)= .0
      ELSE
C ------- CALCULATE VZ, IF VY WAS CORRECTLY INPUT
          VZ(1)=CROSSI(VX(1),VX(2),VX(3),VY(1),VY(2),VY(3))
          VZ(2)=CROSSJ(VX(1),VX(2),VX(3),VY(1),VY(2),VY(3))
          VZ(3)=CROSSK(VX(1),VX(2),VX(3),VY(1),VY(2),VY(3))
      ENDIF
C
C ------- RECALCULATE VY TO INSURE VX,VY AND VZ ARE ORTHOGONAL...
      VY(1)=CROSSI(VZ(1),VZ(2),VZ(3),VX(1),VX(2),VX(3))
      VY(2)=CROSSJ(VZ(1),VZ(2),VZ(3),VX(1),VX(2),VX(3))
      VY(3)=CROSSK(VZ(1),VZ(2),VZ(3),VX(1),VX(2),VX(3))
C
C ------- CHECK ANGLE BETWEEN VX AND VY
      C1=VCOS(VX,VY)
      IF (ABS(C1).GT.999) WRITE (6,*) 'VY IS PARAELL TO VX, REINPUT VY'
C
C ------- NORMALIZE VX,VY,VZ
      AVX=VALUE(VX(1),VX(2),VX(3))
      AVY=VALUE(VY(1),VY(2),VY(3))
      AVZ=VALUE(VZ(1),VZ(2),VZ(3))
      DO 10 I=1,3
         VX(I)=VX(I)/AVX
         VY(I)=VY(I)/AVY
         VZ(I)=VZ(I)/AVZ
         CE(1,I)=VX(I)
         CE(2,I)=VY(I)
 10      CE(3,I)=VZ(I)
C
C ------- CALC THE DIRECTION COSINE MATRIX
      DO 15 I=1,3
      DO 15 J=1,3
         CT(I,J)=0
         DO 15 K=1,3
 15         CT(I,J)=CT(I,J)+COSINE(I,K)*CE(J,K)
C
C ------- REMOVE THE FUZZ FROM THE COSINE MATRIX
      DO 20 I=1,3
      DO 20 J=1,3
 20      IF (ABS(CT(I,J)).LT. 0.0001) CT(I,J)=0.
C
C ------- SET UP THE GLOBAL CONSTRAINT MATRIX
      XYZ(1,1)= .0
      XYZ(1,2)=GLOXYZ(1,1)
      XYZ(1,3)=GLOXYZ(1,2)
      XYZ(2,1)=GLOXYZ(1,3)
      XYZ(2,2)= .0
      XYZ(2,3)=GLOXYZ(1,4)
      XYZ(3,1)=GLOXYZ(1,5)
      XYZ(3,2)=GLOXYZ(1,6)
      XYZ(3,3)= .0
C
      DO 25 I=1,3
        DO 25 J=1,3
          CTXYZ(I,J)=0
```

```
      DO 25 K=1,3                                                       ROT01140
  25    CTXYZ(I,J)=CTXYZ(I,J) + XYZ(I,K)*CT(K,J)                        ROT01150
C                                                                       ROT01160
C ------- SKIP THE MEMBER END ECCENTRICITY MATRIX IF EX=EY=EZ=0         ROT01170
      IF (EX.EQ.0 .AND. EY.EQ.0 .AND. EZ.EQ.0) RETURN                   ROT01180
C                                                                       ROT01190
C ------- CALC THE MEMBER END ECCENTRICITY MATRIX                       ROT01200
      XYZ(1,1)= .0                                                      ROT01210
      XYZ(1,2)=-EZ                                                      ROT01220
      XYZ(1,3)= EY                                                      ROT01230
      XYZ(2,1)= EZ                                                      ROT01240
      XYZ(2,2)= .0                                                      ROT01250
      XYZ(2,3)=-EX                                                      ROT01260
      XYZ(3,1)=-EY                                                      ROT01270
      XYZ(3,2)= EX                                                      ROT01280
      XYZ(3,3)= .0                                                      ROT01290
C                                                                       ROT01300
C ------- CALC THE MEMBER END ECCENTRICITY MATRIX                       ROT01310
      DO 30 I=1,5                                                       ROT01320
        DO 30 J=1,3                                                     ROT01330
          DO 50 K=1,3                                                   ROT01340
  50        CTXYZ(I,J)=CTXYZ(I,J) + CT(I,K)*XYZ(K,J)                    ROT01350
C                                                                       ROT01360
      RETURN                                                            ROT01370
      END                                                               ROT01380
C=======================================================================SKV00010
      SUBROUTINE SKVLIN(IOPT,NDOF,IBUG,TITLE,MDIAG,IELDOF,LM,SE,MD,NAME)SKV00040
      DIMENSION MDIAG(NDOF+1)                                           SKV00050
      DIMENSION MD(IELDOF+1),LM(IELDOF),SE(IELDOF*(IELDOF+1)/2)         SKV00060
      LOGICAL BTEST                                                     SKV00070
      CHARACTER*(*) NAME,TITLE(2)                                       SKV00080
      CHARACTER*1 CX(100)                                               SKV00090
      LOGICAL PT,BUG                                                    SKV00100
C                                                                       SKV00110
      IF (IOPT.EQ.1) THEN                                               SKV00120
        DO 10 I=1,NDOF+1                                                SKV00130
  10      MDIAG(I)=I                                                    SKV00140
        IF ( BTEST(IBUG,12) ) THEN                                      SKV00150
          WRITE (6,*) 'INITIAL VALUES'                                  SKV00160
          WRITE (6,520) (I,MDIAG(I),I=1,NDOF)                           SKV00170
        ENDIF                                                           SKV00180
C                                                                       SKV00190
      ELSE IF (IOPT.EQ.2) THEN                                          SKV00200
C---------- DETERMINE THE SKYLINE OF THIS ELEMENT, AND MODIFY THE       SKV00210
C           GLOBAL SKYLINE...                                           SKV00220
C....... I    = THE ROW OF THE MEMBER STIFFNESS MATRIX                  SKV00230
C....... J    = THE COLUMN OF THE MEMBER STIFFNESS MATRIX               SKV00240
C....... IJ   = THE ADDRESS OF THE MEMBER STIFFNESS TERM I,J            SKV00250
C....... LMI  = THE ROW OF THE GLOBAL STIFFNESS MATRIX CORRESPONDING    SKV00260
C               TO ROW I OF THE MEMBER STIFFNESS MATRIX                 SKV00270
C....... LMJ  = THE COLUMN OF THE GLOBAL STIFFNESS MATRIX CORRESPONDING SKV00280
C               TO COLUMN J OF THE MEMBER STIFFNESS MATRIX              SKV00290
C....... MDIAG(JJ) = THE JJ'TH ELEMENT OF THE GLOBAL SKYLINE MATRIX.    SKV00300
C               AT THIS POINT THE GLOBAL SKYLINE MATRIX CONTAINS THE    SKV00310
C               LOWEST ROW NUMBER THAT CONTAINS A NON-ZERO STIFFNESS    SKV00320
C               TERM.                                                   SKV00330
C                                                                       SKV00340
        DO 250 I=1,IELDOF                                               SKV00350
          LMI=LM(I)                                                     SKV00360
          IF (LMI.LE.0) GO TO 250                                       SKV00370
          DO 240 J=I,IELDOF                                             SKV00380
            LMJ=LM(J)                                                   SKV00390
            IF (LMJ.LE.0) GO TO 240                                     SKV00400
              IJ=MD(J)+J-I                                              SKV00410
              IF (SE(IJ).EQ.0) GO TO 240                                SKV00420
              IF (LMI.LE.LMJ) THEN                                      SKV00430
                  MDIAG(LMJ)=MIN(MDIAG(LMJ),LMI)                        SKV00440
              ELSE                                                      SKV00450
                  MDIAG(LMI)=MIN(MDIAG(LMI),LMJ)                        SKV00460
              ENDIF                                                     SKV00470
  240       CONTINUE                                                    SKV00480
  250     CONTINUE                                                      SKV00490
C-----------------------------------------------------------------------SKV00500
C                                                                       SKV00510
      ELSE IF (IOPT.EQ.3) THEN                                          SKV00520
C---------- CALCULATE THE INDICIES OF THE MAIN DIAGONAL TERMS...        SKV00530
      IF ( BTEST(IBUG,12) ) THEN                                        SKV00540
          WRITE (6,*) 'AFTER COLUMN HEIGHTS'                            SKV00550
          WRITE (6,520) (I,MDIAG(I),I=1,NDOF)                           SKV00560
      ENDIF                                                             SKV00570
      MDIAG(1)=1                                                        SKV00580
      ICOLHT=1                                                          SKV00590
      DO 310 IROW=2,(NDOF+1)                                            SKV00600
        ILOCN=ICOLHT + MDIAG(IROW-1)                                    SKV00610
        ICOLHT=IROW - (MDIAG(IROW)-1)                                   SKV00620
  310   MDIAG(IROW)=ILOCN                                               SKV00630
      IF ( BTEST(IBUG,12) ) THEN                                        SKV00640
          WRITE (6,*) 'ADDRESS OF MAIN DIAGONAL ELEMENTS'               SKV00650
          WRITE (6,520) (I,MDIAG(I),I=1,NDOF)                           SKV00660
  520     FORMAT (' MDIAG',I5,'=',I5)                                   SKV00670
      ENDIF                                                             SKV00680
  521 FORMAT (2I6)                                                      SKV00690
C                                                                       SKV00700
      IF ( .NOT.(BTEST(IBUG,12) .OR. BTEST(IBUG,9)) ) RETURN            SKV00710
C---------- PRINT A MAP OF THE STIFFNESS ARRAY                          SKV00720
      NL= MDIAG(NDOF+1)-MDIAG(1) +1                                     SKV00730
      NP=NDOF/100                                                       SKV00740
      NR=MOD(NDOF,100)                                                  SKV00750
      IF (NR.GT.0) NP=NP+1                                              SKV00760
      DO 270 IP=1,NP                                                    SKV00770
        JC1=(IP-1)*100                                                  SKV00780
        JC2=MIN(IP*100,NDOF)                                            SKV00790
        J2=100                                                          SKV00800
        IF (IP.EQ.NP) J2=NR                                             SKV00810
        WRITE (6,290) TITLE(1),TITLE(2),NAME,NL,(I,I=JC1,JC2,10)        SKV00820
        DO 270 I=1,NDOF                                                 SKV00830
          PT=.FALSE.                                                    SKV00840
          DO 260 JJ=1,J2                                                SKV00850
            J=JJ+(IP-1)*100                                             SKV00860
            IJ=MDIAG(J)+J-I                                             SKV00870
            IF (IJ.LT.MDIAG(J+1) .AND. I.LT.J) THEN                     SKV00880
                CX(JJ)='X'                                              SKV00890
                PT=.TRUE.                                               SKV00900
            ELSE IF (I.EQ.J) THEN                                       SKV00910
                CX(JJ)='0'                                              SKV00920
                PT=.TRUE.                                               SKV00930
            ELSE                                                        SKV00940
                CX(JJ)=' '                                              SKV00950
            ENDIF                                                       SKV00960
  260     CONTINUE                                                      SKV00970
  270   IF (PT) WRITE (6,295) I,(CX(J),J=1,J2)                         SKV00980
C                                                                       SKV00990
  290 FORMAT ('1 STRUCTURE....: ',A,/                                   SKV01000
     &        '  SOLUTION.....: ',A,//                                 SKV01010
     &        '  SKYLINE OF THE ',A,' MATRIX './                       SKV01020
     &  '  THE MATRIX REQUIRES',I6,' STORAGE LOCATIONS',                 SKV01030
     &  /2X,12(I4,'|.....'))                                           SKV01040
  295 FORMAT (I6,132A1)                                                 SKV01050
C                                                                       SKV01060
      ENDIF                                                             SKV01070
C                                                                       SKV01080
      RETURN                                                            SKV01090
      END                                                               SKV01100
C=======================================================================SKV00010
      SUBROUTINE SKVLN2(NDOF,LS,L4,MDS,MD,MDMASS,MDKG,KGFORM)           SKV00040
      DIMENSION MDMASS(NDOF+1),MDG(NDOF+1),MDS(NDOF+1),MDKG(NDOF+1)     SKV00050
C                                                                       SKV00060
C                                                                       SKV00070
C-------- DEVELOP SKYLINE OF DYNAMIC STIFFNESS (S)                      SKV00080
C....... MDS(I)   = ADDRESS OF DYNAMIC STIFF. MAIN DIAGONAL TERM ROW I  SKV00090
C....... MD(I)    = ADDRESS OF STIFFNESS MAIN DIAGONAL TERM ROW I        SKV00100
C....... MDMAS(I) = ADDRESS OF MASS MAIN DIAGONAL TERM ROW I            SKV00110
C....... MDKG(I)  = ADDRESS OF GEOMETRIC STIFF. MAIN DIAGONAL TERM ROW I SKV00120
C                                                                       SKV00130
C....... LSTIFF = # OF STIFFNESS TERMS IN COLUMN J                      SKV00140
C....... LMASS  = # OF MASS TERMS IN COLUMN J                           SKV00150
C....... LKG    = # OF KG TERMS IN COLUMN J                             SKV00160
C....... L      = # OF S  TERMS IN COLUMN J                             SKV00170
C                                                                       SKV00180
      NFREE=L4-LS+1                                                     SKV00190
      MDS(1)=1                                                          SKV00200
      LKG=0                                                             SKV00210
      DO 110 J1=2,NFREE+1                                               SKV00220
        J=J1+LS-1                                                       SKV00230
        LSTIFF=MD(J)-MD(J-1)                                            SKV00240
        LMASS =MDMAS(J)-MDMAS(J-1)                                      SKV00250
        IF (KGFORM.EQ.2)  LKG = MDKG(J)-MDKG(J-1)                       SKV00260
        L=MIN(J1-1,MAX(LSTIFF,LMASS,LKG))                               SKV00270
  110   MDS(J1)=MDS(J1-1)+L                                             SKV00280
C                                                                       SKV00290
      RETURN                                                            SKV00300
      END                                                               SKV00310
C=======================================================================SMA00020
C--- SUBROUTINE SMAX,  FINDS THE MAXIMUM AND MINIMUM VALUES             SMA00030
C                                        (AFTER LAHL)                   SMA00040
      SUBROUTINE SMAX (N,DT,A,YMX,YMN,T1,T2,NLEN,AVE,STDEV,RMS)         SMA00070
      DIMENSION A(NLEN)                                                 SMA00080
      LOGICAL BTEST                                                     SMA00090
      REAL*8  T  ,SUMX,SUMSQ ,ADBL                                      SMA00100
          AVE=0                                                         SMA00110
          STDEV=0                                                       SMA00120
          RMS=0                                                         SMA00130
           YMX=A(1)                                                     SMA00140
           YMN=A(1)                                                     SMA00150
           T=DT                                                         SMA00160
           T1=DT                                                        SMA00170
           T2=DT                                                        SMA00180
          IF (N.LE.1) RETURN                                            SMA00190
           SUMX=A(1)                                                    SMA00200
           ADBL=A(1)                                                    SMA00210
           SUMXSQ=ADBL**2                                               SMA00220
      DO 40 I=2,N                                                       SMA00230
       ADBL=A(I)                                                        SMA00240
       SUMX   = SUMX   + ADBL                                           SMA00250
       SUMXSQ = SUMXSQ + ADBL**2                                        SMA00260
         IF (YMX.LT.A(I)) THEN                                          SMA00270
              YMX=A(I)                                                  SMA00280
              T1=T                                                      SMA00290
         ELSE IF (YMN.GT.A(I)) THEN                                     SMA00300
              YMN=A(I)                                                  SMA00310
              T2=T                                                      SMA00320
         ENDIF                                                          SMA00330
        T=T+DT                                                          SMA00340
   40  CONTINUE                                                         SMA00350
       AVE=SUMX/N                                                       SMA00360
       IF ((N*SUMXSQ-SUMX**2) .GT.0 )                                   SMA00370
     &    STDEV=SQRT( (N*SUMXSQ-SUMX**2)/(N*(N-1)) )                   SMA00380
       RMS=SQRT(SUMXSQ/N)                                               SMA00390
       RETURN                                                           SMA00400
       END                                                              SMA00410
C=======================================================================SOL00010
      SUBROUTINE SOLN(OPTION)                                           SOL00040
      LOGICAL BTEST                                                     SOL00050
      CHARACTER*80 NAME,OPTION                                          SOL00060
      INCLUDE (ZCOMM)                                                   SOL00070
      STEPID=' '                                                        SOL00080
C                                                                       SOL00090
C=========================                                              SOL00100
C= DETERMINE SOLUTION NO. ==                                           SOL00110
C=========================                                              SOL00120
      CALL GETINT(OPTION,'SOL',NSOLN,0,,TRUE.,FALSE.)                   SOL00130
C                                                                       SOL00140
C=======================                                               SOL00150
C= CHOOSE THE SOLUTION ==                                              SOL00160
C=======================                                               SOL00170
      IF (NSOLN.EQ.1) THEN                                              SOL00180
            CALL SOL01                                                  SOL00190
      ELSE IF (NSOLN.EQ.2) THEN                                         SOL00200
            NLOAD=1                                                     SOL00210
            MAXELD=0                                                    SOL00220
            CALL SOL02                                                  SOL00230
      ELSE IF (NSOLN.EQ.3) THEN                                         SOL00240
            NLOAD=1                                                     SOL00250
            MAXELD=0                                                    SOL00260
            CALL SOL03                                                  SOL00270
      ELSE IF (NSOLN.EQ.4) THEN                                         SOL00280
            NLOAD=1                                                     SOL00290
            CALL SOL04                                                  SOL00300
      ELSE                                                              SOL00310
            WRITE (6,*) 'INVALID SOLN#',NSOLN                           SOL00320
            STOP 'INVALID SOLUTION NUMBER'                              SOL00330
      ENDIF                                                             SOL00340
   30 RETURN                                                            SOL00350
      END                                                               SOL00360
C=======================================================================SOL00020
      SUBROUTINE SOL01                                                  SOL00060
      LOGICAL PRINT,DSPFLG,HEAD,AXIAL,BTEST                             SOL00070
      CHARACTER*80 NAME                                                 SOL00080
      DIMENSION RINPUT(100)                                             SOL00090
      INCLUDE (ZCOMM)                                                   SOL00100
C                                                                       SOL00110
      READ (5,*)  NLOAD,MAXELD                                          SOL00120
C                                                                       SOL00130
      WRITE (6,1) TITLE(1),TITLE(2),NLOAD                               SOL00140
    1 FORMAT ('1 STRUCTURE....: ',A,/                                   SOL00150
     &        '  SOLUTION.....: ',A,//                                 SOL00160
     &  &1X,' SOLUTION #1, STATIC - ELASTIC ANALYSIS '/                 SOL00170
     &  &1X,'================================'//                       SOL00180
```

```
     &1X,' NUMBER OF LOAD CASES .............',I5)
C
      IF (NZ(IZKGOT+5).EQ.2) WRITE (6,2)
    2 FORMAT (1X,' GEOMETRIC STIFFNESS IS NOT INCLUDED, ',
     &          ' USE KGDATA(3)=1 TO INCLUDE GEOMETRIC STIFFNESS.')
C
C       ---- INITIALIZE STORAGE FOR STIFFNESS, LOAD AND DISPL... -----
      ELSTIC=.TRUE.
C
C       ---- INITIALIZE STORAGE FOR STIFFNESS, LOAD AND DISPL... -----
C             Z(IZSTIF) = LOCATION OF THE STIFFNESS MATRIX
C             Z(IZLOAD) = LOCATION OF THE LOAD MATRIX
C             Z(IZLOAD) = LOCATION OF THE DISPLACEMENT MATRIX
      IF (IZLOAD.EQ.0) THEN
           IZLOAD = IZ
           IZ     = IZ+NDOF*NLOAD
           ENDIF
      IF (IZDISP.EQ.0) THEN
           IZDISP = IZ
           IZ     = IZ+NDOF*NLOAD
           ENDIF
       IF (IZVEL .EQ.0) THEN
           IZVEL  = IZ
           IZ     = IZ+NDOF
           ENDIF
           DO 3 I=1,NDOF
    3      Z(I+IZVEL -1) =0
       IF (IZDVEL.EQ.0) THEN
           IZDVEL = IZ
           IZ     = IZ+NDOF
           ENDIF
           DO 4 I=1,NDOF
    4      Z(I+IZDVEL-1) =0
       IF (IZFUB .EQ.0) THEN
           IZFUB  = IZ
           IZ     = IZ+NDOF
           DO 5 I=1,NDOF
    5      Z(I+IZFUB -1) =0
           ENDIF
C
C       ---- SET UP CONSTANTS ----------------
      L1=1
      L2=NCOND +NFREE
      L3=NCOND +NFREE+1
      L4=NDOF
      IF (MAXELD.GT.0) THEN
           IZIELD=IZ
           IZ=IZ + MAXELD*5
           IZELD=IZ
           IZ=IZ + MAXELD*12
      ENDIF
      IM=IZ+1
      IMD=NZ(IZLM+L4)
      CALL SOL01A
     &    (L1,L2,L3,L4,IMD,MLOAD,MAXELD,NELD,NNODE,NCOS,NELMT,ABORT,
     &     IZDISP,IZLOAD,MSTOR,IBUG,MAXNOD,TITLE,STEPID,
     &     Z(IZSTIF), Z(IZLOAD), Z(IZDISP), Z(IZVEL ), Z(IZDVEL ),
     &     Z(IZFUB ),NZ(IZIELD), Z(IZELD ), Z(IM    ),NZ(IZMD ),
     &     NZ(IZID  ),NZ(IZIDOF), Z(IZCNST),NZ(IZJFLG), Z(IZCORD),
     &     Z(IZCOS ),NZ(IZJCOS),SUMRCT)
      RETURN
C
      END
C=======================================================================
      SUBROUTINE SOL01A
     &    (L1,L2,L3,L4,IMD,MLOAD,MAXELD,NELD,NNODE,NCOS,NELMT,ABORT,
     &     IZDISP,IZLOAD, MSTOR,IBUG,MAXNOD,TITLE,STEPID,
     &     STIF  , LOAD , DISP , VEL  , DVEL ,
     &     FUB   , IELD , ELD  , WORK , MD   ,
     &     ID    , IDOF , CONST, JTFLG, COORD,
     &     COSINE, JTCOS,SUMRCT)
      CHARACTER*(*) TITLE(2),STEPID
      LOGICAL PRINT,DSPFLG,HEAD,AXIAL,BTEST,BUG,ABORT,BUG3
      CHARACTER*80 NAME
      DIMENSION RINPUT(100),SUMRCT(6)
      REAL LOAD
      DIMENSION STIF(IMD),LOAD(L4,MLOAD),DISP(L4,MLOAD),VEL(L4),DVEL(L4)
      DIMENSION FUB(L4),IELD(MAXELD,5),ELD(MAXELD,12),WORK(1),MD(L4+1)
      DIMENSION COORD(MAXNOD,3),ID(MAXNOD),
     &          IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CONST(MAXNOD,6),
     &          JTFLG(MAXNOD),JTCOS(MAXNOD)
      BUG=BTEST(IBUG,5)
      BUG3=BTEST(IBUG,3)
C
C=======================
C= INPUT LOADING DATA ==
C=======================
C
C       ----- ZERO LOAD AND DISPLACEMENT MATRIX ---------------------
      DO  6 I=1,L4
      DO  6 J=1,MLOAD
         LOAD(I,J)=0
    6    DISP(I,J)=0
C
C       ----- INPUT JOINT LOADS -------------------
C       JOINT LOADS AND SUPPORT DISPLACEMENTS ARE STORED IN THE
C       DISPLACEMENT MATRIX
      CALL JOILOA(BUG5,MAXNOD,L4,NNODE,MLOAD,L2,
     &       ID,IDOF,CONST,DISP,
     &       DSPFLG,JTFLG,TITLE,.FALSE.)
C
C       ----- INPUT ELEMENT LOADS -------------------
C       ELEMENT LOADS ARE STORED IN THE LOAD MATRIX
      IF (MAXELD.GT.0) THEN
           CALL ELELOA(BUG5,L5,NELMT,MAXELD,NELD,L4,MLOAD,
     &          LOAD,IELD,ELD,IZDISP,IZLOAD,NAME,TITLE,.FALSE.)
C
C       ...ADD ELEMENT LOADS ON FREE JOINTS TO THE JOINT LOADS ON FREE
C       ...JOINTS, STORE IN THE DISPLACEMENT MATRIX
C       ...SAVE ELEMENT LOADS ON ALL JOINTS IN LOAD MATRIX, USED LATER
C       ...FOR CALCULATING THE REACTIONS.
           DO 10 I=L1,L2
           DO 10 J=1,MLOAD
   10      DISP(I,J)=DISP(I,J)+LOAD(I,J)
C
C       ...SWAP THE SIGN OF THE FEF AT REACTIONS
           DO 15 I=L3,L4
           DO 15 J=1,MLOAD
   15      LOAD(I,J)= -LOAD(I,J)
C
      ENDIF
C=========================
C= FORMULATE STIFFNESS ====
C=========================
      CALL FORM(1)
C
C=========================================
C= MODIFY LOADS FOR RESTRAINT DISPLACEMENTS ===
C=========================================
      IF (DSPFLG) THEN
        IOPT=1
        CALL REACTM(IOPT,BUG,NNODE,L1,L2,L3,L4,DISP,DISP,
     &  MD,STIF,IMD,L4,NLOAD,1.0,
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,
     &  NCOS,COSINE,JTCOS,JTFLG,SUMRCT)
      ENDIF
C
C=========================
C= SOLVE FOR FREE DISPL ====
C=========================
C
      IF (BTEST(IBUG,5)) THEN
       CALL LMATRX(STIF,MD,L4,IMD,'GLOBAL STIFFNESS DUMP=7')
       CALL MMATRX(DISP,L4,NLOAD,'LOAD MATRIX  DUMP=7')
      ENDIF
C
      IOPT=1
      CALL MSOLL(IOPT,BTEST(IBUG,6),L2,IMD,L4,NLOAD,L1,L2,
     &           MD,STIF,DISP,WORK,
     &          .FALSE.,WORK,WORK,L4,
     &          .FALSE.,WORK,WORK,L4,ABORT)
      IF (ABORT) RETURN
C
      IOPT=3
      CALL MSOLL(IOPT,BTEST(IBUG,6),L2,IMD,L4,NLOAD,L1,L2,
     &           MD,STIF,DISP,WORK,
     &          .FALSE.,WORK,WORK,L4,
     &          .FALSE.,WORK,WORK,L4,ABORT)
      IF (ABORT) RETURN
C
      IF (BTEST(IBUG,5))
     &    CALL MMATRX(DISP,L4,NLOAD,'DISPLACEMENT  DUMP=7')
C
C=========================
C= SOLVE FOR REACTIONS ===
C=========================
      IOPT=2
      CALL REACTM(IOPT,BUG,NNODE,L1,L2,L3,L4,LOAD,DISP,
     &  MD,STIF,IMD,L4,NLOAD,1.0,
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,
     &  NCOS,COSINE,JTCOS,JTFLG,SUMRCT)
      IF (BTEST(IBUG,5))
     &  CALL MMATRX(LOAD,L4,NLOAD,'FINAL LOADS AND REACTIONS')
C
C=============================
C= PRINT GLOBAL DISPLACEMENTS ===
C=============================
      CALL JOIDSP(1,MAXNOD,L4,NNODE,NLOAD,0,L2,ID,
     &  IDOF,CONST,DISP,
     &  TITLE,'DISPLACEMENTS',STEPID,,TRUE.,,
     &  NCOS,COSINE,JTCOS,JTFLG)
C
C=============================
C= PRINT GLOBAL REACTIONS ===
C=============================
      IOPT=5
      CALL REACTM(IOPT,,.TRUE.,NNODE,L1,L2,L3,L4,LOAD,DISP,
     &  MD,STIF,IMD,L4,NLOAD,1.0,
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,,.TRUE.,
     &  NCOS,COSINE,JTCOS,JTFLG,SUMRCT)
C
C=====================================
C= CALCULATE AND PRINT MEMBER FORCES ===
C=====================================
      IOPT=9
      LSTTYP=0
      PRINT=.TRUE.
      HEAD =.TRUE.
      DO 500 IELNO=1,NELMT
  500   CALL ELELIB
     &  (IOPT  ,LSTTYP ,PRINT ,IREL,HEAD,   NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)
C
C=========================================
C= ADD LAST LOAD CASE TO TOTAL MEMBER LOADS ===
C=========================================
C THIS SETS THE TOTAL LOAD IN THE ELEMENT SUBROUTINE EQUAL TO THE
C SUM OF THE TOTAL LOAD AND THE LAST LOAD CASE.
C TWO MAJIOR BENIFITS ARE DERIVED FROM THIS:
C 1) TOTAL ELEMENT LOADS ARE USED AS INITIAL VALUES FOR BUCKLING AND
C    DYNAMIC LOADINGS.
C 2) TOTAL ELEMENT LOADS ARE USED IN NONLINEAR ANALYSIS
C
      IOPT=5
      LSTTYP=0
      PRINT=.FALSE.
      HEAD =.FALSE.
      DO 510 IELNO=1,NELMT
  510   CALL ELELIB
     &  (IOPT  ,LSTTYP ,PRINT ,IREL,HEAD,   NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)
C
C
      IF (NLOAD.EQ.1) RETURN
C
C
C=================================
      WRITE (6,1000) TITLE(1),TITLE(2)
 1000 FORMAT ('1 STRUCTURE.....: ',A,/
     &       '  SOLUTION......: ',A,//
     &      &1X,' MAXIMUN VALUES FOR ALL LOAD CASES '/
     &      &1X,'================================')
C
C=================================
C= GET AND PRINT MAXIMUN DISPL ===
C=================================
C
      DO 1010 I=1,L4
        WORK(I   )=DISP(I,1)
        DO 1010 J=2,NLOAD
        IF (ABS(DISP(I,J)).GT.ABS(WORK(I))) WORK(I)=DISP(I,J)
 1010 CONTINUE
      CALL JOIDSP(2,MAXNOD,L4,NNODE,1,0,L2,ID,
     & IDOF,CONST,WORK,
     & TITLE,'MAXIMUM DISPLACEMENTS',STEPID,,FALSE.,,
     & NCOS,COSINE,JTCOS,JTFLG)
C
C=======================================
```

