NASA Contractor Report 4449

Advanced Composites Structural Concepts and Materials Technologies for Primary Aircraft Structures

Structural Response and Failure Analysis

ISPAN Modules Users Manual

J. W. Hairr, J. T. Huang,
J. E. Ingram, and B. M. Shah
Lockheed Aeronautical Systems Company
Marietta, Georgia

Prepared for
Langley Research Center
under Contract NAS1-18888

National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Program
1992

FOREWORD

This User's Manual covers the work performed in the development of ISPAN (Interactive Stiffened Panel ANalysis) Modules under Phase I, Evaluation and Initial Development, Task 2 Structural Response and Failure Analysis, of NASA Contract NAS1-18888, entitled "Advanced Composite Structural Concepts and Materials Technology for Primary Aircraft Structures", between May 1989 to May 1992. In this respect, this User's Manual compliments the final technical report of the Task 2 - Structural Response and Failure Analysis. This contract is administered under the management direction of Dr. John C. Davis and under the technical direction of Dr. Randall C. Davis, NASA/SPO, NASA Langley Research Center, Hampton, Virginia.

Lockheed Aeronautical Systems Company (LASC) is the prime contractor. Mr. Anthony C. Jackson is the LASC Program Manager, directing all the contract activities. Mr. Bharat M. Shah of Advanced Structures and Materials Division is the technical thrust leader in the performance of this task.

Key LASC (or as indicated) contributors to the ISPAN Modules Development are listed below:

Wayne Brown	(LMSC)
Tony Wei	(LMSC)
Edward Ingram	
Jui-Ten-Haung	
John Hairr	

TABLE OF CONTENTS

SECTION

TABLE OF CONTENTS (Cont'd)

LIST OF FIGURES

SUMMARY

The basic objective of the development of the ISPAN (Interactive Stiffened Panel ANalysis) Modules is to bring finite element analysis into the design of composite structures without the requirement for the user to know much about the techniques and procedures needed to actually perform a finite element analysis from scratch. The programs presented for the analysis of a variety of structural composite panels require the user to simply input basic dimensions of the panel and layup information for the laminates, with each quantity being prompted for. It also markedly simplifies parameter studies, as all analysis can be conducted using the same generic model. Parameter changes from run-to-run can be trivial with the file editing or model revision capabilities.

Section 1.0

Describes the applicable run initialization and input format for all the ISPAN Modules, including user's information in conducting bifurcation buckling and non-linear collapse analysis.

Section 2.0 through 5.0

Describes the flat stiffened panel module; the curved stiffened panel module; the flat tubular panel module: and the geodesic curved stiffened panel module, respectively. Each of the above sections describes the capabilities and restrictions of the respective modules. It also describes the inputs for an example problem using screen echo generated during an interative session. It also presents the output generated, following post-processing the results.

Section 6.0

Describes the post processing module common to all the ISPAN modules. The material properties input and input data file structure is described in Appendices A1 and A2, respectively. The ISPAN module runstream is decribed in Appendix A3 along with the source code and post-processing routines for each module.

1.0 INTRODUCTION

The ISPAN Modules is an interactive design tool that is intended to provide a means of performing simple and self-contained preliminary analysis of aircraft primary structure using composite materials. The advantage of this program lies in that it gives an inexperienced finite element code user a direct way of creating and solving a finite element model without the need to acquire in-depth knowledge about the code; in the mean time it does not limit experienced users from doing any modification and in-depth analysis once the model is created. The term "generic model has been applied to the finite element models created by this process.

This program combines a series of modules with the finite element code DIAL as its backbone. Each module can create a finite element model of a different type of "primary aircraft structure" (i.e. a wing panel, fuselage panel, etc). Each module consists of a set of FORTRAN driver programs which create DIAL generic model runstream files. The DIAL runstreams utilize the syntactic input capability of DIAL which is a FORTRAN-like language within the program. These elements of the generic model processing are transparent to the user.

Users are instructed to input geometric properties, material properties, load information and types of analysis that the user desires. Subsequently, the program would utilize this information to generate the finite element model and perform analysis. The output in the form of summary tables of stress or margins of safety, contour plots of loads or stress and deflected shape plots may be used to determine the viability of the particular design.

The scope of the analysis that can be performed by this program includes conventional linear stress analysis, bifurcation buckling analysis and nonlinear collapse analysis.

1.1 RUN INITIALIZATION AND GENERAL INPUT FORMAT

This program may be installed on any of several different interactive computers such as VAX or Silicon Graphics Personal Iris workstations. Each system requires a somewhat different setup that should be installed by the person responsible for managing this software. For the user a simple one word command is usually sufficient to start the program.

The program begins by displaying a menu showing the various generic models or modules that may be run. Once a particular module is chosen, the program is set up to either use existing data files or ask for new data to be input. Number and types of input varies with the different modules; all data are input in interactive fashion.

Geometric and material data is stored in files using either user

supplied file names or the default file names. Geometric data includes basic dimensions, number of plies, material and individual ply orientations for each layup; the material file contains moduli, Poisson ratios and stress/strain allowables for each material being used.

Users are able to modify any existing data files before or after execution. A simple text editor may be used to modify the contents of either the geometry or material files. File formats are given in the Appendix. Once the data files are completed, the program links the data to the DIAL processors automatically and begins execution of each processor. This can be done interactively or as a batch mode run. At the completion of the execution, the user would have the option of either using the SCOPE processor or using pre-defined post-processing commands to obtain analysis results. A flow diagram of the run format for a typical model is presented in Figure 1.0.

All ISPAN modules follow the same basic steps in creating and processing the analysis. They are: (Not necessarily in the sequence shown)

1) Start ISPAN
2) Input analysis information
3) Input geometry and layup information
4) Input boundary condition information
5) Input loads information
6) Input material properties
7) Run model (Batch or interactive)
8) Post-process results - make geometry plots, stress summaries, contour plots, etc.

1.2 BIFURCATION BUCKLING ANALYSIS

This option can be used to identify initial buckling modes for the structure. No knockdown factors are used. Knockdown factors, if any, need to be determined by the user and applied to the results. Sometimes it is benificial to use the bifurcation mode deflected shape as a starting point for a nonlinear collapse analysis.

1.3 NON-LINEAR COLLAPSE ANALYSIS

The capability to perform a non-linear collapse analysis is an advanced feature of DIAL which can be difficult to use even by those familiar with finite element analysis. Therefore, it is recommended only for advanced users.

This program utilizes the ARCLENGTH method in DIAL to perform non-linear analysis. This method allows users to choose either LOADS or a DEGREE OF FREEDOM as the TARGET POINT in the computation. If LOADS is chosen to be the TARGET POINT type, the input value for the TARGET POINT would be the

FIGURE 1. A TYPICAL GENERIC MODEL FLOW CHART

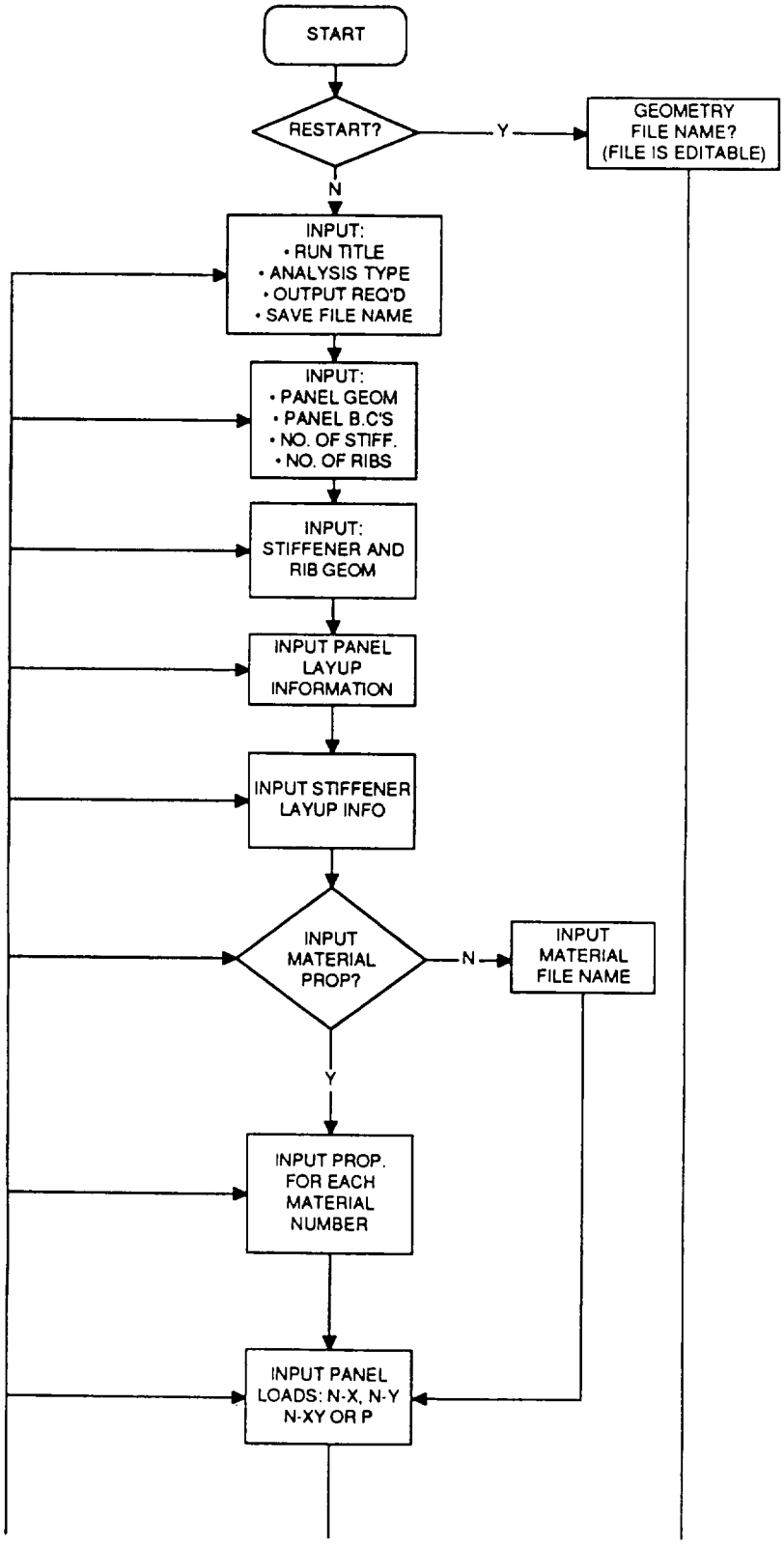
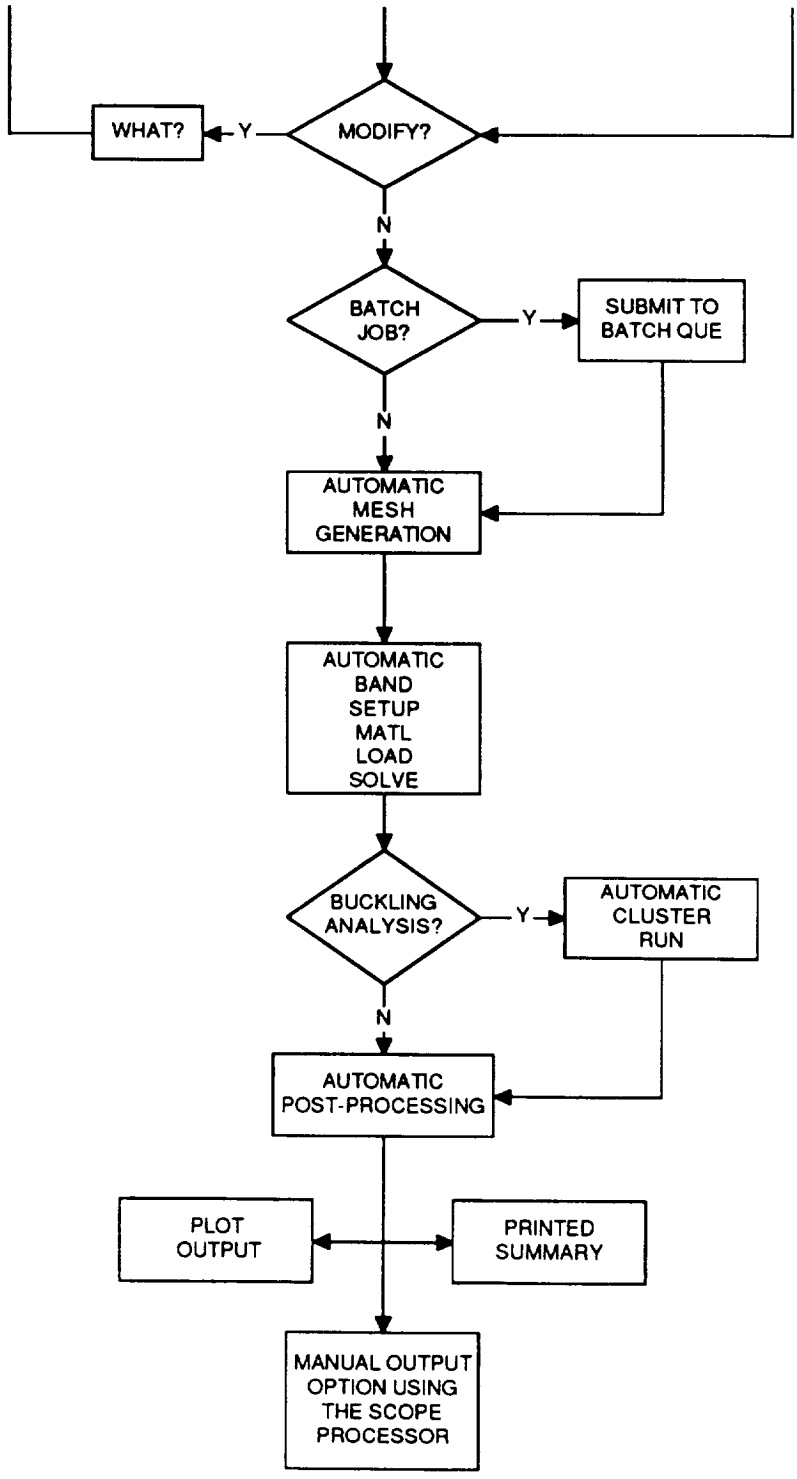


FIGURE 1 WING PANEL GENERIC MODEL FLOW CHART(CONT'D)



load factor (A multiple of the input loads) that the user wishes the program to reach. On the other hand, if any DEGREE OF FREEDOM is chosen, then the TARGET POINT would be the maximum corresponding displacement that the user would like the program to reach at a particular NODAL POINT.

During the course of the SOLVE process, stress and displacement files at each intermediate load steps as well as at the TARGET POINT would be saved in the DIAL database. This information can be retrieved by the user during the post processing session.

Since it is difficult for the user to know in advance about which part of the structure would buckle first (if at all), the DEGREE OF FREEDOM type TARGET POINT is not recommended on the first computer run for the problem on hand. It is also recommended for the user to run the finite element model through the bifurcation buckling analysis first, then use the mode shape obtained as the starting point for the non-linear analysis. This would help to save computer run time on the non-linear analysis. A procedure for this will be included in a future revision of this manual.

Users should be cautioned that a non-linear finite element analysis, depending on the size of the model and the type of machine on which this program is used, could result in the use of substantial CPU time, even on a supercomputer.

2.0 FLAT STIFFENED PANEL MODULE

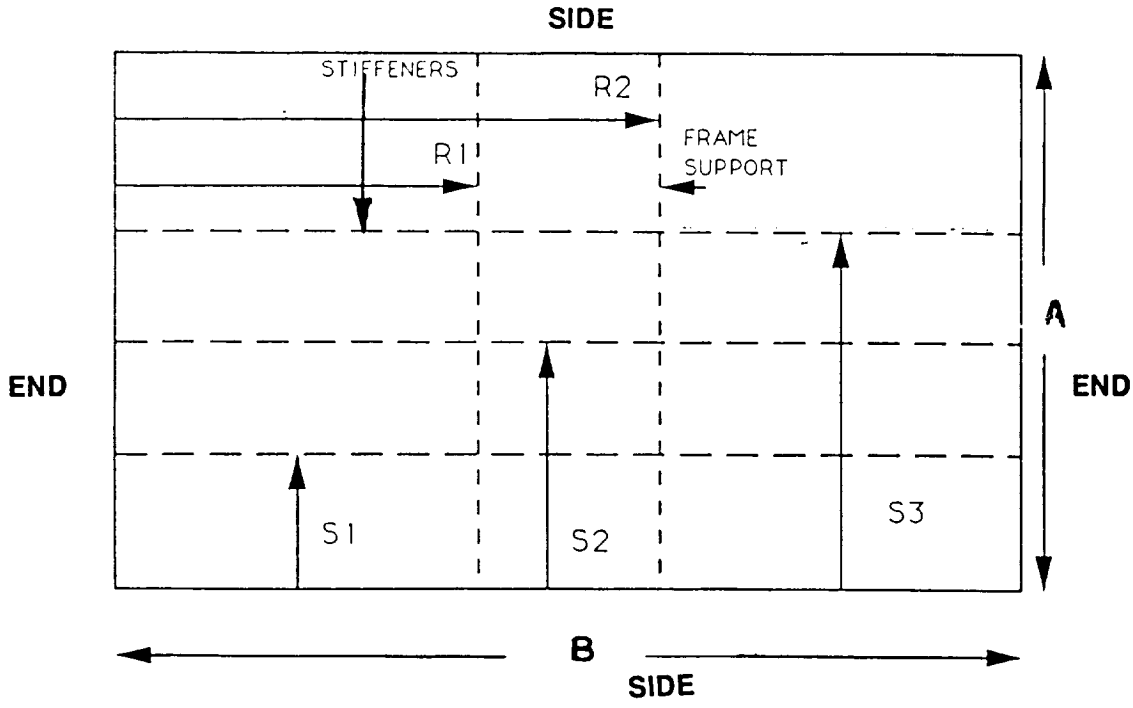
This module creates a model of a flat rectangular panel with axial stiffeners and lateral stiffeners or rib attachments (as shown in Figure 2). This model is representative of an upper or lower wing panel, but without curvature. As illustrated in Figure 3, there are eight different types of stiffener forms available. No actual stiffeners are used in the lateral direction but rib attachments are simulated with local boundary conditions input.

2.1 CAPABILITIES AND RESTRICTIONS

1) Number of axial stiffeners0<N<10
2) Stiffeners may be unequally spaced and of different cross-sections and layups
3) Number of lateral stiffeners or restraints0<N<2
4) Lateral rib or stiffener attachments may provide either a pinned or fixed type support.
5) Boundary conditions options include for the ends either simply supported or fixed against rotation and constrained or free axial displacements and simulated infinite length.

6) Boundary conditions options for the sides include simply supported, fixed rotations, or simulated infinite width.
7) Loading may include any combination of axial, lateral and

FIGURE 2. FLAT STIFFENED PANEL GEOMETRY AND LOAD.