```
C= GET AND PRINT MAXIMUM REACTIONS ===
C=====================================
C
      WRITE (6,1000) TITLE(1),TITLE(2)
      DO 1020 I=1,L4
        WORK(I   )=LOADX(I,1)
        DO 1020 J=2,NLOAD
        IF (ABS(LOADX(I,J)).GT.ABS(WORK(I))) WORK(I)=LOADX(I,J)
 1020 CONTINUE
      DO 1030 I=1,6
 1030  SUMRCT(I)=0
      CALL REACTNK4,.TRUE.,NNODE,L1,L2,L3,L4,WORK,DISP,
     &  MD,STIF,IMD,L4,1    ,1.0,
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,
     &  NCOS,COSINE,JTCOS,JTFLG,SUMRCT)
C
C=====================================
C= PRINT MAXIMUM ELEMENT FORCES ===
C=====================================
C
      WRITE (6,1000) TITLE(1),TITLE(2)
      LSTTYP=0
      PRINT=.TRUE.
      HEAD =.FALSE.
      DO 1510 IELNO=1,NELMT
 1510   CALL ELELIB
     &  (L2  ,LSTTYP ,PRINT ,IREL,HEAD,   NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)
C
      RETURN
C
  220   FORMAT (5X,I5,1P,6G15.6)
  200   FORMAT ('1 GLOBAL ',A,',  LOADING #',I5//
     &  4X,'NODE',7X,'DX',15X,'DY',15X,'DZ',15X,'RX',15X,'RY',15X,'RZ')
      END
C=======================================================================
      SUBROUTINE SOL02
      LOGICAL PRINT,DSPFLG,HEAD,REPEAT,SLMASS,AXIAL,TEST,NEWKDV,BTEST
      LOGICAL UNBAL
      DIMENSION RINPUT(100)
      CHARACTER*1  NAME
      CHARACTER*20 INTEG
      INCLUDE (ZCOMM)
C
C
C==================
C= PRINT HEADER ===
C==================
      WRITE (6,10) TITLE(1),TITLE(2)
   10 FORMAT ('1 STRUCTURE.....: ',A,/
     &        '  SOLUTION.....: ',A,/)
C
C=======================
C= INPUT LOADING DATA ==
C=======================
      READ (5,*) INTEG,THETA, ELSTIC,UNBAL
      READ (5,*) IPRNT,IWRITE,MAXACC,SLMASS
      READ (5,*) TO,DT,TF,GRAV
      IPRNT=MAX(IPRNT,1)
C
      IF (TEST(INTEG,'LINEAR' ,.FALSE.)) THEN
            INTEGO=1
            WRITE (6,11) 'LINEAR ACCELERATION METHOD',
     &                   '========================'
      ELSE IF (TEST(INTEG,'AVERAGE',.FALSE.)) THEN
            INTEGO=2
            WRITE (6,11) 'AVERAGE ACCELERATION METHOD',
     &                   '========================'
      ELSE IF (TEST(INTEG,'WILSON' ,.FALSE.)) THEN
            INTEGO=3
            WRITE (6,11) 'WILSON THETA METHOD',
     &                   '====================='
            WRITE (6,12) THETA
      ELSE IF (TEST(INTEG,'WILSON' ,.FALSE.)) THEN
            WRITE (6,*) 'INVALID INTEGRATION METHOD'
            WRITE (6,11) INTEG,'===================='
            RETURN
      ENDIF
C
C------- CHECK FOR INVALID INTEGRATION WITH PROPORTIONAL DAMPING
      IF ((INTEGO.EQ.1 .OR. INTEGO.EQ.2) .AND. FDAMP.NE.1) THEN
         WRITE (6,19)
   19    FORMAT (5X,'INTEGRATION METHOD IS ONLY VALID FOR ',
     &           'PROPORTIONAL DAMPING'//,5X,'SOLN IS ABORTED')
         RETURN
      ENDIF
C
      WRITE (6,13) TO,DT,TF,GRAV,IPRNT
      IF (ELSTIC) WRITE (6,14)
C
   11 FORMAT (
     &1X,' SOLUTION #2, ',A,' OF NUMERICAL INTEGRATION'/
     &1X,'================',A,'=========================='/)
   12 FORMAT (
     &1X,' THETA.............................',1P,G15.6)
   13 FORMAT (
     &1X,' INITIAL TIME .....................',1P,G15.6/
     &1X,' TIME STEP ........................',1P,G15.6/
     &1X,' FINAL TIME .......................',1P,G15.6/
     &1X,' ACCELERATION DUE TO GRAVITY ......',1P,G15.6/
     &1X,' STEP INTERVAL FOR PRINTING .......',0P,I5)
   14 FORMAT (
     &1X,' THE STRUCTURE IS ASSUMED TO BEHAVE ELASTICALLY')
C
C=====================================
C= SET GEOMETRIC STIFFNESS FLAGS ===
C=====================================
      KGLOAD=NZ(IZKGDT+1)
      KGTYPE=NZ(IZKGDT+2)
      KGFORM=NZ(IZKGDT+3)
      NEWKG=.TRUE.
      IF (KGLOAD.EQ.0 .OR. KGTYPE.EQ.0 .OR. KGFORM.EQ.0) NEWKG=.FALSE.
C
C      ---- SET UP CONSTANTS ------------------
      L1=1
      L2=NCOND
      L3=L2+1
      L4=NCOND +NFREE
      L5=L4+1
      L6=NDOF
C
C      ---- INITIALIZE STORAGE FOR LOAD, DISPL, VELOC, ACCEL
C     Z(IZLOAD) = LOCATION OF THE TOTAL LOAD MATRIX
C     Z(IZDISP) = LOCATION OF THE TOTAL DISPLACEMENT MATRIX
C     Z(IZVEL ) = LOCATION OF THE TOTAL VELOCITY MATRIX
C     Z(IZACC ) = LOCATION OF THE TOTAL ACCELERATION MATRIX
C     Z(IZDLOA) = LOCATION OF THE INCRIMENTAL LOAD MATRIX
C     Z(IZDDSP) = LOCATION OF THE INCRIMENTAL DISPLACEMENT MATRIX
C     Z(IZDVEL) = LOCATION OF THE INCRIMENTAL VELOCITY MATRIX
C     Z(IZDACC) = LOCATION OF THE INCRIMENTAL ACCELERATION MATRIX
C
       IF (IZLOAD.EQ.0) THEN
           IZLOAD = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 1 I=1,NDOF
      1    Z(I+IZLOAD-1) =0
           ENDIF
C
       IF (IZDISP.EQ.0) THEN
           IZDISP = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 2 I=1,NDOF
      2    Z(I+IZDISP-1) =0
           ENDIF
C
       IF (IZVEL .EQ.0) THEN
           IZVEL  = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 3 I=1,NDOF
      3    Z(I+IZVEL -1) =0
           ENDIF
C
       IF (IZACC .EQ.0) THEN
           IZACC  = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 4 I=1,NDOF
      4    Z(I+IZACC-1) =0
           ENDIF
C
       IF (IZDLOA.EQ.0) THEN
           IZDLOA = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 5 I=1,NDOF
      5    Z(I+IZDLOA-1) =0
           ENDIF
       IF (IZDDSP.EQ.0) THEN
           IZDDSP = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 6 I=1,NDOF
      6    Z(I+IZDDSP-1) =0
           ENDIF
       IF (IZDVEL.EQ.0) THEN
           IZDVEL = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 7 I=1,NDOF
      7    Z(I+IZDVEL-1) =0
           ENDIF
       IF (IZDACC.EQ.0) THEN
           IZDACC = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 8 I=1,NDOF
      8    Z(I+IZDACC-1) =0
           ENDIF
       IF (IZFUB .EQ.0) THEN
           IZFUB  = IZ
           IZ     = IZ+NDOF
           CALL CKSTOR(0,0)
           DO 9 I=1,NDOF
      9    Z(I+IZFUB-1) =0
           ENDIF
       IM=IZ+1
C
C---------- FORM MASS MATRIX -----
C
       IMOMS =NZ(IZMOMS+NDOF)
       DO 400 I=1,IMOMS
  400    Z(I+IZMASS-1)=0.
C
       CALL MFORM2,INMASS,NMASS,Z(IZMASS),NZ(IZMOMS),MD,NDOF,IMOMS,
     &             TITLE,NNODE,MAXNOD,NCOS,BTEST(IBUG,5),IBUG,
     &             NZ(IZID),Z(IZCOS),NZ(IZIDOF),Z(IZCNST),NZ(IZJFLG),
     &             NZ(IZJCOS),Z(IZMOAI),Z(IM),NZ(IM+21),Z(IM+27),
     &              Z(IM+ 65),Z(IM+ 99))
C
C---------- FORM SPECIAL MASS MATRIX -----
C
       IF (SLMASS) THEN
           READ (5,*) INMASS,IFMASS
           INMASS=INMASS+1
C          CALL MASSINK INMASS,IFMASS,.FALSE.,BUG,IBUG,TITLE,MAXZ,MAXDZ,
     &    MAXNOD,NCOS,NNODE,IZ,IM,IZID,Z,NZ,DZ,
     &    IZSLMD,IZSLMS,IZLM,IZKE,IZMDT2,
     &    NDOF,NMASS,IZCOS,IZIDOF,IZCNST,IZJFLG,IZJCOS,'GLOBAL  MASS',
     &    NCOND,NFREE,IZMD)
           IMOMSL=NZ(IZSLMD+NDOF)
       ELSE
           IZSLMS=0
           IZSLMD=0
           IMOMSL=1
       ENDIF
       IMOMS =NZ(IZMOMS+NDOF)
C
C-------- SET UP STORAGE FOR MAIN DIAGONAL ADDRESS OF DYNAMIC STIFFNESS
       IZMDS= IZ
       IZ   = IZ + NFREE +1
       CALL CKSTOR(0,IM)
C
C-------- DEVELOP SKYLINE OF DYNAMIC STIFFNESS (S)
       CALL SKYLNZ(NDOF,L5,L4,
     &    NZ(IZMDS),NZ(IZMD),NZ(IZMOMS),NZ(IZMDKG),KGFORM)
C
C---- DETERMINE THE LENGTH OF THE DYNAMIC STIFFNESS MATRIX
       IMDS=NZ(IZMDS+NFREE) - NZ(IZMDS) +1
C
C---- DETERMINE THE ADDRESS OF THE DYNAMIC STIFFNESS MATRIX
       IZS    = IZ
       IZ     = IZ+NZ(IZMDS+NFREE)
C
```

```
C---------- DEFINE STORAGE REQD FOR NUMERICAL INTEGRATION                    SOL02160
C          AND INITTILIZE VALUES                                            SOL02170
      IZAN   = IZ                                                           SOL02180
      IZ     = IZ+NFREE                                                     SOL02190
      IZBN   = IZ                                                           SOL02200
      IZ     = IZ+NFREE                                                     SOL02210
      IZPBAR = IZ                                                           SOL02220
      IZ     = IZ+NFREE                                                     SOL02230
C                                                                           SOL02240
C---- SAVE STORAGE FOR DYNAMIC GROUND ACCELERATIONS                         SOL02250
          IZAGTX=IZ                                                         SOL02260
          IZ    =IZ+MAXACC                                                  SOL02270
          IZAGTY=IZ                                                         SOL02280
          IZ    =IZ+MAXACC                                                  SOL02290
          IZAGTZ=IZ                                                         SOL02300
          IZ    =IZ+MAXACC                                                  SOL02310
          IZAINP=IZ                                                         SOL02320
          NAINP =MAX(NDOF,MAXACC)                                           SOL02330
          IZ    =IZ+NAINP                                                   SOL02340
          IZMCOS=IZ                                                         SOL02350
          IZ    =IZ+NDOF*5                                                  SOL02360
C                                                                           SOL02370
C---- SAVE STORAGE FOR ENERGY CALCULATIONS                                  SOL02380
          IZZERO=IZ                                                         SOL02390
          IZ    =IZ+NDOF                                                    SOL02400
          IZDM  =IZ                                                         SOL02410
          IZ    =IZ+NDOF                                                    SOL02420
          IZVM  =IZ                                                         SOL02430
          IZ    =IZ+NDOF                                                    SOL02440
          IZDBAR=IZ                                                         SOL02450
          IZ    =IZ+NDOF                                                    SOL02460
          IZ    =IZ+NDOF                                                    SOL02470
          IZDL02=IZ                                                         SOL02480
          IZ    =IZ+NDOF                                                    SOL02490
          IZTFD =IZ                                                         SOL02500
          IZ    =IZ+NDOF                                                    SOL02510
          IZDFD =IZ                                                         SOL02520
          IZ    =IZ+NDOF                                                    SOL02530
          IZFG  =IZ                                                         SOL02540
          IZ    =IZ+NDOF                                                    SOL02550
          IZDFG =IZ                                                         SOL02560
          IZ    =IZ+NDOF                                                    SOL02570
      DO 25 I=0,NDOF-1                                                      SOL02580
        Z(IZFG  +I)=0                                                       SOL02590
        Z(IZDFG +I)=0                                                       SOL02600
        Z(IZZERO+I)=0                                                       SOL02610
        Z(IZDM  +I)=0                                                       SOL02620
        Z(IZVM  +I)=0                                                       SOL02630
        Z(IZDBAR+I)=0                                                       SOL02640
        Z(IZDL02+I)=0                                                       SOL02650
        Z(IZTFD +I)=0                                                       SOL02660
   25   Z(IZDFD +I)=0                                                       SOL02670
C                                                                           SOL02680
C=================================                                          SOL02690
      IZDMAX=IZ                                                             SOL02700
      IZ    =IZ+NDOF                                                        SOL02710
      IZVMAX=IZ                                                             SOL02720
      IZ    =IZ+NDOF                                                        SOL02730
      IZAMAX=IZ                                                             SOL02740
      IZ    =IZ+NDOF                                                        SOL02750
      IZRMAX=IZ                                                             SOL02760
      IZ    =IZ+NDOF *6                                                     SOL02770
      DO 28 I=0,NDOF                                                        SOL02780
        Z(IZDMAX+I)=0                                                       SOL02790
        Z(IZVMAX+I)=0                                                       SOL02800
        Z(IZAMAX+I)=0                                                       SOL02810
   28   Z(IZRMAX+I)=0                                                       SOL02820
      DO 29 I=1,6                                                           SOL02830
   29   Z(IZRMAX-1+NDOF+I)=0                                                SOL02840
      DO 50 I=1,6                                                           SOL02850
   50   SUMRCT(I)=0                                                         SOL02860
C                                                                           SOL02870
C=================================                                          SOL02880
C=INPUT FILE OUTPUT CONTROLL DATA ===                                       SOL02890
C=================================                                          SOL02900
      IF (INRITE.GT.0) THEN                                                 SOL02910
        IOPT=1                                                              SOL02920
        CALL DMPDAT(IOPT,INRITE,TO,DT)                                      SOL02930
      ENDIF                                                                 SOL02940
C                                                                           SOL02950
C------ SET UP WORK STORAGE AREA ...                                        SOL02960
      IW     = IZ + 1                                                       SOL02970
C                                                                           SOL02980
C------ CHECK STORAGE REQUIRMENTS ...                                       SOL02990
      CALL CKSTOR(NDOF,IW)                                                  SOL03000
C                                                                           SOL03010
C---- DETERMINE THE LENGTH OF THE STIFFNESS MATRIX                          SOL03020
          IMD=MZ(IZMD+NDOF) - MZ(IZMD) +1                                   SOL03030
C---- DETERMINE THE LENGTH OF THE MASS MATRIX                               SOL03040
          IMDMS=MZ(IZMDMS+NDOF) - MZ(IZMDMS) +1                             SOL03050
C---- DETERMINE THE LENGTH OF THE GEOMETRIC STIFFNESS MATRIX                SOL03060
          IF (MZ(IZKGDT+5).EQ.2) THEN                                       SOL03070
            IMDKG =MZ(IZMDKG+NDOF) - MZ(IZMDKG) +1                          SOL03080
          ELSE                                                              SOL03090
            IMDKG =1                                                        SOL03100
          ENDIF                                                             SOL03110
C                                                                           SOL03120
C------ CALL SOL2A TO CALC DATA ....                                        SOL03130
      CALL SOL02A                                                           SOL03140
     & (L1,L2,L3,L4,L5,L6,IMD,IMDMS,IMDKG,IMDS,IMDMSL,ABORT,ITLCPU,         SOL03150
     &  NLOAD    , MAXELD   , NMODE    , NCOS     , NELMT    ,             SOL03160
     &  IZDISP   , IZLOAD   , MSTOR    , IBUG     , MAXNOD   ,             SOL03170
     &  TITLE    , STEPID   , NAINP    , NDOF     , NFREE    ,             SOL03180
     &  KGFORM   , KGLOAD   , KGTYPE   , NCOND    , INTEGO   ,             SOL03190
     &  MEMKG    , MEMK     , MCOND    , KGCOND   , UNBAL    ,             SOL03200
     &  INTEG    , THETA    , ELSTIC   , IPRNT    , IWRITE   ,GROUND(1),  SOL03210
     &  MAXACC   , SLMASS   , TO       , DT       , TF       ,GROUND(4),  SOL03220
     &  GRAV     , FDAMP    , FMASS    , IZDOSP   ,IZDLOA    ,GROUND(7),  SOL03230
     &  Z(IZDISP), Z(IZVEL ), Z(IZACC ), Z(IZLOAD), Z(IZAN   ),Z(IZDMAX),  SOL03240
     &  Z(IZDOSP), Z(IZOVEL), Z(IZDACC), Z(IZLOLA), Z(IZBN   ),Z(IZVMAX),  SOL03250
     &  Z(IZSTIF), Z(IZMASS), Z(IZSLMS), Z(IZKG  ), Z(IZS    ),Z(IZAMAX),  SOL03260
     & MZ(IZMD ),MZ(IZMDMS),MZ(IZSLMD),MZ(IZMDKG),MZ(IZMDS ),Z(IZRMAX),    SOL03270
     &  Z(IZDAMP), Z(IZPBAR), Z(IZMCOS), Z(IZAGTX),          Z(IZDL02),    SOL03280
     &  Z(IZAGTY), Z(IZAGTZ), Z(IZAINP), Z(IZFUB ), Z(IM    ),  SUMRCT ,  SOL03290
     & MZ(IZID ),MZ(IZIDOF), Z(IZCNST),MZ(IZJFLG), Z(IZCORD),Z(IZDFD ),    SOL03300
     &  Z(IZCOS ),MZ(IZ.KOS), Z(IZTFD ),                                    SOL03310
     &  EIE,ESE,PSE,EKE,EDD,                                                SOL03320
     &  Z(IZZERO), Z(IZFG)  , Z(IZDFG)  )                                   SOL03330
C                                                                           SOL03340
      RETURN                                                                SOL03350
C                                                                           SOL03360
      END                                                                   SOL03370
C=====================================================================      SOL00040
      SUBROUTINE SOL02A                                                     SOL00050
     & (L1,L2,L3,L4,L5,L6,IMD,IMDMS,IMDKG,IMDS,IMDMSL,ABORT,ITLCPU,         SOL00060
     &  NLOAD    , MAXELD   , NMODE    , NCOS     , NELMT    ,             SOL00070
     &  IZDISP   , IZLOAD   , MSTOR    , IBUG     , MAXNOD   ,             SOL00080
     &  TITLE    , STEPID   , NAINP    , NDOF     , NFREE    ,             SOL00090
     &  KGFORM   , KGLOAD   , KGTYPE   , NCOND    , INTEGO   ,             SOL00100
     &  MEMKG    , MEMK     , MCOND    , KGCOND   , UNBAL    ,             SOL00110
     &  INTEG    , THETA    , ELSTIC   , IPRNT    , IWRITE   , GAT    ,   SOL00120
     &  MAXACC   , SLMASS   , TO       , DT       , TF       , GVT    ,   SOL00130
     &  GRAV     , FDAMP    , FMASS    , IZDOSP   ,IZDLOA    , GDT    ,   SOL00140
     &  DISP     , VEL      , ACC      , LOAD     , AN       , DSPMAX,    SOL00150
     &  DDISP    , DVEL     , DACC     , DLOAD    , BN       , VELMAX,    SOL00160
     &  STIFF    , MASS     , MASSSL   , KG       , SBAR     , ACCMAX,    SOL00170
     &  MD       , MDMS     , MDMSL    , MDKG     , MDS      , RCTMAX,    SOL00180
     &  DAMP     , PBAR     , MCOS     , AGTX     ,           DLOAD2,     SOL00190
     &  AGTY     , AGTZ     , AINP     , FUB      , WORK     , SUMRC ,    SOL00200
     &  ID       , IDOF     , CNST     , JTFLG    , COORD    , TFDAMP,    SOL00210
     &  COSINE   , JTCOS    , DFDAMP   ,                                   SOL00220
     &  EIE,ESE,PSE,EKE,EDD,                                               SOL00230
     &   ZERO    , FG       , DFG      )                                   SOL00240
C                                                                          SOL00250
      DOUBLE PRECISION  EIE,ESE,PSE,EDD,EKE                                SOL00260
      CHARACTER*(*) TITLE(2),STEPID                                        SOL00270
      LOGICAL PRINT,DSPFLG,HEAD,REPEAT,SLMASS,AXIAL,TEST,NEWKOV,BTEST      SOL00280
      LOGICAL NEWKG,NEWK,ELSTIC,MCOND,KGCOND,UNBAL,MCOND2,KGCOND2,WRITE    SOL00290
      LOGICAL ABORT,BUG5,BUG4,BUG5,BUG6,BUG11                              SOL00300
      CHARACTER*80 NAME                                                    SOL00310
      CHARACTER*20 INTEG                                                   SOL00320
      REAL LOAD,MASS,MASSSL,XG,MCOS,KE                                     SOL00330
      INTEGER FDAMP,FMASS                                                  SOL00340
      INTEGER ITLCPU,CPUREN,CPUELA                                         SOL00350
      DIMENSION ZERO(L6), DLOAD2(L6), FG(L6), DFG(L6)                      SOL00360
      DIMENSION SUMRC(6),DSUMRC(6),SUMFD(6),DSUMFD(6),SUMUB(6)             SOL00370
      DIMENSION TFDAMP(L6),DFDAMP(L6)                                      SOL00380
      DIMENSION DSPMAX(L6),VELMAX(L6),ACCMAX(L6),RCTMAX(L6+6)              SOL00390
      DIMENSION  GD(5), GV(5), GA(5)                                       SOL00400
      DIMENSION GDT(5),GVT(5),GAT(5), TSUM(6)                              SOL00410
      DIMENSION RINPUT(100)                                                SOL00420
      DIMENSION STIFF(IMD) ,MASS(IMDMS),  KG(IMDKG),SBAR(IMDS),            SOL00430
     &          MASSSL(IMDMSL)                                             SOL00440
      DIMENSION   MD(L6+1),MDMS(L6+1 ),MDKG(L6+1 ), MDS(L6+1),             SOL00450
     &          MDMSL(L6+1)                                                SOL00460
      DIMENSION  LOAD(L6), DISP(L6), VEL(L6), ACC(L6), AN(L6)              SOL00470
      DIMENSION DLOAD(L6),DDISP(L6),DVEL(L6),DACC(L6), BN(L6)              SOL00480
      DIMENSION   FUB(L6), PBAR(L6),MCOS(L6,5),DAMP(5)                     SOL00490
      DIMENSION WORK(2*L6)                                                 SOL00500
      DIMENSION AGTX(MAXACC),AGTY(MAXACC),AGTZ(MAXACC),AINP(NAINP)         SOL00510
      DIMENSION COORD(MAXNOD,5),ID(MAXNOD),                                SOL00520
     &          IDOF(MAXNOD,6) ,COSINE(3,5,MCOS),CNST(MAXNOD,6),           SOL00530
     &          JTFLG(MAXNOD),JTCOS(MAXNOD)                                SOL00540
      SUMUB(1)=0                                                           SOL00550
      SUMUB(2)=0                                                           SOL00560
      SUMUB(3)=0                                                           SOL00570
      SUMUB(4)=0                                                           SOL00580
      SUMUB(5)=0                                                           SOL00590
      SUMUB(6)=0                                                           SOL00600
      SUMFOR=0                                                             SOL00610
      SUMMOM=0                                                             SOL00620
      SUMMFO=0                                                             SOL00630
      SUMMMO=0                                                             SOL00640
C============================================                               SOL00650
      BUG5 =BTEST(IBUG, 5)                                                 SOL00660
      BUG4 =BTEST(IBUG, 4)                                                 SOL00670
      BUG5 =BTEST(IBUG, 5)                                                 SOL00680
      BUG6 =BTEST(IBUG, 6)                                                 SOL00690
      BUG11=BTEST(IBUG,11)                                                 SOL00700
C-------------------------------------------                               SOL00710
        IF (BUG5) THEN                                                     SOL00720
          CALL LMATRX(MASS,MDMS,NDOF,IMDMS,'MASS MATRIX DUMP=7')           SOL00730
        ENDIF                                                              SOL00740
C                                                                          SOL00750
C===========================================                                SOL00760
C= INITIALIZE GROUND ACCELERATION CONSTANTS ===                            SOL00770
C===========================================                                SOL00780
      DO 91 I=1,5                                                          SOL00790
        GD(I)=0                                                            SOL00800
        GV(I)=0                                                            SOL00810
        GA(I)=0                                                            SOL00820
        GDT(I)=0                                                           SOL00830
        GVT(I)=0                                                           SOL00840
   91   GAT(I)=0                                                           SOL00850
      ACCOLD=0                                                             SOL00860
      XGACC=0                                                              SOL00870
      YGACC=0                                                              SOL00880
      ZGACC=0                                                              SOL00890
C                                                                          SOL00900
C================================                                           SOL00910
C= INITIALIZE ENERGY CONSTANTS ===                                         SOL00920
C================================                                           SOL00930
      EIE=0                                                                SOL00940
      ESE=0                                                                SOL00950
      PSE=0                                                                SOL00960
      EKE=0                                                                SOL00970
      EDD=0                                                                SOL00980
      EIEM =0                                                              SOL00990
      ESEM =0                                                              SOL01000
      PSEM =0                                                              SOL01010
      EKEM =0                                                              SOL01020
      EDDM =0                                                              SOL01030
      DO 92 I=1,6                                                          SOL01040
        SUMRC(I)=0                                                         SOL01050
   92   SUMFD(I)=0                                                         SOL01060
C                                                                          SOL01070
C==================================                                        SOL01080
C= SET GEOMETRIC STIFFNESS FLAGS ===                                       SOL01090
C==================================                                        SOL01100
      MEMKG=.TRUE.                                                         SOL01110
      IF (KGLOAD.EQ.0 .OR. KGTYPE.EQ.0 .OR. KGFORM.EQ.0) MEMKG=.FALSE.     SOL01120
C                                                                          SOL01130
C=========================                                                 SOL01140
C= SET STIFFNESS FLAGS ===                                                 SOL01150
C=========================                                                 SOL01160
      MEMK = .TRUE.                                                        SOL01170
      KSAME= 0                                                             SOL01180
C                                                                          SOL01190
C       ---- INPUT INITIAL VALUES FOR DISPL, VELOC, ACCEL                  SOL01200
      CALL INTOVA('BUGS',NDOF,MAXNOD,NNODE,TITLE,IDOF,                     SOL01210
     &  ID,JTFLG,DISP  ,VEL   ,ACC   ,LOAD  )                              SOL01220
C                                                                          SOL01230
C---- INITILIZE TIME TO CALC INITIAL GROUND ACCELERATION                   SOL01240
      T=TO                                                                 SOL01250
      TL=T-DT                                                              SOL01260
C                                                                          SOL01270
C---- READ AND INITILIZE DYNAMIC GROUND ACCELERATIONS                      SOL01280
      IF (SLMASS) THEN                                                     SOL01290
                                                                           SOL01300
                                                                           SOL01310
```

```
         CALL DVLOAC1,BUGS,MAXACC,NDOF,NAINP,T,TL,                      SOL01320
     &   ACCELZ,TOACC,DTACC,GRAV,                                       SOL01330
     &   MAXNOD,NNODE,COSINE,IDOF,JTFLG,JTCOS,NCOS,                     SOL01340
     &   AGTX  ,AGTY  ,AGTZ  ,MASSSL,MOMSSL,INOMSL,                     SOL01350
     &   AINP  ,MCOS  ,OLOAD ,GA    ,GAT   )                            SOL01360
       ELSE                                                             SOL01370
         CALL DVLOAC1,BUGS,MAXACC,NDOF,NAINP,T,TL,                      SOL01380
     &   ACCELZ,TOACC,DTACC,GRAV,                                       SOL01390
     &   MAXNOD,NNODE,COSINE,IDOF,JTFLG,JTCOS,NCOS,                     SOL01400
     &   AGTX  ,AGTY  ,AGTZ  ,MASS  ,MOMS  ,INOMS,                      SOL01410
     &   AINP  ,MCOS  ,OLOAD ,GA    ,GAT   )                            SOL01420
       ENDIF                                                            SOL01430
C                                                                       SOL01440
C---- GET INITIAL DYNAMIC LOAD                                          SOL01450
      IF (SLMASS) THEN                                                  SOL01460
         CALL DVLOAC2,BUGS,MAXACC,NDOF,NAINP,T,TL,                      SOL01470
     &   ACCELZ,TOACC,DTACC,GRAV,                                       SOL01480
     &   MAXNOD,NNODE,COSINE,IDOF,JTFLG,JTCOS,NCOS,                     SOL01490
     &   AGTX  ,AGTY  ,AGTZ  ,MASSSL,MOMSSL,INOMSL,                     SOL01500
     &   AINP  ,MCOS  , LOAD ,GA    ,GAT   )                            SOL01510
       ELSE                                                             SOL01520
         CALL DVLOAC2,BUGS,MAXACC,NDOF,NAINP,T,TL,                      SOL01530
     &   ACCELZ,TOACC,DTACC,GRAV,                                       SOL01540
     &   MAXNOD,NNODE,COSINE,IDOF,JTFLG,JTCOS,NCOS,                     SOL01550
     &   AGTX  ,AGTY  ,AGTZ  ,MASS  ,MOMS  ,INOMS,                      SOL01560
     &   AINP  ,MCOS  , LOAD ,GA    ,GAT   )                            SOL01570
       ENDIF                                                            SOL01580
C                                                                       SOL01590
C---- INITILIZE INTEGRATION ROUTINES                                    SOL01600
C                                                                       SOL01610
      IF (INTEGO.EQ.1) THEN                                             SOL01620
        CALL LINACC(1,BUG*,DT,TO,TF,T,1.00,NEWKDV,                      SOL01630
     &    L1,L2,L3,L4,L5,L6,INOMS,IMD,IMDS,IMDKG,NDOF,NFREE,            SOL01640
     &    KGLOAD,KGFORM,                                                SOL01650
     &     ACC, VEL  , DISP  , LOAD  ,                                  SOL01660
     &     DACC  , DVEL  , DDISP , DLOAD2,WORK,                         SOL01670
     &     MASS  ,MOMS   , DAMP  , STIFF ,MD      ,                    SOL01680
     &     KG    ,MDKG   , FZ    ,GRAV,                                 SOL01690
     &     SBAR  ,MDS    , AN    , BN    , PBAR  ,                      SOL01700
     &    FDAMP,FMASS,C1,C2,C3,C4,C5,C6,C7,ALPHA,BETA,C10,C11,ABORT)     SOL01710
       ELSE IF (INTEGO.EQ.2) THEN                                       SOL01720
        CALL AVEACC(1,BUG*,DT,TO,TF,T,THETA,NEWKDV,                     SOL01730
     &    L1,L2,L3,L4,L5,L6,INOMS,IMD,IMDS,IMDKG,NDOF,NFREE,            SOL01740
     &    KGLOAD,KGFORM,                                                SOL01750
     &     ACC, VEL  , DISP  , LOAD  ,                                  SOL01760
     &     DACC  , DVEL  , DDISP , DLOAD2,WORK,                         SOL01770
     &     MASS  ,MOMS   , DAMP  , STIFF ,MD      ,                    SOL01780
     &     KG    ,MDKG   , FZ    ,GRAV,                                 SOL01790
     &     SBAR  ,MDS    , AN    , BN    , PBAR  ,                      SOL01800
     &    FDAMP,FMASS,C1,C2,C3,C4,C5,C6,C7,ALPHA,BETA,C10,C11,ABORT)     SOL01810
       ENDIF                                                            SOL01920
       IF (ABORT) GO TO 5000                                            SOL01950
C                                                                       SOL01940
C=======================================                                SOL01950
C=WRITE INITIAL DATA TO OUTPUT FILE ===                                 SOL01960
C=======================================                                SOL01970
       IF (IWRITE.GT.0) THEN                                            SOL01980
         IOPT=2                                                         SOL01990
         CALL DMPOAT(IOPT,IWRITE,TO,DT)                                 SOL02000
       ENDIF                                                            SOL02010
C                                                                       SOL02020
C=======================================================================SOL02030
C= LOOP FOR EACH TIME STEP , ISTEP=CURRENT STEP # ======================SOL02040
C=======================================================================SOL02050
C... NSTEP = NUMBER OF TIME STEPS                                       SOL02060
C... ISTEP = TIME STEP NUMBER                                           SOL02070
C... T     = CURRENT TIME                                               SOL02080
C... TL    = TIME AT PREVIOUS TIME STEP                                 SOL02090
C... TO    = INITIAL TIME                                               SOL02100
C... DT    = TIME INCRIMENT                                             SOL02110
C... TL    = FINAL TIME                                                 SOL02120
C                                                                       SOL02130
       MCND2 = MCOND                                                    SOL02140
       NSTEP = (TF-TO)/DT                                               SOL02150
       T     = TO                                                       SOL02160
       DO 500 ISTEP=1,NSTEP                                             SOL02170
         T =TO+ISTEP*DT                                                 SOL02180
         TL=T - DT                                                      SOL02190
         WRITE (STEPID,90) ISTEP,T                                      SOL02200
   90    FORMAT ('STEP:',I6,',  TIME:',1P,G13.5)                        SOL02220
C                                                                       SOL02230
C=========================                                             SOL02240
C= CALC INCRIMENTAL LOADS  =                                            SOL02250
C=========================                                             SOL02260
       I2=2                                                             SOL02270
       CALL DVLOAC2,BUGS,MAXACC,NDOF,NAINP,T,TL,                        SOL02280
     &    ACCELZ,TOACC,DTACC,GRAV,                                      SOL02290
     &   MAXNOD,NNODE,COSINE,IDOF,JTFLG,JTCOS,NCOS,                     SOL02300
     &   AGTX  ,AGTY  ,AGTZ  ,MASS  ,MOMS  ,INOMS,                      SOL02310
     &   AINP  ,MCOS  ,OLOAD ,GA    ,GAT   )                            SOL02320
       FZ    =ACCELZ                                                    SOL02330
C                                                                       SOL02340
C---- ADD UNBALANCED LOAD'S TO THE LOAD VECTOR. ---> STORE AS OLOAD2     SOL02350
C     STORE OLD UNBALANCED LOADS AT REACTIONS IN OLOAD                  SOL02360
       IF (UNBAL) THEN                                                  SOL02370
         DO 195 I=1,L*                                                  SOL02380
           OLOAD2(I)=OLOAD(I)+FUB(I)                                    SOL02390
  195      FUB(I)=0.0                                                   SOL02400
         DO 196 I=L5,L6                                                 SOL02410
           OLOAD2(I)=-FUB(I)                                            SOL02420
C***                                                                    SOL02430
C***   OMIT UNBALANCED LOAD FROM REACTIONS...                           SOL02440
C***   UNBALANCED LOADS ADDED TO THE FRAME ARE RESISTED BY              SOL02450
C***   1) INERTIAL FORCES                                               SOL02460
C***   2) DAMPING FORCES                                                SOL02470
C***   3) MEMBER FORCES                                                 SOL02480
C***   SUBTRACTING UNBALANCED FORCES FROM REACTIONS DUE TO MEMBER FORCES SOL02490
C***   NEGLECTS THE REACTIONS DUE TO INERTIAL AND DAMPING FORCES.       SOL02500
C***           OLOAD(I)= -FUB(I)                                        SOL02510
C***                                                                    SOL02520
  196      FUB(I)=0.0                                                   SOL02530
       ELSE                                                             SOL02540
         DO 197 I=L1,L6                                                 SOL02550
  197      OLOAD2(I)=OLOAD(I)                                           SOL02560
       ENDIF                                                            SOL02570
C                                                                       SOL02580
C=========================                                             SOL02590
C= FORMULATE STIFFNESS ====                                             SOL02600
C=========================                                             SOL02610
C  FORMULATE THE STIFFNESS IF THE FLAG NEWK IS TRUE. THIS FLAG IS       SOL02620
C       INITIALIZED ABOVE, AND SET WHEN THE ELEMENT FORCES ARE CALC.    SOL02630
C       THE STIFFNESS IS ALSO FORMULATED IF KG IS TO BE CALCULATED      SOL02640
C       AND CONDENSATION IS TO BE PREFORMED, (THE STIFFNESS IS NEEDED   SOL02650
C       FOR THE CONDENSATION PROCEDURE...)                              SOL02660
       REPEAT=.TRUE.                                                    SOL02670
       KSAME=0                                                          SOL02680
   100 IF (NEWK) THEN                                                   SOL02690
         I1=1                                                           SOL02700
         CALL FORMK(I1)                                                 SOL02710
C------ CALL LMATRX(STIFF,MD,L6,IMD,'GLOBAL STIFFNESS MATRIX')         SOL02720
         IF (BUGS) THEN                                                 SOL02730
           CALL LMATRX(STIFF,MD,NDOF,IMD,'GLOBAL STIFFNESS DUMP=7')     SOL02740
         ENDIF                                                          SOL02750
       ENDIF                                                            SOL02760
C                                                                       SOL02770
C=============================                                         SOL02780
C= FORM GEOMETRIC STIFFNESS ===                                         SOL02790
C=============================                                         SOL02800
C  FORMULATE THE GEOMETRIC STIFFNESS IF THE FLAG NEWK IS TRUE, THIS     SOL02810
C       FLAG INITIALIZED ABOVE.                                         SOL02820
C       IF KGDATA(1)=1, THE GEOMETRIC STIFFNESS IS FORMED ONCE, AND     SOL02830
C                       MULTIPLIED BY A SCALER TO ACCOUNT FOR CHANGES   SOL02840
C                       IN THE TOTAL VERTICAL LOAD. THUS THE GEOMETRIC  SOL02850
C                       STIFFNESS ONLY NEEDS TO BE FORMED ONCE.         SOL02860
C       IF KGDATA(1)=2, THE GEOMETRIC STIFFNESS IS BASED ON THE AXIAL   SOL02870
C                       FORCE IN THE ELEMENTS, AND IS RECALCULATED EACH SOL02880
C                       TIME THE AXIAL FORCE CHANGES.                   SOL02890
C       IF KGDATA(5)=1, THE GEOMETRIC STIFFNESS IS FORMED WITH THE      SOL02900
C                       STIFFNESS, THUS A SEPERATE CALL TO FORM IS      SOL02910
C                       NOT NEEDED                                      SOL02920
C       IF KGDATA(5)=2, A SEPERATE GEOMETRIC STIFFNESS MATRIX IS FORMED SOL02930
       IF (NEWKG) THEN                                                  SOL02940
C                                                                       SOL02950
C           FORMULATE SEPERATE GEOMETRIC STIFFNESS MATRIX               SOL02960
            IF (KGFORM.EQ.2) THEN                                       SOL02970
              FZ    =0                                                  SOL02980
              I2=2                                                      SOL02990
              CALL FORMK(I2)                                            SOL03000
              KGCND2=KGCOND                                             SOL03010
              IF (BUGS) THEN                                            SOL03020
                CALL LMATRX(KG,MDKG,NDOF,IMDKG,'KG MATRIX DUMP=7')      SOL03030
              ENDIF                                                     SOL03040
            ELSE                                                        SOL03050
              KGCND2=.FALSE.                                            SOL03060
            ENDIF                                                       SOL03070
C                                                                       SOL03080
       ENDIF                                                            SOL03090
C                                                                       SOL03100
C=========================                                             SOL03110
C= CONDENSE OUT FREE DISPL===                                           SOL03120
C=========================                                             SOL03130
C... IF MCOND IS TRUE THE MASS MATRIX IS ALSO REDUCED BY GUYAN REDUCTIONSOL03140
C... IF KGCND2 IS TRUE THEN KG IS ALSO REDUCED                          SOL03150
       IF (MCOND.GT.0 .AND. NEWK ) THEN                                 SOL03160
         CALL MSOLL(1,BUGS,L*,IMD,NDOF,1,L1,L2,                         SOL03170
     &              MD     ,STIFF ,DLOAD2,WORK,                        SOL03180
     &              MCND2,MASS   ,MOMS   ,INOMS,                       SOL03190
     &             KGCND2,KG     ,MDKG   ,IMDKG,ABORT)                 SOL03200
         IF (ABORT) GO TO 5000                                          SOL03210
         MCND2 =.FALSE.                                                 SOL03220
         KGCND2=.FALSE.                                                 SOL03230
       ELSE IF (MCOND.GT.0) THEN                                        SOL03240
C        REDUCE THE DYNAMIC LOAD MATRIX IF OLD STIFFNESS IS USED...     SOL03250
         CALL MSOLL(2,BUGS,L*,IMD,NDOF,1,L1,L2,                         SOL03260
     &              MD     ,STIFF ,DLOAD2,WORK,                        SOL03270
     &              MCND2,MASS   ,MOMS   ,INOMS,                       SOL03280
     &              KGCND2,KG     ,MDKG   ,IMDKG,ABORT)                SOL03290
         IF (ABORT) GO TO 5000                                          SOL03300
       ENDIF                                                            SOL03310
C                                                                       SOL03320
C=======================================================               SOL03330
C= DETERMINE IF A NEW DYNAMIC STIFFNESS NEEDES TO BE CALC ===           SOL03340
C=======================================================               SOL03350
       IF (NEWK .OR. NEWKG .OR. (KGLOAD.EQ.1 .AND. ACCELZ.EQ.ACCOLD))THENSOL03360
         NEWKDV=.TRUE.                                                  SOL03370
         ACCOLD=ACCELZ                                                  SOL03380
       ELSE                                                             SOL03390
         NEWKDV=.FALSE.                                                 SOL03400
       ENDIF                                                            SOL03410
C                                                                       SOL03420
C                                                                       SOL03430
C=============================================                          SOL03440
C= SOLVE DYNAMIC EQUATION FOR INCRIMENTAL DISPL ==                      SOL03450
C=============================================                          SOL03460
       I2=2                                                             SOL03470
       IF (INTEGO.EQ.1) THEN                                            SOL03480
        CALL LINACC(I2,BUG*,DT,TO,TF,T,1.00,NEWKDV,                     SOL03490
     &    L1,L2,L3,L4,L5,L6,INOMS,IMD,IMDS,IMDKG,NDOF,NFREE,            SOL03500
     &    KGLOAD,KGFORM,                                                SOL03510
     &     ACC, VEL  , DISP  , LOAD  ,                                  SOL03520
     &     DACC  , DVEL  , DDISP , DLOAD2,WORK,                         SOL03530
     &     MASS  ,MOMS   , DAMP  , STIFF ,MD      ,                    SOL03540
     &     KG    ,MDKG   , FZ    ,GRAV,                                 SOL03550
     &     SBAR  ,MDS    , AN    , BN    , PBAR  ,                      SOL03560
     &    FDAMP,FMASS,C1,C2,C3,C4,C5,C6,C7,ALPHA,BETA,C10,C11,ABORT)     SOL03570
       ELSE IF (INTEGO.EQ.2) THEN                                       SOL03580
        CALL AVEACC(I2,BUG*,DT,TO,TF,T,THETA,NEWKDV,                    SOL03590
     &    L1,L2,L3,L4,L5,L6,INOMS,IMD,IMDS,IMDKG,NDOF,NFREE,            SOL03600
     &    KGLOAD,KGFORM,                                                SOL03610
     &     ACC, VEL  , DISP  , LOAD  ,                                  SOL03620
     &     DACC  , DVEL  , DDISP , DLOAD2,WORK,                         SOL03630
     &     MASS  ,MOMS   , DAMP  , STIFF ,MD      ,                    SOL03640
     &     KG    ,MDKG   , FZ    ,GRAV,                                 SOL03650
     &     SBAR  ,MDS    , AN    , BN    , PBAR  ,                      SOL03660
     &    FDAMP,FMASS,C1,C2,C3,C4,C5,C6,C7,ALPHA,BETA,C10,C11,ABORT)     SOL03670
       ENDIF                                                            SOL03680
       IF (ABORT) GO TO 5000                                            SOL03690
C                                                                       SOL03700
C================================================                       SOL03710
C= BACK SUBSTIUTION TO SOLVE FOR NON-DYNAMIC DOF ===                    SOL03720
C================================================                       SOL03730
C  DISP_C = T * DISP_F                                                  SOL03740
C  DISP_C = INV(K_CC) * (-K_CF)*DISP(F)                                 SOL03750
C  SUBRTN REA_CTN (IOPT=1):                                             SOL03760
C     PREFORMS THE MULTIPICATION DISP_C=(-K_CF)*DISP_F                  SOL03770
C  SUBRTN MSOLL  (IOPT=5):                                              SOL03780
C     PREFORMS THE BACK SUBSTITUTION TO EQUILIVENT TO                   SOL03790
C     DISP_C=INV(K_CC)*DISP_C                                           SOL03800
C                                                                       SOL03810
       ONE=1.00                                                         SOL03820
       IF (NCOND.GT.0) THEN                                             SOL03830
             I3=3                                                       SOL03840
             I1=1                                                       SOL03850
C------------------ DISPLACEMENTS ---------------------------------     SOL03860
           IF (BUGS) WRITE(6,*)'*---- CALC DISPL'                       SOL03870
           CALL REACTN(I1,BUGS,NNODE,L1,L2,L5,L4,,DDISP,DDISP,          SOL03880
     &          MD     ,STIFF ,MDK,NDOF),NDOF,1.ONE,                  SOL03890
     &         MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,             SOL03900
```

```
     &          NCOS,COSINE,JTCOS,JTFLG,DSUMRC)
           CALL MSOLL(IS,BUGS,L4,IMD,NDOF,I1,L1,L2,MD ,STIFF,
     &          DDISP ,WORK,
     &          .FALSE.,MASS  ,MDMS   ,1,
     &          .FALSE.,KG    ,MDKG   ,1,ABORT)
           IF (ABORT) GO TO 5000
C----------------- VELOCITIES ------------------------------------------
           IF (BUGS) WRITE(6,*)'*---- CALC VELOC'
            CALL REACTNK11,BUGS,NNODE,L1,L2,L5,L4,DVEL,DVEL,
     &          MD     ,STIFF ,MDK(NDOF),NDOF,1,ONE,
     &         MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &          NCOS,COSINE,JTCOS,JTFLG,DSUMRC)
           CALL MSOLL(IS,BUGS,L4,IMD,NDOF,I1,L1,L2,MD ,STIFF,
     &          DVEL  ,WORK,
     &          .FALSE.,MASS  ,MDMS   ,1,
     &          .FALSE.,KG    ,MDKG   ,1,ABORT)
           IF (ABORT) GO TO 5000
C----------------- ACCELERATION ----------------------------------------
           IF (BUGS) WRITE(6,*)'*---- CALC ACCEL'
           CALL REACTNK11,BUGS,NNODE,L1,L2,L5,L4,DACC,DACC,
     &       MD     ,STIFF ,MDK(NDOF),NDOF,1,ONE,
     &       MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &       NCOS,COSINE,JTCOS,JTFLG,DSUMRC)
           CALL MSOLL(IS,BUGS,L4,IMD,NDOF,I1,L1,L2,MD ,STIFF,
     &          DACC  ,WORK,
     &          .FALSE.,MASS  ,MDMS   ,1,
     &          .FALSE.,KG    ,MDKG   ,1,ABORT)
           IF (ABORT) GO TO 5000
      ENDIF
C
C=======================================
C= CALCULATE INCRIMENTAL MEMBER FORCES ===
C=======================================
C ELELIB CALLS THE INDIVIDUAL ELEMENT LIBRARIES, EACH OF WHICH DETERMINS
C  1) DID THE STIFFNESS CHANGE?  IF SO: NEWK=.TRUE.
C  2) SHOULD THE INCRIMENTAL DISPLACEMENT BE RECALCULATED?
C     IF SO; KSAME=KSAME+___ (A VALUE DEPENDING ON ELEMENT TYPE)
C
      IOPT=4
      LSTTYP=0
      PRINT=.FALSE.
      HEAD =.FALSE.
      NEWK=.FALSE.
      NELMT=NELMT
      DO 150 IELNO=1,NELMT
  150   CALL ELELIB
     &   (IOPT  ,LSTTYP ,PRINT ,IREL,HEAD,   NAME ,EESE,EPSE,DAMAGE,
     &    IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDLOA,MSTOR)
C
C=============================================
C= ZERO STIFFNESS CHANGE FLAGS IF ELASTIC ===
C=============================================
      IF (ELSTIC) THEN
         NEWK=.FALSE.
         KSAME=0
      ENDIF
C
C=============================================
C= DETERMINE IF KG IS TO BE UPDATED BASED ON  ===
C=     CHANGING ELEMENT AXIAL FORCE          ===
C=============================================
           IF (KGLOAD.EQ.2) THEN
              NEWKG=.TRUE.
              IF (NCOMD.GT.0) NEWK=.TRUE.
           ELSE
              NEWKG=.FALSE.
           ENDIF
C
C============================
C= REPEAT STEP IF NESSARY ===
C============================
C  IF AN ELEMENT STIFFNESS CHANGE DICTATED THAT THE STEP BE REANALIZED,
C     REFORMULATE THE STIFFNESS AND REANALIZE
C     THE VARIABLE REPEAT IS SET UP TO SUPRESS THE REANALIZING OF A STEP
C     CURRENTLY, EACH STEP MAY ONLY BE REANALIZED ONCE. THIS PREVENTS
C     AN ENDLESS LOOP.
C
      IF (KSAME.NE.0 .AND. REPEAT) THEN
          REPEAT=.FALSE.
          GO TO 100
      ENDIF
C
C=========================
C= SOLVE FOR REACTIONS ===
C=========================
C REACTIONS ARE DUE TO INTERNAL MEMBER FORCES
C
      IF (BUGS) WRITE(6,*)'*---- CALC REACT=K*X'
      CALL REACTNK2,BUGS,NNODE,L1,L4,L5,L6,DLOAD ,DDISP ,
     &  MD     ,STIFF ,MDK(NDOF),NDOF,1,1,0,
     &  MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &          NCOS,COSINE,JTCOS,JTFLG,DSUMRC)
C
C---- GET SUMMATION OF REACTIONS
      CALL REACTNK3,BUGS,NNODE,L1,L4,L5,L6,DLOAD ,DDISP,
     &       MDKG   ,KG    ,MDKG(NDOF),NDOF,1,1,0,
     &      MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &       NCOS,COSINE,JTCOS,JTFLG,DSUMRC)
C
C---- GET SUMMATION OF REACTIONS DUE TO UNBALANCED LOAD
      IF (.NOT.ELSTIC)
     & CALL REACTNK5,BUGS,NNODE,L1,L4,L5,L6,DLOAD ,DDISP,
     &       MDKG   ,KG    ,MDKG(NDOF),NDOF,1,1,0,
     &      MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &       NCOS,COSINE,JTCOS,JTFLG,SUMUB)
C
C=================================================
C= SOLVE FOR INCRIMENTAL GEOMETRIC STIFFNESS FORCE, DFG ===
C=================================================
C
C------- INCRIMENTAL GEOMETRIC STIFFNESS FORCE
      IF (KGFORM.EQ.2) THEN
         IF (BUGS) WRITE(6,*) '*------ INCRIMENTAL FORCE DUE TO KG '
         CALL TMPLVC(BUGS,.TRUE.,IMD,L4,L5,L9,L5,L4,
     &        MDKG,KG,DFG,DDISP)
C        SCALE FORCE TO INCLUDE GROUND ACCELERATION ...
         DO 159 I=L5,L4
  159       DFG(I)= (1+FZ)*DFG(I)
      ENDIF
C
C=================================================
C= SOLVE FOR INCRIMENTAL DAMPING FORCE,  DFDAMP ===
C=================================================
C
      IF (BETA.NE.0) THEN
C------- INCRIMENTAL DAMPING FORCE DUE TO STIFFNESS ...
         IF (BUGS) WRITE(6,*)
     &    ' *------ DAMPING FORCE/REACT DUE TO STIFFNESS'
         CALL TMPLVC(BUGS,.TRUE.,IMD,L6,L5,L4,L5,L4,
     &        MD,STIFF,DFDAMP,DVEL)
         CALL TMPLVC(BUGS,.TRUE.,IMD,L6,L5,L6,L1,L4,
     &        MD,STIFF,DFDAMP,DVEL)
C------- INCRIMENTAL DAMPING FORCE DUE TO KG  ...
         IF (KGFORM.EQ.2) THEN
            IF (BUGS) WRITE(6,*)
     &       ' *------ DAMPING FORCE/REACT DUE TO KG '
            CALL TMPLVC(BUGS,.TRUE.,IMD,L6,L5,L4,L5,L4,
     &           MDKG,KG,WORK,DVEL)
            CALL TMPLVC(BUGS,.TRUE.,IMD,L6,L5,L6,L1,L4,
     &           MDKG,KG,WORK,DVEL)
            DO 158 I=L5,L6
  158          DFDAMP(I)=DFDAMP(I)-WORK(I)
         ENDIF
C---------- CALC SUM OF STIFFNESS PROPORTIONAL DAMPING REACTIONS
         CALL REACTNK5,BUGS,NNODE,L1,L4,L5,L6,DFDAMP,DVEL ,
     &        MDMS   ,MASS  ,MDMS(NDOF),NDOF,1,1.00,
     &       MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &        NCOS,COSINE,JTCOS,JTFLG,DSUMFD)
C------- MULTIPLY BY BETA
         DO 160 I=1,6
           DSUMFD(I)=DSUMFD(I)*BETA
  160      IF (BUGS) WRITE (6,*) 'I,DSUMFD(I),BETA',I,DSUMFD(I),BETA
         DO 161 I=L5,L4
  161      DFDAMP(I)=DFDAMP(I)*BETA
      ENDIF
C
      IF (ALPHA.NE.0) THEN
C------- INCRIMENTAL DAMPING FORCE DUE TO MASS ...
         IF (BUGS) WRITE(6,*) '*------ DAMPING FORCE DUE TO MASS '
         DO 162 I=L1,L6
  162      WORK(I)=0.00
         CALL TMPLVC(BUGS,.FALSE.,IMD,L4,L3,L4,L3,L4,
     &        MDMS,MASS,WORK,DVEL)
C---------- CALC SUM OF FD
         CALL REACTNK3,BUGS,NNODE,L1,L2,L5,L4,WORK,DVEL ,
     &       MDMS   ,MASS  ,MDMS(NDOF),NDOF,1,1.00,
     &       MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &        NCOS,COSINE,JTCOS,JTFLG,TSUM)
C------- MULTIPLY BY ALPHA ADD TO PREVIOUS SUM ...
         IF (BETA.NE.0) THEN
           DO 165 I=L5,L6
  165        DFDAMP(I)=DFDAMP(I)+ALPHA*WORK(I)
           DO 167 I=1,6
  167        DSUMFD(I)=DSUMFD(I)+TSUM(I)*ALPHA
         ELSE
           DO 168 I=L5,L4
  168        DFDAMP(I)=          ALPHA*WORK(I)
           DO 166 I=1,6
  166        DSUMFD(I)=+TSUM(I)*ALPHA
         ENDIF
         IF (BUGS) WRITE (6,*) 'DSUMFD:',DSUMFD
      ENDIF
      IF (BUGS) THEN
         DO 169 I=L5,L6
  169    WRITE (6,*) 'I',I,'DFDAMP',DFDAMP(I),'FDAMP',FDAMP(I)
      ENDIF
C
C=======================================
C= CALC AND ENERGIES AND DUCTILITIES  ===
C=======================================
      WRITE=  MOD(ISTEP,IPRINT).EQ.0
      HEAD = WRITE
C
      IF (HEAD) WRITE (6,250) TITLE(1),TITLE(2),' ',ISTEP,STEPID
      IF (SLMASS) THEN
C------------ ENERGY FORMULATION WITH SPECIAL LOADING MASS ---
C             BASED ON APPLIED LOADING
       CALL ENERGY
     & (L1,L2,L5,L4,L5,L6,NDOF,MAXNOD,NCOS,BUG11,.FALSE.,
     &  IMD,IMDMS,IMDKG,MD,MDMS,MDKG,.TRUE.,WRITE,ABORT,
     &  STIFF,KG,MASS,
     &  EIE,ESE,PSE,EKE,EDD,
     &  ALPHA,BETA,KGFORM,NELMT,NNODE,DT,SUMUB,
     &   SUMRC,DSUMRC, SUMFD,DSUMFD,TFDAMP,DFDAMP, FZ,
     &  ZERO,GV,GD,ZERO,GVT,GDT,  WORK,WORK(L6+1),FG,DFG,
     &  DACC,DVEL,DDISP,ACC,VEL,DISP, LOAD,DLOAD,
     &  ID      , IDOF     , CNST     , JTFLG    , COORD    ,
     &  COSINE  , JTCOS    )
      ELSE
C------------ ENERGY FORMULATION WITH/O SPECIAL LOADING MASS ---
C             BASED ON GROUND MOTION
       CALL ENERGY
     & (L1,L2,L5,L4,L5,L6,NDOF,MAXNOD,NCOS,BUG11,.FALSE.,
     &  IMD,IMDMS,IMDKG,MD,MDMS,MDKG,.TRUE.,WRITE,ABORT,
     &  STIFF,KG,MASS,
     &  EIE,ESE,PSE,EKE,EDD,
     &  ALPHA,BETA,KGFORM,NELMT,NNODE,DT,SUMUB,
     &   SUMRC,DSUMRC, SUMFD,DSUMFD,TFDAMP,DFDAMP, FZ,
     &  GA,GV,GD,GAT,GVT,GDT,  WORK,WORK(L6+1),FG,DFG,
     &  DACC,DVEL,DDISP,ACC,VEL,DISP, ZERO,ZERO,
     &  ID      , IDOF     , CNST     , JTFLG    , COORD    ,
     &  COSINE  , JTCOS    )
      ENDIF
      EIEM =AMAX1(SNGL(EIE),EIEM )
      ESEM =AMAX1(SNGL(ESE),ESEM )
      PSEM =AMAX1(SNGL(PSE),PSEM )
      EKEM =AMAX1(SNGL(EKE),ELEM )
      EDDM =AMAX1(SNGL(EDD),EDDM )
      IF (ABORT) GO TO 5000
C
C=================================================
C= GET TOTAL GLOBAL LOAD, DISPL, VELOC AND ACCEL ===
C=================================================
      IF (BUG4) WRITE (6,520)
      DO 510 I=1,NDOF
        ACC(I)   = DACC(I)   + ACC(I)
        VEL(I)   = DVEL(I)   + VEL(I)
        DISP(I)  = DDISP(I)  + DISP(I)
        LOAD(I)  = DLOAD(I)  + LOAD(I)
        TFDAMP(I)= DFDAMP(I) + TFDAMP(I)
        FG(I)    = DFG(I)    + FG(I)
        IF (ABS( ACC(I)).GT.ABS(ACCMAX(I))) ACCMAX(I)= ACC(I)
```