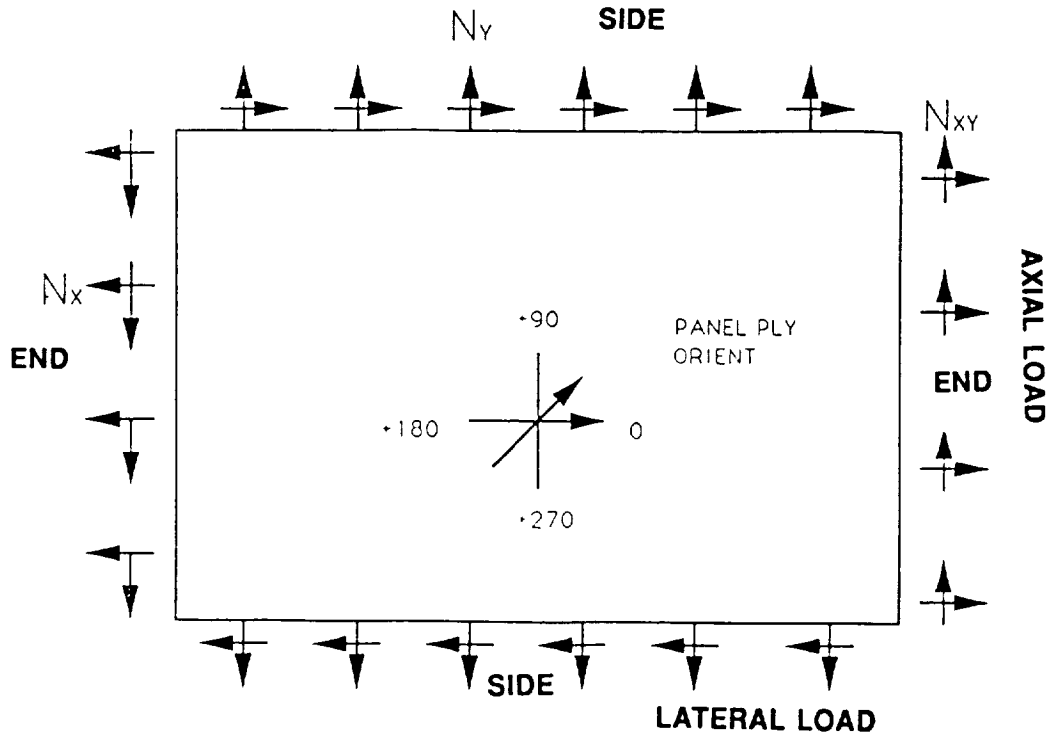
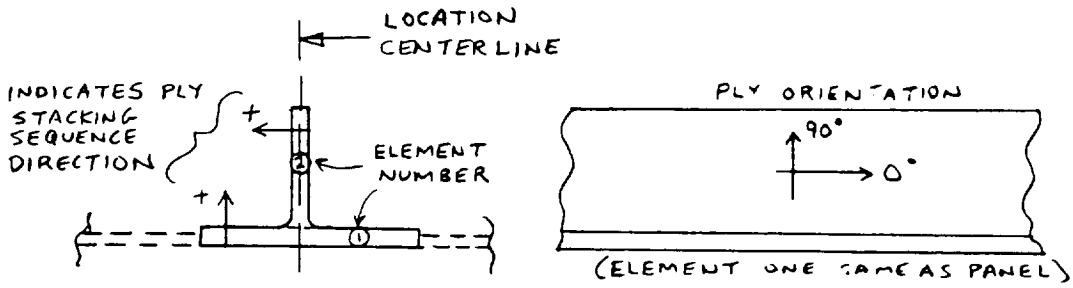
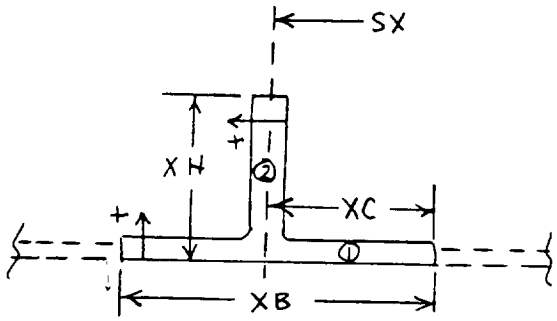


FIGURE 3. STIFFENERS GEOMETRY

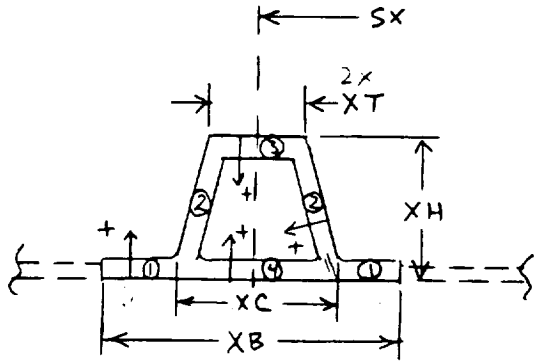
PLY ORIENTATION AND STACKING SEQUENCE



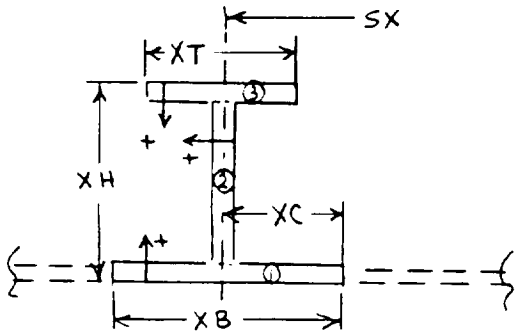
1) BLADE OR 5)ANGLE STIFFENER



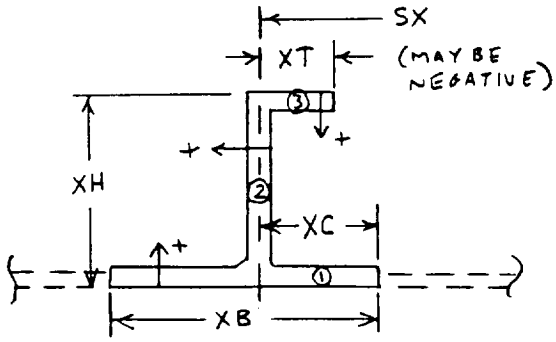
2) CLOSED HAT STIFFENER



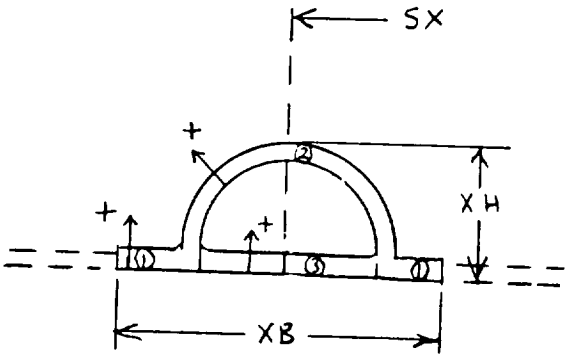
3) "I" STIFFENER



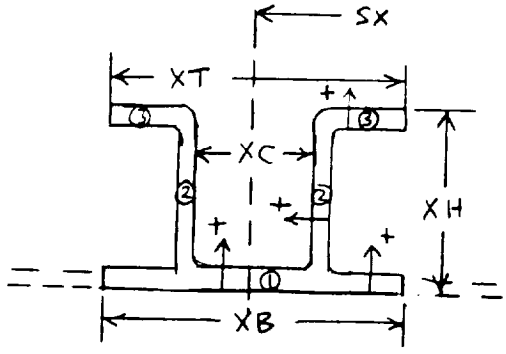
4) J OR 6)Z STIFFENER



7) BEAD STIFFENER



8) OPEN HAT STIFFENER



shear line loads and a lateral pressure.
8) Maximum number of different materials = 10
9) Maximum number of different layups = 20
10) Maximum number of plies per layup = 100
11) Option to perform either a linear stress analysis or buckling analysis. The buckling model is necessarily a more refined mesh than the stress model and as a consequence will require much longer run times.
12) Stiffener layup may include any or all panel plies as integral to the stiffener.

An example input format for this module is presented in Section 2.2. It is accompanied by diagrams which show the required input dimensions and parameters (Figures 2 and 3). A plot of the model generated for this example is shown in Figure 4. Figure 5 illustrates a panel with an alternate type of stiffener.

USAGE NOTES

1) The skin panel at the base of the stiffener is considered part of the stiffener base. The laminate input should include contributions from both sources.
2) Panel sub-divisions define the model element density which is program controlled unless requested by the user.
3) Definition of full 3-D material properties is not required since only shell elements are used. The material property input is described in Appendix A.1.

2.2 FLAT STIFFENED PANEL MODULE - EXAMPLE INPUT

Following input prompts would be shown on the terminal, if there are no pre-assigned data files when the program is called. Numbers after the prompts~ are the actual inputs for the model shown in Figure 4. User input is shown in bold type.

$ @DIALAMATIC

Advanced Composites Structural Concept and Materials Technologies

Automated DIAL Finite Element Analysis

DIALAMATIC
VAX VERSION 1.2
JUNE 1991

FIGURE 4. BLADE STIFFENED FLAT PANEL - EXAMPLE PLOT

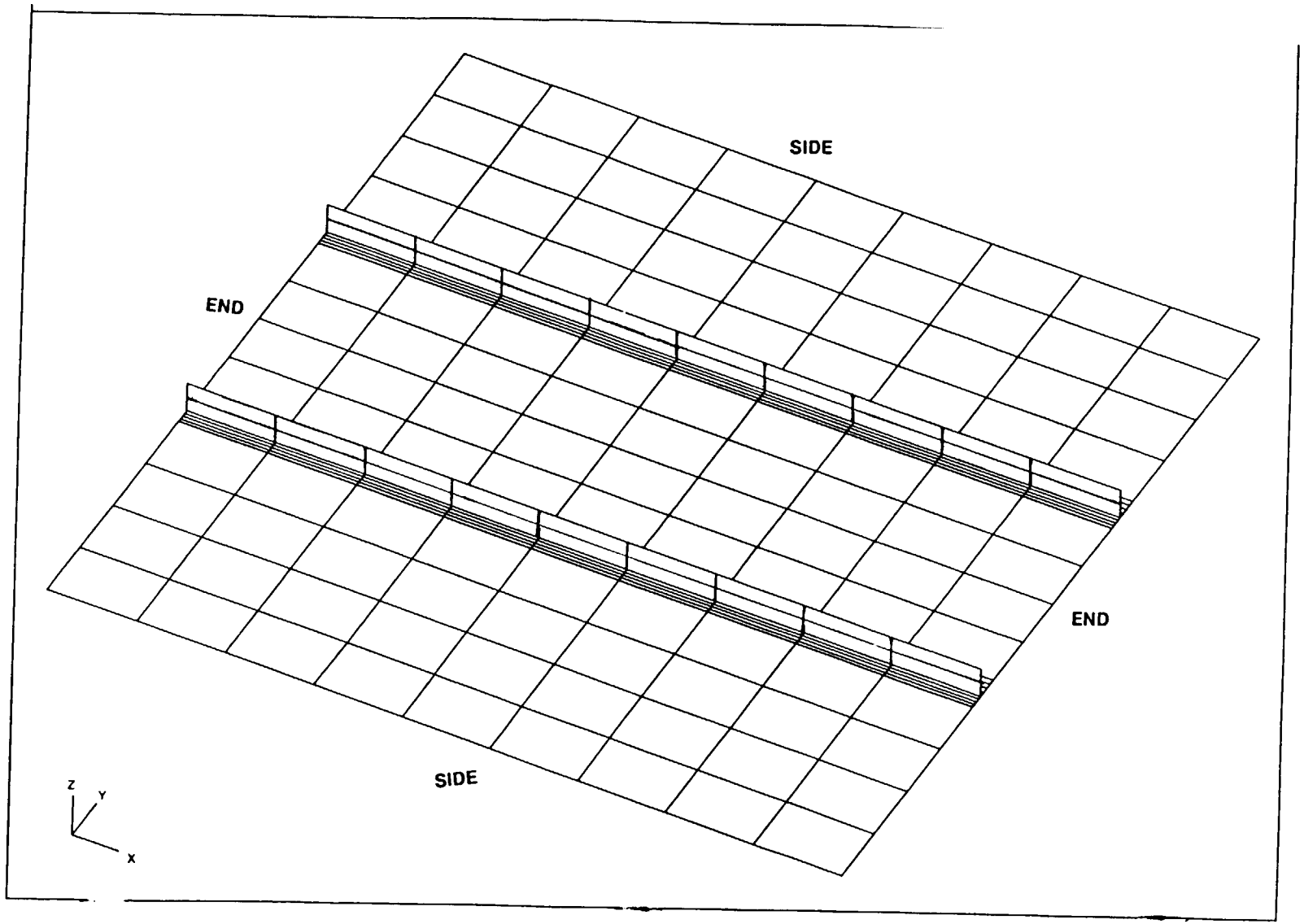
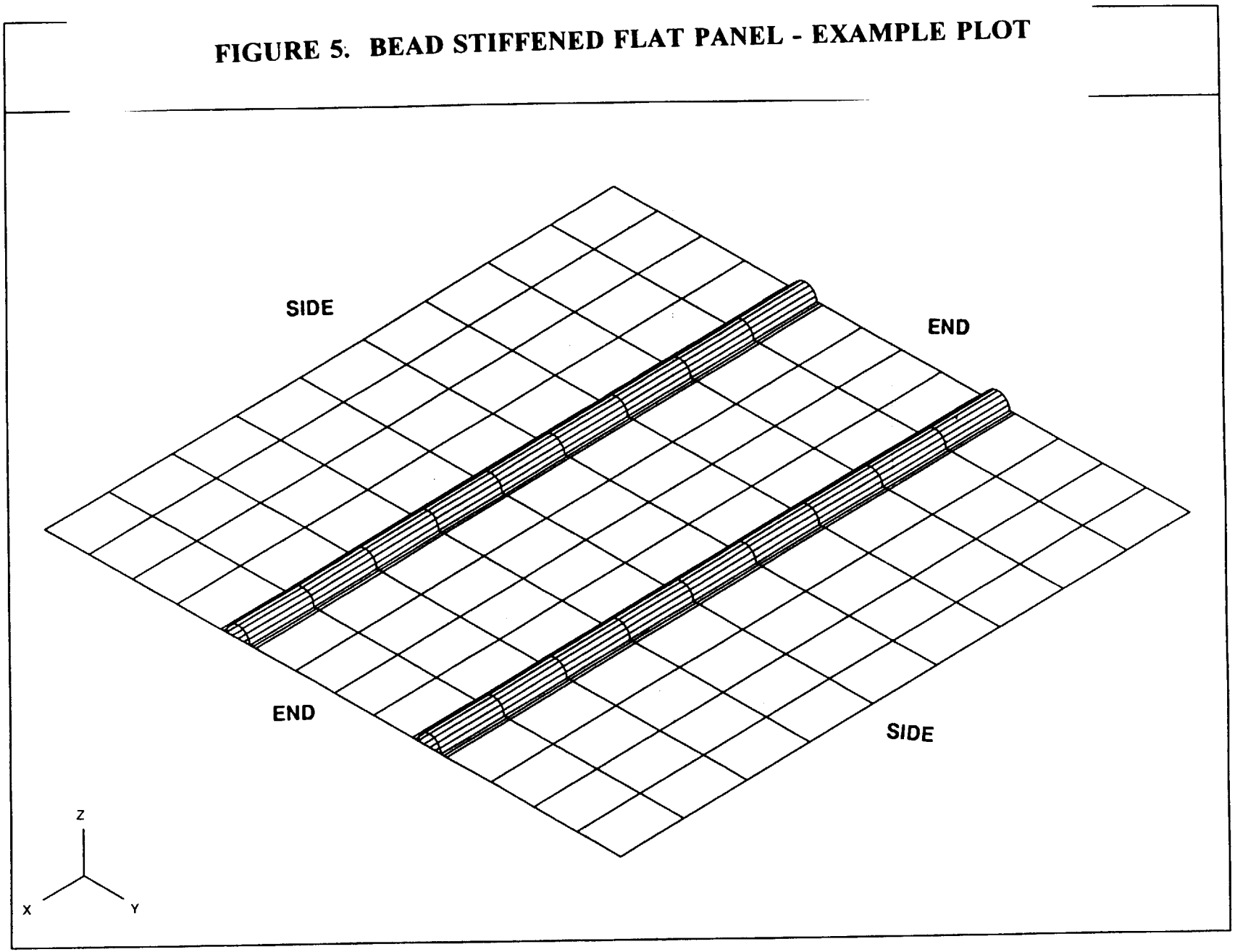


FIGURE 5. BEAD STIFFENED FLAT PANEL - EXAMPLE PLOT



```
The following analysis modules are available at this time:


MODULE NO.      GENERIC MODEL
1Flat rectangular stiffened panel (Wing panel)
2Curved Stiffened Panel
3 Flat rectangular tubular panel
4Geodesic Panel


Select a module number >>: 1


        Generic Model for Flat Stringer Stiffened Panel


                    MODULE 1
                VAX VERSION 1.2
                    JUNE 1991

BASIC ANALYSIS INFORMATION:


Does a DIAL database file exist? (Y/N) Default is FIL002.
>>: n
Would you like to give the data base file a unique name? (Y/N)
>>: f l a t
Does a geometry file already exist for this analysis? (Y/N)
>>: n
Input a name for the geometry file. >>: g1
Does a material properties file exist?(Y/N) >>: n
Input a name for the material roperties file. >>: m 1


*************************************************************
        *Advanced Composite Structural Program by LMSC
         *Flat Stiffener Panel: Revision 1.2, 6/28/91
*************************************************************


Input Panel Width and Length >>( 10.000 10.000)> 10 10
Input No. of Stiffeners and Ribs >>( 2 1)> 2 2
Does the user want to specify Panel Sub-divisions? (Y/N)
Enter >> n

 Sub-Grid Between Stiffeners are 2.
 Sub-Grid Between Ribs are 3.

    Stiffeners equally spaced (Y/N)? >>(Y)> y
    Ribs equally spaced (Y/N)? >>(Y)> y
```

```
----  Boundary Condition Definition -------

Are there ribs at Ends ? (Y/N)>>(Y)> n
Ribs on:(1)Skin (2)STIF (3) Both ?>>(1)>  3
Choose a B.C. Type no. at Ribs:
(1) Simply Supported; (2) S.S. and Rotation; ?>>(1)> 2

Choose a B.C. type no. at Ends:
(1) Part of a Larger Panel (2) Free (3) Simply Supported
(4) Semi-Clamped-1 (axial & lateral free)
(5) Semi-Clamped-2 (axial only free)
(6) Fully Clamped (all DOF fixed) --->>(1)> 1
Choose a B.C. type no. at Sides:
(1) Part of a Larger Panel
(2) Free
(3) Simply Supported
(4) Semi-Clamped-1 (axial & lateral free)
(5) Semi-Clamped-2 (axial only free)
(6) Fully Clamped (all DOF fixed) --->>(2)> 1

    Loading Condition Definition

Choose input loads type (Total=1,Line=2)?>>(2)> 1

following total load unit: lb

   Input axial load >>(1 000.0)> 1 000
   Input lateral load >>( 0.0)> 0
   Input shear load along ends >>( 0.0)> 0
   Input lateral pressure in psi >> 0.00)> 0

     Type of Analysis

Choose ONE of the following type of analysis>>
1> Linear Analysis.
2> Bifurcation Buckling Analysis.
3> Post-Buckling Analysis.
      Enter >> 3
  Type of Target Point ?>> 0
O> Load Factor.
1-6> Displacement DOF.
   Enter>> Target Point Value = 5.7

Input No. of Materials >>(1)> 1
Input Matl# 1 Elastic & Shear Modulus, Poison #, AML +/-
Allowables
    E1,E2      >>( 0.30E+08 0.30E+08)> 8.9e6 8.9e6
    G12,G23,G31 >>( 0.1 2E+08 0.1 2E+08 0.1 2E+08)~0.89e6
0.3e6 0.3e6
  V12      >>( 0.3043)> 0.095
Do you want to input AML Failure Parameters (Y/N)? y
   + Allow.>>(-60.0 0.001 0.0 0.002 60.0 0.003)> -60 0.011 0
```

```
0.022 60 0.011
   - Allow.>>(-60.0 -0.002 0.0 -0.003 60.0 -0.004)> -60
-0.011 0 -0.022 60 -0.011

     ------------------------------------------------

Input No. of plies for the basic panel >>( 2)> 8
Input Matl#, Thickness, Angle
Are the information the same for ALL plies (Y/N)? y
Ply# 1>>( 1 0.1000  0.00)> 1 0.0142 45
     ------------------------------------------------

Stiffener Type: 1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,
         5)Angle, 6)Zee, 7)Bead, 8)0pen Hat.
  Select axial stiffener type >>(1)> 1
  Input XH,XB,XC >>( 0.400 0.500 0.350)> 0.4 0.5 0.2
     ------------------------------------------------

Input No. of plies for stiffener ELEM #1>>( 2)> 8
Input Matl#, Thickness, Angle

Are the information the save for ALL plies (Y/N)? y
Ply# 1>>(1 0.1 000 0.00)> 1 0.0142 -45
     ------------------------------------------------

Input No. of plies for stiffener ELEM #2>>( 2)> 4
Input Matl#, Thickness, Angle

Are the information the save for ALL plies (Y/N)? y
Ply# 1>>(1 0.1 000 0.00)> 1 0.0142 90


     ************************************************
            *Advanced Composite Structural Program by LMSC*
             *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ************************************************


m,n) >>> MAIN MENU <<< (general MESH data)
 0,0) Geometry & General Data
 0,1) Base Panel Width and Length (AX,BY): 10.000
10.000
 0,2) No. of Stiffeners and Ribs (M,N): 2 2
 0,3) No. of GRID for Stiffeners and Ribs(Ml,NI): 2 3
 0,4) Stiffener locations (A): 3.333 6.667
 0,5) Rib locations (B): 3.333 6.667


   >>> OTHER MENUS <<< (MATL, BASE Ply, STIF Ply, BC & LC)
 1,0) Total No. of Materials (NMAT): 1
 2,0) No. of Plies for The Base Panel(NLAY): 8
 3,0) Axial stiffener type No.(MTYP): 1
```

```
Blade=1,Close Hat=2,"I"=3,"J"=4,Angle=5,"ZH=6,Bead=7,
Open
Hat=8.
    XH= 0.400 XB= 0.500 XC= 0.200
 4,0) Boundary & Loading Conditions
Select m,n=> -1:END; 9:QUIT; m,n:UPDATE ~> 1,0


   ******************************************************
            *Advanced Composite Structural Program by LMSC*
             *Flat Stiffener Panel: Revision 1.2, 6/28/91*
   ******************************************************


MATL MENU (MENU1):
 1,0) Total No. of Materials (NMAT): 1
 1,1) Material Number 1
Selection=> 1,n: modify MATL #n;
 -1: Exit; 9: Quit; m,0: other MENUS >> 1,1


    ******************************************************
      *Advanced Composite Structural Program by LMSC*
       *Flat Stiffener Panel: Revision 1.2, 6/28/91*
    ******************************************************

   Properties for MATL No. :1
  1>>Elastic Modulus (E1,E2): 0.89E+07 0.89E+07
  2>>Shear Modulus (G12,G23,G31): 0.89E+06 0.30E+06 0.30E+06
  3>>Poisson Ratio (V12): 0.0950
  4>>AML + Allowables:-60.0 0.0110 0.0 0.0220 60.0 0.0110
  5>>AML- Allowables:-60.0 -0.0110 0.0 -0.0220 60.0 -0.0110

  Modify Above Material Properties (Y/N)? >>(N)> n


    ******************************************************
      *Advanced Composite Structural Program by LMSC*
       *Flat Stiffener Panel: Revision 1.2, 6/28/91*
    ******************************************************


MATL MENU (MENU1):
 1,0) Total No. of Materials (NMAT): 1
 1,1) Material Number 1
Selection=> 1,n: modify MATL #n;
 -1: Exit; 9: Quit; m,0: other MENUS >> 2,0


    ******************************************************
             *Advanced Composite Structural Program by LMSC*
              *Flat Stiffener Panel: Revision 1.2, 6/28/91*
    ******************************************************
```

```
Base Ply MENU (MENU2):
 2,0) No. of Plies for The Base Panel (NLAY): 8
     Ply # Material # Thickness Angle
      1        1       0.0142     45.0
      2        1       0.0142     45.0
      3        1       0.0142     45.0
      4        1       0.0142     45.0
      5        1       0.0142     45.0
      6        1       0.0142     45.0
      7        1       0.0142     45.0
      8        1       0.0142     45.0

Modify the no. of plies @ the panel (Y/N)? >>(N)>
n Choose PLY NUMBER to modify (Y/N)? n Selection
=> 2,n: modify panel plies;
       -1 :End; 9:Quit; m,0: other MENUS >> 3,0



    ******************************************************
                *Advanced Composite Structural Program by
LMSC*
       *Flat Stiffener Panel: Revision 1.2, 6/28/91*
    ******************************************************

STIF MENU (MENU3): 1)Blade, 2)Close Hat,
3)I-shape, 4)J-shape, 5)Angle, 6)Zee, 7)Bead,
8)0pen Hat.

3,0) Type of Stiffener: 1
     XH= 0.400 XB= 0.500 XC= 0.200
3,1) No. of Plies for STIF,ELEM(1): 8
3,2) No. of Plies for STIF,ELEM(2): 4
Selection => 3,n: modify panel plies;
        -1:End; 9:Quit; m,0: other MENUS; -1: exit. >> 3,1



******************************************************
            *Advanced Composite Structural Program by LMSC*
             *Flat Stiffener Panel: Revision 1.2, 6/28/91*
    ******************************************************

3,1 ) No. of Plies for STIF,ELEM(1): 8
    Ply # Material # Thickness Angle
    1     1        0.0142     -45.0
    2     1        0.0142     -45.0
    3     1        0.0142     -45.0
    4     1        0.0142     -45.0
    5     1        0.0142     -45.0
    6     1        0.0142     -45.0
    7     1        0.0142     -45.0
```

```
     8     1        0.0142      -45.0
     00 No Change

Modify the number of above plies (Y/N)? >>(N)> n
Choose the PLY NUMBER you wish to change >> 0


      ****************************************************
              *Advanced Composite Structural Program by LMSC*
               *Flat Stiffener Panel: Revision 1.2, 6/28/91*
      ****************************************************

STIF MENU (MENU3):
1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,
5)Angle, 6)Zee, 7)Bead, 8)Open Hat.

3,0) Type of Stiffener: 1
     XH= 0.400 XB= 0.500 XC= 0.200
3,1) No. of Plies for STIF,ELEM(1): 8
3,2) No. of Plies for STIF,ELEM(2): 4

         ------------------------------------------
Selection => 3,n: modify panel plies;
          -1 :End; 9:Quit; m,0: other MENUS; -1: exit. >>
4,0


      ****************************************************
              *Advanced Composite Structural Program by LMSC*
               *Flat Stiffener Panel: Revision 1.2, 6/28/91*
      ****************************************************

    --- Boundary Condition Definition -----
4,1)  There are NO ribs on the ends.
4,2)  Ribs on both the Skin and the Stiffener.
4,3)  Ribs are simulated using Simple Support plus
Rotational
      Constraints.
4,4)  Ends are Part of a Larger Panel.
4,5)  Sides are Part of a Larger Panel.

      -----Loading Condition Definition------
4,6)  Total Loads (Ibs.):
4,7)  Axial= 1000.0
4,8)  Lateral= 0.0
4,9)  Shear Along the Ends = 0.0
4,10) Lateral Pressure (psi.)= 0.0

--- Solution Technique ---
4,11) Post-Buckling Analysis.
```

```
TYPE OF TARGET >LOAD FACTOR TARGET= 5.70000

 Select m,n=> -1 :Exit; 9:Quit; m,n: Other Menu or
Update.>>-1
```