```
      IF (ABS( VEL(I)).GT.ABS(VELMAX(I))) VELMAX(I)= VEL(I)
      IF (ABS(DISP(I)).GT.ABS(DSPMAX(I))) DSPMAX(I)=DISP(I)
      IF (ABS(LOAD(I)).GT.ABS(RCTMAX(I))) RCTMAX(I)=LOAD(I)
      IF (BUG4)
     & WRITE (6,330) I,DDISP(I),DVEL(I),DACC(I),DLOAD(I),
     &                 DISP(I), VEL(I), ACC(I), LOAD(I)
      DACC(I)=0.00
      DVEL(I)=0.00
      DDISP(I)=0.00
      DFDAMP(I)=0.00
  310 DLOAD(I) =0.00
C
      DO 309 I=1,6
        SUMRC(I) = DSUMRC(I) + SUMRC(I)
  309   SUMFC(I) = DSUMFC(I) + SUMFC(I)
      SUMFOR=SQRT(SUMRC(1)**2+SUMRC(2)**2+SUMRC(3)**2)
      SUMMOM=SQRT(SUMRC(4)**2+SUMRC(5)**2+SUMRC(6)**2)
      SUMMFO=MAX(SUMFOR,SUMMFO)
      SUMMMO=MAX(SUMMOM,SUMMMO)
C
  320 FORMAT (' DOF',6X,'DD',12X,'DV',12X,'DA',12X,'DP',
     &          12X,' D',12X,' V',12X,' A',12X,' P')
  330 FORMAT (1X,I6,1P,5G14.6)
C
C=====================================================================
C= PRINT TOTAL GLOBAL DISPLACEMENTS, VELOCITIES, ACCELERATIONS ===
C=====================================================================
      IF (WRITE) THEN
           CALL JOIDSP(1,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &           IDOF,CNST  ,DISP  ,
     &           TITLE,'DISPLACEMENTS',STEPID..TRUE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
           CALL JOIDSP(1,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &           IDOF,CNST  ,VEL   ,
     &           TITLE,'VELOCITIES',STEPID,.TRUE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
           CALL JOIDSP(1,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &           IDOF,CNST  ,ACC   ,
     &           TITLE,'ACCELERATIONS',STEPID,.TRUE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
      ENDIF
C
C=================================
C= PRINT TOTAL GLOBAL REACTIONS ===
C=================================
      IOPT=5
      CALL REACTNK(IOPT,WRITE,NNODE,L1,L4,L5,L6,LOAD,ACC ,
     &     MOMS   ,MASS  ,MOMS(NDOF),NDOF,1,ONE,
     &     MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.TRUE.,
     &     NCOS,COSINE,JTCOS,JTFLG,SUMRC)
      IF (WRITE) WRITE (6,315) SUMFOR,SUMMOM
  315 FORMAT (/4X,'RESULTANT OF REACTIONS,   FORCE=',1P,G12.4,
     &                                    ' MOMENT=',G12.4)
      DO 311 I=1,6
        J=NDOF+I
        IF (ABS(SUMRC(I)).GT.ABS(RCTMAX(J))) RCTMAX(J)=SUMRC(I)
  311   CONTINUE
C
C=============================================
C= CALCULATE AND PRINT TOTAL MEMBER FORCES ===
C=============================================
      IOPT=5
      LSTTYP=0
      IF (WRITE) THEN
           PRINT=.TRUE.
           HEAD =.TRUE.
      ELSE
           PRINT=.FALSE.
           HEAD =.FALSE.
      ENDIF
      DO 300 IELNO=1,NELMT
  300   CALL ELELIB
     &  (IOPT  ,LSTTYP ,PRINT ,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDLOA,MSTOR)
C
C=============================================================
C= CALCULATE AND PRINT DUCTILITY AND EXCURSION RATIOS ===
C=============================================================
      IOPT=11
      LSTTYP=0
      IF (WRITE) THEN
        PRINT=.TRUE.
        HEAD =.TRUE.
        DO 312 IELNO=1,NELMT
  312     CALL ELELIB
     &    (IOPT  ,LSTTYP ,PRINT ,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &     IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDLOA,MSTOR)
      ENDIF
C
C=================================
C= WRITE DATA TO OUTPUT FILES  ===
C=================================
      IF (IWRITE.GT.0) THEN
        IF (MOD(ISTEP,IWRITE).EQ.0) THEN
          IOPT=2
          CALL DMPDAT(IOPT,IWRITE,TO,DT)
        ENDIF
      ENDIF
C
C=============================================
C= CHECK ELAPSED CPU TIME ON THE UNIBUS SYSTEM ===
C=============================================
      CALL CPUTIM(CPUREM,CPUELA)
      ITLCPU=CPUREM+CPUELA
      IF (ITLCPU.GT.0) THEN
        IF (CPUREM.LT.0.05*ITLCPU .OR. CPUREM.LT.5) THEN
          WRITE (6,4891) ITLCPU,CPUELA,CPUREM,ISTEP,T
          ABORT=.TRUE.
          GO TO 5000
 4891     FORMAT (//
     &    ' ==========================================================='/
     &    ' ==                                                       ==='/
     &    ' ==            TIME LIMIT IS EXCEEDED                     ==='/
     &    ' ==              SOLUTION IS ABORTED                      ==='/
     &    ' ==                                                       ==='/
     &    ' ==  CPU TIME AT BEGINING OR EXECUTION...',I7,' (SEC)    ==='/
     &    ' ==  ELAPSED CPU TIME.....',I7,' (SEC)                   ==='/
     &    ' ==  CPU TIME REMAINING.....',I7,' (SEC)                 ==='/
     &    ' ==  NUMBER OF STEPS COMPLETED.....',I7,'                ==='/
     &    ' ==  SOLUTION STOP TIME.....',1P,G14.7  ,'               ==='/
     &    ' ==                                                       ==='/
     &    ' ==========================================================')
        ENDIF
      ENDIF
C
C=============================================
C= PRINT MAXIMUM GLOBAL DISPLACEMENTS ===
C=============================================
  251 FORMAT ('1 STRUCTURE.....: ',A,/
     &        '  SOLUTION......: ',A,//
     &        '  MAXIMUM VALUES FOR ALL STEPS '/
     &        ' ==========================')
      STEPID=' '
      ISTEP = 0
      WRITE (6,251) TITLE(1),TITLE(2)
           CALL JOIDSP(2,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &           IDOF,CNST  ,DSPMAX,
     &           TITLE,'MAXIMUM DISPLACEMENTS',STEPID,.FALSE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
      WRITE (6,251) TITLE(1),TITLE(2)
           CALL JOIDSP(2,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &           IDOF,CNST  ,VELMAX  ,
     &           TITLE,'MAXIMUM VELOCITIES',STEPID,.FALSE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
      WRITE (6,251) TITLE(1),TITLE(2)
           CALL JOIDSP(2,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &           IDOF,CNST  ,ACCMAX  ,
     &           TITLE,'MAXIMUM ACCELERATIONS',STEPID,.FALSE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
C
C=================================
C= PRINT MAXIMUM GLOBAL REACTIONS ===
C=================================
      WRITE (6,251) TITLE(1),TITLE(2)
      DO 5100 I=1,6
        SUMRC(I)=RCTMAX(I+NDOF)
 5100 CONTINUE
      CALL REACTNK(4,.TRUE.,NNODE,L1,L4,L5,L6,RCTMAX,DSPMAX,
     &     MOMS   ,MASS  ,MOMS(NDOF),NDOF,1,ONE,
     &     MAXNOD,IDOF,COORD ,ID,TITLE,STEPID,.FALSE.,
     &     NCOS,COSINE,JTCOS,JTFLG,SUMRC)
      WRITE (6,5211) SUMMFO,SUMMMO
 5111 FORMAT (/4X,'MAXIMUM RESULTANT OF REACTIONS,   FORCE=',1P,G12.4,
     &                                    ' MOMENT=',G12.4)
C
C=================================
C= PRINT MAXIMUM MEMBER FORCES ===
C=================================
      LSTTYP=0
      PRINT=.TRUE.
      HEAD =.FALSE.
      WRITE (6,251) TITLE(1),TITLE(2)
      DO 5300 IELNO=1,NELMT
 5300   CALL ELELIB
     &  (12    ,LSTTYP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDLOA,MSTOR)
C
C=============================================
C= MEMBER DUCTILITY AND EXCURSION RATIOS ===
C=============================================
      IF (.NOT. ELSTIC) THEN
           LSTTYP=0
           PRINT=.TRUE.
           HEAD =.FALSE.
           WRITE (6,251) TITLE(1),TITLE(2)
           DO 5310 IELNO=1,NELMT
 5310        CALL ELELIB
     &       (11    ,LSTTYP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &       IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDLOA,MSTOR)
      ENDIF
C
C=============================
C= PRINT DAMAGE INDICIES  ===
C=============================
      IF (.NOT. ELSTIC) THEN
           LSTTYP=0
           HEAD =.FALSE.
           PRINT=.TRUE.
           WRITE (6,251) TITLE(1),TITLE(2)
           DAMAGE=0
           DO 5320 IELNO=1,NELMT
             CALL ELELIB
     &       (14    ,LSTTYP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,EDAM,
     &       IELNO,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDLOA,MSTOR)
 5320        DAMAGE=DAMAGE+EDAM
           DAMAGE=DAMAGE/(ESE+PSE)
           WRITE (6,5330) DAMAGE
 5330      FORMAT (/10X,'STRUCTURE DAMAGE INDEX=',1P,G15.5)
      ENDIF
C
C=============================
C= PRINT PEAK ENERGY INFO ===
C=============================
      WRITE (6,251) TITLE(1),TITLE(2)
      WRITE (6,5340) EIEN,ESEN,PSEN,EKEN,EDOM
 5340 FORMAT ('   PEAK ENERGY VALUES ' / 1P,
     & '   MAX INPUT ENERGY.....',G12.5/
     & '   MAX ELASTIC STRAIN ENERGY.....',G12.5/
     & '   MAX PLASTIC STRAIN ENERGY.....',G12.5/
     & '   MAX KINETIC ENERGY.....',G12.5/
     & '   MAX DAMPING ENERGY.....',G12.5/)
C
C
      RETURN
C
  250 FORMAT ('1 STRUCTURE.....: ',A,/
     &        '  SOLUTION......: ',A,//
     &        2X,A,':',  LOADING #',I5,5X,A)
C
      END
C=====================================================================
      SUBROUTINE SOL05
      LOGICAL TEST,QUIT,BTEST
      INCLUDE (ZCOMM)
C
C     ---- SET UP CONSTANTS ----------------
      L1=1
```

```
      L2=NCOND
      L3=L2+1
      L4=NCOND +NFREE
      L5=L4+1
      L6=NDOF
      ELSTIC=.TRUE.
C
C      ---- INITIALIZE STORAGE
C     Z(IZEVAL) = LOCATION OF THE EIGENVALUES
C     Z(IZEVEC) = LOCATION OF THE EIGENVECTORS
C     Z(IZA   ) = LOCATION OF THE SYMMETRIC STIFFNESS
C     Z(IZB   ) = LOCATION OF THE SYMMETRIC MASS OR KG
C
       IZEVAL = IZ
       IZ     = IZ+NFREE
       IZEVEC = IZ
       IZ     = IZ+NFREE*NFREE
       IZA    = IZ
       IZ     = IZ+NFREE*(NFREE+1)/2
       IZB    = IZ
       IZ     = IZ+NFREE*(NFREE+1)/2
       IZTMP= IZ
       IZ     = IZ+NDOF
       IM=IZ
C
      CALL CKSTOR(NDOF,IM)
C
      IMD  =NZ(IZMD  +NDOF)
      IMDMS=NZ(IZMDMS+NDOF)
      IMDKG=NZ(IZMDKG+NDOF)
      IMDAB=NFREE*(NFREE+1)/2
C
C---------- FORM MASS MATRIX -----
C
      DO 400 I=1,IMDMS
  400     Z(I+IZMAS3-1)=0.
C
      CALL MFORM(2,IMMASS,NMASS,Z(IZMASS),NZ(IZMDMS),MD,NDOF,IMDMS,
     &              TITLE,NNODE,MAXNOD,NCOS,BTEST(IBUG,5),IBUG,
     &              NZ(IZID),Z(IZCOS),NZ(IZIDOF),Z(IZCNST),NZ(IZJFLG),
     &              NZ(IZJCOS),Z(IZMDAT),Z(IM),NZ(IM+21),Z(IM+27),
     &               Z(IM+ 65),Z(IM+ 99))
C
C
C
      CALL SOLOSA
     & (L1,L2,L3,L4,L5,L6,IMD,IMDMS,IMDKG,IMDAB,KGCOND,NCOND,ABORT,
     &  Z(IZKGDT),NZ(IZKGDT+1),NZ(IZKGDT+2),NZ(IZKGDT+5),
     &  NDOF     , NCOND    , NFREE    , TITLE    , STEPID   ,
     &  Z(IZSTIF), Z(IZMASS), Z(IZKG  ), Z(IZA   ), Z(IZB   ),
     & NZ(IZMD  ),NZ(IZMDMS),NZ(IZMDKG), Z(IM    ), Z(IZTMP ),
     &  NNODE    , NCOS     , NELMT    , MAXNOD   ,IBUG      ,
     & NZ(IZID  ),NZ(IZIDOF), Z(IZCNST),NZ(IZJFLG), Z(IZCORD),
     &  Z(IZCOS ),NZ(IZJCOS), Z(IZEVAL), Z(IZEVEC) )
C
      RETURN
      END
C=======================================================================
      SUBROUTINE SOLOSA
     & (L1,L2,L3,L4,L5,L6,IMD,IMDMS,IMDKG,IMDAB,KGCOND,NCOND,ABORT,
     &  KGDATA,KGLOAD,KGTYPE,KGFORM,
     &  NDOF     , NCOND    , NFREE    , TITLE    , STEPID   ,
     &  STIFF    , MASS     , KG       , A        , B        ,
     &  MD       , MDMS     , MDKG     , WORK     , TMP      ,
     &  NNODE    , NCOS     , NELMT    , MAXNOD   , IBUG     ,
     &  ID       , IDOF     , CNST     , JTFLG    , CORD     ,
     &  COSINE   , JTCOS    , EVAL     , EVEC     )
      LOGICAL TEST,QUIT,BTEST,MCOND,KGCOND,MCOND1,KCOND1,ABORT
      REAL MASS,KG, KGDATA
      DIMENSION STIFF(IMD),MASS(IMDMS),KG(IMDKG),A(IMDAB),B(IMDAB)
      DIMENSION MD(L6+1),MDMS(L6+1),MDKG(L6+1),WORK(L6),TMP(L6)
      DIMENSION EVAL(NFREE),EVEC(NFREE,NFREE),SUBRCT(6)
      DIMENSION COORD(MAXNOD,5),ID(MAXNOD),
     &          IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CNST(MAXNOD,6),
     &          JTFLG(MAXNOD),JTCOS(MAXNOD)
C
      CHARACTER*1   NAME
      CHARACTER*40  OPTION
      CHARACTER*(*) TITLE(2),STEPID
C
C=============
C= INPUT DATA ===
C=============
      READ (5,*) OPTION,IPRT,IPRT2
C
      IF (TEST(OPTION,'FREQ',.FALSE.)) THEN
         IOPT=1
         OPTION='NATURAL FREQUENCIES'
      ELSE IF (TEST(OPTION,'BUCK',.FALSE.)) THEN
         IOPT=2
         OPTION='ELASTIC BUCKLING LOAD'
      ELSE
         WRITE (6,*) 'SOL-05 INVALID OPTION: ',OPTION
         RETURN
      ENDIF
C
      IF (IPRT.LE.0) IPRT=NFREE
      IPRT=MIN(NFREE,IPRT)
      IPRT2=MIN(IPRT2,IPRT)
C
      WRITE (6,10) TITLE(1),TITLE(2),OPTION,IPRT
   10 FORMAT ('1 STRUCTURE.....: ',A,/
     &        '  SOLUTION......: ',A,//
     &1X,' SOLUTION #5, DETERMINE EIGENVALUES AND EIGENVECTORS ',/
     &1X,'============================================================'//
     &1X,' OPTION:.....',A/
     &1X,' NUMBER OF EIGENVALUES:.....',I5)
C
C=========================
C= FORMULATE STIFFNESS ===
C=========================
      II=1
      CALL FORM(II)
       IF (BTEST(IBUG,5)) CALL LMATRX(STIFF,MD,NDOF ,IMD,
     &    'GLOBAL STIFFNESS MATRIX')
C
C==============================
C= FORM GEOMETRIC STIFFNESS ===
C==============================
C  FORMULATE THE GEOMETRIC STIFFNESS IF THE FLAG NENK IS TRUE, THIS
C       FLAG INITIALIZED ABOVE.
C       IF KGDATA(1)>=1, THE GEOMETRIC STIFFNESS IS FORMED ONCE, AND
C                        MULTIPLIED BY A SCALER TO ACCOUNT FOR CHANGES
C                        IN THE TOTAL VERTICAL LOAD. THUS THE GEOMETRIC
C                        STIFFNESS ONLY NEEDS TO BE FORMED ONCE.
C       IF KGDATA(1)>=2, THE GEOMETRIC STIFFNESS IS BASED ON THE AXIAL
C                        FORCE IN THE ELEMENTS, AND IS RECALCULATED EACH
C                        TIME THE AXIAL FORCE CHANGES.
C       IF KGDATA(3)>=1, THE GEOMETRIC STIFFNESS IS FORMED WITH THE
C                        STIFFNESS, THUS A SEPERATE CALL TO FORM IS
C                        NOT NEEDED
C       IF KGDATA(3)>=2, A SEPERATE GEOMETRIC STIFFNESS MATRIX IS FORMED
C
C     FORMULATE SEPERATE GEOMETRIC STIFFNESS MATRIX
      IF (IOPT.EQ.2) THEN
            KGFORM=2
            KGDATA=0
            I2=2
            CALL FORM(I2)
          IF (BTEST(IBUG,5)) CALL LMATRX(KG,MDKG,NDOF,IMDKG,
     &      ' GLOBAL GEOMETRIC STIFFNESS MATRIX')
      ELSE
          IF (BTEST(IBUG,5)) CALL LMATRX(MASS,MDMS,NDOF,IMDMS,
     &      ' GLOBAL MASS MATRIX')
      ENDIF
C
C=============================
C= CONDENSE OUT NON-FREE DOF ===
C=============================
C... IF NCOND IS TRUE THE MASS MATRIX IS ALSO REDUCED BY GUYAN REDUCTION
C... IF KGCOND IS TRUE THEN KG IS ALSO REDUCED
      IF (NCOND.GT.0) THEN
          II=1
          IIA=1
          IF (IOPT.EQ.2) THEN
             MCOND1 =.FALSE.
             KCOND1 =KGCOND
          ELSE IF (IOPT.EQ.1) THEN
             KCOND1 =.FALSE.
             MCOND1 =MCOND
          ENDIF
C         SET UP DUMMY ZERO LOAD VECTOR
          DO 28 I=1,NDOF
   28        TMP(I)=0
          CALL MSOLL(1,BTEST(IBUG,6),L4,IMD,NDOF,1,L1,L2,
     &              MD,STIFF,IMP,WORK,
     &              MCOND1,MASS,MDMS,IMDMS,
     &              KCOND1,KG  ,MDKG,IMDKG,ABORT)
          IF (ABORT) RETURN
      ENDIF
C
C=====================================
C= TRANSFER STIFFNESS INTO A MATRIX ===
C=====================================
C  NOTE: IMSL ROUTINE EIGZS IS USED TO SOLVE FOR EIGENVALUES.
C        THIS ROUTINE USES A 'SYMMETRIC MATRIX' DEFINED BELOW
C
C        |  1                 |
C        |  2  3              |     NOTE: THE STORAGE METHOD USED
C        |  4  5  6           |           FOR STIFFNESS, MASS AND
C   A =  |  7  8  9 10        |           GEOMETRIC STIFFNESS ARE
C        | 11 12 13 14 15     |           DIFFERENT.
C        | 16 17 18 19 20 21  |
C        | 22 23 24 25 26 27 28 |
C
      QUIT=.FALSE.
      L=0
      DO 50 I=L3,L4
      DO 50 J=L5,I
         L=L+1
         JI=MD(I)+I-J
         J2=MD(I+1)
         IF (JI.GE.J2) THEN
            A(L)=0.
         ELSE
            A(L)=STIFF(JI)
         ENDIF
         IF (BTEST(IBUG,5)) WRITE (6,29) I,J,L,JI,J2,A(L)
   29    FORMAT (' I:',I5,' J:',I5,' L:',I5,' JI',I5,' J2:',I5,' A:',
     &        1P,G15.5)
C
C--------- CHECK FOR ZERO'S ON MAIN DIAGONAL
         IF (I.EQ.J .AND. A(L).LE.0 ) THEN
            QUIT=.TRUE.
            WRITE (6,52) I,A(L)
         ENDIF
C
   50    CONTINUE
   52 FORMAT (5X,'DOF #',I5,' IS .LE. 0 IN THE STIFFNESS MATRIX...',
     &        5X,'K(I,I)=',1P,G12.4,'  SOLN IS ABORTED')
C
C=============================================================
C= TRANSFER MASS OR GEOMETRIC STIFFNESS INTO B MATRIX ===
C=============================================================
      IF (IOPT.EQ.1) THEN
          L=0
          DO 40 I=L3,L4
          DO 40 J=L5,I
             L=L+1
             JI=MDMS(I)+I-J
             J2=MDMS(I+1)
             IF (JI.GE.J2) THEN
                B(L)=0.
             ELSE
                B(L)=MASS(JI)
             ENDIF
          IF (BTEST(IBUG,5)) WRITE (6,59) I,J,L,JI,J2,B(L)
   59     FORMAT (' I:',I5,' J:',I5,' L:',I5,' JI',I5,' J2:',I5,' B:',
     &        1P,G15.5)
C
C--------- CHECK FOR ZERO'S ON MAIN DIAGONAL
          IF (I.EQ.J .AND. B(L).LE.0 ) THEN
             QUIT=.TRUE.
             WRITE (6,42) I,B(L)
          ENDIF
C
   40     CONTINUE
   42 FORMAT (5X,'DOF #',I5,' IS .LE. 0 IN THE MASS MATRIX...',
     &        5X,'K(I,I)=',1P,G12.4,'  SOLN IS ABORTED')
C
      ELSE
          L=0
          DO 50 I=L3,L4
          DO 50 J=L5,I
             L=L+1
             JI=MDKG(I)+I-J
             J2=MDKG(I+1)
             IF (JI.GE.J2) THEN
```

```
                B(L)=0.
              ELSE
                B(L)=KG(JI)
              ENDIF
          IF (BTEST(IBUG,5)) WRITE (6,49) I,J,L,JI,J2,B(L)
   49     FORMAT (' I:',IS,' J:',IS,' L:',IS,' JI',IS,' J2:',IS,' B:',
     &       1P,G15.5)
C
C--------- CHECK FOR ZERO'S ON MAIN DIAGONAL
          IF (I.EQ.J .AND. B(L).LE.0 ) THEN
              QUIT=.TRUE.
              WRITE (6,52) I,B(L)
          ENDIF
C
   50         CONTINUE
   52 FORMAT (5X,'DOF #',IS,
     &     ' IS .LE. 0 IN THE GEOMETRIC STIFFNESS MATRIX...',
     &       5X,'K(I,I)=',1P,G12.4,'  SOLN IS ABORTED?')
C
      ENDIF
C
C---------- RETURN IF A MAIN DIAGONAL IS LE ZERO
      IF (QUIT) RETURN
C
C=============================================
C= SOLVE FOR EIGENVALUES AND EIGENVECTORS ===
C=============================================
C---------- IMSL SU_BROUTINE EIGZS ---------------------------------------------
C  EIGENVALUES AND EIGENVECTORS OF A*X-LAMBDA*B*X=0
C  A  = STIFFNESS - REAL, SYMMETRIC MATRIX
C  B  = MASS OR GEOMETRIC STIFFNESS - REAL, SYMMETRIC MATRIX
C  N  = NUMBER OF DOF OF A AND B
C  IJOB  = 0 - CALC. EIGENVALUES ONLY
C        = 1 - CALC. EIGENVALUES AND EIGENVECTORS
C        = 2 - CALC. EIGENVALUES, EIGENVECTORS AND PREFORMANCE INDEX, PI
C        = 3 - CALC. PREFORMANCE INDEX, PI
C          PI=W(1),    PI<1    - ROUTINE PREFORMED WELL
C                    1<PI<100  - ROUTINE PREFORMED SATISFACTORILY
C                    100<PI    - ROUTINE PREFORMED POORLY
C  D   = EIGENVALUES (LAMBDA), N BY 1
C  Z   = EIGENVECTORS, N BY N
C  IZ  = ROW DIMENSION OF Z MATRIX
C  WK  = WORK AREA, N*(N+1)
C  IER = ER_ROR PARAMETER
C      = 129 - A IS NOT POSITIVE DEFINATE
C      = 129+J - ROUTINE FAILED ON EIGENVALUE J, EIG'S 1 TO J-1 ARE OK
C-------------------------------------------------------------------------------
      IJOB=2
      IER=0
      CALL EIGZS(A,B,NFREE,IJOB,EVAL,EVEC,NFREE,WORK,IER)
      IF (IER.NE.0) WRITE (6,*)
     &    'ER','ROR CODE',IER,' IN IMSL ROUTINE EIGZS'
      WRITE (6,*) ' PREFORMANCE INDEX:',WORK(1)
C
C=============================================
C= NORMALIZE MODE SHAPE TO MAX VALUE OF ONE ===
C=============================================
      DO 90 J=1,IPRT
      AMIN=0
      AMAX=0
      DO 80 I=1,NFREE
         AMIN=MIN(AMIN,EVEC(I,J))
   80    AMAX=MAX(AMAX,EVEC(I,J))
      IF (-AMIN.GT.AMAX) AMAX=AMIN
      IF (AMAX.EQ.0) AMAX=1
C
      DO 90 I=1,NFREE
   90    EVEC(I,J)=EVEC(I,J)/AMAX
C
C=========================================
C= PRINT EIGENVALUES AND EIGENVECTORS ===
C=========================================
      IST =1
  100 IEND=MIN(IST+6,IPRT)
C
      WRITE (6,110) (I            ,I=IST,IEND)
      IF (IOPT.EQ.2) THEN
          WRITE (6,120) (EVAL(I),I=IST,IEND)
      ELSE IF (IOPT.EQ.1) THEN
          DO 125 I=IST,IEND
  125       EVAL(I)=SQRT( EVAL(I) )
          WRITE (6,126) (EVAL(I),I=IST,IEND)
          DO 127 I=IST,IEND
  127       EVAL(I)= EVAL(I)  / 6.2831853
          WRITE (6,128) (EVAL(I),I=IST,IEND)
          DO 129 I=IST,IEND
            IF (EVAL(I).EQ.0) EVAL(I)=1.E-15
  129       EVAL(I)= 1.00 / EVAL(I)
          WRITE (6,130) (EVAL(I),I=IST,IEND)
      ENDIF
      WRITE (6,135)
  110 FORMAT('              MODE:',7(I10,5X))
  120 FORMAT('      EIGENVALUE:',1P,7(G14.5,1X) )
  126 FORMAT(' FREQ (RAD/SEC):',1P,7(G14.5,1X) )
  128 FORMAT(' FREQUENCY (HZ):',1P,7(G14.5,1X) )
  130 FORMAT('    PERIOD (SEC):',1P,7(G14.5,1X) )
  135 FORMAT('    EIGENVECTORS:*)
C
C-------- EIGENVECTORS
      DO 140 I=1,NFREE
        II=I+NCOND
  140   WRITE (6,150) II,(EVEC(I,J),J=IST,IEND)
  150 FORMAT('      DOF(',I6,')',1P,7(G14.5,1X) )
C
      IF (IEND.LT.IPRT ) THEN
         WRITE (6,10) TITLE(1),TITLE(2),OPTION,IPRT
         IST=IEND+1
         GO TO 100
      ENDIF
C
C=============================================
C= CALC EIGEN VECTORS FOR NONDYNAMIC DOF ===
C=============================================
C
C------ TRANSFER EIGENVECTORS TO TEMP STORAGE
C
      DO 500 IFREQ=1,IPRT2
C
C------ TRANSFER EIGENVECTORS TO TEMP STORAGE
          DO 200 J=L1,L2
  200       WORK(J)=0
          DO 210 J=L5,L4
  210       WORK(J)=EVEC(J-L2,IFREQ)
          DO 220 J=L5,L6
```

```
  220       WORK(J)=0
C
C------ CALC EIGENVECTORS FOR CONDENSED OUT DOF
          CALL REACTNK(I1,BTEST(IBUG,6),NNODE,L1,L2,L3,L4,TMP,WORK,
     &         MD,STIFF,IMD,NDOF,1,1,0,
     &         MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,
     &         NCOS,COSINE,JTCOS,JTFLG,SUMRCT)
          CALL MSOLL(5,BTEST(IBUG,6),L4,IMD,NDOF,I1,L1,L2,
     &         MD,STIFF,TMP,WORK,
     &         .FALSE.,MASS,MDMS,1,
     &         .FALSE.,KG  ,MDKG,1,ABORT)
        IF (ABORT) RETURN
C
C------- NORMALIZE MODE SHAPE TO MAX VALUE OF ONE
          AMIN=0
          AMAX=0
          DO 250 I=1,L6
             AMIN=MIN(AMIN,WORK(I))
  250        AMAX=MAX(AMAX,WORK(I))
          IF (-AMIN.GT.AMAX) AMAX=AMIN
C
          DO 240 I=1,L6
  240        WORK(I)=WORK(I)/AMAX
C
C------- PRINT EIGENVALUE AT EACH NODE
          IF (IOPT.EQ.2) WRITE (STEPID,250) IFREQ,EVAL(IFREQ)
             IF (EVAL(IFREQ).EQ.0) EVAL(IFREQ)=1.E-15
          FREQ=1.00/EVAL(IFREQ)
          IF (IOPT.EQ.1) WRITE (STEPID,260) IFREQ,FREQ
C
          CALL JOIDSPC(1,MAXNOD,NDOF,NNODE,1,L5,L4,ID,
     &         IDOF,CUST,WORK,
     &         TITLE,'EIGENVECTORS',STEPID,.TRUE.,
     &         NCOS,COSINE,JTCOS,JTFLG)
  250     FORMAT ('MODE #',I3,'   EIGENVALUE:',1P,G15.6)
  260     FORMAT ('MODE #',I3,'   FREQUENCY:',1P,G15.6)
  500 CONTINUE
C
      RETURN
C
      END
C=======================================================================
      SUBROUTINE SOL0
      LOGICAL BTEST,UNBAL
      INCLUDE (ZCOMM)
C
C=================================
C= SET GEOMETRIC STIFFNESS FLAGS ===
C=================================
      Z(IZKGDT)=0.00
      KGLOAD=NZ(IZKGDT+1)
      KGTYPE=NZ(IZKGDT+2)
      KGFORM=NZ(IZKGDT+5)
C
C=============================
C= READ INFO AND SET OPTIONS ===
C=============================
      READ  (5,*) MAXELD,IPRNT,IWRITE,UNBAL
      NLOAD=1
      IPRNT=MAX(IPRNT,1)
C
C==================
C= PRINT HEADER ===
C==================
      WRITE (6,101) TITLE(1),TITLE(2),IPRNT,IWRITE
  101 FORMAT ('1 STRUCTURE....: ',A,/
     &        '  SOLUTION.....: ',A,/
     &       &1X,' SOLUTION #=. STATIC NONLINEAR SOLUTION '/
     &       &1X,'=============================================='/
     &       &1X,' INTERVAL FOR PRINTING DATA.............',I5/
     &       &1X,' INTERVAL FOR WRITING DATA TO FILE......',I5/)
C
      IF (KGFORM.EQ.2) WRITE (6,11)
   11 FORMAT (1X,' GEOMETRIC STIFFNESS IS NOT INCLUDED. ',
     &        ' USE KGDATA(5)=1 TO INCLUDE GEOMETRIC STIFFNESS.')
      IF (     UNBAL) WRITE (6,15)  ' ARE '
      IF (.NOT.UNBAL) WRITE (6,15)  ' ARE NOT '
   15 FORMAT(' UNBALANCED JOINT FORCES',A,'ADDED TO THE NEXT CYCLE')
C
C=========================================
C=INPUT FILE OUTPUT CONTROLL DATA ===
C=========================================
      IF (IWRITE.GT.0) THEN
         IOPT=1
         TO=0
         DT=0
         CALL DMPDAT(IOPT,IWRITE,TO,DT)
      ENDIF
C
C=========================================
C=-INITILIZE STORAGE REQ'D FOR THIS SOLN ===
C=========================================
C     Z(IZLOAD) = LOCATION OF THE TOTAL LOAD MATRIX
C     Z(IZDISP) = LOCATION OF THE TOTAL DISPLACEMENT MATRIX
C     Z(IZVEL ) = LOCATION OF THE TOTAL VELOCITY MATRIX
C     Z(IZLOA) = LOCATION OF THE INCRIMENTAL LOAD MATRIX
C     Z(IZDDSP) = LOCATION OF THE INCRIMENTAL DISPLACEMENT MATRIX
C     Z(IZDVEL) = LOCATION OF THE INCRIMENTAL VELOCITY MATRIX
C     Z(IZQ   ) = LOCATION OF THE JOINT LOADS
C     Z(IZR   ) = LOCATION OF THE ELEMENT LOADS
C
       IF (IZLOAD.EQ.0) THEN
          IZLOAD = IZ
          IZ     = IZ+NDOF
          DO 1 I=1,NDOF
    1     Z(I+IZLOAD-1) =0
          ENDIF
C
       IF (IZDISP.EQ.0) THEN
          IZDISP = IZ
          IZ     = IZ+NDOF
          DO 2 I=1,NDOF
    2     Z(I+IZDISP-1) =0
          ENDIF
C
       IF (IZVEL .EQ.0) THEN
          IZVEL  = IZ
          IZ     = IZ+NDOF
          DO 5 I=1,NDOF
    5     Z(I+IZVEL -1) =0
          ENDIF
C
       IF (IZDLOA.EQ.0) THEN
          IZDLOA = IZ
```

```
            IZ     = IZ+NDOF
            DO 5 I=1,NDOF
    5       Z(I+IZDLOA-1) =0
            ENDIF
C
        IF (IZDDSP.EQ.0) THEN
            IZDDSP = IZ
            IZ     = IZ+NDOF
            DO 6 I=1,NDOF
    6       Z(I+IZDDSP-1) =0
            ENDIF
C
        IF (IZDVEL.EQ.0) THEN
            IZDVEL = IZ
            IZ     = IZ+NDOF
            DO 7 I=1,NDOF
    7       Z(I+IZDVEL-1) =0
            ENDIF
C
            IZQ    = IZ
            IZ     = IZ+NDOF
            DO 8 I=1,NDOF
    8       Z(I+IZQ-1) =0
C
            IZR    = IZ
            IZ     = IZ+NDOF
            DO 9 I=1,NDOF
    9       Z(I+IZR-1) =0
C
        IF (IZFUB .EQ.0) THEN
            IZFUB  = IZ
            IZ     = IZ+NDOF
            ENDIF
            DO 10 I=1,NDOF
   10       Z(I+IZFUB-1) =0
C
            IZUBLD = IZ
            IZ     = IZ+NDOF
            IZUBDP = IZ
            IZ     = IZ+NDOF
            DO 12 I=1,NDOF
            Z(I+IZUBLD-1) =0
   12       Z(I+IZUBDP-1) =0
C
        IH=IZ+1
C
C---- DETERMINE THE LENGTH OF THE STIFFNESS MATRIX
            IMO=NZ(IZMO+NDOF) - NZ(IZMO) +1
C
C       ---- SET UP CONSTANTS ----------------
        L1=1
        L2=NCOND
        L3=L2+1
        L4=NCOND +NFREE
        L5=L4+1
        L6=NDOF
C
        NLOAD=1
C
        ELSTIC=.FALSE.
C
        IF (MAXELD.GT.0) THEN
            IZIELD=IZ
            IZ=IZ + MAXELD*5
            IZELD=IZ
            IZ=IZ + MAXELD*12
        ENDIF
C
C==========================================
        IZFG  = IZ
        IZ    = IZ+NDOF
        IZDFG = IZ
        IZ    = IZ+NDOF
        IZDMAX= IZ
        IZ    = IZ+NDOF
        IZRMAX= IZ
        IZ    = IZ+NDOF
        IZDRLO= IZ
        IZ    = IZ+NDOF
        IZRLOA= IZ
        IZ    = IZ+NDOF
        DO 27 I=0,NDOF-1
          Z(IZFG  +I)=0
          Z(IZDFG +I)=0
          Z(IZDMAX+I)=0
          Z(IZRMAX+I)=0
          Z(IZRLOA+I)=0
   27     Z(IZDRLO+I)=0
C
C==========================================
C
C------ CHECK MEMORY
        CALL CKSTOR(NDOF,IH)
C
C------ SET UP CONSTANTS
        IMO   =NZ(IZMO  +NDOF)
        IMOMS =NZ(IZMOMS+NDOF)
        IMOKG =NZ(IZMOKG+NDOF)
        IMONSL=1
        IMOS  =1
C
C=========================
C= ZERO GROUND MOTION ===
C=========================
        DO 18 I=1,6
   18   SUMRCT(I)=0.
C
C=========================
C= ZERO GROUND MOTION ===
C=========================
        DO 19 I=1,9
   19   GROUNDX(I)=0.
C
C=========================
C= CALC PRINTING INFO ===
C=========================
        IPDISP=6+NMODE
        IPRCTN=7
        DO 20 I=1,NMODE
          DO 15 J=0,5
          IF (BTEST(NZ(IZJFLG-1+I),J)) THEN
            IPRCTN=IPRCTN+1
            GO TO 20
          ENDIF
   15     CONTINUE
   20   CONTINUE
        IPELEM=NELMT
        LSTTYP=0
      * DO 26 I=1,NELMT
          J=NZ(IZELE+I)
          IF (NZ(J).NE.LSTTYP) IPELEM=IPELEM+4
   26     LSTTYP=NZ(J)
  111 FORMAT (OP,' ZMATRX(',I6,'):',I15,1P,G20.10)
C
        CALL SOL04A
     & (L1,L2,L3,L4,L5,L6,IMO,IMOMS,IMOKG,IMOS,IMONSL,ABORT,
     &  MAXELD  , IPRNT   , IWRITE  , UNBAL   ,
     &  NLOAD   , NMODE   , NCOS    , NELMT   ,
     &  IZDDSP  , IZDLOA  , IBUG    , MAXNOD  ,
     &  TITLE   , STEPID  , NDOF    , NFREE   , Z(IZRLOA),
     &  KGFORM  , KGLOAD  , KGTYPE  , NCOND   , Z(IZDRLO),
     &  NEWKG   , NEWK    , MCOND   , KGCOND  ,   SUMRCT ,
     &  IPDISP  , IPRCTN  , IPELEM  , Z(IZDMAX), Z(IZRMAX),
     &  Z(IZDISP), Z(IZVEL ), Z(IZLOAD), Z(IZQ   ), Z(IZFUB ),
     &  Z(IZDDSP), Z(IZDVEL), Z(IZDLOA), Z(IZR   ), Z(IM    ),
     &  Z(IZSTIF), Z(IZMASS), Z(IZKG  ), Z(IZIELD), Z(IZUBLD),
     & NZ(IZMO  ),NZ(IZMOMS),NZ(IZMOKG),NZ(IZIELD),NZ(IZUBDP),
     & NZ(IZIID ),NZ(IZIDOF),NZ(IZCNST),NZ(IZJFLG),Z(IZCORD),
     &  Z(IZCOS ),NZ(IZJCOS), EIE,ESE,PSE,EKE,EDD ,
     &  Z(IZFG  ),NZ(IZDFG ))
        RETURN
C
        END
C=======================================================================
        SUBROUTINE SOL04A
     & (L1,L2,L3,L4,L5,L6,IMO,IMOMS,IMOKG,IMOS,IMONSL,ABORT,
     &  MAXELD  , IPRNT   , IWRITE  , UNBAL   ,
     &  NLOAD   , NMODE   , NCOS    , NELMT   ,
     &  IZDDSP  , IZDLOA  , IBUG    , MAXNOD  ,
     &  TITLE   , STEPID  , NDOF    , NFREE   , RLOAD   ,
     &  KGFORM  , KGLOAD  , KGTYPE  , NCOND   , DRLOA   ,
     &  NEWKG   , NEWK    , MCOND   , KGCOND  , SUMRC   ,
     &  IPDISP  , IPRCTN  , IPELEM  , DSPMAX  , RCTMAX  ,
     &  DISP    , VEL     , LOAD    , Q       , FUB     ,
     &  DDISP   , DVEL    , DLOAD   , R       , WORK    ,
     &  STIFF   , MASS    , KG      , ELD     , UBLOAD  ,
     &  MQ      , MOMS    , MOKG    , IELD    , UBDISP  ,
     &  ID      , IDOF    , CNST    , JTFLG   , COORD   ,
     &  COSINE  , JTCOS   ,  EIE,ESE,PSE,EKE,EDD,
     &  FG      , DFG     )
C
        DOUBLE PRECISION  EIE,ESE,PSE,EDD,EKE
C
        CHARACTER*(*) TITLE(2),STEPID
        LOGICAL PRINT,DSPFLG,HEAD,REPEAT,AXIAL,TEST,NEWKDY,BTEST,ABORT
        LOGICAL NEWKG,NEWK,MCOND,KGCOND,UNBAL,MCND2,KGCND2,WRITE
        LOGICAL BUG5,BUG5,BUG6,BUG11
        CHARACTER*80 NAME
        CHARACTER*20 INTEG
        REAL LOAD,MASS,MASSSL,KG,MCOS
        INTEGER FOAMP,FMASS
        DIMENSION SUMRC(6),DSUMRC(6),SUMFD(6),DSUMFD(6),SUMRCM(6),SUMUB(6)
        DIMENSION DSPMAX(L6),RCTMAX(L6),DRLOA(L6),RLOAD(L6)
        DIMENSION RINPUT(100)
        DIMENSION STIFF(IMO) ,MASS(IMOMS),  KG(IMOKG)
        DIMENSION    MQ(L6+1),MOMS(L6+1) ),MOKG(L6+1 )
        DIMENSION  LOAD(L6), DISP(L6), VEL(L6), Q(L6),FG(L6),DFG(L6)
        DIMENSION DLOAD(L6),DDISP(L6),DVEL(L6), R(L6)
        DIMENSION   FUB(L6),UBLOAD(L6),UBDISP(L6)
        DIMENSION WORK(2*L6)
        DIMENSION COORD(MAXNOD,3),ID(MAXNOD),
     &            IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CNST(MAXNOD,6),
     &            JTFLG(MAXNOD),JTCOS(MAXNOD)

        DIMENSION IELD(1),ELD(1)
        DIMENSION  GA(3), GV(3), GD(3)
        DIMENSION GAT(3),GVT(3),GDT(3)
        LOGICAL NEWFAC
        CHARACTER*10 STEP,STEP2
C
        SUMUB(1)=0
        SUMUB(2)=0
        SUMUB(3)=0
        SUMUB(4)=0
        SUMUB(5)=0
        SUMUB(6)=0
        SUMFOR=0
        SUMMOM=0
        SUMFD=0
        SUMMMO=0
C=================================
        BUG3 =BTEST(IBUG, 3)
        BUG5 =BTEST(IBUG, 5)
        BUG6 =BTEST(IBUG, 6)
        BUG11=BTEST(IBUG,11)
C=================================
C
C=================================
C= INITIALIZE ENERGY CONSTANTS ===
C=================================
        EIE=0
        ESE=0
        PSE=0
        EKE=0
        EDD=0
C
C=================================
C= SET GEOMETRIC STIFFNESS FLAGS ===
C=================================
        NEWKG=.TRUE.
        IF (KGLOAD.EQ.0 .OR. KGTYPE.EQ.0 .OR. KGFORM.EQ.0) NEWKG=.FALSE.
C
C=========================
C= SET STIFFNESS FLAGS ===
C=========================
        NEWK = .TRUE.
        KSAME= 0
C
C=========================
C= INPUT LOADING DATA ==
C=========================
C--------- MATRICIES Q AND R ARE ZEROED IN THE CALLING PGM.
C
C       ---- INPUT JOINT LOADS ----------------
C       JOINT LOADS AND SUPPORT DISPLACEMENTS ARE STORED IN Q
        CALL JOILOA(BUG5,MAXNOD,NDOF,NMODE,NLOAD,L4,
     &      ID,IDOF,CNST,Q, DSPFLG,JTFLG,TITLE,.FALSE.)
C
   15   CONTINUE                                                    SOL02110
   20 CONTINUE                                                      SOL02120
      IPELEM=NELMT                                                  SOL02130
      LSTTYP=0                                                      SOL02140
      DO 26 I=1,NELMT                                               SOL02150
        J=NZ(IZELE+I)                                               SOL02160
        IF (NZ(J).NE.LSTTYP) IPELEM=IPELEM+4                        SOL02170
   26   LSTTYP=NZ(J)                                                SOL02180
  111 FORMAT (OP,' ZMATRX(',I6,'):',I15,1P,G20.10)                  SOL02190
C                                                                   SOL02200
      CALL SOL04A                                                   SOL02210
     & (L1,L2,L3,L4,L5,L6,IMO,IMOMS,IMOKG,IMOS,IMONSL,ABORT,        SOL02220
     &  MAXELD  , IPRNT   , IWRITE  , UNBAL   ,                     SOL02230
     &  NLOAD   , NMODE   , NCOS    , NELMT   ,                     SOL02240
     &  IZDDSP  , IZDLOA  , IBUG    , MAXNOD  ,                     SOL02250
     &  TITLE   , STEPID  , NDOF    , NFREE   , Z(IZRLOA),          SOL02260
     &  KGFORM  , KGLOAD  , KGTYPE  , NCOND   , Z(IZDRLO),          SOL02270
     &  NEWKG   , NEWK    , MCOND   , KGCOND  ,   SUMRCT ,         SOL02280
     &  IPDISP  , IPRCTN  , IPELEM  , Z(IZDMAX), Z(IZRMAX),         SOL02290
     &  Z(IZDISP), Z(IZVEL ), Z(IZLOAD), Z(IZQ   ), Z(IZFUB ),      SOL02300
     &  Z(IZDDSP), Z(IZDVEL), Z(IZDLOA), Z(IZR   ), Z(IM    ),      SOL02310
     &  Z(IZSTIF), Z(IZMASS), Z(IZKG  ), Z(IZIELD), Z(IZUBLD),      SOL02320
     & NZ(IZMO  ),NZ(IZMOMS),NZ(IZMOKG),NZ(IZIELD),NZ(IZUBDP),      SOL02330
     & NZ(IZIID ),NZ(IZIDOF),NZ(IZCNST),NZ(IZJFLG),Z(IZCORD),       SOL02340
     &  Z(IZCOS ),NZ(IZJCOS), EIE,ESE,PSE,EKE,EDD ,                 SOL02350
     &  Z(IZFG  ),NZ(IZDFG ))                                       SOL02360
      RETURN                                                        SOL02380
C                                                                   SOL02380
      END                                                           SOL02390
C=======================================================================SOL00040
      SUBROUTINE SOL04A                                             SOL00050
     & (L1,L2,L3,L4,L5,L6,IMO,IMOMS,IMOKG,IMOS,IMONSL,ABORT,        SOL00060
     &  MAXELD  , IPRNT   , IWRITE  , UNBAL   ,                     SOL00070
     &  NLOAD   , NMODE   , NCOS    , NELMT   ,                     SOL00080
     &  IZDDSP  , IZDLOA  , IBUG    , MAXNOD  ,                     SOL00090
     &  TITLE   , STEPID  , NDOF    , NFREE   , RLOAD   ,           SOL00100
     &  KGFORM  , KGLOAD  , KGTYPE  , NCOND   , DRLOA   ,           SOL00110
     &  NEWKG   , NEWK    , MCOND   , KGCOND  , SUMRC   ,           SOL00120
     &  IPDISP  , IPRCTN  , IPELEM  , DSPMAX  , RCTMAX  ,           SOL00130
     &  DISP    , VEL     , LOAD    , Q       , FUB     ,           SOL00140
     &  DDISP   , DVEL    , DLOAD   , R       , WORK    ,           SOL00150
     &  STIFF   , MASS    , KG      , ELD     , UBLOAD  ,           SOL00160
     &  MQ      , MOMS    , MOKG    , IELD    , UBDISP  ,           SOL00170
     &  ID      , IDOF    , CNST    , JTFLG   , COORD   ,           SOL00180
     &  COSINE  , JTCOS   ,  EIE,ESE,PSE,EKE,EDD,                   SOL00190
     &  FG      , DFG     )                                        SOL00200
C                                                                   SOL00210
      DOUBLE PRECISION  EIE,ESE,PSE,EDD,EKE                        SOL00220
C                                                                   SOL00230
      CHARACTER*(*) TITLE(2),STEPID                                 SOL00240
      LOGICAL PRINT,DSPFLG,HEAD,REPEAT,AXIAL,TEST,NEWKDY,BTEST,ABORT SOL00250
      LOGICAL NEWKG,NEWK,MCOND,KGCOND,UNBAL,MCND2,KGCND2,WRITE      SOL00260
      LOGICAL BUG5,BUG5,BUG6,BUG11                                  SOL00270
      CHARACTER*80 NAME                                             SOL00280
      CHARACTER*20 INTEG                                            SOL00290
      REAL LOAD,MASS,MASSSL,KG,MCOS                                 SOL00300
      INTEGER FOAMP,FMASS                                           SOL00310
      DIMENSION SUMRC(6),DSUMRC(6),SUMFD(6),DSUMFD(6),SUMRCM(6),SUMUB(6)SOL00320
      DIMENSION DSPMAX(L6),RCTMAX(L6),DRLOA(L6),RLOAD(L6)           SOL00330
      DIMENSION RINPUT(100)                                         SOL00340
      DIMENSION STIFF(IMO) ,MASS(IMOMS),  KG(IMOKG)                 SOL00350
      DIMENSION    MQ(L6+1),MOMS(L6+1) ),MOKG(L6+1 )                SOL00360
      DIMENSION  LOAD(L6), DISP(L6), VEL(L6), Q(L6),FG(L6),DFG(L6)  SOL00370
      DIMENSION DLOAD(L6),DDISP(L6),DVEL(L6), R(L6)                 SOL00380
      DIMENSION   FUB(L6),UBLOAD(L6),UBDISP(L6)                     SOL00390
      DIMENSION WORK(2*L6)                                          SOL00400
      DIMENSION COORD(MAXNOD,3),ID(MAXNOD),                         SOL00410
     &          IDOF(MAXNOD,6) ,COSINE(3,3,NCOS),CNST(MAXNOD,6),    SOL00420
     &          JTFLG(MAXNOD),JTCOS(MAXNOD)                         SOL00430
C                                                                   SOL00440
      DIMENSION IELD(1),ELD(1)                                      SOL00450
      DIMENSION  GA(3), GV(3), GD(3)                                SOL00460
      DIMENSION GAT(3),GVT(3),GDT(3)                                SOL00470
      LOGICAL NEWFAC                                                SOL00510
      CHARACTER*10 STEP,STEP2                                       SOL00520
C                                                                   SOL00540
      SUMUB(1)=0                                                    SOL00550
      SUMUB(2)=0                                                    SOL00560
      SUMUB(3)=0                                                    SOL00570
      SUMUB(4)=0                                                    SOL00580
      SUMUB(5)=0                                                    SOL00590
      SUMUB(6)=0                                                    SOL00600
      SUMFOR=0                                                      SOL00610
      SUMMOM=0                                                      SOL00620
      SUMFD=0                                                       SOL00630
      SUMMMO=0                                                      SOL00640
C=================================                                  SOL00650
      BUG3 =BTEST(IBUG, 3)                                          SOL00660
      BUG5 =BTEST(IBUG, 5)                                          SOL00670
      BUG6 =BTEST(IBUG, 6)                                          SOL00680
      BUG11=BTEST(IBUG,11)                                          SOL00690
C=================================                                  SOL00700
C                                                                   SOL00710
C=================================                                  SOL00720
C= INITIALIZE ENERGY CONSTANTS ===                                  SOL00730
C=================================                                  SOL00740
      EIE=0                                                         SOL00750
      ESE=0                                                         SOL00760
      PSE=0                                                         SOL00770
      EKE=0                                                         SOL00780
      EDD=0                                                         SOL00790
C                                                                   SOL00800
C=================================                                  SOL00810
C= SET GEOMETRIC STIFFNESS FLAGS ===                                SOL00820
C=================================                                  SOL00830
      NEWKG=.TRUE.                                                  SOL00840
      IF (KGLOAD.EQ.0 .OR. KGTYPE.EQ.0 .OR. KGFORM.EQ.0) NEWKG=.FALSE. SOL00850
C                                                                   SOL00860
C=========================                                          SOL00870
C= SET STIFFNESS FLAGS ===                                          SOL00880
C=========================                                          SOL00890
      NEWK = .TRUE.                                                 SOL00900
      KSAME= 0                                                      SOL00910
C                                                                   SOL00920
C=========================                                          SOL00950
C= INPUT LOADING DATA ==                                            SOL00940
C=========================                                          SOL00950
C--------- MATRICIES Q AND R ARE ZEROED IN THE CALLING PGM.         SOL00960
C                                                                   SOL00970
C       ---- INPUT JOINT LOADS ----------------                     SOL00980
C       JOINT LOADS AND SUPPORT DISPLACEMENTS ARE STORED IN Q       SOL00990
      CALL JOILOA(BUG5,MAXNOD,NDOF,NMODE,NLOAD,L4,                  SOL01000
     &      ID,IDOF,CNST,Q, DSPFLG,JTFLG,TITLE,.FALSE.)             SOL01010
C                                                                   SOL01020
```