3.0 CURVED STIFFENED PANEL MODULE

This module creates a model of a curved rectangular panel with longitudinal axial stiffeners and circumferential frame attachments (as shown in Figure 6). This model is representative of a stiffened fuselage shell. There are eight different types of stiffeners available. No actual frames are modeled in the circumferential direction. Frame attachments are simulated with local boundary conditions.

3.1 CAPABILITIES AND RESTRICTIONS

1) Number of axial stiffeners0<N<10
2) Stiffeners may be unequally spaced and of different cross-sections and layups
3) Number of lateral stiffeners or restraints0<N<2
4) Lateral rib or stiffener attachments may provide either a pinned or fixed type support.
5) Boundary conditions options include for the ends either simply supported, clamped or free.
6) Boundary conditions options for the sides include simply supported, clamped or free.
7) Loading may include any combination of axial, lateral and shear line loads and a lateral pressure.
8) Maximum number of different materials = 10
9) Maximum number of different layups = 20
10) Maximum number of plies per layup = 100
11) Option to perform either a linear stress analysis or buckling analysis. The buckling model is necessarily a more refined mesh than the stress model and as a consequence will require much longer run times.
12) Stiffener layup may include any or all panel plies as integral to the stiffener.

An example input format for this module is presented in Section 3.2. The required input dimensions and parameters are the same as shown in Figures 2 and 3. A plot of the model generated for this example is shown in Figure 6 and Figure 7.

USAGE NOTES

1) The skin panel at the base of the stiffener is considered part of the stiffener base. The laminate input should include

FIGURE 6. CURVED STIFFENED PANEL GEOMETRY AND LOAD

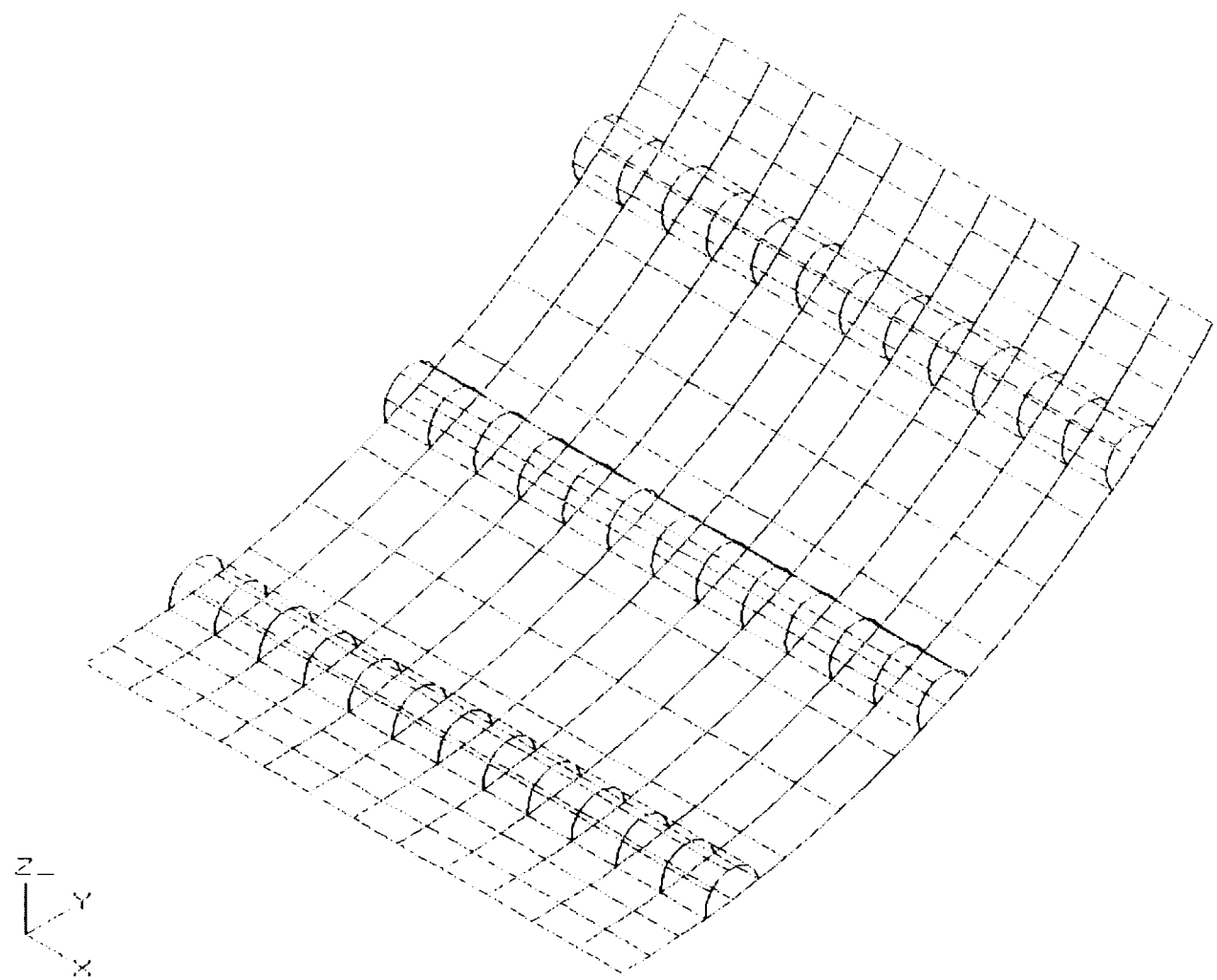
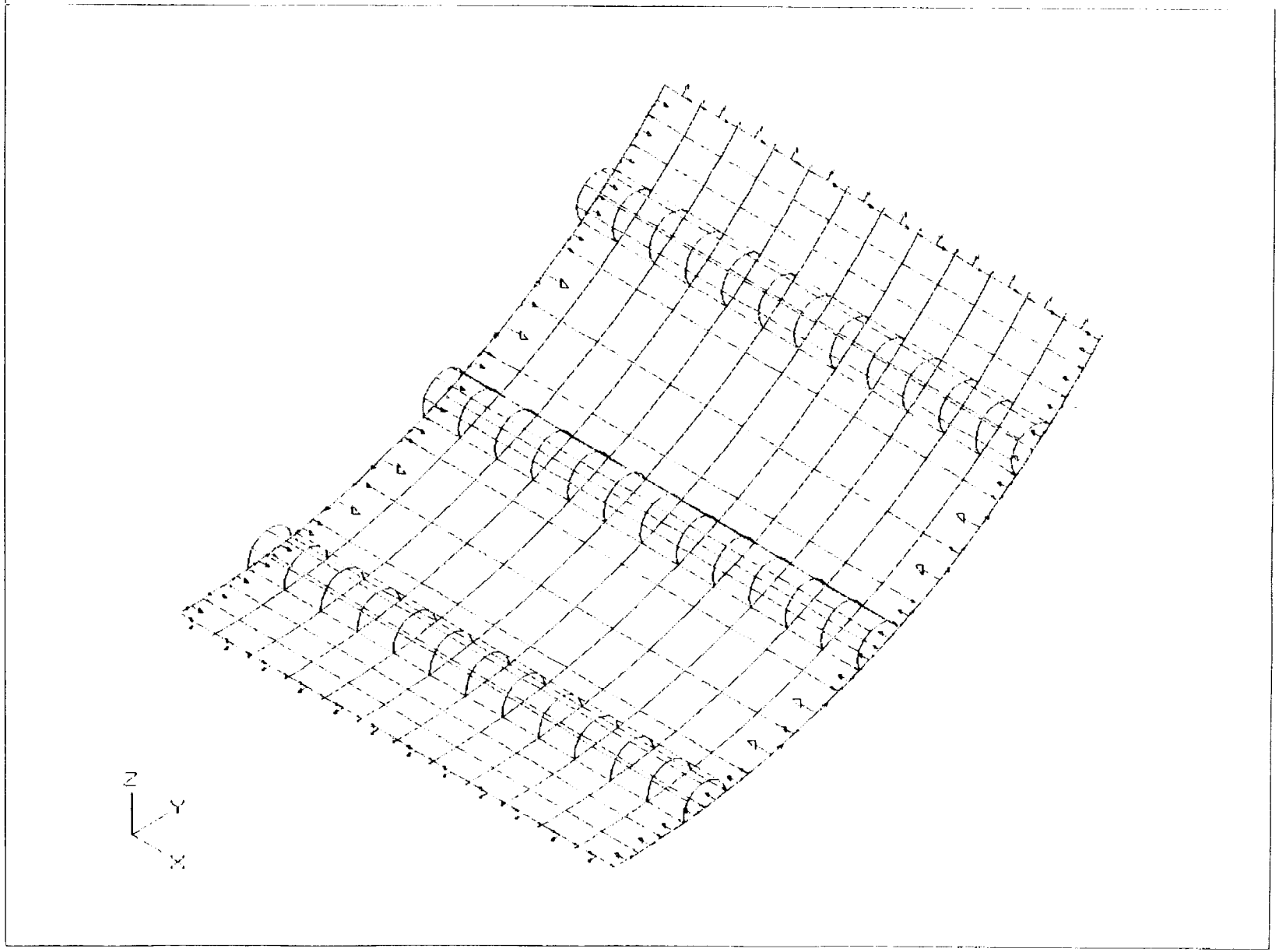


FIGURE 7. BEAD STIFFENED CURVED PANEL - EXAMPLE PLOT



contributions from both sources.
2) Panel sub-divisions define the model element density which is program controlled unless requested by the user.
3) Definition of full 3-D material properties is not required since only shell elements are used. The material property input is described in Appendix A.1.

3.2 CURVED STIFFENED PANEL MODULE - EXAMPLE INPUT

Following input prompts would be shown on the terminal, if there are no pre-assigned data files when the program is called. Numbers after the prompts~ are the actual inputs for the model shown in Figure 6. User input is shown in bold type.

$ @DIALAMATIC

Advanced Composites Structural Concept and Materials Technologies

Automated DIAL Finite Element Analysis

DIALAMATIC
VAX VERSION 1.2
JUNE 1991

The following analysis modules are available at this time:

MODULE NO.	GENERIC MODEL
1	Flat rectangular stiffened panel (Wing panel)
2	Curved Stiffened Panel
3	Flat rectangular tubular panel
4	Geodesic Panel

Select a module number >>: **2**

Generic Model for Curved Stiffened Panel

MODULE 2
VAX VERSION 1.2
JUNE 1991

BASIC ANALYSIS INFORMATION:

Does a DIAL database file exist? (Y/N) Default is FIL002. >>:**N**
Would you like to give the data base file a unique name? (Y/N) >>:

CURVED

```
Does a geometry file already exist for this analysis?(Y/N)
>>: N
Input a name for the geometry file. >>: BEAD.DAT
Does a material properties file exist?(Y/N) >>: n
Input a name for the material roperties file. >> m1


     ***************************************************
        *Advanced Composite Structural Program by LMSC
                    *Flat Stiffener Panel: Revision 1. 6/28/91
     ***************************************************

Input Panel Width,Length and Radius  >>( 10.000 10.000
122.0)> 50. 40. 60.

Input No. of Stiffeners and Ribs >>( 2 1)> 31
Does the user want to specify Panel Sub-divisions? (Y/N)
Enter >> Y
Input no. of Sub-Grid between stif/Rib>>(2m2)> 2  6

    Stiffeners equally spaced (Y/N)? >>(Y) > y
    Ribs equally spaced (Y/N)? >>(Y)> y

----  Boundary Condition Definition -------

Are there ribs at Ends ?(Y/N)>>(Y)> n
Ribs on:(1)Skin (2)STIF (3) Both ?>>(1)> 3

Choose a B.C. Type no. at Ribs:

Choose a B.C. type no. at z = 0:
    (1)  Simploy Supported
    (2)  S. S. and Rotation
    (3)  Free --->>(1)> 1

Choose a B.C. type no. at z = L:
    (1)  Simploy Supported
    (2)  S. S. and Rotation
    (3)  Free --->>(1)> 1

Choose a B.C. type no. at theta = 0:
    (1)  Simploy Supported
    (2)  S. S. and Rotation
    (3)  Free --->>(1)> 1

Choose a B.C. type no. at theta = theta_max:
    (1)  Simploy Supported
    (2)  S. S. and Rotation
    (3)  Free --->>(1)> 1
```

```
   Loading Condition Definition ~

Choose input loads type (Total=1,Line=2)?>>(2)> 2
following total load unit: lb

   Input axial load >>(1000.0)> -5000
   Input lateral load >>( 0.0)> 2500.
   Input shear load along ends >>( 0.0)> 1000.
   Input lateral pressure in psi >> (0.00)> 8.3

     Type of Analysis ~
Choose ONE of the following type of analysis>>
1> Linear Analysis.
2, Bifurcation Buckling Analysis.
3> Post-Buckling Analysis.
      Enter >> 1

Input No. of Materials >>( 1)> 1
Input Matl# 1 Elastic & Shear Modulus, Poison #, AML +/-
Allowables
    E1,E2      >>( 0.30E+08 0.30E+08)> 8.9e6 8.9e6
    G12,G23,G31 >>( 0.1 2E+08 0.12E+08 0.1 2E+08)~0.89e6
0.3e6 0.3e6
     V12      >>( 0.3043)> 0.095
Do you want to input AML Failure Parameters (Y/N)? y
   + Allow.>>(-60.0 0.001 0.0 0.002 60.0 0.003)> -60 0.011 0
0.022 60.0.011
   - Allow.>>(-60.0 -0.002 0.0 -0.003 60.0 -0.004)>
-60-0.011 0-0.022 60 -0.011


     ------------------------------------------------

Input No. of plies for the basic panel >>( 2)> 6

Input Matl#, Thickness, Angle

Are the information the same for ALL plies (Y/N)? y

 Ply# 1>>( 1 0.1000  0.00), 1 0.005

           ------------------------------------------

Stiffener Type: 1)Blade, 2)Close Hat, 3)1-shape, 4)J-shape,
         5)Angle, 6)Zee, 7)Bead, 8)0pen Hat.
   Select axial stiffener type >>(1)> 7
  Input XH,XB, >>( 0.400 0.500)> 2.0 8.0
     ------------------------------------------------


Input No. of plies for stiffener ELEM #1>>( 2)> 6
Input Matl#, Thickness, Angle

Are the information the savm for ALL plies (Y/N)? y
Ply# 1>,( 1 0.1 000 0.00)> 1 0.005   45
```

```
      ------------------------------------------------

 Input No. of plies for stiffener ELEM #2>>( 2)> 6
 Input Matl#, Thickness, Angle

 Are the information the same for ALL plies (Y/N)? y
 Ply# 1>,( 1 0.1 000 0.00)> 1 0.006  45

         ------------------------------------------------
 Input No. of plies for stiffener ELEM #3>>( 2)> 6
 Input Matl#, Thickness, Angle

 Are the information the same for ALL plies (Y/N)? y
 Ply# 1>,(.1 0.1000 0.00)> 1 0.005  45



     ******************************************************
           *Advanced Composite Structural Program by LMSC *
              *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ******************************************************

m,n) >>> MAIN MENU <<< (general MESH data)
  0,0) Geometry & General Data
  0,1) Base Panel Width, Length and Radius (AX,BY:) 50.000
90.000 60.000
  0,2) No. of Stiffeners and Ribs (M,N): 3 1
  0,3) No. of GRID for Stiffeners and Ribs(Ml,NI): 2 6
  0,4) Stiffener locations (A): 8.333  25.000 41.667
  0,5) Rib locations (B): 20.000

         ------------------------------------------------


    >>> OTHER MENUS <<< (MATL, BASE Ply, STIF Ply, BC & LC)

  1,0) Total No. of Materials (NMAT): 1
  2,0) No. of Plies for The Base Panel(NLAY): 6
  3,0) Axial stiffener type No.(MTYP): 7

Blade=1,Close Hat=2,"1"=3,"J"=4,Angle=5,"ZH=6,Bead=7 Open
Hat=8.
     XH= 2.00 XB= 8.000
  4,0) Boundary & Loading Conditions
Select m,n=> -1:END; 9:QUIT; m,n:UPDATE ~> 1,0



     ******************************************************
           *Advanced Composite Structural Program by LMSC *
              *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ******************************************************
```

```
MATL MENU (MENU1):
 1,0) Total No. of Materials (NMAT): 1
 1,1) Material Number 1
Selection=> 1,n: modify MATL #n;
 -1: Exit; 9: Quit; m,0: other MENUS >> 1,1


     ****************************************************
       *Advanced Composite Structural Program by LMSC*
       *Flat Stiffener Panel: Revision 1.2, 6/28/91*
      ****************************************************

    Properties for MATL No. :1
 1>>Elastic Modulus (E1,E2): 0.89E+07 0.89E+07
 2>>Shear Modulus (G12,G23,G31): 0.89E+06 0.30E+06 0.30E+06
 3>>Poisson Ratio (V12): 0.0950
 4>>AML + Allowables:-60.0 0.0110 0.0 0.0220 60.0 0.0110
 5>>AML- Allowables:-60.0 -0.0110 0.0 -0.0220 60.0 -0.0110

  Modify Above Material Properties (Y/N)? >>(N)> Y


     ****************************************************
              *Advanced Composite Structural Program by LMSC*
               *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ****************************************************


MATL MENU (MENU1):
 1,0) Total No. of Materials (NMAT): 1
 1,1) Material Number 1
Selection=> 1,n: modify MATL #n;
 -1: Exit; 9: Quit; m,0: other MENUS >> 2,0


     ****************************************************
              *Advanced Composite Structural Program by LMSC*
               *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ****************************************************


Base Ply MENU (MENU2):
 2,0) No. of Plies for The Base Panel (NLAY): 6
      Ply # Material # Thickness Angle

       1        1       0.005     45.0
       2        1       0.005     45.0
       3        1       0.005     45.0
       4        1       0.005     45.0
       5        1       0.005     45.0
```