```
C                                                                       SOL01050
C       ------ INPUT ELEMENT LOADS ------------------                   SOL01040
C       ELEMENT LOADS ARE STORED IN R                                   SOL01050
        IF (MAXELD.GT.0)                                                SOL01060
     &   CALL ELELOAC(BUGS,L5,NELMT,MAXELD,NELD,NDOF,NLOAD,            SOL01070
     &   R,IELO,ELD,IZDISP,IZLOAD,   NAME,TITLE,.FALSE.)               SOL01080
C                                                                       SOL01090
C                                                                       SOL01100
C=======================================                                SOL01110
C=WRITE INITIAL DATA TO OUTPUT FILE ===                                 SOL01120
C=======================================                                SOL01130
        IF (IWRITE.GT.0) THEN                                           SOL01140
          IOPT=2                                                        SOL01150
          CALL DMPOAT(IOPT,IWRITE,TO,DT)                                SOL01160
        ENDIF                                                           SOL01170
C                                                                       SOL01180
C                                                                       SOL01190
C=======================================================================SOL01200
C= LOOP FOR EACH TIME STEP , ISTEP=CURRENT STEP # ======================SOL01210
C=======================================================================SOL01220
C... ISTEP = STEP NUMBER                                                SOL01230
C    FACTOR= CURRENT TARGET LOAD = FACTOR*(INPUT LOAD)                  SOL01240
C    FACTL = PREVIOUS TRAGET LOAD                                       SOL01250
C    FACT2 = NEXT TRAGET LOAD                                           SOL01260
C    DF    = LOAD STEP                                                  SOL01270
C    DFL   = PREVIOUS LOAD STEP                                         SOL01280
C    DFN   = NEXT LOAD STEP                                             SOL01290
C    PMAX  = MAXIMUN ALLOWABLE INCRIMENTAL LOAD / STEP                  SOL01300
C    DMAX  = MAXIMUN ALLOWABLE INCRIMENTAL DISPLACMENT / STEP           SOL01310
C                                                                       SOL01320
        ISTEP=0                                                         SOL01330
        FACTL=0                                                         SOL01340
        DF=0                                                            SOL01350
        READC5,*) STEP2,FACT2,PMAX2,DMAX2,NUMB2                         SOL01360
 1000 STEP =ISTEP2                                                      SOL01370
        IF (TEST(STEP,'END',.FALSE.)) GO TO 5000                        SOL01380
        FACTOR=FACT2                                                    SOL01390
        PMAX  =PMAX2                                                    SOL01400
        DMAX  =DMAX2                                                    SOL01410
        NUMB  =NUMB2                                                    SOL01420
        NUMBOF=NUMB2                                                    SOL01430
        READC5,*,END=1010) STEP2,FACT2,PMAX2,DMAX2,NUMB2                SOL01440
        GO TO 1020                                                      SOL01450
 1010 STEP2='END'                                                       SOL01460
        FACT2=FACTOR                                                    SOL01470
 1020 IF (TEST(STEP,'END',.FALSE.)) GO TO 5000                          SOL01480
        IF (FACTOR.EQ.FACTL) GO TO 1000                                 SOL01490
C                                                                       SOL01500
 1100   ISTEP=ISTEP+1                                                   SOL01510
        WRITE (STEPID,90) STEP,ISTEP,FACTOR                             SOL01520
        IF (BUGS.OR.BUGS.OR.BUG6.OR.BUG11) THEN                         SOL01530
          WRITE (6,91) STEP,ISTEP,FACTOR                                SOL01540
          WRITE (6,*) 'NEWK: ',NEWK,'   NEWKG: ',NEWKG                  SOL01550
        ENDIF                                                           SOL01560
   90   FORMAT (A,' STEP:',I5,' FACTOR:',1P,G11.4)                      SOL01570
   91   FORMAT ('1 ',A,' STEP:',I5,' FACTOR:',1P,G11.4)                 SOL01580
C                                                                       SOL01590
C============================                                           SOL01600
C= CALC INCRIMENTAL LOADS  ==                                           SOL01610
C============================                                           SOL01620
C 1) ADD ELEMENT LOADS, AND JOINT LOADS ON FREE JOINTS                  SOL01630
C    SAVE IN DLOA FOR CALCULATING UNBALANCED FORCES FOR THE NEXT STEP,  SOL01640
C 2) PUT JOINT DISPLACEMENTS IN DOISP FOR RESTRAINED JOINTS.            SOL01650
C 3) PUT THE NEGAITVE ELEMENT LOAD IN DLOA FOR RESTRAINED JOINTS, THIS  SOL01660
C    IS USED TO CALCULATE REACTIONS.                                    SOL01670
C 4) ZERO THE UNBALANCED LOAD VECTOR                                    SOL01680
C    IS USED TO CALCULATE REACTIONS.                                    SOL01690
        DFL=DF                                                          SOL01700
        DF =FACTOR-FACTL                                                SOL01710
        DFN=FACT2 -FACTOR                                               SOL01720
        IF (PMAX.EQ.0 .AND. DMAX.EQ.0 .AND. NUMB.GT.0) DF=DF/NUMB       SOL01730
        IF (DF.EQ.0) GO TO 1000                                         SOL01740
C                                                                       SOL01750
          DO 106 I=L1,L4                                                SOL01760
  106       DLOAD(I)=(Q(I)+R(I))*DF                                     SOL01770
C                                                                       SOL01780
C       ...SWAP THE SIGN OF THE FEF AT REACTIONS                        SOL01790
        DO 110 I=L5,L6                                                  SOL01800
             DLOAD(I)=-R(I)*DF                                          SOL01810
  110        DOISP(I)= Q(I)*DF                                         SOL01820
C                                                                       SOL01830
C============================                                           SOL01840
C= FORMULATE STIFFNESS ====                                             SOL01850
C============================                                           SOL01860
C  FORMULATE THE STIFFNESS IF THE FLAG NEWK IS TRUE, THIS FLAG IS       SOL01870
C       INITIALIZED ABOVE, AND SET WHEN THE ELEMENT FORCES ARE CALC.    SOL01880
C       THE STIFFNESS IS ALSO FORMULATED IF KG IS TO BE FORMED          SOL01890
      REPEAT=.TRUE.                                                     SOL01900
      KSAME=0                                                           SOL01910
  100 IF (NEWK.OR.NEWKG) THEN                                           SOL01920
        I1=1                                                            SOL01930
        CALL FORMK(I1)                                                  SOL01940
        IF (KGLOAD.EQ.1) NEWKG=.FALSE.                                  SOL01950
      ENDIF                                                             SOL01960
C                                                                       SOL01970
C=============================================                          SOL01980
C= MODIFY LOADS FOR RESTRAINT DISPLACEMENTS ===                         SOL01990
C=============================================                          SOL02000
      IF (DSPFLG) THEN                                                  SOL02010
C       ------ ZERO INCRIMENTAL REACTION LOAD VECTOR ----               SOL02020
        DO 113 I=L1,L4                                                  SOL02030
  113     DRLOA(I)=0                                                    SOL02040
        CALL REACTN(1,BUGS,NNODE,L1,L4,L5,L6,DRLOA,DOISP,               SOL02050
     &  MD,STIFF,IMD,NDOF,1,1.D,                                        SOL02060
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,                      SOL02070
     &           NCOS,COSINE,JTCOS,JTFLG,SUMRC)                         SOL02080
C       ------ ADD INCRIMENTAL REACTION LOADS TO OTHER LOADS           SOL02090
        DO 114 I=L1,L4                                                  SOL02100
  114     DLOAD(I) = DLOAD(I) + DRLOA(I)                                SOL02110
      ENDIF                                                             SOL02120
C                                                                       SOL02130
C   SAVE IN LOADS IN DOSP FOR CALCULATING FREE DISPL FOR THIS STEP.     SOL02140
        DO 115 I=L1,L4                                                  SOL02150
  115     DOISP(I)= DLOAD(I)                                            SOL02160
C                                                                       SOL02170
C============================                                           SOL02180
C= SOLVE FOR FREE DISPL =====                                           SOL02190
C============================                                           SOL02200
      IF (BUGS) THEN                                                    SOL02210
        IF (NEWK .OR. NEWKG)                                            SOL02220
     &  CALL LMATRX(STIFF,MD,NDOF,IMD,'GLOBAL STIFFNESS DUMP=7')        SOL02230
        CALL MMATRX(DLOAD,NDOF,NLOAD,'LOAD MATRIX  DUMP=7')             SOL02240
      ENDIF                                                             SOL02250
C                                                                       SOL02260
        IF (NEWK .OR. NEWKG) THEN                                       SOL02290
          CALL MSOLL(1,.FALSE.,L4,IMD,NDOF,NLOAD,L1,L4,                 SOL02300
     &    MD,STIFF,DDISP,WORK,.FALSE.,MASS,MDMS,1,.FALSE.,KG,MDKG,1,    SOL02310
     &    ABORT)                                                        SOL02320
          IF (ABORT) GO TO 5000                                         SOL02330
        ELSE                                                            SOL02340
          CALL MSOLL(2,.FALSE.,L4,IMD,NDOF,NLOAD,L1,L4,                 SOL02350
     &    MD,STIFF,DDISP,WORK,.FALSE.,MASS,MDMS,1,.FALSE.,KG,MDKG,1,    SOL02360
     &    ABORT)                                                        SOL02370
          IF (ABORT) GO TO 5000                                         SOL02380
        ENDIF                                                           SOL02390
        CALL MSOLL(5,.FALSE.,L4,IMD,NDOF,NLOAD,L1,L4,                   SOL02400
     &    MD,STIFF,DDISP,WORK,.FALSE.,MASS,MDMS,1,.FALSE.,KG,MDKG,1,    SOL02410
     &    ABORT)                                                        SOL02420
        IF (ABORT) GO TO 5000                                           SOL02430
C                                                                       SOL02440
       IF (BUGS)                                                        SOL02450
     &  CALL MMATRX(DDISP,NDOF,NLOAD,'DISPLACEMENT  DUMP=7')            SOL02460
C                                                                       SOL02470
C=============================================                          SOL02480
C= SCALE SOLN SUCH THAT PM LE PMAX, DM LE DMAX ==                       SOL02490
C=============================================                          SOL02500
        DM=0                                                            SOL02510
        PM=0                                                            SOL02520
        DO 130 I=L5,L4                                                  SOL02530
          PM=MAX(DLOAD(I),-DLOAD(I),PM)                                 SOL02540
  130     DM=MAX(DDISP(I),-DDISP(I),DM)                                 SOL02550
        IF ((DM.GT.DMAX .AND. DMAX.NE.0).OR.                            SOL02560
     &     (PM.GT.PMAX .AND. PMAX.NE.0) ) THEN                          SOL02570
          F1=0                                                          SOL02580
          F2=0                                                          SOL02590
          IF (DMAX.NE.0) F1=DM/DMAX                                     SOL02600
          IF (PMAX.NE.0) F2=PM/PMAX                                     SOL02610
          F=MAX(F1,F2)                                                  SOL02620
          DF=DF/F                                                       SOL02630
          NEWFAC=.FALSE.                                                SOL02640
          DO 140 I=L1,L6                                                SOL02650
            DRLOA(I)=DRLOA(I)/F                                         SOL02660
            DLOAD(I)=DLOAD(I)/F                                         SOL02670
  140       DDISP(I)=DDISP(I)/F                                         SOL02680
            DFDFN=1                                                     SOL02690
        ELSE IF (PMAX.EQ.0 .AND. DMAX.EQ.0 .AND. NUMB.GT.1) THEN        SOL02700
          NEWFAC=.FALSE.                                                SOL02710
          NUMB=NUMB-1                                                   SOL02720
        ELSE                                                            SOL02730
          NEWFAC=.TRUE.                                                 SOL02740
          DFDFN=DF*DFN                                                  SOL02750
        ENDIF                                                           SOL02760
C                                                                       SOL02770
        AFACT=FACTL+DF                                                  SOL02780
C                                                                       SOL02790
C=============================================                          SOL02800
C= SOLVE FOR DISPL DUE TO UNBALANCED FORCES ===                         SOL02810
C=============================================                          SOL02820
C                                                                       SOL02830
C...... USE FIRST STIFFNESS TO CALCULATE UNBALANCED FORCES.             SOL02840
C     IF THE STEP IS REPEATED, DO NOT USE THE NEW STIFFNESS.            SOL02850
        IF (UNBAL.AND.REPEAT) THEN                                      SOL02860
          DO 104 I=L1,L4                                                SOL02870
            UBLOAD(I) = -FUBC(I)                                        SOL02880
  104       UBDISP(I) = -FUBC(I)                                        SOL02890
          DO 105 I=L5,L6                                                SOL02900
            UBLOAD(I) = -FUBC(I)                                        SOL02910
  105       UBDISP(I) = 0.00                                            SOL02920
          IF (BUG5)                                                     SOL02930
     &      CALL MMATRX(UBDISP,NDOF,NLOAD,'UNBALANCED LOAD MATRIX')     SOL02940
          CALL MSOLL(2,.FALSE.,L4,IMD,NDOF,NLOAD,L1,L4,                 SOL02950
     &      MD,STIFF,UBDISP,WORK,.FALSE.,MASS,MDMS,1,.FALSE.,KG,MDKG,1,  SOL02960
     &      ABORT)                                                      SOL02970
          IF (ABORT) GO TO 5000                                         SOL02980
          CALL MSOLL(5,.FALSE.,L4,IMD,NDOF,NLOAD,L1,L4,                 SOL02990
     &      MD,STIFF,UBDISP,WORK,.FALSE.,MASS,MDMS,1,.FALSE.,KG,MDKG,1,  SOL03000
     &      ABORT)                                                      SOL03010
          IF (ABORT) GO TO 5000                                         SOL03020
          IF (BUG5)                                                     SOL03030
     &    CALL MMATRX(UBDISP,NDOF,NLOAD,'UNBALANCED DISPLACEMENT')      SOL03040
        ENDIF                                                           SOL03050
C                                                                       SOL03060
C...... ADD UNBALANCED FORCES AND DISPL TO STRUCTURAL LOAD AND DISPL.   SOL03070
        IF (UNBAL) THEN                                                 SOL03080
          DO 107 I=L1,L4                                                SOL03090
           DDISP(I)=DDISP(I)+ UBDISP(I)                                 SOL03100
  107      CONTINUE                                                     SOL03110
          DO 109 I=L5,L6                                                SOL03120
            DLOAD(I)=DLOAD(I)- UBLOAD(I)                                SOL03130
  109     CONTINUE                                                      SOL03140
        ENDIF                                                           SOL03150
C                                                                       SOL03160
C...... CALCULATE THE NORM OF THE UNBALANCED FORCES                     SOL03170
        SUM=0                                                           SOL03180
        DO 108 I=L1,L6                                                  SOL03190
          SUM=SUM+FUBC(I)**2                                            SOL03200
  108     FUBC(I)=0.00                                                  SOL03210
          FNORM = SQRT(SUM/L6)                                          SOL03220
C                                                                       SOL03230
C============================                                           SOL03240
C= CALC PSUDO-VELOCITY ===                                              SOL03250
C============================                                           SOL03260
        IF (ISTEP.EQ.1) THEN                                            SOL03270
          DO 120 I=L1,L6                                                SOL03280
            VEL(I)=Q(I)+R(I)                                            SOL03290
  120       DVEL(I)=Q(I)+R(I)                                           SOL03300
          PSUVEL=2                                                      SOL03310
          DVEL1 =1                                                      SOL03320
        ELSE IF (FACTOR.EQ.AFACT .AND.                                  SOL03330
     &     .NOT.((FACTL.LT.FACTOR .AND. FACTOR.LT.FACT2) .OR.           SOL03340
     &           (FACTL.GT.FACTOR .AND. FACTOR.GT.FACT2))) THEN         SOL03350
          PSUVEL=-PSUVEL                                                SOL03360
          DO 121 I=L1,L6                                                SOL03370
  121       DVEL(I)= PSUVEL * (Q(I)+R(I))                               SOL03380
          DVEL1 =1                                                      SOL03390
        ELSE IF (DVEL1.NE.0) THEN                                       SOL03400
          DO 122 I=L1,L6                                                SOL03410
  122       DVEL(I)=0                                                   SOL03420
            DVEL1 =0                                                    SOL03430
        ENDIF                                                           SOL03440
C                                                                       SOL03450
C=======================================                                SOL03460
C= CALCULATE INCRIMENTAL MEMBER FORCES ===                              SOL03470
C=======================================                                SOL03480
C ELE_LIB CALLS THE INDIVIDUAL ELEMENT LIBRARIES, EACH OF WHICH DETERMINSOL03490
C  1) DID THE STIFFNESS CHANGE?  IF SO; NEWK=.TRUE.                     SOL03500
C  2) SHOULD THE INCRIMENTAL DISPLACEMENT BE RECALCULATED?              SOL03510
C     IF SO; KSAME=KSAME+___ (A VALUE DEPENDING ON ELEMENT TYPE)        SOL03520
```

```
C  3) THE INCRIMENTAL ELEMENT FORCES
C  4) THE TOTAL ELEMENT FORCES- STORED IN FUB
C
      LSTTYP=0
      PRINT=.FALSE.
      HEAD =.FALSE.
      NEWK=.FALSE.
      DO 150 IELNO=1,NELMT
        IOPT=4
        CALL ELELIB
     & (IOPT  ,LSTTYP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &  IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDL0A,MSTOR)
        IOPT=IOPT
  150 CONTINUE
C
C=========================
C= REPEAT STEP IF NESSARY ===
C=========================
C  IF AN ELEMENT STIFFNESS CHANGE DICTATED THAT THE STEP BE REANALIZED,
C     REFORMULATE THE STIFFNESS AND REANALIZE
C     THE VARIABLE REPEAT IS SET UP TO SUPRESS THE REANALIZING OF A STEP
C     CURENTLY, EACH STEP MAY ONLY BE REANALIZED ONCE, THIS PREVENTS
C     AN ENDLESS LOOP.
C
      IF (KSAME.NE.0 .AND. REPEAT) THEN
          REPEAT=.FALSE.
          GO TO 100
      ENDIF
C
C=========================
C= SOLVE FOR REACTIONS ===
C=========================
      IOPT=2
      CALL REACTNK2,BUGS,NNODE,L1,L4,L5,L6,DLOAD,DDISP,
     &  MD,STIFF,IMD,NDOF,1,1.0,
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,
     &           NCOS,COSINE,JTCOS,JTFLG,SUMRC)
C
C=============================================================
C= SOLVE FOR INCRIMENTAL GEOMETRIC STIFFNESS FORCE, DFG ===
C=============================================================
C
C------- INCRIMENTAL GEOMETRIC STIFFNESS FORCE
      IF (KGFORM.EQ.2) THEN
        IF (BUGS) WRITE(6,*) '<------ INCRIMENTAL FORCE DUE TO KG '
        CALL TMPLVC(BUGS,.TRUE.,IMD,L4,L5,L4,L4,
     &           MDKG,KG,DFG,DDISP)
CC      SCALE FORCE TO INLCUDE GROUND ACCELERATION ...
      ENDIF
C
C=========================
C= PRINT STEP SIZE ===
C=========================
      ISTEP=ISTEP
      IPRNT=IPRNT
      WRITE=(MOD(ISTEP,IPRNT).EQ.0)
      IF (WRITE) THEN
        WRITE (6,245) TITLE(1),TITLE(2),STEPID,FACTOR,AFACT,DF,FACTL,
     &           PMAX,DMAX,FNORM
  245   FORMAT ('1 STRUCTURE....: ',A,/
     &        '  SOLUTION.....: ',A,//,2X,A/
     &  '   TARGET FACTOR...:',1P,G12.4,
     &  '  FACTOR....:',1P,G12.4,'  INC. FACTOR....:',G12.4,
     &  '  PREV. FACTOR.:',G12.4/
     &  '    PMAX............:',1P,G12.4,
     &  '  DMAX......:',G12.4,
     &  '  NORM OF UNBALANCED FORCE:',G12.4)
        LINE=6
      ENDIF
C
C=========================================
C= CALC AND ENERGIES AND DUCTILITIES  ===
C=========================================
C
      IF (WRITE)  LINE=LINE+9
C---------- PULL JOINT DISPL FORCES OUT OF LOAD MATRIX ...
      IF (DSPFLG) THEN
        DO 157 I=1,L4
          LOAD(I)  = LOAD(I)  - RLOAD(I)
  157     DLOAD(I) = DLOAD(I) - DRLOA(I)
      ENDIF

      CALL ENERGY
     & (L1,L2,L3,L4,L5,L6,NDOF,MAXNOD,NCOS,BUG11,.FALSE.,
     &  IMD,IMOMS,IMDKG,MD,MOMS,MDKG,.FALSE.,WRITE,ABORT,
     &  STIFF,KG,MASS,
     &  EIE,ESE,PSE,EKE,EDD,
     &  ALPHA,BETA,KGFORM,NELMT,NNODE,DT,SUMUB,
     &   SUMRC,DSUMRC,  SUMFD,DSUMFD,FDAMP,DFDAMP, FZ,
     &  GA,GV,GD,GAT,GVT,GDT,  WORK,WORK(L6+1),ZERO,ZERO,
     &  DACC,DVEL,DDISP,ACC,VEL,DISP, LOAD,DLOAD,
     &  ID      , IDOF     , CAST     , JTFLG    , COORD    ,
     &  COSINE  , JTCOS    )

      IF (ABORT) RETURN
C
C=============================================================
C= GET TOTAL GLOBAL LOADS AND DISPL                        ==
C=============================================================
      DO 259 I=1,L4
        LOAD(I)=LOAD(I)+DLOAD(I) + RLOAD(I)
        RLOAD(I) = RLOAD(I) + DRLOA(I)
        DISP(I)=DISP(I)+DDISP(I)
        FG(I)  =FG(I)  +DFG(I)
  259   CONTINUE
      DO 240 I=L5,L6
        LOAD(I)=LOAD(I)+DLOAD(I)
        DISP(I)=DISP(I)+DDISP(I)
        FG(I)  =FG(I)  +DFG(I)
  240   CONTINUE
C
C..... GET MAXIMUM LOADS AND DISPL
      DO 1007 I=L1,L6
       IF (ABS(DISP(I)).GT.ABS(DSPMAX(I))) DSPMAX(I)=DISP(I)
       IF (ABS(LOAD(I)).GT.ABS(RCTMAX(I))) RCTMAX(I)=LOAD(I)
 1007   CONTINUE
C
C=========================================
C= PRINT TOTAL GLOBAL DISPLACEMENTS ===
C=========================================
      IF (WRITE) THEN
        IF (IPDISP+LINE.LE.60 ) THEN
          LINE=IPDISP+LINE
        ELSE
          LINE=IPDISP+4
          WRITE (6,250) TITLE(1),TITLE(2),     ISTEP,STEPID
        ENDIF

  250 FORMAT ('1 STRUCTURE....: ',A,/
     &        '  SOLUTION.....: ',A,//
     &        2X,  '  LOADING #',I5,5X,A)
        CALL JOIDSPC1,MAXNOD,NDOF,NNODE,1,L5,L6,ID,
     &           IDOF,CNST,DISP,
     &           TITLE,'DISPLACEMENTS',STEPID,.FALSE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
      ENDIF
C
C
C=========================
C= PRINT TOTAL GLOBAL REACTIONS ===
C=========================
      IF (WRITE) THEN
        IF (IPRCTN+LINE.LE.60 ) THEN
          LINE=IPRCTN+LINE+4
        ELSE
          LINE=IPRCTN+8
          WRITE (6,250) TITLE(1),TITLE(2),     ISTEP,STEPID
        ENDIF
      ENDIF

      CALL REACTNC3,WRITE,NNODE,L1,L4,L5,L6,LOAD,DISP,
     &  MD,STIFF,IMD,NDOF,1,1.0,
     &  MAXNOD,IDOF,COORD,ID,TITLE,STEPID,.FALSE.,
     &  NCOS,COSINE,JTCOS,JTFLG,SUMRC)
        SUMFOR=SQRT(SUMRC(1)**2+SUMRC(2)**2+SUMRC(5)**2)
        SUMMOM=SQRT(SUMRC(4)**2+SUMRC(5)**2+SUMRC(6)**2)
        SUMFO=MAX(SUMFOR,SUMFO)
        SUMMO=MAX(SUMMOM,SUMMO)
      IF (WRITE) WRITE (6,315) SUMFOR,SUMMOM
  315 FORMAT (/4X,'RESULTANT OF REACTIONS,   FORCE=',1P,G12.4,
     &                                    MOMENT=',G12.4)
C
      DO 1008 I=1,6
        IF (ABS(SUMRC(I)).GT.ABS(SUMRCMX(I))) SUMRCMX(I)=SUMRC(I)
 1008 CONTINUE
C
C=========================================
C= CALCULATE AND PRINT TOTAL MEMBER FORCES ===
C=========================================
      IOPT=5
      LSTTYP=0
      HEAD =.FALSE.
      IF (WRITE) THEN
          PRINT=.TRUE.
          IF (IPELEM+LINE.LE.60 ) THEN
            LINE=IPELEM+LINE
          ELSE
            LINE=IPELEM+4
            WRITE (6,250) TITLE(1),TITLE(2),     ISTEP,STEPID
          ENDIF
      ELSE
          PRINT=.FALSE.
      ENDIF
      DO 500 IELNO=1,NELMT
  300   CALL ELELIB
     & (IOPT  ,LSTTYP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &  IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDL0A,MSTOR)
C
C=============================================
C= PRINT MEMBER DUCTILITY AND EXCURSION RATIOS ===
C=============================================
      IOPT=11
      LSTTYP=0
      HEAD =.FALSE.
      IF (WRITE) THEN
          PRINT=.TRUE.
          IF (IPELEM+LINE.LE.60 ) THEN
            LINE=IPELEM+LINE
          ELSE
            LINE=IPELEM+4
            WRITE (6,250) TITLE(1),TITLE(2),     ISTEP,STEPID
          ENDIF
      ELSE
          PRINT=.FALSE.
      ENDIF
      DO 510 IELNO=1,NELMT
  510   CALL ELELIB
     & (IOPT  ,LSTTYP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,
     &  IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDOSP,IZDL0A,MSTOR)
C
C
C=========================
C= WRITE DATA TO OUTPUT FILES  ===
C=========================
      IF (IWRITE.GT.0) THEN
        IF (MOD(ISTEP,IWRITE).EQ.0) THEN
          IOPT=2
          CALL OMPDAT(IOPT,IWRITE,TO,DT)
        ENDIF
      ENDIF
C
C=============================================================
C= EITHER GET AN NEW FACTOR, OR KEEP WORKING ON THE CURRENT ONE ==
C=============================================================
      FACTL=FACTL+DF
      IF (NEWFAC) GO TO 1000
      GO TO 1100
C
C=========================
C= CLOSE OUTPUT FILES  ===
C=========================
 5000 IF (IWRITE.GT.0) THEN
          IOPT=3
          CALL OMPDAT(IOPT,IWRITE,TO,DT)
      ENDIF
C
C=========================
C= PRINT MAXIMUM  DISPLACEMENTS ===
C=========================
      LINE=IPDISP+4
      WRITE (6,250) TITLE(1),TITLE(2),0,'MAXIMUM DISPLACEMENTS'
      CALL JOIDSPC2,MAXNOD,NDOF,NNODE,1,L5,L6,ID,
     &           IDOF,CNST,DSPMAX,
     &           TITLE,'DISPLACEMENTS','MAXIMUM VALUES',.FALSE.,
     &           NCOS,COSINE,JTCOS,JTFLG)
C
C=========================
```