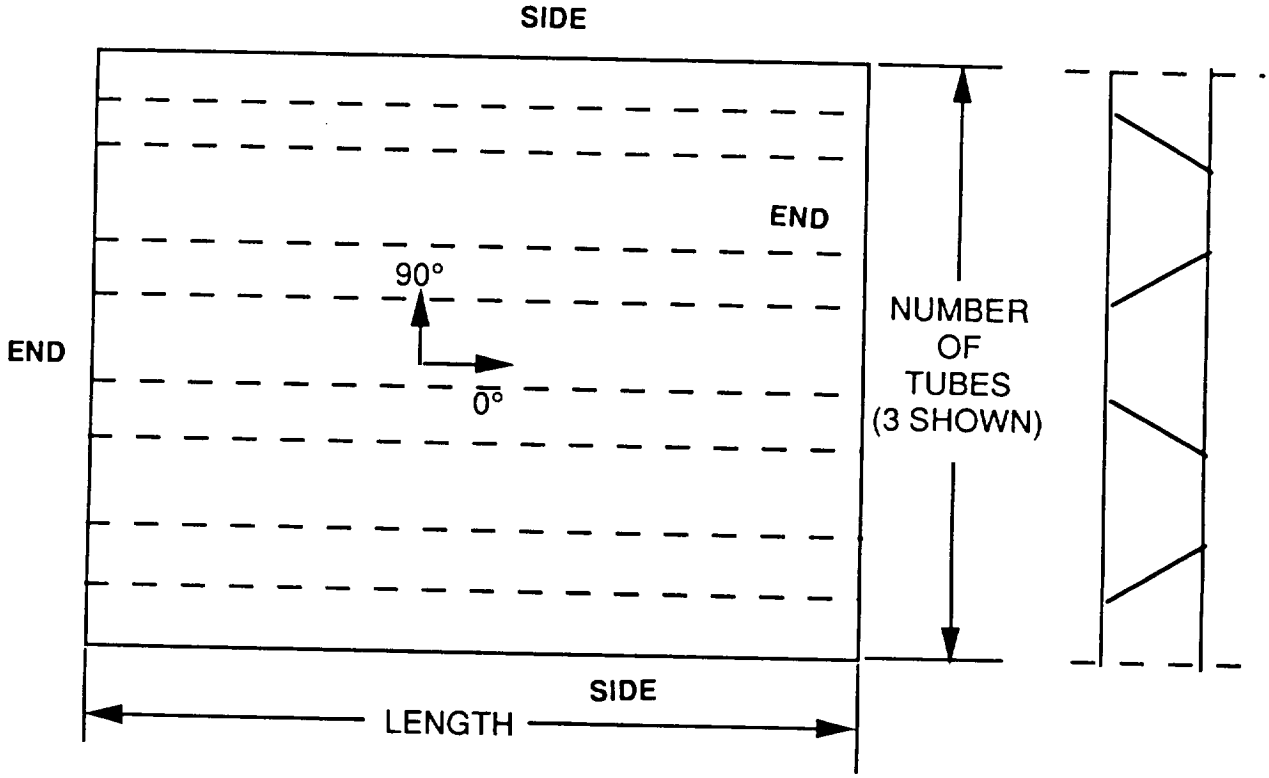
```
       6       1       0.005     45.0

Modify the no. of plies @ the panel (Y/N)? >>(N)>
n
Choose PLY NUMBER to modify (Y/N)? n Selection =>
2,n: modify panel plies;
       -1 :End; 9:Quit; m,0: other MENUS >> 3,0


     ***************************************************
            *Advanced Composite Structural Program by LMSC*
             *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ***************************************************

STIF MENU (MENU3): 1)Blade, 2)Close Hat,
3)1-shape, 4)J-shape, 5)Angle, 6)Zee, 7)Bead,
8)Open Hat.

3,0) Type of Stiffener: 7
     XH= 2.000 XB= 8.000
3,1) No. of Plies for STIF,ELEM(1): 6
3,2) No. of Plies for STIF,ELEM(2): 6
3,3)  No. of Plies for STIF, ELEM(3):  6
Selection => 3,n: modify panel plies;
        -1:End; 9:Quit; m,0: other MENUS; -1: exit. >> 3,1


    ***************************************************
           *Advanced Composite Structural Program by LMSC*
            *Flat Stiffener Panel: Revision 1.2, 6/28/91*
    ***************************************************

3,1 ) No. of Plies for STIF,ELEM(1): 6
    Ply # Material # Thickness Angle
    1     1        0.005     -45.0
    2     1        0.005     -45.0
    3     1        0.005     -45.0
    4     1        0.005     -45.0
    5     1        0.005     -45.0
    6     1        0.005     -45.0
    00 No Change

 Modify the number of above plies (Y/N)? >>(N)> n
Choose the PLY NUMBER you wish to change >> 0


     ***************************************************
            *Advanced Composite Structural Program by LMSC*
             *Flat Stiffener Panel: Revision 1.2, 6/28/91*
     ***************************************************
```

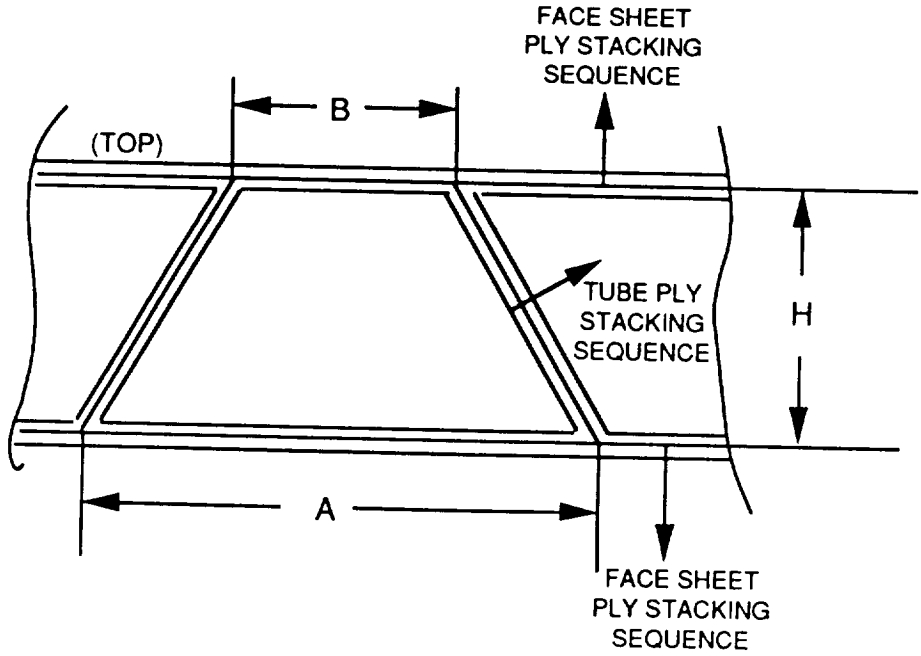
```
STIF MENU (MENU3):
1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,
5)Angle, 6)Zee, 7)Bead, 8)0pen Hat.

3,0) Type of Stiffener: 7
     XH= 2.000 XB= 8.000
3,1) No. of Plies for STIF,ELEM(1): 6
3,2) No. of Plies for STIF,ELEM(2): 6
3,3) No. of Plies for STIF,ELEM(3):  6
        ----------------------------------------

Selection = >,n: modify panel plies;
          -1 :End; 9:Quit; m,0: other MENUS; -1


     ***************************************************
             *Advanced Composite Structural Program by LMSC*
              *Flat Stiffener Panel: Revision 1.2, 6/28/91*
      ***************************************************

       --- Boundary Condition Definition -----
4,1)  There are NO ribs on the ends.
4,2)  Ribs on both the Skin and the Stiffener.
4,3)  Ribs are simulated using Simple Support plus
Rotational
      Constraints.
4,4)  Ends are Part of a Larger Panel.
4,5)  Sides are Part of a Larger Panel.


       -----Loading Condition Definition------
4,6)  Total Loads (lbs.):
4,7)  Axial= 1000.0
4,8)  Lateral= 0.0
4,9)  Shear Along the Ends = 0.0
4,10) Lateral Pressure (psi.)= 0.0

         --- Solution Technique ---
4,11) Post-Buckling Analysis.

TYPE OF TARGET >LOAD FACTOR TARGET= 5.70000

 Select m,n=> -1 :Exit; 9:Quit; m,n: Other Menu or Update.>>
-1
```

4.0 FLAT RECTANGULAR TUBULAR PANEL

This module creates a model of a flat rectangular panel made up of trapezoidal tubes bonded side by side with face sheets bonded on the top and bottom surface, see Figure 8. All tubular stiffeners are identical but the top and bottom face

FIGURE 8. TUBULAR FLAT PANEL GEOMETRY





sheets may be different layups. The model may be loaded with any combination of axial, lateral and shear load. The model may be used to perform a stress analysis of the laminates or local buckling analysis of the stiffener/face sheet combination. Overall buckling of the panel may also be determined but care in the selection of boundary conditions is important to obtaining meaningful output.

4.1 CAPABILITIES AND RESTRICTIONS

1) Any number of tubular stiffeners of any size as long as the basic shape remains trapezoidal.
2) All tubular stiffeners are identical in shape and layup.
3) Stress and bifurcation buckling analyses may be performed.

A full 3-D set of orthotropic properties is not required since all models use shell elements for analysis. Through-the-thickness properties are not required. See Appendix A.1 for definition of material directions and requested properties.

An option to put a filler material along the open sides of this model exists. This would simulate the use of a potting material in this area used to prevent local buckling in a test article. Input may be entered to define the filler material. Otherwise properties for a lightweight syntactic filled epoxy material are used by default.

An example input format for this module is preser ed in the Section 4.2. Figures 8 and 9 are included to explain some of terminology and conventions used for the input. A typical model generated using this module is shown in Figures 10 and 11.

USAGE NOTES

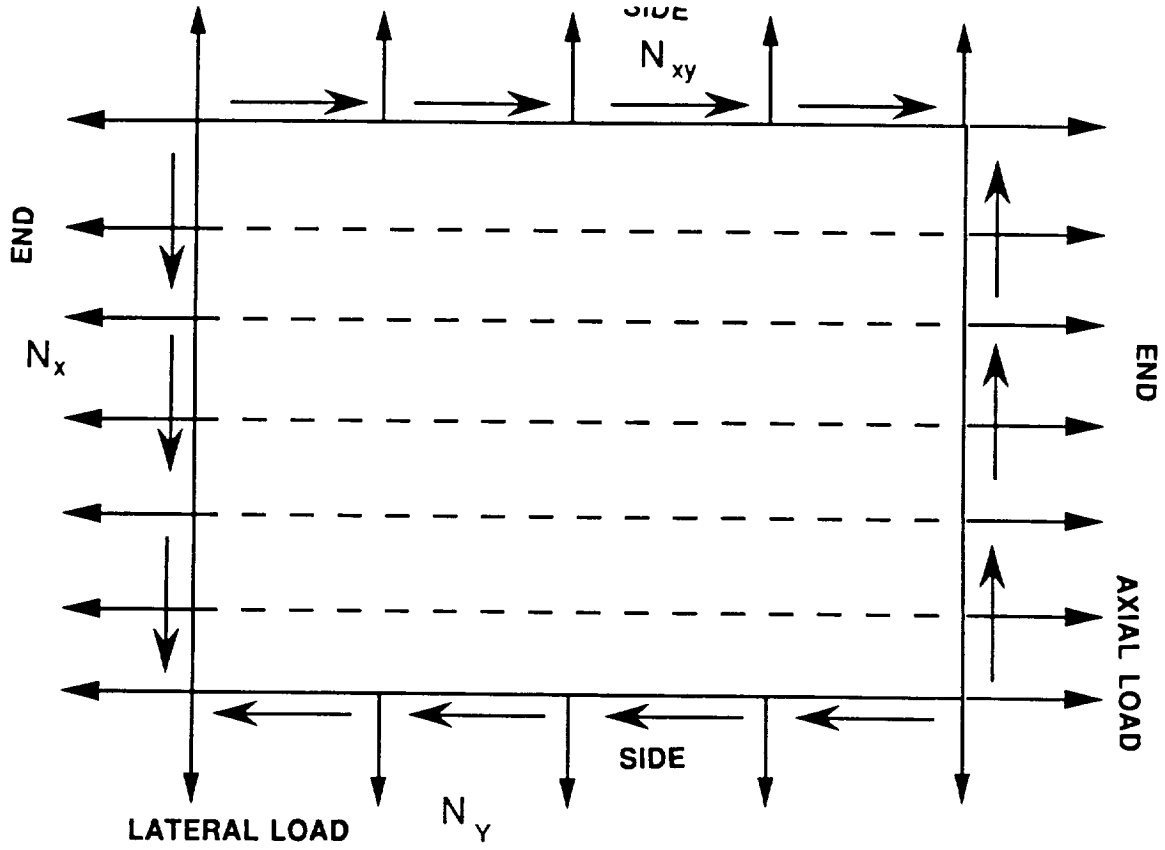
1) The tubes and skins are handled as separate parts in the input, they are combined within the model as if they are fully bonded to each other.

4.2 TUBULAR MODULE - EXAMPLE INPUT

The following input prompts would be shown on the terminal for a typical run. Refer-to Figures and for guidance on the meaning of various prompts. User input is shown in bold type.

$ @dialamatic

FIGURE 9. TUBULAR FLAT PANEL LOADING



- LINE LOADS ARE AVERAGE DISTRIBUTED OVER BOTH SKIN AND STIFFENERS. PROGRAM WILL AUTOMATICALLY REDISTRIBUTE LOADS.

FIGURE 10. TUBULAR FLAT PANEL - EXAMPLE PLOT

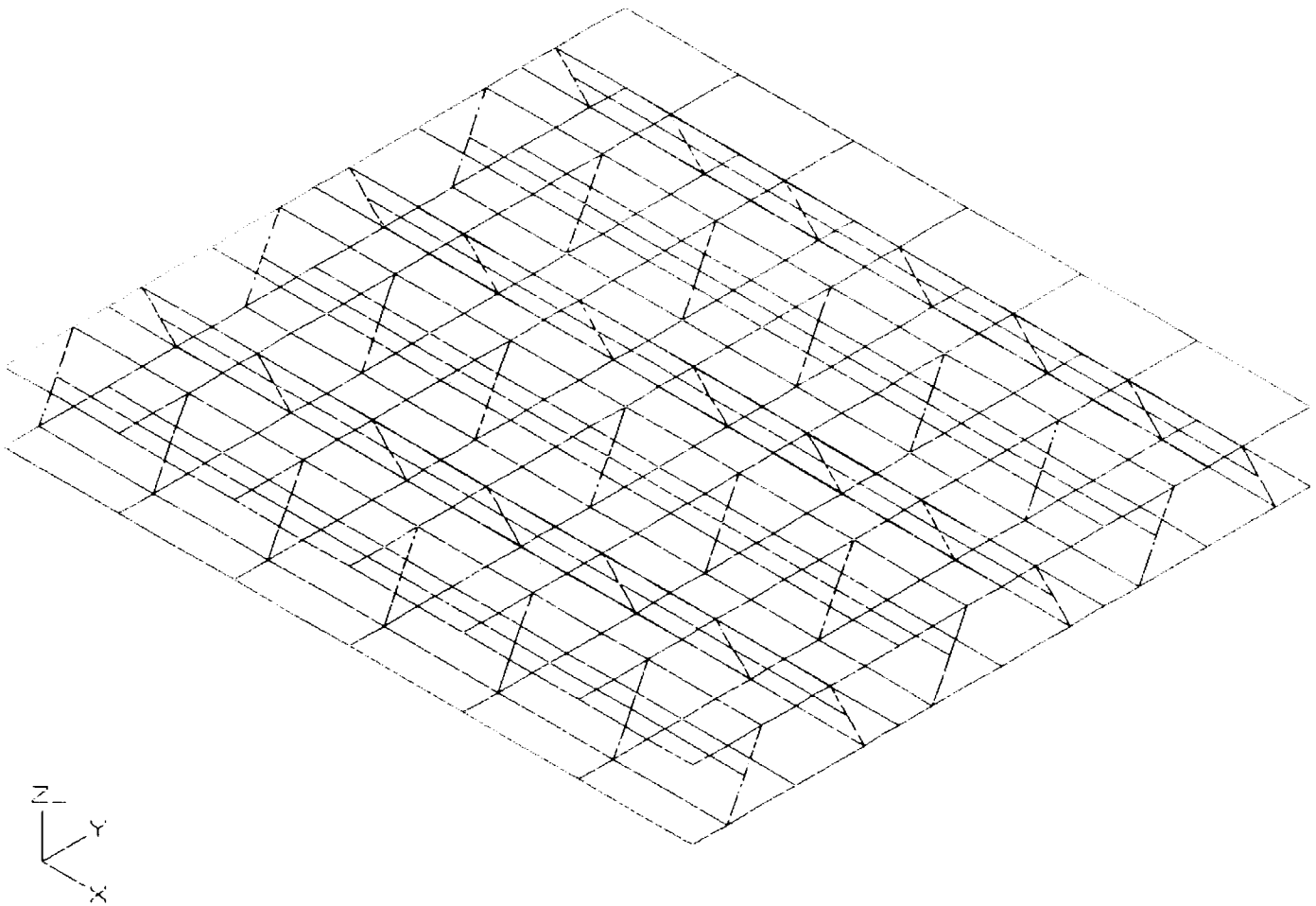
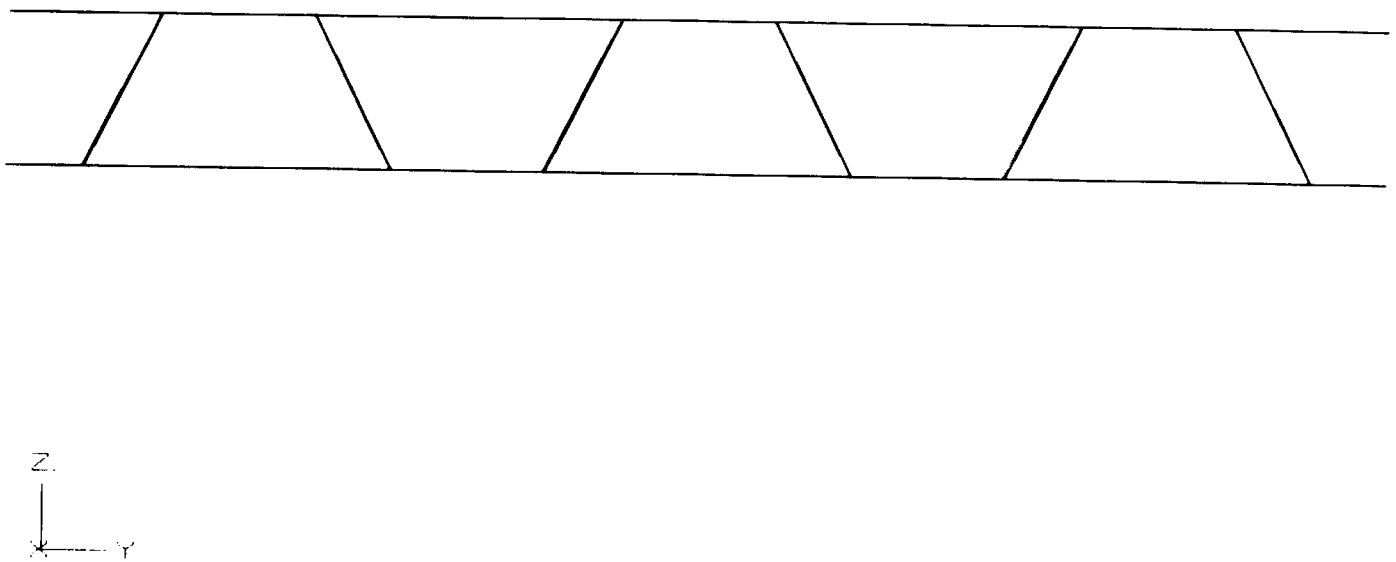


FIGURE 11. TUBULAR FLAT PANEL - EXAMPLE END PLOT



Advanced Composites Structural Concept and Materials Technologies
Automated DIAL Finite Element Analysis

DIALAMATIC
VAX VERSION 1.2
JUNE 1991

The following analysis modules are available at this time:

MODULE NO.	GENERIC MODEL
1	Flat rectangular stiffened panel (Wing panel)
2	Curved Stiffened Panel
3	Flat rectangular tubular panel
4	Geodesic Curved Panel

Select a module number >>: **3**

Generic Model for Flat Stringer Stiffened Panel

MODULE 3
VAX VERSION 1.2
JUNE 1991

BASIC ANALYSIS INFORMATION:

Does a DIAL database file exist?(Y/N) Default is FIL002.>>: **n**
Would you like to give the data base file a unique name?(Y/N)>>: **y**
Input database file name >>: **tube**
Does a geometry file already exist for this analysis?(Y/N)>>:**n**
Input a name for the geometry file. >>: **g1.dat**
Does a material properties file exist?(Y/N) >>: n
Input a name for the material properties file. >>: **m1.dat**

Advanced Composite Structural Program

TUBULAR PANEL MODULE
Version 1.2 6/28/91

Updated by Tony Wei. LMSC Bld. 157 Ext:6-1137

Geometric Parameter Inputs

Input 1) for Stress Analysis.
2) for Buckling Analysis.

```
      3) for Post Buckling Analysis.
       Enter >> 1

Input Top Width of the Tube (0.5000in.) >> 0.5
Input Bottom Width of the Tube (1.0000in.) >> 1.0
Input Height of the Tubes (0.5000in.) >> 0.5
Input Length of the Panel (5.0000in.) >> 5.
Input Number of Tubes in the Width Direction (Y) (5) >> 5
Input Boundary Conditions for the Panel.
1-Infinite Panel.
2-Sides Simply Supported. (with Filler Material; ends free)
3-Sides Simply Supported. (w/o Filler Material; ends free)
4-Ends Clamped; w/o Filler at the Sides. (Sides Free)
5-Ends Clamped; with Filler at the Sides. (Sides Free)
Select Your Choice >> 4

**************REMINDER***************

**Please Keep the Total Number of Materials
 (Excluding Filler Material) at Maximum of 6**

Input No. of Materials Used.(Exclude Filler Material)>> 1
Input Information for Material 1 >>

Name (No More Than 50 Characters) >> 976 GRAPHITE EPOXY CLOTH
Input Elastic Moduli: E11,E22 >>8.9e6 8.9e6

Input Shear Moduli: G12,G23,G31 >> 0.89e6 0.3e6 0.3e6
Input Poisson's Ratio: V12 >> 0.095

Do You Want to Utilize the AML Failure Criteria? [Y or N] >> y
Input AML Parameters: ***for further definition, please consult
the user's manual.***

Input AML Paramters for Material 1 >>

*** Tension Strain Values ***
Point 1-- Angle (Degree) and Strain Allowable.
  Enter Angle. (deg.) >> -60
  Enter Strain Allowable. >> 0.011
Point 2-- Angle (Degree) and Strain Allowable.
  Enter Angle. (deg.) >> 0
  Enter Strain Allowable. >> 0.022
Point 3-- Angle (Degree) and Strain Allowable.
  Enter Angle. (deg.) >> 60
  Enter Strain Allowable. >> 0.011

***Compression Strain Values ***
Point 4-- Angle (Degree) and Strain Allowable.
  Enter Angle. (deg.) >> -60
  Enter Strain Allowable. >> -0.011
Point 5-- Angle (Degree) and Strain Allowable.
```

```
  Enter Angle. (deg.) >> 0
  Enter Strain Allowable. >> -0.022
Point 6-- Angle (Degree) and Strain Allowable.
  Enter Angle. (deg.) >> 60
  Enter Strain Allowable. >> -0.011

***Wall Layup Information Input***

Input No. of Plies for Top Face Sheet (4) >> 4
Input No. of Plies for Bottom Face Sheet (4) >> 4
Input No. of Plies for Tube Core ( 4) >> 4
Input Material Numbers for Top Face Sheet Lay-up >>
Is it the Same for All Plies? [Y or N] >> y
Enter >> 1

Input Thicknesses for Top Face Sheet Lay-up(in.) >>
Is it the Same for All Plies? [Y or N] >> y
Enter >> 0.0142

Input Orientations of Plies for Top Face Sheet (deg.) >>
Is it the Same for All Plies? [Y or N] >> n
Enter >>  0
Enter >> 45
Enter >>-45
Enter >> 90

Input Material Numbers for Bottom Face Sheet Lay-up >>
Is it the Same for All Plies? [Y or N] >> y
Enter >> 1

Input Thicknesses for Bottom Face Sheet Lay-up(in.) >>
Is it the Same for All Plies? [Y or N] >> y
Enter >> 0.0142

Input Orientations of Plies for Bottom Face Sheet (deg.) >
Is it the Same for All Plies? [Y or N] >> n
Enter >>  0
Enter >> 45
Enter >>-45
Enter >> 90

Input Material Numbers Tube Core Lay-up>>
Is it the Same for All Plies? [Y or N] >> y
Enter >> 1

Input Thicknesses for Tube Core Lay-up(in.) >>
Is it the Same for All Plies? [Y or N] >> y
Enter >> 0.0142

Input Orientations of Plies for Core Wall (deg.) >>
Is it the Same for All Plies? [Y or N] >> n
Enter >>  0
```

```
Enter >> 45
Enter >>-45
Enter >> 90
Apply loads to the model:
Load may be input as total load or as an average line load.
How would you like to input loads?
  1>  Total Load.
  2>  Line Load.
  Enter >> 1
Total Tensile (+) or Compressive (-) Load (lb.) on the Ends.
  Enter >> -1000
Total Tensile (+) or Comppressive (-) Load (lb.) on the Sides.

Enter >> 0
Total Shear Load (+ CCW and - CW) (lb.)
  1) On the Ends.
  2) On the Sides.
  Enter >> 1
  Shear Load on the Ends = 0

Choose the Info. to be Listed:
  1> Panel Geometry.
  2> Material Data.
  3> Wall Layups.
  4> Load.
  5> Continue.
  Enter >> 1

Width at the Tube Bottom = 1.000in.
Width at the Tube Top= 0.500in.
Number of Tubes = 5
Panel Width = 4.500in.
Panel Length, PL= 5.000in.
Total Height of the Panel=0.613600in.
Tube Height, PH= 0.500in.
Top Cover Sheet Thickness= 0.056800in.
Bottom Cover Sheet Thickness= 0.056800in.
Tube Wall Thickness= 0.056800in.
Boundary Condition: Ends clamped; w/o filler at the sides.
Stress Analysis.

***Hit <<Return>> for Next Page.***

Choose the Info. to be Listed:
  1> Panel Geometry.
  2> Material Data.
  3> Wall Layups.
  4> Load.
  5> Continue.
  Enter>> 2

Material Number: 1
```

```
Material Name = 976 GRAPHITE EPOXY CLOTH
Orthotropic Material
E11 (psi)= 8900000.0 E22(psi)=  8900000.0
G12 (psi)=  890000.0 G23(psi)=  300000.0 G31 (psi)= 300000.0
V1 2=0.09500
                    ------------------------

AML parameters for this material.

    ***Tension Strain Values***
  Point 1 Angle(Degree)=-60.00 Strain Allowable= 0.01100
  Point 2 Angle(Degree)= 0.00 Strain Allowable= 0.02200
  Point 3 Angle(Degree)= 60.00 Strain Allowable= 0.01100

   ***Compression Strain Values***
  Point 4 Angle(Degree)=-60.00 Strain Allowable= -0.01100
  Point  5 Angle(Degree)= 0.00 Strain Allowable= -0.02200
  Point  6 Angle(Degree)= 60.00 Strain Allowable= -0.01100

   ***Hit <<Return>> for Next Page***

 AML Failure Calculation is invoked.

Choose the Info. to be Listed:
  1> Panel Geometry.
  2> Material Data.
  3> Wall Layups.
  4> Load.
  5> Continue.
  Enter>> 3
Top Face Sheet Layup.
Ply Number 1
  Material Number: 1
   Ply Thickness(in)=0.01420
   Ply Angle(degree)= 0.00
Ply Number 2
   Material Number: 1
   Ply Thickness(in)=0.01420
   Ply Angle(degree)= 45.00
Ply Number 3
   Material Number: 1
   Ply Thickness(in)=0.01420
   Ply Angle(degree)=-45.00
Ply Number 4
   Material Number: 1
   Ply Thickness(in)=0.01420
   Ply Angle(degree)= 90.00

***Hit <<Return>> for Next Page.***

Total Thickness= 0.056800in.
```

```
***Hit <<Return>> for Next Page.***

Bottom Face Sheet Layup.
Ply Number 1
  Material Number: 1
  Ply Thickness(in)=0.01420
  Ply Angle(degree)= 0.00
Ply Number 2
  Material Number: 1
  Ply Thickness(in)=0.01420
  Ply Angle(degree)= 45.00
Ply Number 3
  Material Number: 1
  Ply Thickness(in)=0.01420
  Ply Angle(degree)=-45.00
Ply Number 4
  Material Number: 1
  Ply Thickness(in)=0.01420
  Ply Angle(degree)= 90.00

***Hit <<Return>> for Next Page.***

Total Thickness= 0.056800in.

***Hit <<Return>> for Next Page.***

Tube Wall Layup.
Ply Number 1
  Material Number: 1
  Ply Thickness(in)=0.01420
  Ply Angle(degree)= 0.00
Ply Number 2
   Material Number: 1
   Ply Thickness(in)=0.01420
   Ply Angle(degree)= 45.00
Ply Number 3
   Material Number: 1
   Ply Thickness~in)=0.01420
   Ply Angle(degree)=-45.00
Ply Number 4
   Material Number: 1
   Ply Thickness(in)=0.01420
   Ply Angle(degree)= 90.00

***Hit <<Return>> for Next Page.***

Total Thickness= 0.056800in.

***Hit <<Return>> for Next Page.***

Choose the Info. to be Listed:
  1> Panel Geometry.
```

```
  2> Material Data.
  3> Wall Layups.
  4> Load.
  5> Continue.
  Enter >> 4

Axial Line Load on the Ends. (lb./in)= -222.222
  Total Axial Load on the Ends. (lb.)= -1000.000
Axial Line Load on the Sides. (lb./in)=  0.000
  Total Axial Load on the Sides. (lb.)=  0.000
Shear Line Load (lb./in)= 0.000
  Total Shear Load on the Ends. (lb.)=   0.000
  Total Shear Load on the Sides. (lb.)=  0.000

***Hit <<Return>> for Next Page.***

Choose the Info. to be Listed:
  1> Panel Geometry.
  2> Material Data.
  3> Wall Layups.
  4> Load.
  5> Continue.
  Enter >> 5

Choose the Info. to be Modified: 1>Panel Geometry.
2>Material Data. 3>Wall Layups. 4>Load. 5>Return to List
Menu. 6>Save the Changes and Exit. Enter>> 6
Do you really want to exit this program? [Y or N] Enter
>> y

 *****End of Data Input Session.*****

*** Begin Dial Runstream Generation ***

*** Dial Runstream Generation Complete ***

FORTRAN STOP

Would you like to rename the geometry file? (Y/N) >>: n
Did you change geometry data?  (Y/N) >>: y
The TGEOM. dat file was deleted.
Your designated geometry file was not purged.
Would you like to rename the material file? (Y/N) >>: n
Did you change material data?  (Y/N): >> y
The TMAT. dat file was deleted.
Your designated material file was not purged.
Do you want to proceed with code execution? (Y/N) >> y

MESH GENERATION STARTED
FORTRAN STOP
MESH GENERATON COMPLETED
FORTRAN STOP
```

```
FOPTRAN STOP
BANDWIDTH OPTIMIZATION COMPLETED
FORTRAN STOP
MATERIAL PROCESSOR COMPLETED
FORTRAN STOP

LOAD PROCESSOR COMPLETED

Do you wish to C)ontinue solve processor interactively?
                B)submit as a batch job?
                P)go to post processor directly?
                E)xit this module?
Please Select>>: E

End of this session for TUBULAR PANEL. Bye!!
Do you want to execute another module? (Y/N) >>: n
END OF THIS SESSION. BYE!!
```

5.0 GEODESIC CURVED STIFFENED PANEL

This module creates a model of a curved panel with geodesically shaped stiffeners. The panel can be curved to represent a section of a fuselage. The stiffeners are assumed to be filament wound with systactic layers such that at the intersections the syntactic filler squeezes out to create continuous filament intersections without layer thickness buildup. The general structural arrangement and panel loading is shown in Figures 12 and 13. The stiffeners may have non-filament wound overwraps if desired. Because of the curvature, normal pressures may be applied to the panel which will be reacted with a hoop tension load in the panel. All elements of the model are created with thick shell elements and analysis can include linear stress analysis and a bifurcation buckling analysis if desired.

5.1 CAPABILITIES AND RESTRICTIONS

1) Number of axial bays (Must be even number) N>2
2) Number of lateral bays (Hoop direction) 1<N<5
3) stiffener angle 40<Alpha<60
4) All stiffeners, both diagonal and hoop, must have similar layups.
5) Model cannot exceed 180 degree arc circumferentally.
6) All bays are geometrically similar
7) Any hoop load (Ny) is determined based on the applied pressure.
8) Option to perform either a linear stress analysis or buckling analysis. The buckling model is necessarily more reefined than the stress model and as a consequence will requirs longer solution times.

FIGURE 12. GEODESIC CURVED PANEL GEOMETRY

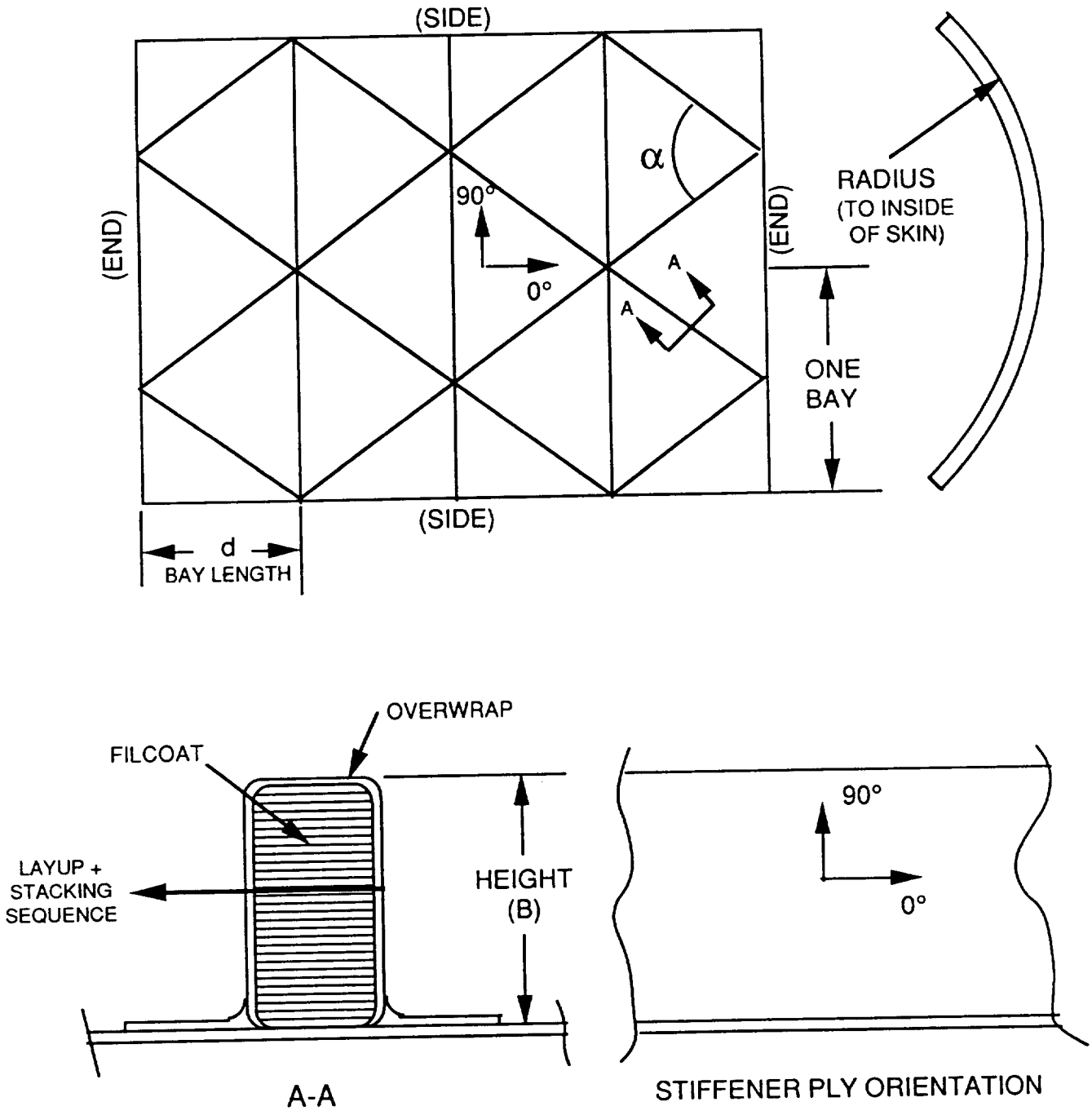
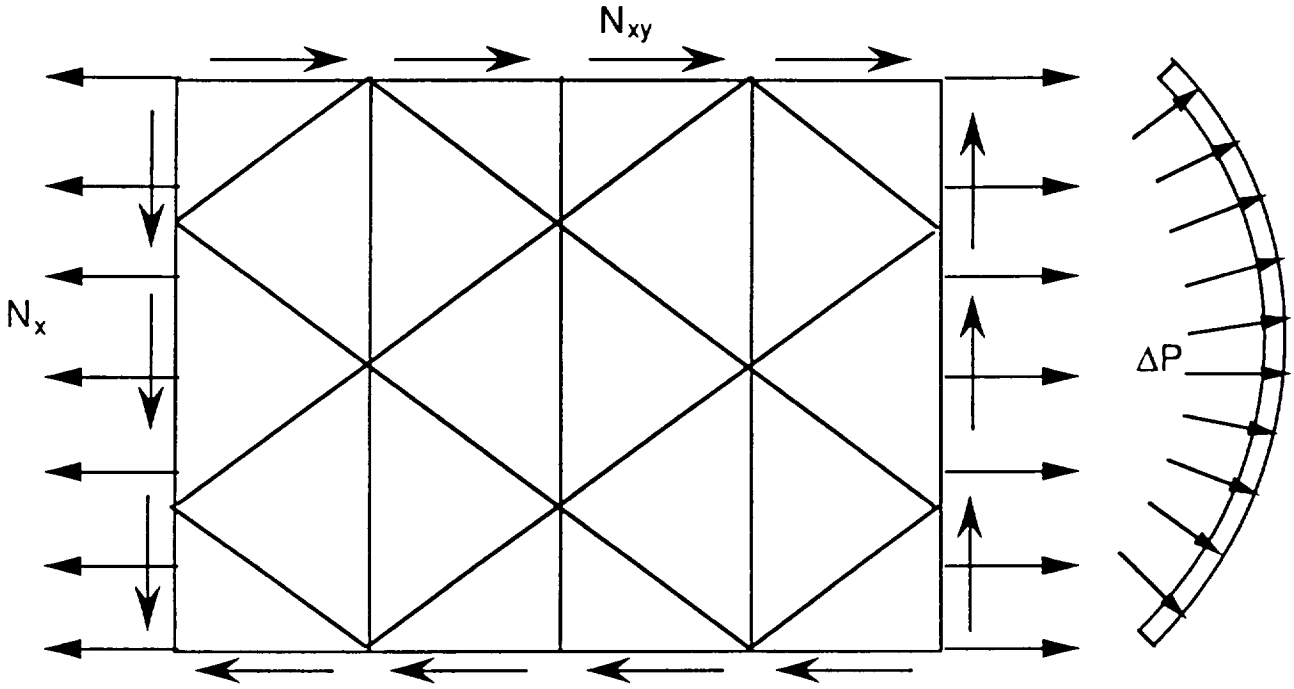


FIGURE 13. GEODESIC CURVED PANEL LOADING



- Ny LOADS ARE GENERATED AUTOMATICALLY WHEN PRESSURE IS APPLIED.
- LINE LOADS ARE AVERAGE DISTRIBUTED OVER BOTH SKIN AND STIFFENERS. PROGRAM WILL AUTOMATICALLY REDISTRIBUTE LOADS.

Material input procedure is common to all modules and uses a common material file (Library). A full 3-D set of orthotropic properties is not required since all models use shell elements for analysis. Through-the-thickness properies are not required. See appendix A.1 for definition of material directions and requested properties.

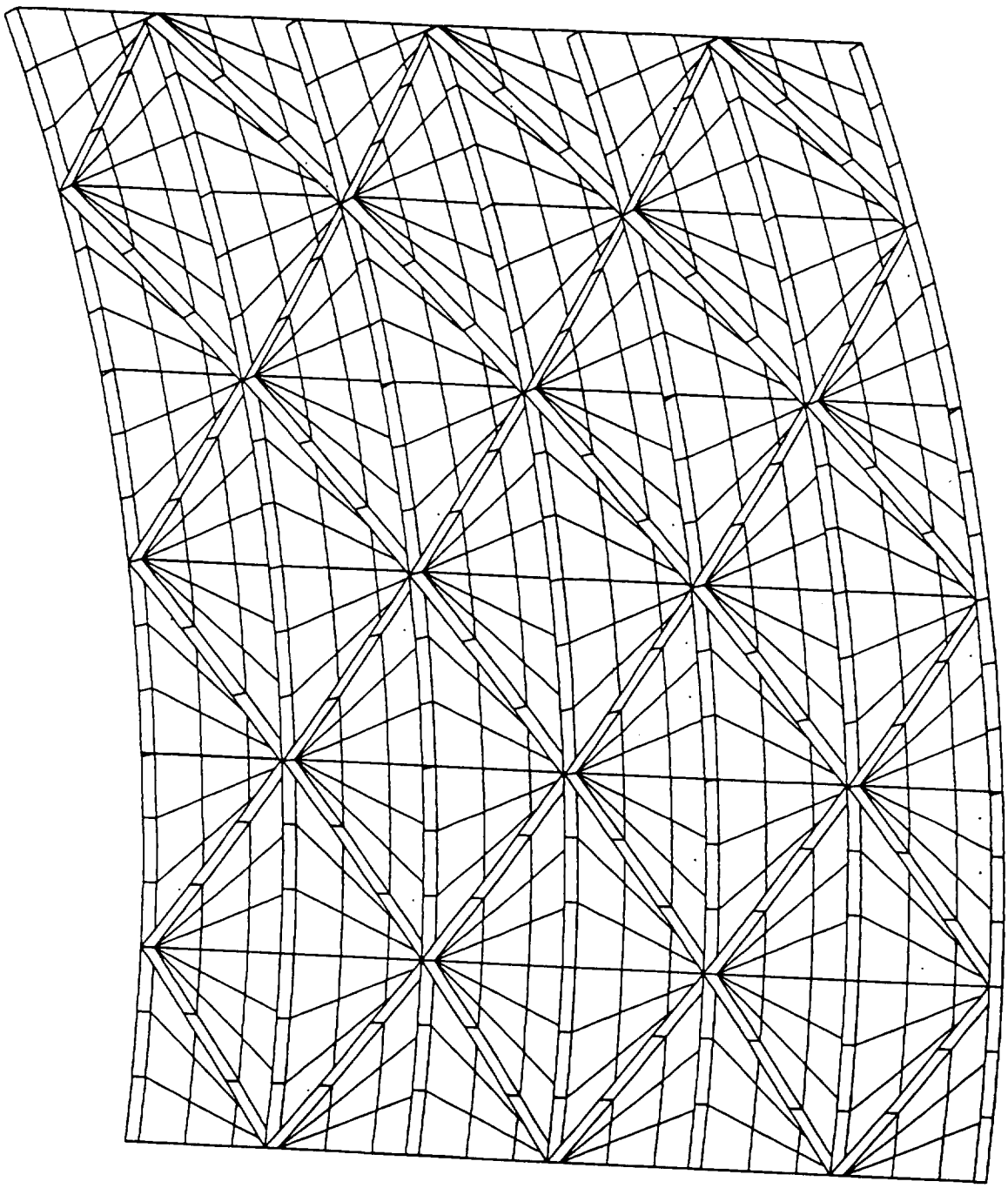
An example input format for this module is presented on the following pages. Additional figures are included to explain some of the terminology and conventions used for the input. A typical model generated using this module is shown in Figure 14.