```
C= PRINT MAXIMUM GLOBAL REACTIONS ===                                  SOL06060
C=====================================                                 SOL06070
      WRITE (6,250) TITLE(1),TITLE(2),0,'MAXIMUM REACTIONS'            SOL06080
        CALL REACTN(4,.TRUE.,NNODE,L1,L4,L5,L6,RCTMAX,DSPMAX,          SOL06090
     &       MD,STIFF,IMD,NDOF,1,1.0,                                  SOL06100
     &       MAXNOD,IDOF,COORD,ID,TITLE,'MAXIMUM VALUES',.FALSE.,      SOL06110
     &       NCOS,COSINE,JTCOS,JTFLG,SUMRCN)                           SOL06120
      WRITE (6,5111) SUMMFO,SUMMMO                                     SOL06130
 5111 FORMAT (/4X,'MAXIMUM RESULTANT OF REACTIONS,   FORCE=',1P,G12.4,  SOL06140
     &                                 '   MOMENT=',G12.4)            SOL06150
C                                                                      SOL06160
C=====================================                                 SOL06170
C= PRINT MAXIMUM MEMBER FORCES ===                                     SOL06180
C=====================================                                 SOL06190
      LSTTVP=0                                                         SOL06200
      HEAD =.FALSE.                                                    SOL06210
      PRINT=.TRUE.                                                     SOL06220
      WRITE (6,250) TITLE(1),TITLE(2),0,'PEAK ELEMENT FORCES'          SOL06230
      DO 5300 IELNO=1,NELMT                                            SOL06240
 5300   CALL ELELIB                                                    SOL06250
     &  (12    ,LSTTVP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,     SOL06260
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDDSP,IZDLOA,MSTOR)   SOL06270
C                                                                      SOL06280
C=============================================                         SOL06290
C= PRINT DUCTILITIES AND EXCURSION RATIOS ===                          SOL06300
C=============================================                         SOL06310
      LSTTVP=0                                                         SOL06320
      HEAD =.FALSE.                                                    SOL06330
      PRINT=.TRUE.                                                     SOL06340
      WRITE (6,250) TITLE(1),TITLE(2),0,                               SOL06350
     &    'PEAK DUCTILITIES AND EXCURSION RATIO''S'                    SOL06360
      DO 5310 IELNO=1,NELMT                                            SOL06370
 5310   CALL ELELIB                                                    SOL06380
     &  (11    ,LSTTVP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,DAMAGE,     SOL06390
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDDSP,IZDLOA,MSTOR)   SOL06400
C                                                                      SOL06410
C============================                                          SOL06420
C= PRINT DAMAGE INDICIES  ===                                          SOL06430
C============================                                          SOL06440
      LSTTVP=0                                                         SOL06450
      HEAD =.FALSE.                                                    SOL06460
      PRINT=.TRUE.                                                     SOL06470
      WRITE (6,250) TITLE(1),TITLE(2),0,                               SOL06480
     &    'DAMAGE INDEX '                                              SOL06490
      DAMAGE=0                                                         SOL06500
      DO 5320 IELNO=1,NELMT                                            SOL06510
        CALL ELELIB                                                    SOL06520
     &  (14    ,LSTTVP ,PRINT,IREL ,HEAD  ,NAME ,EESE,EPSE,EDAM,      SOL06530
     &   IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDDSP,IZDLOA,MSTOR)   SOL06540
 5320   DAMAGE=DAMAGE+EDAM                                             SOL06550
      DAMAGE=DAMAGE/(ESE+PSE)                                          SOL06560
      WRITE (6,5330) DAMAGE                                            SOL06570
 5330 FORMAT (//10X,'STRUCTURE DAMAGE INDEX=',1P,G15.5)                SOL06580
      RETURN                                                           SOL06590
C                                                                      SOL06600
      END                                                              SOL06610
C=======================================================================STR00050
      SUBROUTINE STRENG (ESE,PSE,PSEOLD,P,PL,D,DL,S,SE)                STR00060
C                                                                      STR00070
      IMPLICIT REAL(A-H,O-Z)                                           STR00080
      DIMENSION ESE(3)                                                 STR00090
C-----------------------------------------------------------------------STR00100
C VARIABLES:                                                           STR00110
C-----------------------------------------------------------------------STR00120
C TSE    = TOTAL ELEMENT STRAIN ENERGY                                 STR00130
C ESE(1) = ELASTIC ELEMENT STRAIN ENERGY                               STR00140
C ESE(2) = MAXIMUM ELASTIC ELEMENT STRAIN ENERGY, POSITIVE LOADING     STR00150
C ESE(3) = MAXIMUM ELASTIC ELEMENT STRAIN ENERGY, NEGATIVE LOADING     STR00160
C PSE    = PLASTIC ELEMENT STRAIN ENERGY                               STR00170
C PSEOLD = PLASTIC ELEMENT STRAIN ENERGY AT LAST ZERO CROSSING         STR00180
C P      = CURRENT LOAD                                                STR00190
C PL     = PREVIOUS LOAD                                               STR00200
C D      = CURRENT DISPLACEMENT                                        STR00210
C DL     = PREVIOUS DISPLACEMENT                                       STR00220
C S      = CURENT STIFFNESS                                            STR00230
C SE     = UNLOADING STIFFNESS                                         STR00240
C-----------------------------------------------------------------------STR00250
C                                                                      STR00260
      PSEL  = PSE                                                      STR00270
C             PREVENT UNDERFLOW FOR VERY SMALL PSUM OR DDIF            STR00280
            PSUM=P+PL                                                  STR00290
            DDIF=D-DL                                                  STR00300
            TOL=1.00E-25                                               STR00310
            IF (ABS(PSUM).LE.TOL .OR. ABS(DDIF).LE.TOL) THEN           STR00320
               DTSE=0                                                  STR00330
            ELSE                                                       STR00340
               DTSE  = PSUM*DDIF/2.00                                  STR00350
            ENDIF                                                      STR00360
      TSE   = ESE(1)+ PSE + DTSE                                       STR00370
C               PREVENT UNDERFLOW FOR VERY SMALL P, OR  SE=0           STR00380
            IF (ABS(P).LE.TOL) THEN                                    STR00390
               ESE(1)=0.                                               STR00400
            ELSE IF (SE.EQ.0) THEN                                     STR00410
               ESE(1)=TSE                                              STR00420
            ELSE                                                       STR00430
               ESE(1)= P**P/(2*SE)                                     STR00440
            ENDIF                                                      STR00450
      ESE(1)= MIN(TSE,ESE(1))                                          STR00460
      PSE   = TSE-ESE(1)                                               STR00470
C                                                                      STR00480
C........ CALC PLASTIC STRAIN ENERGY AT ZERO CROSSING ...              STR00490
      IP =SIGN(1.,P )                                                  STR00500
      IPL=SIGN(1.,PL)                                                  STR00510
      IF (IP*IPL.LE.0) THEN                                            STR00520
         DP=P-P                                                        STR00530
         DP=PSEL+PSE                                                   STR00540
         IF (ABS(DP).GT.TOL .AND. ABS(P).GT.TOL .AND.                  STR00550
     &       ABS(DP).GT.TOL                        ) THEN              STR00560
            PSEOLD=PSE+ DP**P/DP                                       STR00570
         ELSE                                                          STR00580
            PSEOLD=PSE                                                 STR00590
         ENDIF                                                         STR00600
      ENDIF                                                            STR00610
C                                                                      STR00620
C........ DETERMINE MAXIMUM ELASTIC STRAIN ENERGY/SIDE...              STR00630
      IF (P.GE.0) THEN                                                 STR00640
         ESE(2)=MAX(ESE(2),ESE(1))                                     STR00650
      ELSE                                                             STR00660
         ESE(3)=MAX(ESE(3),ESE(1))                                     STR00670
      ENDIF                                                            STR00680
C                                                                      STR00690
      RETURN                                                           STR00700
      END                                                              STR00710
C=======================================================================STR00040
      SUBROUTINE STRUCT                                                STR00050
      DIMENSION NOINP(8,2),RINPUT(100),GINPUT(100),DUCT(3),EXCR(5)     STR00060
      CHARACTER*80 NAME,TYPE                                           STR00070
      DIMENSION SEC(100)                                               STR00080
      LOGICAL FIRST,PT,HEAD,TEST,AXIAL,HEAD2,BTEST                     STR00090
      INCLUDE (ZCOMN)                                                  STR00100
      WRITE (6,5) TITLE(1),TITLE(2)                                    STR00110
    5 FORMAT ('1 STRUCTURE.....: ',A,/                                 STR00120
     &        ' SOLUTION......: ',A,/)                                 STR00130
      WRITE (6,6)                                                      STR00140
    6 FORMAT (' *** PROGRAM FEN ***          DOUBLE PRECISION VERSION'//)STR00150
C                                                                      STR00160
C------- INPUT # OF NODES, # OF SUPTS, # OF CONST., AND SCALE FACTOR   STR00170
      READ (5,*) NNODE,NCOS,NSUPT,NCOND,NCONST,SCALE                   STR00180
      MAXNOD=NNODE                                                     STR00190
C                                                                      STR00200
      NLOAD=1                                                          STR00210
C=====================                                                 STR00220
C= INPUT NODE DATA ====                                                STR00230
C=====================                                                 STR00240
C       ------ NODE DATA STORAGE ------------------------             STR00250
C            NZ(IZID  ) = ID NUMBERS FOR EACH NODE                     STR00260
C            NZ(IZIDOF) = DEGREE OF FREEDOM FOR EACH NODE              STR00270
C             Z(IZCORD) = GLOBAL COORDINATE OF EACH NODE (X,Y,Z)       STR00280
C             Z(IZCNST) = GLOBAL CONSTRAINT MATRIX FOR EACH NODE       STR00290
C            NZ(IZWK  ) = TEMPORARY STORAGE                            STR00300
C                                                                      STR00310
      IZID   = IZ                                                      STR00320
      IZ     = IZ+MAXNOD                                               STR00330
      IZIDOF = IZ                                                      STR00340
      IZ     = IZ+MAXNOD*6                                             STR00350
      IZCORD = IZ                                                      STR00360
      IZ     = IZ+MAXNOD*3                                             STR00370
      IZCOS  = IZ                                                      STR00380
      IZ     = IZ+MAXNOD*9                                             STR00390
      IZCNST = IZ                                                      STR00400
      IZ     = IZ+MAXNOD*6                                             STR00410
      IZJFLG = IZ                                                      STR00420
      IZ     = IZ+MAXNOD                                               STR00430
      IZJCOS = IZ                                                      STR00440
      IZ     = IZ+MAXNOD                                               STR00450
      IZWK   = IZ                                                      STR00460
      IZWK2  = IZ+MAXNOD*6                                             STR00470
C                                                                      STR00480
      CALL NODDOF(NDOF,NCOND,NFREE,NREST,MAXNOD,NNODE,NCOS,            STR00490
     &    SCALE,NSUPT,NCONST,                                          STR00500
     &    NZ(IZID),NZ(IZIDOF),NZ(IZWK2),Z(IZCORD),Z(IZCOS),Z(IZCNST),  STR00510
     &    NZ(IZJFLG),NZ(IZJCOS),BTEST(IBUG,0),NZ(IZWK) )               STR00520
C                                                                      STR00530
C------- SET UP TEMPORARY ARRAY FOR MAIN DIAGONAL INDICIES...          STR00540
         IZMD =IZ                                                      STR00550
         IZ   =IZ+NDOF+1                                               STR00560
         CALL SKYLIN(1,NDOF,IBUG,TITLE,NZ(IZMD),                       STR00570
     &          IELDOF,NZ(IZLM),Z(IZKE),MD,'GLOBAL STIFFNESS')        STR00580
C                                                                      STR00590
C                                                                      STR00600
C=====================                                                 STR00610
C= INPUT MATERIAL DATA =                                               STR00620
C=====================                                                 STR00630
      READ (5,*) NMAT                                                  STR00640
C       ------ MATERIAL DATA STORAGE ------------------               STR00650
C            NZ(IZMAT     )=          # OF MATERIAL PORPERTIES (NMAT) STR00660
C            NZ(IZMAT+  1)=          ABSOLUTE ADDRESS OF MATL #1      STR00670
C            NZ(IZMAT+  2)=          ABSOLUTE ADDRESS OF MATL #2      STR00680
C            NZ(IZMAT+  3)=          ABSOLUTE ADDRESS OF MATL #3      STR00690
C                  .                        .                         STR00700
C                  .                        .                         STR00710
C                  .                        .                         STR00720
C            NZ(IZMAT+NMAT )=        ABSOLUTE ADDRESS FO MATL # NMAT  STR00730
C             Z(NZ(IZMAT+1))=        MATERIAL DATA FOR MATERIAL #1    STR00740
C                  .                        .                         STR00750
C                  .                        .                         STR00760
C                  .                        .                         STR00770
C             Z(NZ(IZMAT+2))=        MATERIAL DATA FOR MATERIAL #2    STR00780
C                  .                        .                         STR00790
C                  .                        .                         STR00800
C                  .                        .                         STR00810
C                                                                      STR00820
C                                                                      STR00830
C                                                                      STR00840
      IOPT=1                                                           STR00850
      IZMAT =IZ                                                        STR00860
      IZ    =IZ +NMAT+1                                                STR00870
      NZ(IZMAT)=NMAT                                                   STR00880
      PT=.TRUE.                                                        STR00890
      FIRST=.TRUE.                                                     STR00900
      HEAD=.TRUE.                                                      STR00910
      MAT   =0                                                         STR00920
      LSTTVP=0                                                         STR00930
C------------ LOOP FOR EACH MATERIAL ------------------------------    STR00940
  110 MAT   =MAT   +1                                                  STR00950
         READ (5,*) TYPE                                               STR00960
         BACKSPACE (5)                                                 STR00970
C                                                                      STR00980
         IF (TEST(TYPE,'3D-BEAM',.FALSE.)) THEN                        STR00990
            MATTYP= 1                                                  STR01000
         ELSE IF (TEST(TYPE,'AXLMOD' ,.FALSE.)) THEN                   STR01010
            MATTYP= 2                                                  STR01020
         ELSE IF (TEST(TYPE,'BEND1'  ,.FALSE.)) THEN                   STR01030
            MATTYP= 3                                                  STR01040
         ELSE IF (TEST(TYPE,'SHEAR1' ,.FALSE.)) THEN                   STR01050
            MATTYP= 4                                                  STR01060
         ELSE IF (TEST(TYPE,'ROMBERG',.FALSE.)) THEN                   STR01070
            MATTYP= 5                                                  STR01080
         ELSE IF (TEST(TYPE,'TAKEDA' ,.FALSE.)) THEN                   STR01090
            MATTYP= 6                                                  STR01100
         ELSE IF (TEST(TYPE,'ELSPLS' ,.FALSE.)) THEN                   STR01110
            MATTYP= 7                                                  STR01120
         ELSE IF (TEST(TYPE,'BILINEAR',.FALSE.)) THEN                  STR01130
            MATTYP= 7                                                  STR01140
C                                                                      STR01150
C.......... TO ADD ADDITIONAL MATERIALS, COPY THE NEXT TWO LINES       STR01160
C..             CHANGE THE FOLLOWING VALUES                            STR01190
C..             (1) YOUR MATERIAL NAME                                 STR01200
C..             (2) MATERIAL ID NUMBER = XX                            STR01210
C..             ADD CALL MATXX TO SUBPROGRAM   MATLIB                  STR01220
C..             ADD SUBPROGRAM MATXX                                   STR01230
C..........                                                            STR01240
         ELSE IF (TEST(TYPE,'END',.FALSE.)) THEN                       STR01250
            NMAT=MAT-1                                                 STR01260
            NZ(IZMAT)=NMAT                                             STR01270
            GO TO 120                                                  STR01280
         ELSE                                                          STR01290
            WRITE (6,*) 'INVALID MATERIAL TYPE: ',HEAD                 STR01500
            GO TO 110                                                  STR01510
```

```
         ENDIF
C
C.......RESERVE STORAGE
       NZ(IZMAT+MAT   )=IZ
       IZWK=IZ+200
C
C.......CALL MATERIAL LIBRARY TO INITIALIZE MATERIAL DATA
       CALL MATLIB
     & (IOPT  ,LSTTYP ,PT    ,IREL,HEAD,NAME,EESE,EPSE,DUCT,EXCR,
     & P     ,PL    ,D     ,DL    ,V     ,VL    ,LELEM ,LMAT  ,
     & MAT   ,MATTYP,SE,-1,DAMAGE)
C
C.......RESERVE STORAGE
       IZ=IZ+LMAT+1
C
      IF (MAT   .LT.NMAT) GO TO 110
  120 CONTINUE
C
C=====================================
C= INPUT GEOMETRIC STIFFNESS DATA ===
C=====================================
C   Z(IZKGDT+0) = VERTICAL GROUND ACCELERATION
C  NZ(IZKGDT+1) = TYPE OF AXIAL LOAD USED
C               = 0, GEOMETRIC STIFFNESS IS NOT CONSIDERED
C               = 1, LOAD= F(INPUT)*(1.0 + ACCEL)
C               = 2, PREVIOUS LOAD STEP'S AXIAL LOAD IS USED
C  NZ(IZKGDT+2) = TYPE OF ELEMENT GEOMETRIC STIFFNESS TO BE USED
C               = 0, GEOMETRIC STIFFNESS IS NOT CONSIDERED
C               = 1, LUMPED FORMULATION
C               = 2, CONSISTENT FORMULATION
C  NZ(IZKGDT+3) = FORM OF SOLUTION
C               = 0, GEOMETRIC STIFFNESS IS NOT CONSIDERED
C               = 1, KG IS SUBTRACTED FROM ELEMENT STIFFNESS
C               = 2, SEPERATE GLOBAL KG IS FORMED
C  KGCOND       = LOGICAL FLAG, TRUE IF KG IS TO BE CONDENSED
C
      IZKGDT=IZ
      IZ     =IZ +4
      READ (5,*)  (NZ(IZKGDT+I),I=1,3) ,KGCOND
      I=IZKGDT
      IF (NZ(I+1).EQ.0 .OR. NZ(I+2).EQ.0 .OR. NZ(I+3).EQ.0) THEN
         NZ(I+1)=0
         NZ(I+2)=0
         NZ(I+3)=0
       ELSE
         WRITE (6,150) TITLE(1),TITLE(2)
         IF (NZ(I+1).EQ.1)  WRITE (6,160)
     &    'KGLOAD   = 1, LOAD= F(INPUT)*(1.00+ACCEL)',
     &    '                  F(INPUT) IS POSITIVE IN COMPRESSION',
     &    '                  ACCEL IS POSITIVE GLOBAL Z-AXIS ACCELERATION'
         IF (NZ(I+1).EQ.2)  WRITE (6,160)
     &       'KGLOAD   = 2, PREVIOUS LOAD STEP''S AXIAL LOAD IS USED'
C
         IF (NZ(I+2).EQ.1)  WRITE (6,160)
     &       'KGTYPE   = 1, LUMPED FORMULATION   '
         IF (NZ(I+2).EQ.2)  WRITE (6,160)
     &       'KGTYPE   = 2, CONSISTENT FORMULATION '
C
C
         IF (NZ(I+3).EQ.1)  WRITE (6,160)
     &       'KGFORM   = 1, KG IS SUBTRACTED FROM ELEMENT STIFFNESS '
         IF (NZ(I+3).EQ.2)  WRITE (6,160)
     &       'KGFORM   = 2, SEPERATE GLOBAL KG IS FORMED'
C
         WRITE (6,160) ' '
         IF (     KGCOND)  WRITE (6,160)
     &     '- THE GEOMETRIC STIFFNESS MATRIX IS CONDENSED '//
     &                'WITH THE STRUCTURAL STIFFNESS MATRIX.'
         IF (.NOT.KGCOND)  WRITE (6,160)
     &     '- THE GEOMETRIC STIFFNESS MATRIX IS NOT CONDENSED '//
     &                'WITH THE STRUCTURAL STIFFNESS MATRIX.'
       ENDIF
C
  150 FORMAT ('1 STRUCTURE.....: ',A,/
     &        '  SOLUTION.....: ',A,//
     &     5X,' GEOMETRIC STIFFNESS DATA '/
     &     5X,'=========================='/)
  160 FORMAT(5X,A)
C
C
C=================================================
C= SET UP STORAGE FOR GEOMETRIC STIFFNESS DATA ===
C=================================================
C      ---- GLOBAL STIFFNESS STORGE ----------------------
C           NZ(IZMDKG) = INDICIES OF THE MAIN DIAGONAL TERMS
C             Z(IZKG  ) = GLOBAL GEOMETRIC STIFFNESS MATRIX
       IF (NZ(IZKGDT+3).EQ.2) THEN
          IZMDKG=IZ
          IZ     =IZ +NDOF + 1
          CALL SKYLINK(1,NDOF,IBUG,TITLE,NZ(IZMDKG),
     &   KGDOF,NZ(IZLMKG),Z(IZKEKG),MD,'GLOBAL GEOMETRIC STIFFNESS')
       ELSE
C         DUMMY LOCATION FOR KG TO AVOID ADDRESSING EXCEPTIONS
C          KG DATA IS NOT STORED IN THESE LOCATIONS...
          IZMDKG=IZKGDT
          IZKG  =IZKGDT
       ENDIF
C
C=======================
C= INPUT ELEMENT DATA ==
C=======================
       READ (5,*) NELMT
C      ---- ELEMENT STORAGE ------------------------------
C           NZ(IZELE      )=          # OF MATERIAL PORPERTIES (NELMT)
C           NZ(IZELE+   1)=          ABSOLUTE ADDRESS OF ELEMENT #1
C           NZ(IZELE+   2)=          ABSOLUTE ADDRESS OF ELEMENT #2
C           NZ(IZELE+   3)=          ABSOLUTE ADDRESS OF ELEMENT #3
C                 .                                .
C                 .                                .
C                 .                                .
C           NZ(IZMAT+NELMT)=         ABSOLUTE ADDRESS FOR ELEM # NELMT
C            Z(NZ(IZELE+1))=         MATERIAL DATA FOR ELEMENT #1
C                 .                                .
C                 .                                .
C                 .                                .
C            Z(NZ(IZELE+2))=         MATERIAL DATA FOR ELEMENT #1
C                 .                                .
C                 .                                .
C                 .                                .
C
       IOPT=1
       IZELE =IZ
       IZ     =IZ +NELMT+1
       NZ(IZELE)=NELMT
       FIRST=.TRUE.
       HEAD=.TRUE.
       IGEN=0
       IELNO=0
       LSTTYP=0
C------------ LOOP FOR EACH ELEMENT ------------------------------
  210 IELNO=IELNO+1
        IF (IGEN.LE.0) THEN
          READ (5,*) TYPE
C
          IF (TEST(TYPE,'3D-BEAM',.FALSE.)) THEN
            NINPUT= 10
            NINZ  = 305
            NELTYP= 1
C
          ELSE IF (TEST(TYPE,'SPRING',.FALSE.)) THEN
            NINPUT= 10
            NINZ  = 155
            NELTYP= 2
C
          ELSE IF (TEST(TYPE,'SHEAR WALL',.FALSE.)) THEN
            NINPUT= 9
            NINZ  = 769
            NELTYP= 3
C
C..... TO ADD ADDITIONAL ELEMENTS, COPY THE NEXT FOUR LINES
          ELSE IF (TEST(TYPE,'(1)',.FALSE.)) THEN
            NINPUT= (2)
            NINZ  = (5)
            NELTYP= (4)
C..                     CHANGE THE FOLLOWING VALUES
C..                     (1) YOUR ELEMENT NAME
C..                     (2) # OF VALUES TO BE INPUT
C..                     (5) # OF STORAGE LOCATIONS REQD FOR ELEMENT
C..                     (4) ELEMENT ID NUMBER = XX
C..                     ADD CALL ELEXX TO SUBPROGRAM   ELE_LIB
C..                     ADD SUBPROGRAM ELEXX
C.............
          ELSE IF (TEST(TYPE,'END',.FALSE.)) THEN
            NELMT=IELNO-1
            NZ(IZELE)=NELMT
            GO TO 500
          ELSE
            WRITE (6,*) 'INVALID ELEMENT TYPE: ',HEAD
            GO TO 210
          ENDIF
          BACKSPACE (5)
          READ (5,*) TYPE,NAME,       ( RINPUT(I),I=1,NINPUT),IGEN
          IF (IGEN.GT.0)  READ (5,*)  ( GINPUT(I),I=1,NINPUT)
        ELSE
          IGEN=IGEN-1
          NAME='GENERATED'
          DO 220 I=1,NINPUT
  220       RINPUT(I)=RINPUT(I)+GINPUT(I)
        ENDIF
  111 FORMAT (OP,' ZMATRIX',I6,'):',I15,1P,G20.10)
C.......RESERVE STORAGE
        NZ(IZELE+IELNO)=IZ
        IC=IZ
        IZWK=IZ
C
C.......STORE ELEMENT TYPE
        NZ(IC) =NELTYP
C
C.......CALL ELEMENT LIBRARY TO DETERMINE THE STORAGE FOR MATL DATA
        IOPT=9
        HEAD2=.FALSE.
        LSTTP2=0
        CALL ELELIB
     & (IOPT  ,LSTTP2 ,.FALSE.,IREL,HEAD2, NAME , EESE,EPSE,DAMAGE,
     &  IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)
C
C.......RESERVE STORAGE
        IZ=IZ+NINZ+MSTOR
        IZWK=IZ
C
C.......CALL ELEMENT LIBRARY TO INITIALIZE ELEMENT DATA AND RETURN
C.......INFO ON ELEMENT DOF...
        IOPT=1
        CALL ELELIB
     & (IOPT  ,LSTTYP ,.TRUE.,IREL,HEAD,NAME , EESE,EPSE,DAMAGE,
     &  IELNO ,IELDOF,KGDOF,PGEOM,RINPUT,AXIAL,IZDISP,IZLOAD,MSTOR)
C
C.......RELEASE UNUSED STORAGE .....
        IZ=IZ-IREL
C
C.......DUMP ELEMENT DATA IF DEBUGING
        IF (BTEST(IBUG,8)) THEN
         WRITE (6,*) 'IZLM=',IZLM,' IZKE=',IZKE
         JJ=-1
         DO 7000 I=IC,(IZWK-1-IREL)
           JJ=JJ+1
 7000      IF (NZ(I).NE.0) WRITE (6,7010) IELNO,JJ,I,NZ(I),Z(I)
 7010    FORMAT (' IELNO=',I3,'  #',I6,'   Z(',I6,') =',1P,G18.8)
        ENDIF
C
C--------- RESERVE SPACE IN THE SKYLINE FOR THE STIFFNESS
        CALL SKYLINK(2,NDOF,IBUG,TITLE,NZ(IZMD),
     &            IELDOF,NZ(IZLM),Z(IZKE),MD,'GLOBAL STIFFNESS')
C
C--------- RESERVE SPACE IN THE SKYLINE FOR THE GEOMETRIC STIFFNESS
       IF (NZ(IZKGDT+3).EQ.1 .AND. AXIAL) THEN
          CALL SKYLINK(2,NDOF,IBUG,TITLE,NZ(IZMD),
     &    KGDOF,NZ(IZLMKG),Z(IZKEKG),MD,'GLOBAL GEOMETRIC STIFFNESS')
       ELSE IF (NZ(IZKGDT+3).EQ.2 .AND. AXIAL) THEN
          CALL SKYLINK(2,NDOF,IBUG,TITLE,NZ(IZMDKG),
     &    KGDOF,NZ(IZLMKG),Z(IZKEKG),MD,'GLOBAL GEOMETRIC STIFFNESS')
        ENDIF
C
        IF (IELNO.LT.NELMT) GO TO 210
  500 CONTINUE
C
C
C=============================================================
C= MODIFY GEOMETRIC STIFFNESS MATRIX FOR CONDENSATION  ===
C=============================================================
C THE CONDENSATION PROCESS MAY EXPAND THE GEOMETRIC STIFFNESS MATRIX'S
C SKYLINE.  THE MAXIMUM SKYLINE OF BOTH THE GEOMETRIC STIFFNESS AND
C THE STIFFNESS MATRIX IS USED.
       IF (NZ(IZKGDT+3).EQ.2 .AND. KGCOND .AND. NCOND.GT.0)THEN
         DO 310 I=0,NDOF
           J=I+1
           MDKG1=NZ(IZMDKG+I)
           NZ(IZMDKG+I)=MIN( NZ(IZMDKG+I),NZ(IZMD+I) )
```

```
  310     CONTINUE                                                      STR05870
        ENDIF                                                           STR05880
C                                                                       STR05890
C                                                                       STR05900
C=====================================                                  STR05910
C= SET UP GLOBAL STIFFNESS STORAGE ==                                   STR05920
C=====================================                                  STR05930
C             NZ(IZMD)   = INDICIES OF THE MAIN DIAGONAL TERMS          STR05940
C             ZC(IZSTIF) = GLOBAL STIFFNESS MATRIX                      STR05950
C                                                                       STR05960
      CALL SKYLINK(5,NDOF,IBUG,TITLE,NZ(IZMD),                          STR05970
     &             IELDOF,NZ(IZLM),ZC(IZKE),MD,'GLOBAL STIFFNESS')      STR05980
      IZSTIF=IZ                                                         STR05990
      IZ     = IZ+NZ(IZMD+NDOF) +1                                      STR06000
C                                                                       STR06010
C=================================================                      STR06020
C= SET UP STORAGE FOR GEOMETRIC STIFFNESS DATA ===                      STR06030
C=================================================                      STR06040
C             NZ(IZMDKG) = INDICIES OF THE MAIN DIAGONAL TERMS          STR06050
C              ZC(IZKG  ) = GLOBAL GEOMETRIC STIFFNESS MATRIX           STR06060
      IF (NZ(IZXGDT+5).EQ.2) THEN                                       STR06070
         CALL SKYLINK(5,NDOF,IBUG,TITLE,NZ(IZMDKG),                     STR06080
     &   KGDOF,NZ(IZLMKG),ZC(IZKEKG),MD,'GLOBAL GEOMETRIC STIFFNESS')   STR06090
         IZKG  =IZ                                                      STR06100
         IZ    =IZ +NZ(IZMDKG+NDOF)                                     STR06110
      ENDIF                                                             STR06120
C                                                                       STR06130
C                                                                       STR06140
C=======================                                               STR06150
C= INPUT MASS MATRIX ===                                                STR06160
C=======================                                               STR06170
C     NMASS = NUMBER OF NODAL MASSES TO BE INPUT                        STR06180
C     FMASS = 0, NO MASS MATRIX                                         STR06190
C             1, MASS MATRIX DUE TO NODE MASSES ONLY                    STR06200
C     MCOND = LOGICAL FLAG, TRUE IF MASS IS TO BE CONDENSED...          STR06210
C                                                                       STR06220
      READ (5,*) INMASS,FMASS,MCOND                                     STR06230
C                                                                       STR06240
      CALL MASSINK(INMASS,FMASS,MCOND,BUG,IBUG,TITLE,MAXZ,MAXDZ,        STR06250
     &     MAXNOD,MCOS,NNODE,IZ,IW,IZID,Z,NZ,DZ,                        STR06260
     &     IZMDMS,IZMASS,IZLM,IZKE,IZMDAT,                              STR06270
     &     NDOF,NMASS,IZCOS,IZIDOF,IZCNST,IZJFLG,IZJCOS,'GLOBAL MASS',  STR06280
     &     MCOND,NFREE,IZMD)                                            STR06290
      IF (INMASS.GT.0) THEN                                             STR06300
         WRITE (6,160) ' '                                              STR06310
         IF (MCOND) THEN                                                STR06320
            WRITE (6,160) '- THE MASS MATRIX IS '//                     STR06330
     &      'CONDENSED WITH THE STRUCTURAL STIFFNESS MATRIX.'           STR06340
         ELSE                                                           STR06350
            WRITE (6,160) '- THE MASS MATRIX IS NOT '//                 STR06360
     &      'CONDENSED WITH THE STRUCTURAL STIFFNESS MATRIX.'           STR06370
         ENDIF                                                          STR06380
      ELSE                                                              STR06390
         WRITE (6,499)                                                  STR06400
  499    FORMAT(//5X,' ZERO MASS MATRIX '/                              STR06410
     &          5X,'==================')                                STR06420
      ENDIF                                                             STR06430
C                                                                       STR06440
C=======================                                               STR06450
C= INPUT DAMPING DATA ===                                               STR06460
C=======================                                               STR06470
C     NDAMP = NUMBER OF NODAL DAMPING COIEF TO BE INPUT ≠0              STR06480
C     FDAMP = 0, NO DAMPING MATRIX                                      STR06490
C             1, ALPHA AND BETA INPUT                                   STR06500
C                                                                       STR06510
      NDAMP=0                                                           STR06520
      FDAMP=1                                                           STR06530
C     READ (5,*) NDAMP,FDAMP                                            STR06540
      IF (FDAMP.EQ.1) THEN                                              STR06550
        IZDAMP=IZ                                                       STR06560
        IZ=IZ + 2                                                       STR06570
        READ (5,*) ALPHA,BETA                                           STR06580
        ZC(IZDAMP)   =ALPHA                                             STR06590
        ZC(IZDAMP+1) =BETA                                              STR06600
        WRITE(6,500) ALPHA,BETA                                         STR06610
      ENDIF                                                             STR06620
  500 FORMAT(//5X,' PROPORTIONAL DAMPING COIEFICIENTS'/                 STR06630
     &         5X,'================================='/,                 STR06640
     &         5X,'ALPHA=',1P,G13.5,'   BETA=',G13.5/)                  STR06650
C                                                                       STR06660
C=================================                                      STR06670
C= ZERO LOAD AND DISPL INDICIES ===                                     STR06680
C=================================                                      STR06690
      IZLOAD=0                                                          STR06700
      IZDISP=0                                                          STR06710
      IZVEL =0                                                          STR06720
      IZACC =0                                                          STR06730
      IZDLOA=0                                                          STR06740
      IZDDSP=0                                                          STR06750
      IZDVEL=0                                                          STR06760
      IZDACC=0                                                          STR06770
C                                                                       STR06780
      RETURN                                                            STR06790
      END                                                               STR06800
C=======================================================================TES00010
      LOGICAL FUNCTION TEST(OPTION,TARGET,DEFLT)                        TES00040
      CHARACTER*(*) OPTION,TARGET                                       TES00050
      LOGICAL DEFLT,BTEST                                               TES00060
         J= INDEX(OPTION,TARGET)                                        TES00070
         IF (DEFLT) THEN                                                TES00080
              IF (J.LT.2) THEN                                          TES00090
                   TEST=.TRUE.                                          TES00100
              ELSE IF (OPTION(J-2:J-1).NE.'NO') THEN                    TES00110
                   TEST=.TRUE.                                          TES00120
              ELSE                                                      TES00130
                   TEST=.FALSE.                                         TES00140
              ENDIF                                                     TES00150
         ELSE                                                           TES00160
              IF (J.EQ.0) THEN                                          TES00170
                   TEST=.FALSE.                                         TES00180
              ELSE IF (J.LT.2 ) THEN                                    TES00190
                   TEST=.TRUE.                                          TES00200
              ELSE IF (OPTION(J-2:J-1).EQ.'NO') THEN                    TES00210
                   TEST=.FALSE.                                         TES00220
              ELSE                                                      TES00230
                   TEST=.TRUE.                                          TES00240
              ENDIF                                                     TES00250
         ENDIF                                                          TES00260
      RETURN                                                            TES00270
      END                                                               TES00280
C=======================================================================TMP00010
C   TRIANGLAR MATRIX MULTIPLICATION ....                                TMP00050
C   MATRIX A IS AN UPPER TRIANGULAR MATRIX, WITH MAIN DIAGONAL ADDRESS  TMP00060
C   MATRIX MD.  X IS A VECTOR.  P=AX                                    TMP00070
C   THE MULTIPLICATION IS CARRIED OUT FROM ROW IR1 TO IR2  AND FROM     TMP00080
C   COLUMN JC1 TO JC2.                                                  TMP00090
C=======================================================================TMP00100
      SUBROUTINE TMPLY(PRINT,ZERO,IMD,N,IR1,IR2,JC1,JC2,MD,A,P,X)       TMP00110
      DIMENSION A(IMD),X(N),P(N),MD(N+1)                                TMP00120
      LOGICAL BTEST,PRINT,ZERO                                          TMP00130
C                                                                       TMP00140
      IJI(I,J)=MD(J)+J-I                                                TMP00150
      IF (PRINT) THEN                                                   TMP00160
         WRITE (6,500) 'TRIANGULAR MULTIPLACATION',                     TMP00170
     &                 IR1,IR2,JC1,JC2,N                                TMP00180
         CALL LMATRX(A,MD,N,MD(N+1),'MATRIX A ')                        TMP00190
      ENDIF                                                             TMP00210
C       ---- ZERO P ----                                                TMP00220
      IF (ZERO) THEN                                                    TMP00230
        DO 520 J=IR1,IR2                                                TMP00240
  520     P(J)=0                                                        TMP00250
      ENDIF                                                             TMP00260
C                                                                       TMP00400
C-------- MULTIPLY TERMS ABOVE AND INCLUDING THE MAIN DIAGONAL          TMP00410
        DO 20 J=JC1,JC2                                                 TMP00420
          IRC=J-MD(J+1)+MD(J)+1                                         TMP00450
          MDJ=MD(J)+J                                                   TMP00440
          I1=MAX(IRC,IR1)                                               TMP00450
          I2=MIN(J,IR2)                                                 TMP00460
          IF (I2.GE.I1) THEN                                            TMP00470
            DO 10 I=I1,I2                                               TMP00480
   10         P(I)=P(I)+A(MDJ-I)*X(J)                                   TMP00490
          ENDIF                                                         TMP00500
   20   CONTINUE                                                        TMP00510
C                                                                       TMP00520
C-------- MULTIPLY TERMS BELOW THE MAIN DIAGONAL                        TMP00530
        DO 30 I=IR1,IR2                                                 TMP00540
          IRC=I-MD(I+1)+MD(I)+1                                         TMP00550
          MDI=MD(I)+I                                                   TMP00560
          J1=MAX(IRC,JC1)                                               TMP00570
          J2=MIN(I-1,JC2)                                               TMP00580
          IF (J2.GE.J1) THEN                                            TMP00590
            DO 40 J=J1,J2                                               TMP00600
   40         P(I)=P(I)+A(MDI-J)*X(J)                                   TMP00610
          ENDIF                                                         TMP00620
   30   CONTINUE                                                        TMP00630
C                                                                       TMP00640
  500 FORMAT (/2X,A/,2X,' IR1:',I5,' IR2:',I5,                          TMP00650
     &        ' JC1:',I5,' JC2:',I5,' N:',I5)                           TMP00660
      IF (PRINT) THEN                                                   TMP00670
        DO 515 I=IR1,IR2                                                TMP00680
  515     WRITE(6,520) I,X(I),P(I)                                      TMP00690
        WRITE (6,*) ' '                                                 TMP00700
  520 FORMAT (2X,'DOF:',I6,'  X(DOF)',1P,G14.5,' A*X=P(DOF):',G14.5)    TMP00710
      ENDIF                                                             TMP00720
      RETURN                                                            TMP00730
      END                                                               TMP00740
C=======================================================================UTM00050
C    UPPER TRIANGLAR MATRIX MULTIPLICATION ....                         UTM00060
C   MATRIX A IS AN UPPER TRIANGULAR MATRIX, WITH MAIN DIAGONAL ADDRESS  UTM00070
C   MATRIX MD.  X IS A VECTOR.  P=AX                                    UTM00080
C   THE MULTIPLICATION IS CARRIED OUT FROM ROW L1 TO L2  AND FROM       UTM00090
C  COLUMN L1 TO N.                                                      UTM00100
C=======================================================================UTM00110
      SUBROUTINE UTMPLY(PRINT,ZERO,IMD,N,IR1,IR2,JC1,JC2,MD,A,P,X)      UTM00120
      DIMENSION A(IMD),X(N),P(N),MD(N+1)                                UTM00140
      LOGICAL BTEST,PRINT,ZERO                                          UTM00150
C                                                                       UTM00160
      IJI(I,J)=MD(J)+J-I                                                UTM00170
      IF (PRINT) THEN                                                   UTM00180
         WRITE (6,500) 'UPPER TRIANGULAR MULTIPLACATION',               UTM00190
     &                 IR1,IR2,JC1,JC2,N                                UTM00200
         CALL LMATRX(A,MD,N,MD(N+1),'MATRIX A ')                        UTM00210
      ENDIF                                                             UTM00230
C       ---- ZERO P ----                                                UTM00240
      IF (ZERO) THEN                                                    UTM00250
        DO 520 I=L1,L2                                                  UTM00260
  520     P(I)=0                                                        UTM00270
      ENDIF                                                             UTM00280
C                                                                       UTM00290
C-------- MULTIPLY TERMS ABOVE AND INCLUDING THE MAIN DIAGONAL          UTM00300
        DO 20 J=JC1,JC2                                                 UTM00310
          IRC=J-MD(J+1)+MD(J)+1                                         UTM00320
          MDJ=MD(J)+J                                                   UTM00330
          I1=MAX(IRC,IR1)                                               UTM00340
          I2=MIN(J,IR2)                                                 UTM00350
          IF (I2.GE.I1) THEN                                            UTM00360
            DO 10 I=I1,I2                                               UTM00370
   10         P(I)=P(I)+A(MDJ-I)*X(J)                                   UTM00380
          ENDIF                                                         UTM00390
   20   CONTINUE                                                        UTM00400
C                                                                       UTM00410
C---------- MULTIPLY S_RED * DISP = P                                   UTM00420
C                                                                       UTM00430
  500 FORMAT (/2X,A/,2X,' IR1:',I5,' IR2:',I5,                          UTM00520
     &        ' JC1:',I5,' JC2:',I5,' N:',I5)                           UTM00530
      IF (PRINT) THEN                                                   UTM00540
        DO 515 I=IR1,IR2                                                UTM00550
  515     WRITE(6,520) I,X(I),P(I)                                      UTM00560
        WRITE(6,*) ' '                                                  UTM00570
  520 FORMAT (2X,'DOF:',I6,'  X(DOF)',1P,G14.5,'  A*X=P(DOF):',G14.5)   UTM00580
      ENDIF                                                             UTM00590
      RETURN                                                            UTM00600
      END                                                               UTM00610
C=======================================================================VC000010
      REAL FUNCTION VCOS(V1,V2)                                         VC000040
      DIMENSION V1(5),V2(5)                                             VC000050
      VALUE (A1,A2,A5)=SQRT(A1**2+A2**2+A5**2)                          VC000060
      DOT   (A1,B1,C1,A2,B2,C2)=A1*A2+B1*B2+C1*C2                       VC000070
C                                                                       VC000080
      AV1=VALUE(V1(1),V1(2),V1(5))                                      VC000090
      AV2=VALUE(V2(1),V2(2),V2(5))                                      VC000100
      VDOT=DOT(V1(1),V1(2),V1(5) ,V2(1),V2(2),V2(5))                    VC000110
      VCOS=VDOT/(AV1*AV2)                                               VC000120
      RETURN                                                            VC000150
      END                                                               VC000140
C=======================================================================VCR00010
C      V3 = V1 X V2,  VECTOR CROSS PRODUCT ...                          VCR00020
C=======================================================================VCR00050
      SUBROUTINE  VCROSS(V3,R1,R2)                                      VCR00040
      DIMENSION V1(5),V2(5),V3(5)                                       VCR00050
      DIMENSION R1(5),R2(5)                                             VCR00060
C                                                                       VCR00070
      DO 10 I=1,5                                                       VCR00080
      V1(I)=R1(I)                                                       VCR00090
```

```
   10 V2(I)=R2(I)                                                       VCR00100
C                                                                       VCR00110
      V3(1) = V1(2)*V2(3) - V1(3)*V2(2)                                 VCR00120
      V3(2) = V1(3)*V2(1) - V1(1)*V2(3)                                 VCR00130
      V3(3) = V1(1)*V2(2) - V1(2)*V2(1)                                 VCR00140
C                                                                       VCR00150
      RETURN                                                            VCR00160
      END                                                               VCR00170
C=======================================================================MMA00010
      SUBROUTINE MMATRX(A,IR,IC,NAME)                                   MMA00040
      DIMENSION A(IR,IC)                                                MMA00050
      CHARACTER*(*) NAME                                                MMA00060
      LOGICAL DMP,BTEST                                                 MMA00070
C                                                                       MMA00080
      CALL GETINT('NAME,'DUMP=',IDUMP,0,,TRUE.,,FALSE.)                 MMA00090
      IF (IDUMP.NE.0)  THEN                                             MMA00100
          DMP=.TRUE.                                                    MMA00110
      ELSE                                                              MMA00120
          DMP=.FALSE.                                                   MMA00130
      ENDIF                                                             MMA00140
C                                                                       MMA00150
     *WRITE (6,*) ' '                                                   MMA00160
      WRITE (6,*) 'PRINTING MATRIX: ',NAME                              MMA00170
      DO 50 ICG=1,IC,6                                                  MMA00180
         IF (ICG.GT.1) WRITE (6,*) ' '                                  MMA00190
         K=MIN(ICG+5),IC)                                               MMA00200
         WRITE (6,10) (J,J=ICG,K)                                       MMA00210
         DO 20 I=1,IR                                                   MMA00220
           IF (DMP) THEN                                                MMA00230
             DO 15 J=ICG,K                                              MMA00240
  15         IF (A(I,J).NE.0) WRITE (IDUMP,*) I,J,A(I,J),NAME           MMA00250
             ENDIF                                                      MMA00260
  20         WRITE (6,30)  I,(A(I,J),J=ICG,K)                           MMA00270
  50  CONTINUE                                                          MMA00280
      RETURN                                                            MMA00290
  10  FORMAT(' COLUMN',T17,I5,5(15X,I5))                                MMA00500
  30  FORMAT(' ROW',I4,6F20.6)                                          MMA00510
      END                                                               MMA00520
C=======================================================================MMT00010
      SUBROUTINE MMTRX2(A,IR,IC,NAME,IROW)                              MMT00040
      DIMENSION A(IROW,IC)                                              MMT00050
      CHARACTER*(*) NAME                                                MMT00060
      LOGICAL DMP,BTEST                                                 MMT00070
C                                                                       MMT00080
      CALL GETINT('NAME,'DUMP=',IDUMP,0,,TRUE.,,FALSE.)                 MMT00090
      IF (IDUMP.NE.0)  THEN                                             MMT00100
          DMP=.TRUE.                                                    MMT00110
      ELSE                                                              MMT00120
          DMP=.FALSE.                                                   MMT00130
      ENDIF                                                             MMT00140
C                                                                       MMT00150
      WRITE (6,*) ' '                                                   MMT00160
      WRITE (6,*) 'PRINTING MATRIX: ',NAME                              MMT00170
      DO 50 ICG=1,IC,6                                                  MMT00180
         IF (ICG.GT.1) WRITE (6,*) ' '                                  MMT00190
         K=MIN(ICG+5),IC)                                               MMT00200
         WRITE (6,10) (J,J=ICG,K)                                       MMT00210
         DO 20 I=1,IR                                                   MMT00220
           IF (DMP) THEN                                                MMT00230
             DO 15 J=ICG,K                                              MMT00240
  15         IF (A(I,J).NE.0) WRITE (IDUMP,*) I,J,A(I,J),NAME           MMT00250
             ENDIF                                                      MMT00260
  20         WRITE (6,30)  I,(A(I,J),J=ICG,K)                           MMT00270
  50  CONTINUE                                                          MMT00280
      RETURN                                                            MMT00290
  10  FORMAT(' COLUMN',T17,I5,5(15X,I5))                                MMT00500
  30  FORMAT(' ROW',I4,6F20.6)                                          MMT00510
      END                                                               MMT00520
```

C. PROGRAM SEE

FILE: SEE FORTRAN

```
C=======================================================================SEE00010
C         DEBUG UNIT(6),TRACE,SUBCHK,INIT,SUBTRACE                      SEE00020
C         END DEBUG                                                     SEE00030
      PARAMETER (MMAX=1500)                                             SEE00040
      LOGICAL PAXIS                                                     SEE00050
      CHARACTER*20 LABEL(3)                                             SEE00060
      CHARACTER*80 TITLE                                                SEE00070
      CHARACTER*42 TEMP                                                 SEE00080
      DIMENSION VX(3),VV(3),VZ(3),X(4),Y(4)                             SEE00090
      DIMENSION S(3,MMAX),SR(3,MMAX),KEY(MMAX),RMAX(3),RMIN(3)          SEE00100
      DIMENSION E(3,MMAX),ER(3,MMAX)                                    SEE00110
      CROSSI(A1,B1,C1,A2,B2,C2)=B1*C2-B2*C1                             SEE00120
      CROSSJ(A1,B1,C1,A2,B2,C2)=C1*A2-C2*A1                             SEE00130
      CROSSK(A1,B1,C1,A2,B2,C2)=A1*B2-A2*B1                             SEE00140
      VALUE (A1,A2,A3)=SQRT(A1**2+A2**2+A3**2)                          SEE00150
      DOT   (A1,B1,C1,A2,B2,C2)=A1*A2+B1*B2+C1*C2                       SEE00160
      LABEL(1)='ROTATED X AXIS'                                         SEE00170
      LABEL(2)='ROTATED Y AXIS'                                         SEE00180
      LABEL(3)='ROTATED Z AXIS'                                         SEE00190
C                                                                       SEE00200
C------ INITIALIZE THE PLOTTING DEVICE ----                            SEE00210
      CALL PLOTS(0,0,7)                                                 SEE00220
C                                                                       SEE00230
C------ INPUT & PRINT POINTS --------                                   SEE00240
      DO 10 I=1,MMAX                                                    SEE00250
      READ  (10,*,END=15) KEY(I),(S(J,I),J=1,3),(E(J,I),J=1,3)          SEE00260
   10 WRITE (6,20) KEY(I),(S(J,I),J=1,3),(E(J,I),J=1,3),'INPUT POINT'   SEE00270
   15 NUMB=I                                                            SEE00280
      IF (I.LE.MMAX) NUMB=I-1                                           SEE00290
   20 FORMAT (1X,I5,1P,6G12.4,:,A)                                      SEE00300
C                                                                       SEE00310
C------ LOOP FOR EACH VIEW ----------                                   SEE00320
      READ(5,*) NVIEW                                                   SEE00330
      DO 2000 IVIEW=1,NVIEW                                             SEE00340
C                                                                       SEE00350
C------ INPUT VIEWING DATA ----------                                   SEE00360
      READ (5,*)  TITLE,XLEN,YLEN,PAXIS,VX,VV                          SEE00370
C                                                                       SEE00380
C------ CALCULATE DIRECTION COSINES -                                   SEE00390
      VZ(1)= VX(2)*VV(3)-VX(3)*VV(2)                                    SEE00400
      VZ(2)=-VX(1)*VV(3)+VX(3)*VV(1)                                    SEE00410
      VZ(3)= VX(1)*VV(2)-VX(2)*VV(1)                                    SEE00420
      VV(1)= VZ(2)*VX(3)-VZ(3)*VX(2)                                    SEE00430
      VV(2)=-VZ(1)*VX(3)+VZ(3)*VX(1)                                    SEE00440
      VV(3)= VZ(1)*VX(2)-VZ(2)*VX(1)                                    SEE00450
      DX=VALUE(VX(1),VX(2),VX(3))                                       SEE00460
      DV=VALUE(VV(1),VV(2),VV(3))                                       SEE00470
      DZ=VALUE(VZ(1),VZ(2),VZ(3))                                       SEE00480
      DO 50 I=1,3                                                       SEE00490
        VX(I)=VX(I)/DX                                                  SEE00500
        VV(I)=VV(I)/DV                                                  SEE00510
   50   VZ(I)=VZ(I)/DZ                                                 SEE00520
C                                                                       SEE00530
C------ PRINT DIRECTION COSINES ----                                    SEE00540
      WRITE (6,*) ' VIEW NUMBER.............:',IVIEW                   SEE00550
      WRITE (6,*) ' TITLE...................: ',TITLE                  SEE00560
      WRITE (6,*) ' X-LENGTH................:',XLEN                    SEE00570
      WRITE (6,*) ' V-LENGTH................:',YLEN                    SEE00580
      WRITE (6,*) ' VX......................:',VX                      SEE00590
      WRITE (6,*) ' VV......................:',VV                      SEE00600
      WRITE (6,*) ' VZ......................:',VZ                      SEE00610
C                                                                       SEE00620
C------ CALCULATE & PRINT ROTATED COORDINATES ----                      SEE00630
      DO 100 J=1,NUMB                                                   SEE00640
        DO 40 I=1,3                                                     SEE00650
          SR(I,J)=0                                                     SEE00660
   40     ER(I,J)=0                                                     SEE00670
        DO 50 I=1,3                                                     SEE00680
          SR(1,J)= SR(1,J) + VX(I)*S(I,J)                               SEE00690
          SR(2,J)= SR(2,J) + VV(I)*S(I,J)                               SEE00700
          SR(3,J)= SR(3,J) + VZ(I)*S(I,J)                               SEE00710
          ER(1,J)= ER(1,J) + VX(I)*E(I,J)                               SEE00720
          ER(2,J)= ER(2,J) + VV(I)*E(I,J)                               SEE00730
   50     ER(3,J)= ER(3,J) + VZ(I)*E(I,J)                               SEE00740
        IF (J.EQ.1) THEN                                                SEE00750
          DO 60 I=1,3                                                   SEE00760
            RMAX(I)=MAX(SR(I,J),ER(I,J))                                SEE00770
   60       RMIN(I)=MIN(SR(I,J),ER(I,J))                               SEE00780
        ELSE                                                            SEE00790
          DO 70 I=1,3                                                   SEE00800
            RMAX(I)=MAX(SR(I,J),ER(I,J),RMAX(I))                        SEE00810
   70       RMIN(I)=MIN(SR(I,J),ER(I,J),RMIN(I))                       SEE00820
        ENDIF                                                           SEE00830
  100   WRITE (6,20) KEY(J),(SR(I,J),I=1,3),(ER(I,J),I=1,3),           SEE00840
     &          'ROTATED POINT'                                         SEE00850
C                                                                       SEE00860
C=======================================================================SEE00870
C------ PLOT POINTS. USING CALCOMP 1051 PLOTTER COMMANDS...             SEE00880
C=======================================================================SEE00890
C                                                                       SEE00900
C                                                                       SEE00910
C------ LOOP FOR DIFFENINET VIEWS ---                                   SEE00920
      DO 500 I=1,3                                                      SEE00950
        IF (I.EQ.1) THEN                                                SEE00960
            IX=1                                                        SEE00950
            IV=2                                                        SEE00960
        ELSE IF (I.EQ.2) THEN                                           SEE00970
            IX=2                                                        SEE00980
            IV=3                                                        SEE00990
        ELSE                                                            SEE01000
            IX=3                                                        SEE01010
            IV=1                                                        SEE01020
        ENDIF                                                           SEE01050
C                                                                       SEE01040
C------ DRAW TRIM LINE ---------------                                  SEE01050
      CALL PLOT(0.,11.,2)                                               SEE01060
      CALL PLOT(0., 0.,2)                                               SEE01070
C                                                                       SEE01080
C------ MOVE ORIGIN TO 0.5,0.5 ------                                   SEE01090
      CALL PLOT(.75,.5,-3)                                              SEE01100
C                                                                       SEE01110
C------ GET SCALE FACTORS FOR X-V PLOT ---                              SEE01120
      X(1)=RMIN(IX)                                                     SEE01150
      X(2)=RMAX(IX)                                                     SEE01140
      Y(1)=RMIN(IV)                                                     SEE01150
      Y(2)=RMAX(IV)                                                     SEE01160
      CALL SCALE(X,XLEN,2,1)                                            SEE01170
      CALL SCALE(Y,YLEN,2,1)                                            SEE01180
      X3=X(3)                                                           SEE01190
      X4=X(4)                                                           SEE01200
      Y3=Y(3)                                                           SEE01210
      Y4=Y(4)                                                           SEE01220
C                                                                       SEE01230
C------ PLOT AXIS ------------------------                              SEE01240
      IF (PAXIS) THEN                                                   SEE01250
          CALL AXIS(0.00,-.05,LABEL(IX),-20,XLEN, 0.,X(3),X(4))        SEE01260
          CALL AXIS(-.05,0.00,LABEL(IV), 20,YLEN,90.,Y(3),Y(4))        SEE01270
      ENDIF                                                             SEE01280
C                                                                       SEE01290
C------ PLOT MEMBERS ----------------------                             SEE01500
      DO 200 J=1,NUMB                                                   SEE01510
        X(1)=SR(IX,J)                                                   SEE01520
        X(2)=ER(IX,J)                                                   SEE01530
        Y(1)=SR(IV,J)                                                   SEE01540
        Y(2)=ER(IV,J)                                                   SEE01550
  200   CALL LINE(X,Y,2,1,0,0)                                          SEE01560
C                                                                       SEE01570
C------ PLOT TITLE ------------------------                             SEE01580
      CALL SYMBOL(0.1,(YLEN+.15),.14,TITLE ,0.,80)                      SEE01590
      TEMP=LABEL(IX)//'--'//LABEL(IV)                                   SEE01400
      CALL SYMBOL(0.1,(YLEN+.01),.05,TEMP  ,0.,42)                      SEE01410
C                                                                       SEE01420
C------ MOVE ORIGIN BACK TO 0,0 AND ADVANCE PAGE -----                  SEE01450
      CALL PLOT((XLEN+.25),-.5,-3)                                      SEE01440
C                                                                       SEE01450
  500 CONTINUE                                                          SEE01460
C                                                                       SEE01470
 1000 CONTINUE                                                          SEE01480
 2000 CONTINUE                                                          SEE01490
C                                                                       SEE01500
C------ DRAW TRIM LINE ---------------                                  SEE01510
      CALL PLOT(0.,11.,2)                                               SEE01520
      CALL PLOT(0., 0.,2)                                               SEE01530
C                                                                       SEE01540
C------ CLOSE THE PLOTTING DEVICE ----                                  SEE01550
      CALL PLOT(0.,0.,999)                                              SEE01560
C                                                                       SEE01570
      STOP                                                              SEE01580
      END                                                               SEE01590
```