5.2 Geodesic Curved Stiffened Panel - Example Input

The following input prompts would be shown on the terminal for a typical run. The actual prompts will vary for different panel designs. Numbers in parenthes after the prompts are the default values and may be used simply by hitting a return. Refer to Figures 12 and 13 for guidance on the meaning of various prompts.

```
$ @DIALAMATIC

Advanced Composites Structural Concept and Materials Technologies
          Automated DIAL Finite Element Analysis

                    DIALAMATIC
                  VAX VERSION 1.1
                   DECEMBER 1990

The following analysis modules are available at this time:

MODULE NO.    GENERIC MODEL

     1      Flat rectangular stiffened panel (Wing panel)
     2      Curved Stiffened Panel
     3      Flat rectangular tubular panel
     4      Curved geodesically stiffened rectangular panel

Select a module number >> :  4

 Generic Model for Geodesically Reinforced Cylindrical Panel

                     MODULE 4
                  VAX VERSION 1.1
```

FIGURE 14. GEODESIC CURVED PANEL - EXAMPLE MODEL



```
BASIC ANALYSIS INFORMATION:

Does a DIAL database file exist?(Y/N) Default is FIL002. >>: N
Does a geometry file already exist for this analysis?(Y/N) >>: N

Input a name for the geometry file.
Do not use GEOM or GEOM.DAT >>: G1
Does a material properties file exist?(Y/N) >>: N
Input a name for the material properties file.
Do not use MAT or MAT.DAT >>: M1


                Advanced Composite Structural Program
                      Geodesic Fuselage Panel
                       Updated Version - 1.3

Input 1> for Stress Analysis, 2> for Buck. Analysis.>>1
Input Bay Length (20. in.) >>15
Input No. of Bays in Axial Direction.
    (Must be Even Number) >>2
Input No. of Bays in Circumferential Direction. >>1
Input Panel Radius. (in.) >>30
Input Stiffener Angle (deg.) >>50
Input Stiffener Height.  (in.) >>1
Input Boundary Conditions for the panel.
1-Infinite Panel.
2-Sides Clamped.
3-Ends Clamped.
Select Your Choice >>1
Input No. of Layers for Panel Wall. >>3
Input No. of Layers for Stiffener Wall. >>3
Input Panel Lay-up Material Num.
Layer Number  1. >> 1
Layer Number  2. >> 1
Layer Number  3. >> 1
Input Panel Lay-Up Thickness.
Layer Number  1. >> .014
Layer Number  2. >> .014
Layer Number  3. >> .014
Input Panel Lay-up Orientation.
Layer Number  1. >>  0
Layer Number  2. >> 45
Layer Number  3. >>  0

    ***Warning!! Stiffener Lay-up Must be Symmetric.***
Input Stiffener Lay-up Material Number.
Layer Number  1. >> 1
Layer Number  2. >> 1
Layer Number  3. >> 1
Input Stiffener Lay-up Thickness.
```

```
Layer Number  1. >> .014
Layer Number  2. >> .2
Layer Number  3. >> .014
Input Stiffener Lay-up Orientation.
Layer Number  1. >> 45
Layer Number  2. >>  0
Layer Number  3. >> 45
Input No. of Materials Used. >>2
Input 1> for Iso. Material. 2> for Ortho. Materials. >>
Enter >>2
Input Elastic Moduli:  E11, E22 (psi) >>7.E6,6.8E6
Input Shear Moduli: G12,G23,G31 (psi) >>3.E6,500000.,400000.
Input Poisson's Ratios: V12 >>.15
Input Properties for Matateriał Number 2 >>
Input  1> for Iso. Material. 2> for Ortho. Material. >>
Enter >>2
Input Elastic Moduli: E11,E22 (psi) >>20.E6,2.E6
Input Shear Moduli: G12,G23,G31(psi)
>>1.E6,300000.,350000.
Input Poisson's Ratios: V12 >>.06

Apply Loads to the Model:
  How Would You Like to Input Loads?
  1>as Total Load.
  2>as Line Load.
  Enter >>2

Input Axial Line Load (lb./in)=1000
Input Shear Line Load (lb./in)=500
Input Pressure Load (psi)=14.7
Choose the Info. to be Listed:
  1> Panel and Stiffener Geometry.
  2> Material Data.
  3> Wall Lay-ups.
  4> Load.
  5> Exit.
  Enter >>1

Panel Radius(in)= 30.000
Panel Length, D(in)= 15.000
Stiffener Height(in)= 1.000
Stiffener Angle, ALPHA(degree)=50.00
Number of Panels in: Axial Direction=  2 Hoop Direction =1
Panel Thickness(in)= 0.042000
Stiffener Thickness(in)= 0.228000
Boundary Condition: Infinite Panel.
Stress Analysis.

***Hit <<Return>> for Next Page.***

Choose the Info. to be Listed:
  1> Panel and Stiffener Geometry.
```

```
  2> Material Data.
  3> Wall Lay-ups.
  4> Load.
  5> Exit.
  Enter >>2

Material Number: 1
Orthotropic Material
E11(psi)=    7000000.0  E22(psi)= 6800000.0
G12(psi)=    3000000.0  G23(psi)=  500000.0 G31(psi)=  400000.0
V12=    0.150

Material Number: 2
Orthotropic Material
E11(psi)= 20000000.0  E22(psi)= 2000000.0
G12(psi)=  1000000.0  G23(psi)=  300000.0 G31(psi)=  350000.0
V12=    0.060

***Hit <<Return>> for Next Page.***

Choose the Info. to be Listed:
  1> Panel and Stiffener Geometry.
  2> Material Data.
  3> Wall Lay-ups.
  4> Load.
  5> Exit.
  Enter >>3

Panel Wall Lay-ups.
Ply Number 1
  Material Number: 1
  Ply Thickness(in)= 0.014000
  Ply Angle(deg.)=   0.00
Ply Number 2
  Material Number: 1
  Ply Thickness(in)= 0.014000
  Ply Angle(deg.)=  45.00
Ply Number 3
  Material Number: 1
  Ply Thickness(in)= 0.014000
  Ply Angle(deg.)=   0.00
Panel Thickness(in)= 0.042000

***Hit <<Return>> for Next Page.***

   Stiffener Wall Lay-ups.
   Ply Number 1
    Material Number: 1
    Ply Thickness(in)= 0.014000
    Ply Angle(deg.)=  45.00
   Ply Number 2
    Material Number: 2
```

```
     Ply Thickness(in)= 0.200000
     Ply Angle(deg.)=   0.00
    Ply Number 3
     Material Number: 1
     Ply Thickness(in)= 0.014000
     Ply Angle(deg.)=  45.00
    Stiffener Thickness(in)= 0.228000

***Hit <<Return>> for Next Page.***

Choose the Info. to be Listed:
  1> Panel and Stiffener Geometry.
  2> Material Data.
  3> Wall Lay-ups.
  4> Load.
  5> Exit.
  Enter >>4
Axial Line Load (lb./in)= 1000.000
Shear Line Load (lb./in)= 500.000
Pressure Load (lb./in**2)= 14.700
###Hit <<Return>> for Next Page.***
Choose the Info. to be Listed:
  1> Panel and Stiffener Geometry.
  2> Material Data.
  3> Wall Lay-ups.
  4> Load.
  5> Exit.
  Enter >>5

Choose the Info. to be Modified: 1>Panel and
Stiffener Geometry. 2>Material Data. 3>Wall
Lay-ups. 4>Load. 5>Update Info. and List.
6>Exit. Enter >>6

   Do You Really Want to Exit This Program? [Y or N]
   HAVE YOU SAVED YOUR DATABASE ???
   Enter >>Y

   *****End of data input session.*****

   Begin DIAL Runstream Generation

   DIAL Runstream Generation Complete

FORTRAN STOP

CREATION OF DIAL RUNSTREAMS COMPLETED, MESH GENERATION STARTED.
FORTRAN STOP
MESH GENERATION COMPLETED.
FORTRAN STOP
FORTRAN STOP
BANDWIDTH OPTIMIZATION COMPLETED.
```

```
FORTRAN STOP
MATERIAL PROCESSOR COMPLETED.
FORTRAN STOP
LOAD PROCESSOR COMPLETED.

Do you wish to enter the solve processor now or submit as a batch job?
(Continue=C, Batch job=B) >> : B

AUTOMATIC BATCH PROCESSING NOT IMPLEMENTED AT THIS TIME
USER MUST SUBMIT DIAL SOLVE PROCESSOR IN FILE FLOAD.COM MANUALLY.
THE DIAL DATA BASE GENERATED MUST BE USED OR REGENERATED ON
WHATEVER MACHINE IS CHOSEN FOR BATCH PROCESSING.

DUE YOU WISH TO PROCEED INTO POST PROCESSING TO VIEW GEOMETRY OR
LOADS? (Y/N) >> : N

Do you want to end this session?(Y/N) >> : Y
%SYSTEM-F-NOLOGNAM, no logical name match
$
```

6.0 POST PROCESSING

Whether the generic model is run interactively or as a batch job a data base file is created which contains the results of the analysis. Other files are also generated which contain the print output of each DIAL processor. The print output files may be edited to obtain general information on the run. The primary means of obtaining results of the analysis is through an automated post processing program which queries the user for what he would like to see and generates either plots or summary tables containing the pertinant information. The added flexibility of performing other post processing tasks is possible through the actual execution of the normal DIAL SCOPE processor. This requires knowledge of the commands of that processor as well as certain node and element numbering information of the model.

Each generic model module will generate a unique set of plots for that particular model, but all models have the same basic categories of plots and summaries available. They include:

1) Geometry plots of the model
2) Geometry plots showing applied loads .
3) Deflected geometry plots for the applied load case for either the linear stress state or the buckled shape.
4) Load, stress or margin of safety contour plots for individual laminates or plies of the model.
5) Load summary tables giving the max/min laminate loads and strains for individual parts of the model.
6) Stress summary tables giving the maximum ply stresses and strains for individual laminates of the model.
7) Margin of safety summary table for individual laminates of the model using the AML failure criteria.

The program that is used to generate this output is similar to the programs that generate the model to start with. That is, all data is queried for interactively in a simple straight-forward fashion. This provides for maximum ease of use with some flexibility in what output is desired.

6-1 POST PROCESSING SESSION EXAMPLE

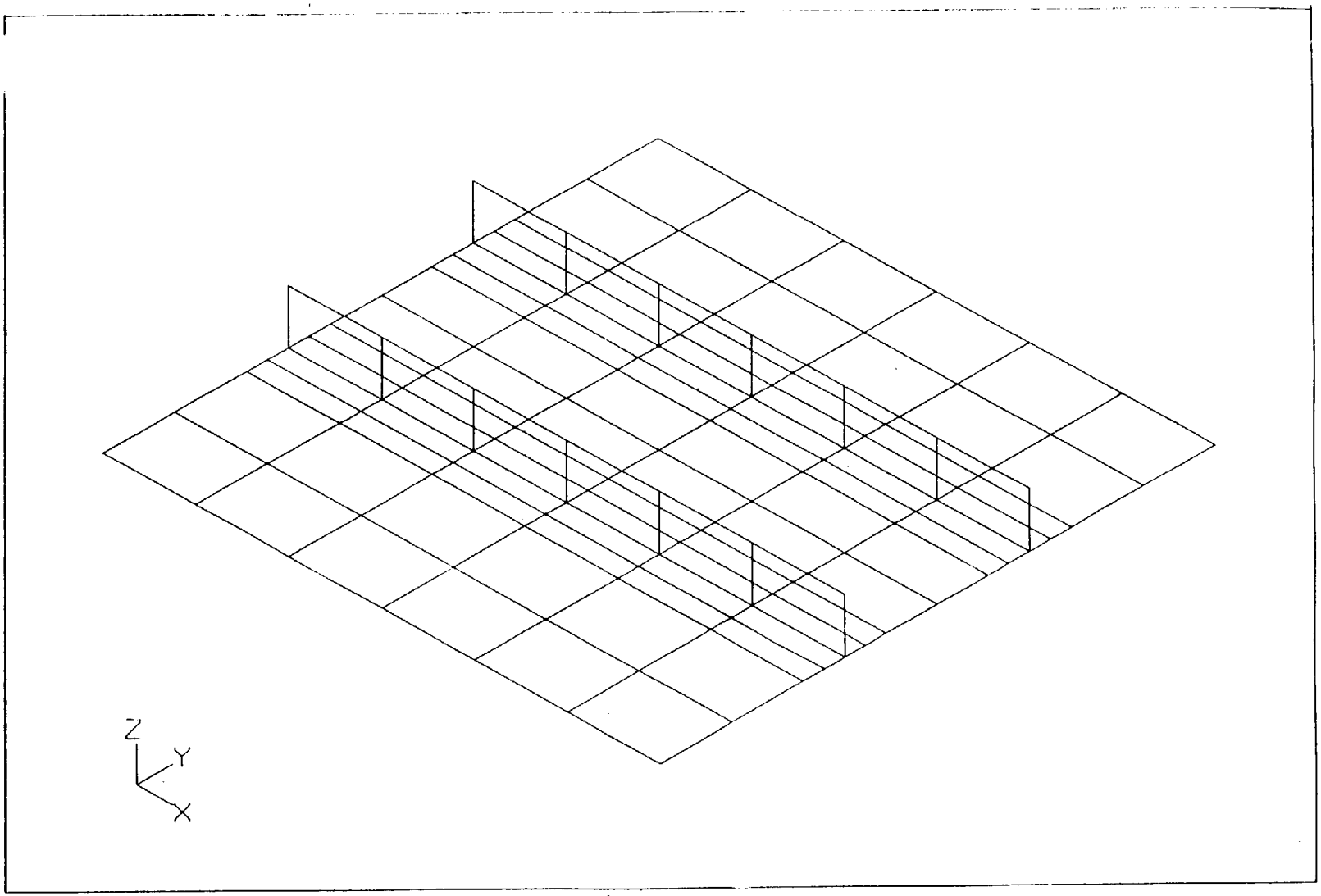
The following is an example session for generating post processing output. The actual output devices will depend on the local hardware configuration and user selection. For the pupose of this example it is assumed that tl~e post processing executable runstream is given the name PP for execution. If this is a continuation of an interactive run no assignments to the data base file will be necessary(Just hit return), but if the data base file was previously created the program will ask for the name of the file. Example plots generated by this module for the stringer stiffened panel model follow the listing.

```
$ @dialamatic



Advanced Composites Structural Concept and Materials Technologies
          Automated DIAL Finite Element Analysis


                    DIALAMATIC
                  VAX VERSION 1.2
                     JUNE 1991

The following analysis modules are available at this time:

MODULE NO.          GENERIC MODEL
     1        Flat rectangular stiffened panel (Wing panel)
     2        Curved Stiffened Panel
     3        Flat rectangular tubular panel
     4        Curved Geodesically Stiffened Panel

Select a module number >> : 1

         Generic Model for Flat Stringer Stiffened Panel


                     MODULE 1
                  VAX VERSION 1.2
                     JUNE 1991

BASIC ANALYSIS INFORMATION:
```

```
Does a DIAL database file exist?(Y/N) Default is FIL002. >> : y

What is the name of the DIAL database? >> : FIL002.

Do you wish to proceed into post processing of analysis results? (Y/N)
>> : y


                    Advanced Composites Structural Concept
                          and Materials Technologies
                    Automated DIAL finite element analysis
                          POST PROCESSING MODULE


                              VAX VERSION 1.2
                                JUNE 1991

What is the name of the DIAL data base file?: FIL002.
Which module was used to create this data base?
  1) FLAT RECTANGULAR STRINGER STIFFENED PANEL
  2) FLAT RECTANGULAR TUBULAR PANEL
 NUMBER?>> : 1

                          SCOPE Processor

Original Run Time: 12:29:38 Date:      2-JUL-91    Version: L3D3
Current Run Time: 10:38:59 Date:       3-JUL-91    Version: L3D2

BLANK COMMON Space   1000000
GIVE A SHORT NAME FOR THIS ANALYSIS
PRECEDE MULTIPLE WORDS WITH APOSTROPHE >> 'EXAMPLE SESSION
WHAT TYPE OF TERMINAL?(TEK,SEL,RETR,VT24) >> VT24
WILL YOU WANT TO PLOT HARD COPIES?(Y/N) >> y
INPUT HARD COPY DEVICE(VERS,QMS) >> VERS
WHAT OUTPUT OPTION WOULD YOU LIKE?
 1) SCREEN PLOTS ONLY, NO HARD COPIES
 2) ONLY HARD COPY PLOTS
 3) SCREEN PLOTS WITH HARD COPY PLOTS OPTIONAL
    OPTION NUMBER? >> 3


MENU FOR PLOTS AND SUMMARY TABLES:
 1)  BASIC GEOMETRY PLOT
 2)  GEOMETRY PLOT WITH LOAD VECTORS SHOWN
 3)  DEFLECTED GEOMETRY PLOT
 4)  CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS
 5)  CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN
 6)  CONTOUR PLOT OF SKIN MARGIN OF SAFETY
 7)  CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR STRAINS
 8)  CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN
 9)  CONTOUR PLOT OF STIFF. MARGIN OF SAFETY
```

```
10)  SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS
11)  SUMMARY TABLE OF PLY STRESSES AND STRAINS
12)  SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY
99)  TERMINATE POST PROCESSING SESSION

INPUT NUMBER  OF MENU ITEM DESIRED >> 1


AFTER EACH PLOT IS GENERATED HIT A RETURN TO CONTINUE

CURRENT VIEWPOINT IS: X =  1.0  Y = -1.0  Z =  1.0

DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT?(Y/N) >>n
PICK THE PARTS TO BE DISPLAYED:
      ALL PARTS (SKIN + STIFFENERS) =  0
      O SKIN ONLY                   = -1
      SINGLE STIFFENER (1 THRU N)    = ?

 INPUT OPTION >> 0

Element Set  20    INSERT  Keyword ALL

PICK ONE OF THE FOLLOWING LABEL OPTIONS:
       NO NODE OR ELEMENT NUMBERING =  0
       NODE NUMBERING ONLY          =  1
       ELEMENT NUMBERING ONLY       =  2
       NODE AND ELEMENT NUMBERING   =  3

 INPUT OPTION >> 0

Save plot on metafile, plot number=  1  (Ref. Figure 15)

MENU FOR PLOTS AND SUMMARY TABLES:
  1)  BASIC GEOMETRY PLOT
  2)  GEOMETRY PLOT WITH LOAD VECTORS SHOWN
  3)  DEFLECTED GEOMETRY PLOT
  4)  CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS
  5)  CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN
  6)  CONTOUR PLOT OF SKIN MARGIN OF SAFETY
  7)  CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR STRAINS
  8)  CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN
  9)  CONTOUR PLOT OF STIFF. MARGIN OF SAFETY
 10)  SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS
 11)  SUMMARY TABLE OF PLY STRESSES AND STRAINS
 12)  SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY
 99)  TERMINATE POST PROCESSING SESSION

INPUT NUMBER OF MENU ITEM DESIRED >> 2
CURRENT VIEWPOINT IS: X =  1.0  Y = -1.0  Z =  1.0
DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT?(Y/N) >>n

Set Nodal Quantities NQD= 1, 2, 3, 4, 5, 6
```

FIGURE 15. POST PROCESSING - GEOMETRY PLOT



```
    File [FIL.2]UL.SV   Fourier No.= 0 Sequence No.=  1
Save plot on metafile, plot number=   2  (Ref. Figure 16)

MENU FOR PLOTS AND SUMMARY TABLES:
         1) BASIC GEOMETRY PLOT
         2) GEOMETRY PLOT WITH LOAD VECTORS SHOWN
         3) DEFLECTED GEOMETRY PLOT
         4) CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS
         5) CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN
         6) CONTOUR PLOT OF SKIN MARGIN OF SAFETY
         7) CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR
            STRAINS
         8) CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN
         9) CONTOUR PLOT OF STIFF. MARGIN OF SAFETY
        10) SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS
        11) SUMMARY TABLE OF PLY STRESSES AND STRAINS
        12) SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY
        99) TERMINATE POST PROCESSING SESSION

INPUT NUMBER OF MENU ITEM DESIRED >> 3
SELECT DESIRED DEFLECTION PLOT TYPE:
           LINEAR ANALYSIS                - 0
           BUCKLING ANALYSIS MODE SHAPE   = 1
           NONLINEAR ANALYSIS             = 2

INPUT SELECTION >> 0

SELECT DEFLECTION MAGNIFICATION FACTOR TO BE USED:
             LOW (.10)          = 0
             MED (.20)          = 1
             HIGH (.50)         = 2
             USER SELECTABLE    = 3

INPUT SELECTION >> 1

Set Nodal Quantities NQD= 1, 2, 3, 4, 5, 6
File [FIL.2]D.SV   Fourier No.=  0  Sequence No. =  1

CURRENT VIEWPOINT IS: X =  1.0  Y = -1.0  Z =  2.0
DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT? (Y/N) >>n
WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE? (Y/N)>> y

Save plot on metafile, plot number=   3  (Refer Figure 17)

MENU FOR PLOTS AND SUMMARY TABLES:
  1)  BASIC GEOMETRY PLOT
  2)  GEOMETRY PLOT WITH LOAD VECTORS SHOWN
  3)  DEFLECTED GEOMETRY PLOT
  4)  CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS
  5)  CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN
  6)  CONTOUR PLOT OF SKIN MARGIN OF SAFETY
  7)  CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR STRAINS
```

FIGURE 16. POST PROCESSING - GEOMETRY PLOT WITH LOAD VECTORS

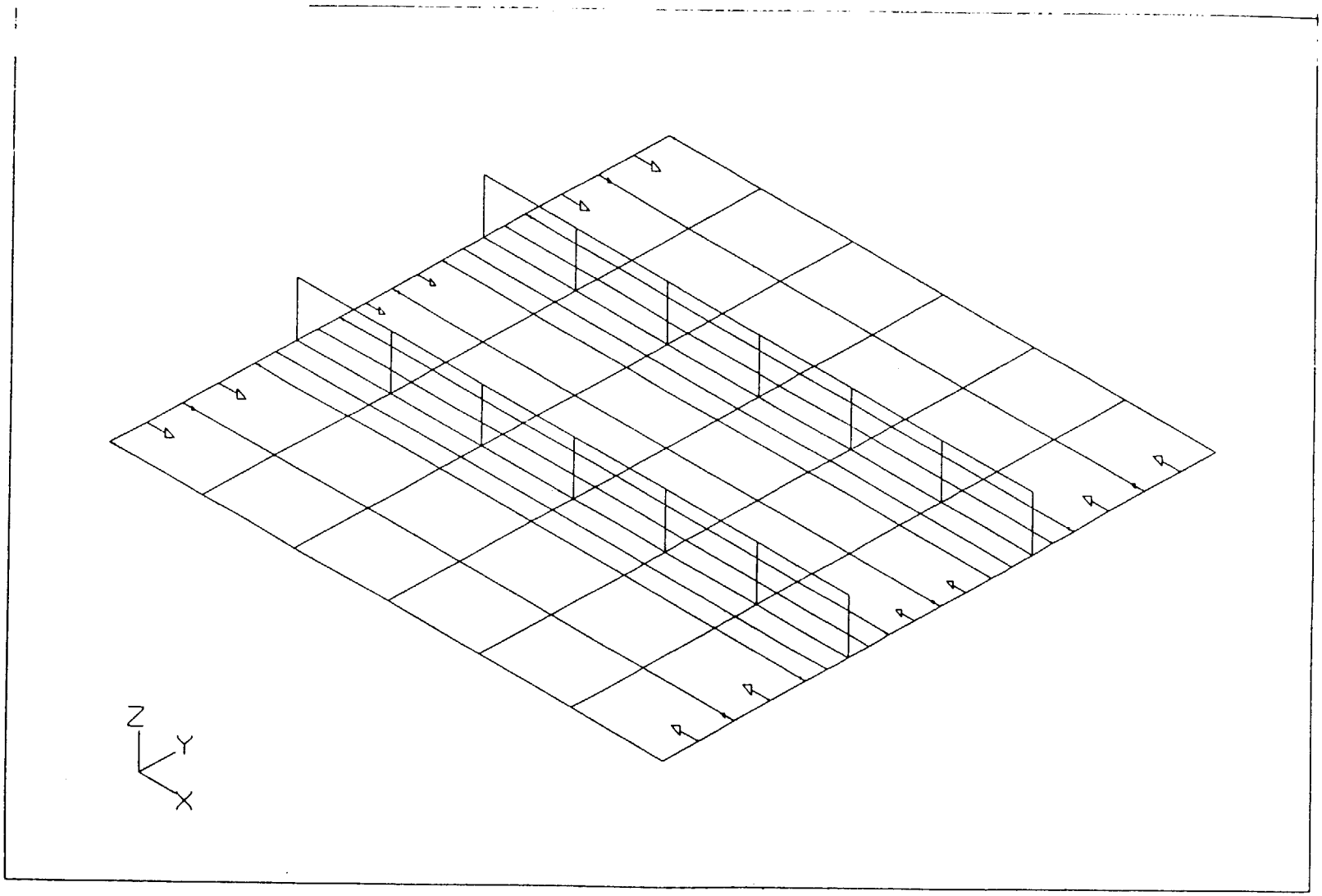


FIGURE 17. POST PROCESSING 0F DEFLECTED GEOMETRY PLOT

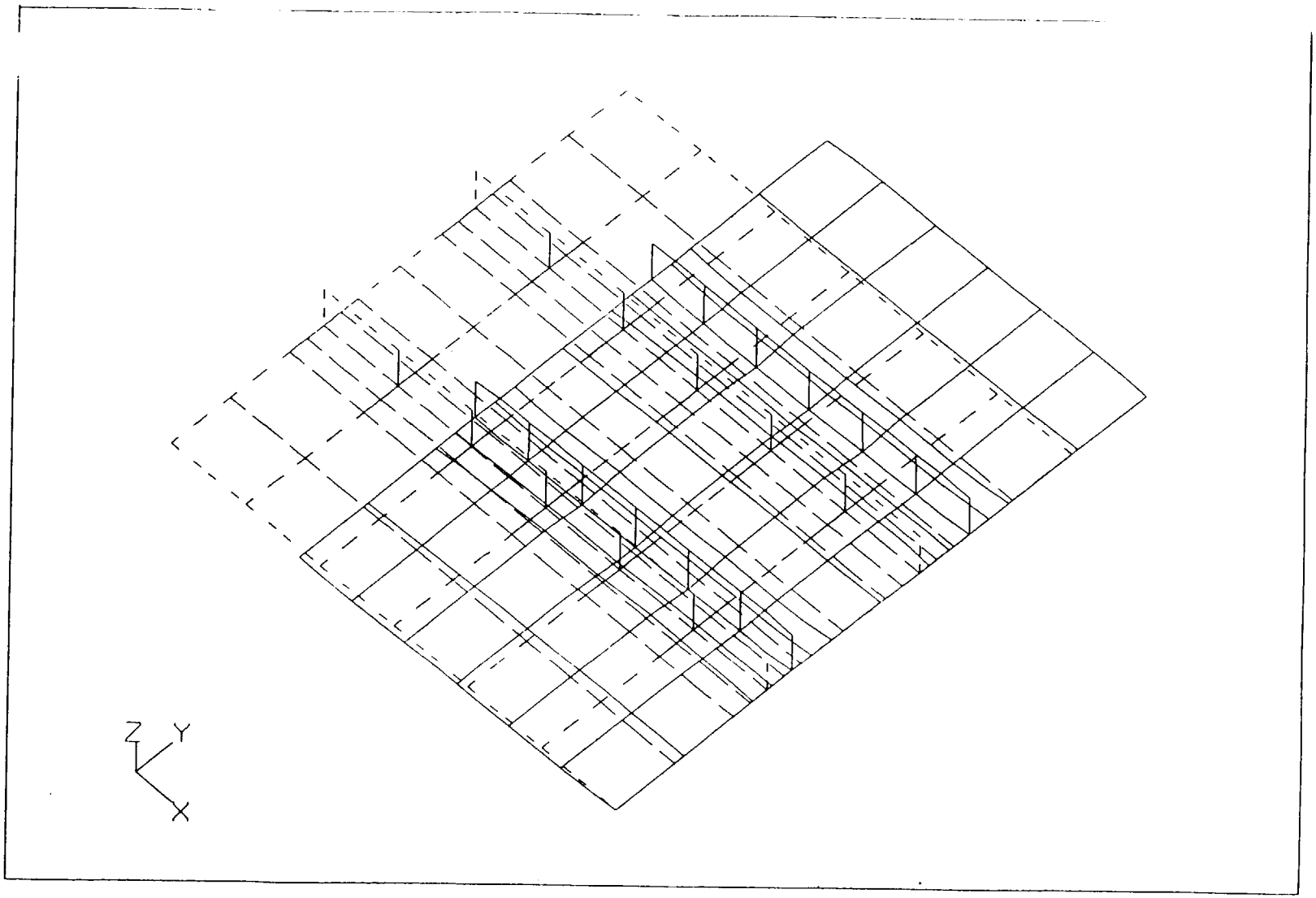
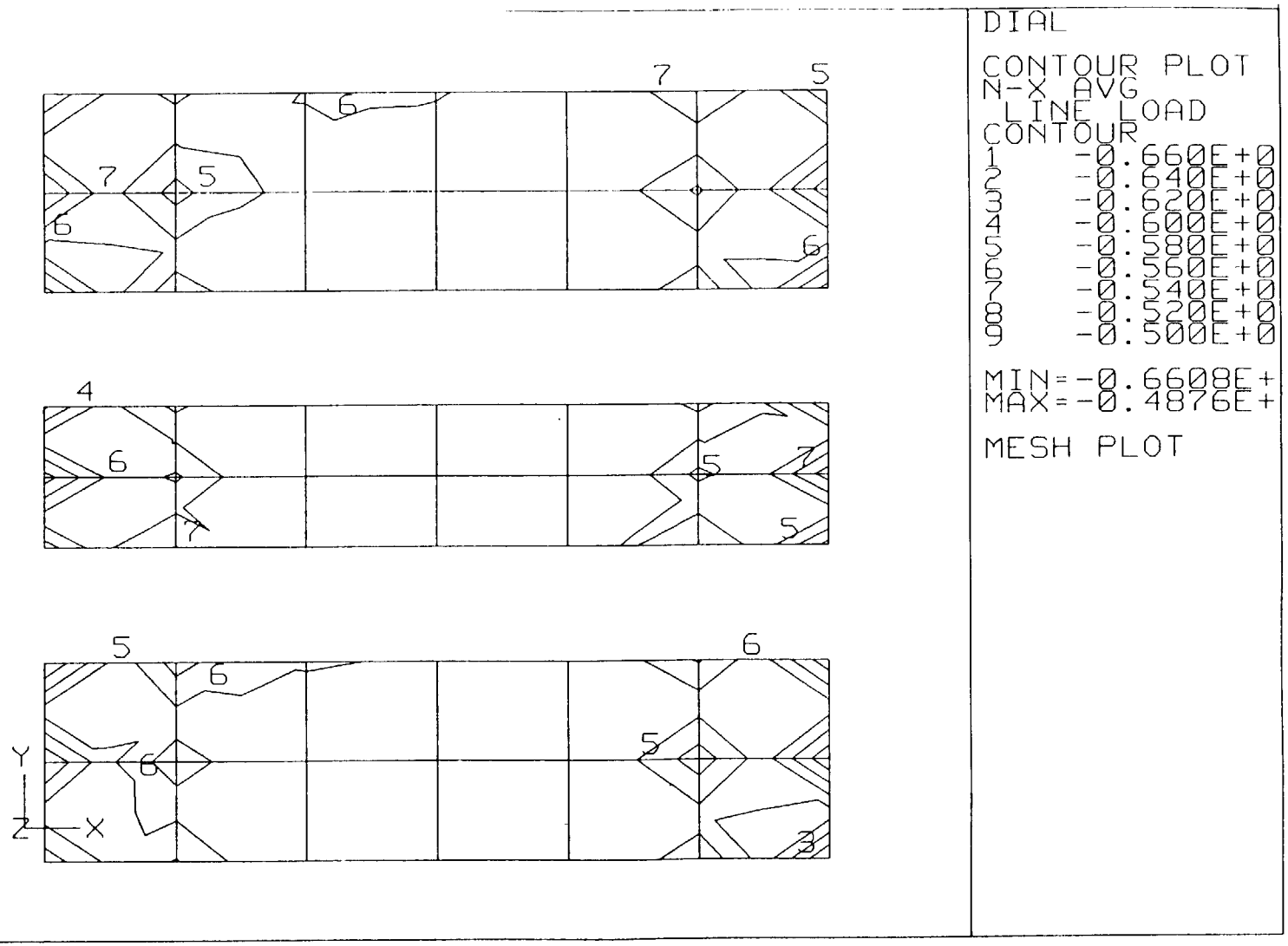


FIGURE 18. SKIN STRAIN CONTOUR PLOT



```
   8)  CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN
   9)  CONTOUR PLOT OF STIFF. MARGIN OF SAFETY
  10)  SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS
  11)  SUMMARY TABLE OF PLY STRESSES AND STRAINS
  12)  SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY
  99)  TERMINATE POST PROCESSING SESSION

 INPUT NUMBER OF MENU ITEM DESIRED >> 4

 CURRENT VIEWPOINT IS: X =  0.0  Y = 0.0  Z = 1.0
 DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT?(Y/N) >>n
 SELECT LOAD FACTOR LEVEL TO BE PLOTTED:
               LINEAR SOLUTION          = 0
               MAX LOAD STEP ATTAINED   = 1
               INTERMEDIATE LOAD FACTOR = 2

  INPUT SELECTION >> 0

 Set Element Integration Point Quantities
 File [FIL.2]S.EIP  Fourier No.= 0 Sequence No.= 1

 QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3
               (LINE MOMENT) M-X =  4, M-Y =  5
               (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13
         -->   INPUT 99 TO DISCONTINUE PLOTTING
 INPUT DESIRED QUANTITY NUMBER >> 1

 Save plot on metafile, plot number=   4  (Refer. Figure 18)

 QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3
               (LINE MOMENT) M-X =  4, M-Y =  5
               (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13
         -->   INPUT 99 TO DISCONTINUE PLOTTING
 INPUT DESIRED QUANTITY NUMBER >> 99

 MENU FOR PLOTS AND SUMMARY TABLES:
          1) BASIC GEOMETRY PLOT
          2) GEOMETRY PLOT WITH LOAD VECTORS SHOWN
          3) DEFLECTED GEOMETRY PLOT
          4) CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS
          5) CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN
          6) CONTOUR PLOT OF SKIN MARGIN OF SAFETY
          7) CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR
             STRAINS
          8) CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN
          9) CONTOUR PLOT OF STIFF. MARGIN OF SAFETY
         10) SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS
         11) SUMMARY TABLE OF PLY STRESSES AND STRAINS
         12) SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY
         99) TERMINATE POST PROCESSING SESSION
```

```
INPUT NUMBER OF MENU ITEM DESIRED >> 99

* Mapped-vector algorithm *

*****PLOT OPTIONS IN EFFECT*****

o . oo 10.56 0 . 00 1 . 00 0 . 00 -1 . 00  2112

MODEL  =     1200    ,XMIN   =     0.00   ,XMAX   =    10.56
YMIN   =     0.00    ,YMAX   =    10.56   ,MSGLVL =        1
XSTART =     0.00    ,YSTART =     0.00   ,SCALE  =     1.00
XFACT  =     1.00    ,YFACT  =     1.00   ,UNITS  =     1.00
STRIP  =    10.56    ,STRIPO =     0.00   ,SPACE  =    10.56
12FLG  =        2    ,OUT    =    -1.00   ,LYNES  =      600
NSCN   =       48    ,NIBS   =     2112   ,DEN    =   200.00
MODELP =        6

SAVED PLOTS ARE NOW BEING TRANSFORMED TO PLOT FILES
PLEASE WAIT UNTIL COMPLETION IS INDICATED.

* PLOT 1 summary:
     941. plottable vectors
        0 clipped vectors
        0 unplottable vectors
* PLOT 2 summary:
    1031. plottable vectors
       0 clipped vectors
       0 unplottable vectors
* PLOT 3 summary:
   2299. plottable vectors
      0 clipped vectors
      0 unplottable vectors

*S* METAPL Metafile plotting completed, 4 plots generated

~LOT FILE TRANSFORMATION COMPLETED

THIS COMPLETES THE POST PROCESSING SESSION.

ANY HARD COPY PLOTS GENERATED HERE WILL RESIDE ON FILES IN THE
WORKING DIRECTORY, USE NORMAL SUBMITTAL PROCEDURES TO ACTUALLY
PRINT THEM.

* PLOT 4 summary:
    1902. plottable vectors
       0 clipped vectors
       0 unplottable vectors

Data Base Updated
Processor Stop
```

```
FORTRAN STOP
Do you want to end this session? (Y/N) >> : y
END OF THIS SESSION FOR FLAT STRINGER STIFFENED PANEL. Bye!!!
Do you want to execute another module? (Y/N) >> : n
END OF THIS SESSION. Bye!!!
```

APPENDIX

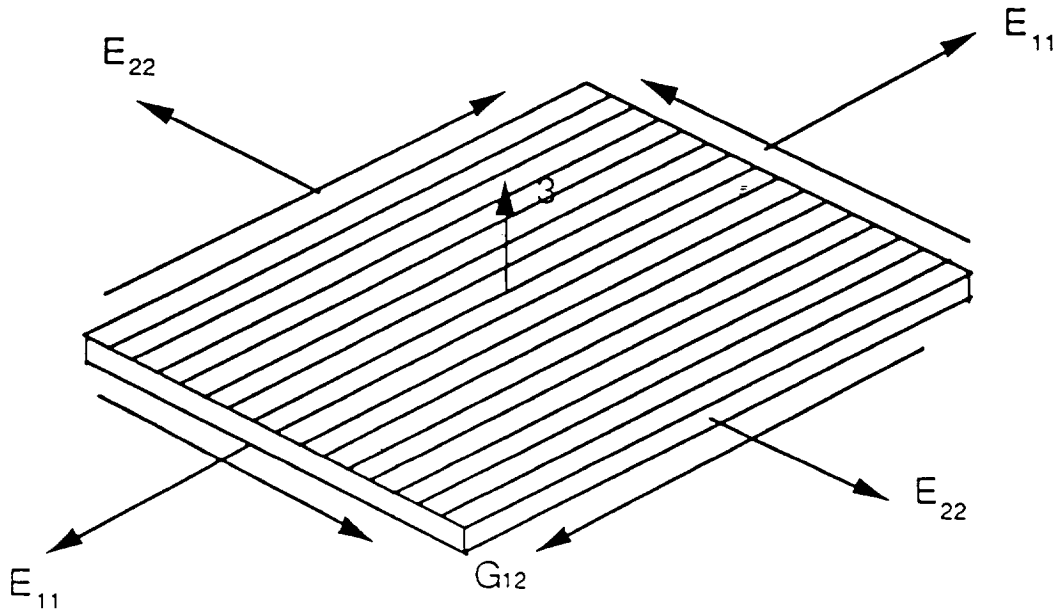
A.1 Material Properties Input

The material input section is common to all generic model modules and is intended to create a library type file from which material properties for the lamina are obtained. The modules will query the user for material input numbers for the materials being used and if they do not already exist on the material file, the program will ask for the appropriate input. One should be careful of the numbering sequence and revising materials that previously existed. This could cause previously generated models to pick up the wrong values if rerun. The input required is shown below.

Material number (From 1 to 99)
In plane moduli E_{11} and E_{22} (psi)

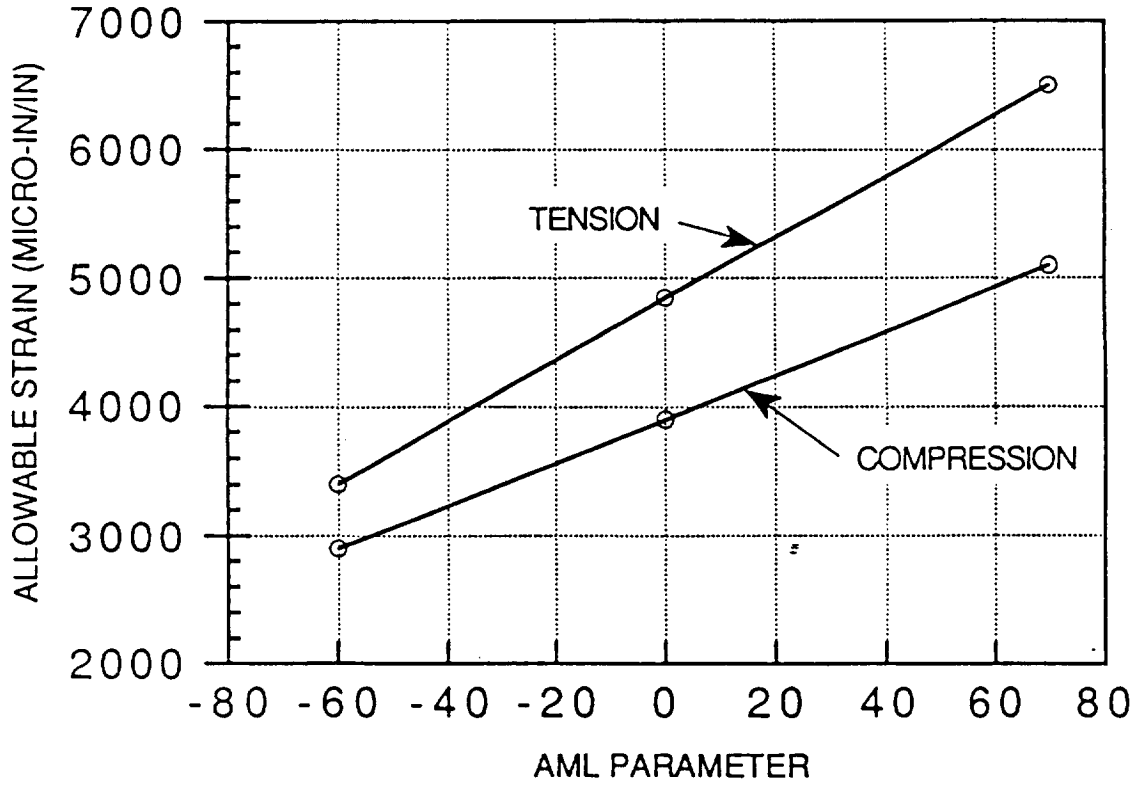
Shear moduli G_{12}, G_{13}, and G_{23} (In plane and transverse - psi)

In plane Poissons ratio (μ(12)



AML Failure Methodology Input

If the user desires to calculate margins of safety for the structure being analyzed, the AML (Angle Minus Loaded) method has been implemented. The allowable strain in both tension and compression is required versus the AML parameter (Percentage plies at 45 deg to the direction being checked minus the percentage of plies in the direction being analyzed). A typical figure is shown below. Input to the program consists of six number pairs of AML values and allowable strain values. This allows for a bilinear curve for both tension and compression allowables. The user is prompted for these number pairs. The points are shown graphically below.



INPUT PAIRS:	AML PARAMETER	ALLOWABLE STRAIN
	-60	+0.0034
	0	+0.0048
	+70	+0.0065
	-60	-0.0029
	0	-0.0039
	+70	-0.0051

APPENDIX

A.2 Input Data File Structure

BRIEF DESCRIPTIONS OF DATA IS GIVEN IN PARENTHESIS.
ACTUAL DATA VARIES FROM ONE MODEL TO ANOTHER.

MODULE 1:

GEOMETRY FILE:

```
  1    1    2    3   20.0000  15.000  (PANEL GEOM. & MESH REFINEMENT)
  0.0000   10.0000   20.0000                    (STRINGER SPACING)
  0.0000    7.5000   15.0000                      (FRAME SPACING)
  2    1    1    4    3    1                 (BOUNDARY CONDITIONS)
0.5000E+04  0.0000E+00  0.0000E+00  0.0000E+00              (LOADS)
3  0      1.00000                  (ANALYSIS TYPE AND PARAMETERS)
```

MATERIAL FILE:

```
   1                                                  (MATERIAL NUMBER)
 0.7000E+07  0.8000E+07  0.000E+00  0.2000E+00 0.0000E+00  0.0000E+00*
 0.2000E+07  0.5000E+06  0.5000E+06 0.0000E+00 0.0000E+00  0.0000E+00*
-0.6000E+02  0.1000E-02  0.0000E+00 0.2000E-02 0.6000E+02  0.3000E-02*
-0.6000E+02 -0.2000E-02  0.0000E+00-3.000E-02  0.6000E+02 -0.4000E-02*
   2                                          (SKIN LO. OF LAYERS)
   1    0.10000   45.0000      (SKIN LAYER MATL NO., THICK. & ORIENT.)
   1    0.10000   45.0000
   8                                            (STIFFENER TYPE)
 0.3000E+01  0.4000E+01  0.2000E+00  0.3000E+01    (STIFF. DIMENSIONS)
   1                              (NO. OF LAYERS IN STIFF. ELEMENT #1)
   1    0.1000     0.0000           (LAYER MATL NO., THICK. & ORIENT.)
   2                               (NO. OF LAYERS IN STIFF ELEMENT #1)
   1    0.0500   -45.0000             (LAYER MATL NO., THICK & ORIENT.)
   1    0.0500    45.0000
   1                               (NO. OF LAYERS IN STIFF ELEMENT #1)
   1    0.1000     0.0000           (LAYER MATL NO.,THICK. & ORIENT.)
```

* SIX COLUMNS ABOVE CONTAIN MATERIAL LAMINA E1, E2, E3, G12, G23, V12 AND ALM ALLOWABLE INFORMATION.

A.2 Input Data File Structure (Continued)

MODULE 2:

GEOMETRY FILE:

```
1.20000   0.50000   0.75000   3.00000          (TUBE DIMENSIONS)
 3    5      1    1               (NO. OF TUBES, NO. OF ELEM., B.C.'S)
  3.00000    3 50                    (NONLINEAR ANLAYSIS PARAMETERS)
0.06000    0.15000 0.05000        (SKIN AND TUBE LAMINATE THICKNESSES)
3  3  1                                (NO. OF PLIES IN EACH PART)
1  1 0.02000  45.00
2  1 0.02000   0.00     (TOP SKIN LAYER MATL NO., THICKNESS, & ORIENT)
3  1 0.02000  45.00

1  1 0.05000  45.00
2  1 0.05000   0.00     (bOP SKIN LAYER MATL NO., THICKNESS, & ORIENT)
3  1 0.05000  45.00

1  1 0.0500    0.00         (TUBEL LAYER MATL NO., THICKNESS & ORIENT)
```

MATERIAL FILE:

```
    1                              (MATERIAL NUMBER)
GR/EP  976                         (MATERIAL NAME)
         7000000.000      8000000.000          100.000 (E1,E2,E3)
         1000000.000        50000.000        50000.000 (G12,G,13,G23)
               0.200            0.000            0.000 (V12)

 ¿
       0.000
    1470.588
   -1470.588
    1470.588
       0.000                                (LOADS)
    1470.588
   -1470.588
    1470.588
   0.00000
   0.00000
   0.00000
   0.00000
   0.00000
   0.00000                              (AML ALLOWABLES)
   0.00000
   0.00000
   0.00000
   0.00000
   0.00000
   0.00000
 ¿
```

APPENDIX

A.3 GENERIC MODEL RUNSTREAMS

RUNSTREAM	LOCATION
MAIN DRIVER PROGRAM	A.3.1
MODULE 1 DRIVER	A.3.2.1
MODULE 2 DRIVER	A.3.2.2
MODULE 3 DRIVER	A.3.2.3
MODULE 4 DRIVER	A.3.3.4
MODULE 1 PROGRAM	A.3.3.1
MODULE 2 PROGRAM	A.3.3.2
MODULE 3 PROGRAM	A.3.3.3
MODULE 4 PROGRAM	A.3.3.4
POST PROCESSING DRIVER	

APPENDIX A.3

A.3.1 MAIN DRIVER PROGRAM

```
$!
$!                  DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO ACT AS THE
$!  BASIC RUN DRIVER FOR THE
$!  FLAT, CURVED,TUBULAR AND GEODESICALLY
$!  STIFFENED PANEL GENERIC MODELS, AS PART OF
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.2   6-28-91
$!  AUTHOR: A. Y. WEI AND W. M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#########################################################################
$START:
$ W :=WRITE SYS$OUTPUT
$W ""
$W "
$W "          DIAL-A-MATIC Stringer Stiffened Panels "
$W ""
$W "                         MODULES 1,2,3,4
$W "                         VAX VERSION 1.2"
$W "                            JUNE 1991"
$W ""
$W ""
$W "                  1) FLAT STIFFENED PANEL
$W "                  2) CURVED STIFFENED PANEL
$W "                  3) TUBULAR TRUSS CORE PANEL
$W "                  4) GEODESICALLY STIFFENED PANEL
$W ""
$INQUIRE MOD " PLEASE CHOOSE THE APPROPRIATE MODULE, 1-4 >> "
$IF MOD .EQS. "1" THEN GOTO MOD1
$IF MOD .EQS. "2" THEN GOTO MOD2
$IF MOD .EQS. "3" THEN GOTO MOD3
$IF MOD .EQS. "4" THEN GOTO MOD4
$MOD1:
$@MOD1.COM
$GOTO Q1
$MOD2:
$@MOD2.COM
$GOTO Q1
$MOD3:
$@MOD3.COM
$GOTO Q1
$MOD4:
$@MOD4.COM
$GOTO Q1
$Q1:
$INQUIRE RTN " WOULD YOU LIKE TO RUN ANOTHER MODULE ? >> "
$IF RTN .EQS. "Y" GOTO START
```

A3.2.1 FLAT STIFFENED PANEL MODULE # 1 DRIVER ############

```
$!
$!                 DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO ACT AS THE
$!  BASIC RUN DRIVER FOR THE STRINGER STIFFENED
$!  FLAT PANEL GENERIC MODEL AS PART OF
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.2   6-28-91
$!  AUTHOR: A. Y. WEI AND W. M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#########################################################################
$START:
$  W := WRITE SYS$OUTPUT
$FILE=F$SEARCH("TAPE10.DAT")
$IF FILE .EQS. "" THEN GOTO FD1
$DELETE TAPE10.DAT;*
$FD1:
$FILE=F$SEARCH("TAPE11.DAT")
$IF FILE .EQS. "" THEN GOTO FD2
$DELETE TAPE11.DAT;*
$FD2:
$W ""
$W "
$W "          Generic Model for Flat Stringer Stiffened Panel"
$W ""
$W "                            MODULE 1
$W "                         VAX VERSION 1.2"
$W "                            JUNE 1991
$W ""
$W ""
$W "BASIC ANALYSIS INFORMATION:"
$W ""
$INQUIRE DB "Does a DIAL database file exist?(Y/N) Default is FIL002. >> "
$IF DB .EQS. "Y" THEN GOTO STP1
$IF DB .EQS. "N" THEN GOTO STP2

$STP1:
$W ""
$INQUIRE D1 "What is the name of the DIAL database? >> "
$ASSIGN 'D1' FIL002.
$GOTO END1

$STP2:
$W ""
$INQUIRE DBN "Would you like to give the data base file a unique name?(Y/N) >> "
$IF DBN .EQS. "Y" THEN GOTO STP2A
$ASSIGN FIL002. FIL002.
$GOTO STP2B
$STP2A:
$INQUIRE D1 "Input database file name >> "
$ASSIGN 'D1' FIL002.
$STP2B:
$W ""
$INQUIRE D2 "Does a geometry file already exist for this analysis?(Y/N) >> "
$IF D2 .EQS. "Y" THEN GOTO STP3
$IF D2 .EQS. "N" THEN GOTO STP4
```



```
$STP3:
$W ""
```

```
$INQUIRE GEOM "What is the name of the existing geometry file? >> "
$COPY  'GEOM' TAPE10.DAT
$GOTO STP4A

$STP4:
$W ""
$INQUIRE GEOM "Input a name for the geometry file. >> "
$FILE=F$SEARCH(GEOM)
$IF FILE .EQS. "" THEN GOTO FD3
$DELETE 'GEOM';*
$FD3:

$STP4A:
$W ""
$W ""
$INQUIRE MAT "Does a material properties file exist?(Y/N) >> "
$IF MAT .EQS. "Y" THEN GOTO STP5
$IF MAT .EQS. "N" THEN GOTO STP6

$STP5:
$W ""
$INQUIRE MAT "What is the name of the existing material properties file? >> "
$COPY  'MAT' TAPE11.DAT
$GOTO STP6A

$STP6:
$W ""
$INQUIRE MAT "Input a name for the material properties file. >> "
$FILE=F$SEARCH(MAT)
$IF FILE .EQS. "" THEN GOTO FD4
$DELETE 'MAT';*
$FD4:

$STP6A:
$define/user sys$input sys$command
$!RUN GSTIFVAX.EXE
$!RUN UD11:[SCRATCH.E586018.SPAN]SP910.EXE
$RUN SP910.EXE
$W ""
$W "ASSEMBLY OF DIAL INPUT RUNSTREAMS COMPLETED"

$W ""
$INQUIRE SAVE "Would you like to rename the geometry file?(Y/N) >> "
$IF SAVE .EQS. "N" THEN GOTO STEP6B
$W ""
$INQUIRE NG "Input new name for geometry file >> "
$COPY TAPE10.DAT TAPE10.
$COPY TAPE10. 'NG'
$STEP6B:
$COPY TAPE10.DAT TAPE10.
$COPY TAPE10. 'GEOM'
$W ""
$INQUIRE SAVE "Would you like to rename the material file?(Y/N) >> "
$IF SAVE .EQS. "N" THEN GOTO STEP6C
$W ""
$INQUIRE NM "Input new name for material file >> "
$COPY TAPE11.DAT TAPE11.
$COPY TAPE11. 'NM'
$STEP6C:
$COPY TAPE11.DAT TAPE11.
$COPY TAPE11. 'MAT'
$W ""
```

```
$INQUIRE DEXT "Do you want to proceed with code execution?(Y/N) >>"
$IF DEXT .EQS. "N" THEN GOTO CLOSEA
$W ""
$W "MESH GENERATION STARTED."
$@TAPE12A.COM
$W "MESH GENERATION COMPLETED."
$@TAPE12B.COM
$W "BANDWIDTH OPTIMIZATION COMPLETED."
$@TAPE12C.COM
$W "MATERIAL PROCESSOR COMPLETED."
$@TAPE12D.COM
$W "LOAD PROCESSOR COMPLETED."
$W ""
$W " Do you want to C)ontinue solve processor interactively?"
$W "                B)submit as a batch job?"
$W "                P)go to post processor directly?"
$W "                E)exit this module? (option to write CRAY COM file)"
$INQUIRE SOL "Please select >>"
$IF SOL .EQS. "B" THEN GOTO BATCH
$IF SOL .EQS. "P" THEN GOTO END1
$IF SOL .EQS. "E" THEN GOTO CLOSE
$@TAPE12E.COM
$GOTO END1

$BATCH:
$W ""
$W "AUTOMATIC BATCH PROCESSING NOT IMPLEMENTED AT THIS TIME"
$W "USER MUST SUBMIT DIAL SOLVE PROCESSOR IN FILE TAPE12E.COM MANUALLY."
$W "THE DIAL DATA BASE GENERATED HERE MUST BE USED OR REGENERATED ON"
$W "WHATEVER MACHINE IS CHOSEN FOR BATCH PROCESSING."
$W ""
$W "Due you wish to proceed into post processing to view geometry or loads?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$@PPMOD1.COM
$GOTO END2

$END1:
$W ""
$W "Due you wish to proceed into post processing of analysis results?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$!@PP.COM
$@PPMOD1.COM

$END2:
$W ""
$INQUIRE P "Do you want to end this session?(Y/N) >> "
$IF P .EQS. "Y" THEN GOTO CLOSE
$IF P .EQS. "N" THEN GOTO STP2B

$CLOSE:
$PUR *.OUT
$CLOSEA:
$FILE=F$SEARCH("TAPE10.DAT")
$IF FILE .EQS. "" THEN GOTO FD10
$DELETE TAPE10.DAT;*
$FD10:
$FILE=F$SEARCH("TAPE11.DAT")
$IF FILE .EQS. "" THEN GOTO FD11
$DELETE TAPE11.DAT;*
$FD11:
```

```
$FILE=F$SEARCH("TAPE10.")
$IF FILE .EQS. "" THEN GOTO FD12
$DELETE TAPE10.;*
$FD12:
$FILE=F$SEARCH("TAPE11.")
$IF FILE .EQS. "" THEN GOTO FD13
$DELETE TAPE11.;*
$FD13:
$PUR 'GEOM'
$PUR 'MAT'
$PUR 'NG'
$PUR 'NM'
$PUR 'CNAM'
$DEASSIGN FIL002.
$W ""
$INQUIRE CR "Do you want to write a CRAY COM file?(Y/N) >> "
$IF CR .EQS. "Y" THEN GOTO CRAY
$IF CR .EQS. "N" THEN GOTO NOCR
$CRAY:
$W ""
$INQUIRE CNAM "What name would you give the CRAY runstream? >> "
$RUN CONVERT.EXE
$COPY  CRAY.COM 'CNAM'
$W ""
$NOCR:
$W ""
$W "END OF THIS SESSION FOR FLAT STRINGER STIFFENED PANEL.   Bye!!!"
$W ""
```

A3.2.2 CURVED STIFFENED PANEL MODULE # 2 DRIVER ##############

```
$!
$!                 DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO ACT AS THE
$!  BASIC RUN DRIVER FOR THE STRINGER STIFFENED
$!  CURVED PANEL GENERIC MODEL AS PART OF
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.2   6-28-91
$!  AUTHOR: A. Y. WEI AND W. M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#########################################################################
$START:
$  W := WRITE SYS$OUTPUT
$FILE=F$SEARCH("TAPE10.DAT")
$IF FILE .EQS. "" THEN GOTO FD1
$DELETE TAPE10.DAT;*
$FD1:
$FILE=F$SEARCH("TAPE11.DAT")
$IF FILE .EQS. "" THEN GOTO FD2
$DELETE TAPE11.DAT;*
$FD2:
$W ""
$W "
$W "          Generic Model for Curved Stringer Stiffened Panel"
$W ""
$W "                            MODULE 2
$W "                         VAX VERSION 1.2"
$W "                            JUNE 1991
$W ""
$W ""
$W "BASIC ANALYSIS INFORMATION:"
$W ""
$INQUIRE DB "Does a DIAL database file exist?(Y/N) Default is FIL002. >> "
$IF DB .EQS. "Y" THEN GOTO STP1
$IF DB .EQS. "N" THEN GOTO STP2

$STP1:
$W ""
$INQUIRE D1 "What is the name of the DIAL database? >> "
$ASSIGN 'D1' FIL002.
$GOTO END1

$STP2:
$W ""
$INQUIRE DBN "Would you like to give the data base file a unique name?(Y/N) >> "
$IF DBN .EQS. "Y" THEN GOTO STP2A
$ASSIGN FIL002. FIL002.
$GOTO STP2B
$STP2A:
$INQUIRE D1 "Input database file name >> "
$ASSIGN 'D1' FIL002.
$STP2B:
$W ""
$INQUIRE D2 "Does a geometry file already exist for this analysis?(Y/N) >> "
$IF D2 .EQS. "Y" THEN GOTO STP3
$IF D2 .EQS. "N" THEN GOTO STP4
```



```
$STP3:
$W ""
```

```
$INQUIRE GEOM "What is the name of the existing geometry file? >> "
$COPY  'GEOM' TAPE10.DAT
$GOTO STP4A

$STP4:
$W ""
$INQUIRE GEOM "Input a name for the geometry file. >> "
$FILE=F$SEARCH(GEOM)
$IF FILE .EQS. "" THEN GOTO FD3
$DELETE 'GEOM';*
$FD3:

$STP4A:
$W ""
$W ""
$INQUIRE MAT "Does a material properties file exist?(Y/N) >> "
$IF MAT .EQS. "Y" THEN GOTO STP5
$IF MAT .EQS. "N" THEN GOTO STP6

$STP5:
$W ""
$INQUIRE MAT "What is the name of the existing material properties file? >> "
$COPY  'MAT' TAPE11.DAT
$GOTO STP6A

$STP6:
$W ""
$INQUIRE MAT "Input a name for the material properties file. >> "
$FILE=F$SEARCH(MAT)
$IF FILE .EQS. "" THEN GOTO FD4
$DELETE 'MAT';*
$FD4:

$STP6A:
$define/user sys$input sys$command
$!RUN GSTIFVAX.EXE
$!RUN UD11:[SCRATCH.E586018.SPAN]SP910.EXE
$RUN curvep.EXE
$W ""
$W "ASSEMBLY OF DIAL INPUT RUNSTREAMS COMPLETED"

$W ""
$INQUIRE SAVE "Would you like to rename the geometry file?(Y/N) >> "
$IF SAVE .EQS. "N" THEN GOTO STEP6B
$W ""
$INQUIRE NG "Input new name for geometry file >> "
$COPY TAPE10.DAT TAPE10.
$COPY TAPE10. 'NG'
$STEP6B:
$COPY TAPE10.DAT TAPE10.
$COPY TAPE10. 'GEOM'
$W ""
$INQUIRE SAVE "Would you like to rename the material file?(Y/N) >> "
$IF SAVE .EQS. "N" THEN GOTO STEP6C
$W ""
$INQUIRE NM "Input new name for material file >> "
$COPY TAPE11.DAT TAPE11.
$COPY TAPE11. 'NM'
$STEP6C:
$COPY TAPE11.DAT TAPE11.
$COPY TAPE11. 'MAT'
$W ""
```

```
$INQUIRE DEXT "Do you want to proceed with code execution?(Y/N) >>"
$IF DEXT .EQS. "N" THEN GOTO CLOSEA
$W ""
$W "MESH GENERATION STARTED."
$@TAPE12A.COM
$W "MESH GENERATION COMPLETED."
$@TAPE12B.COM
$W "BANDWIDTH OPTIMIZATION COMPLETED."
$@TAPE12C.COM
$W "MATERIAL PROCESSOR COMPLETED."
$@TAPE12D.COM
$W "LOAD PROCESSOR COMPLETED."
$W ""
$W " Do you want to C)ontinue solve processor interactively?"
$W "                B)submit as a batch job?"
$W "                P)go to post processor directly?"
$W "                E)exit this module? (option to write CRAY COM file)"
$INQUIRE SOL "Please select >>"
$IF SOL .EQS. "B" THEN GOTO BATCH
$IF SOL .EQS. "P" THEN GOTO END1
$IF SOL .EQS. "E" THEN GOTO CLOSE
$@TAPE12E.COM
$GOTO END1

$BATCH:
$W ""
$SUBMIT/NOPRINTER/NOTIFY/QUE=SYS$SLOW TAPE12E.COM
$W "BATCH JOB SUBMITTED ON QUE SYS$SLOW"
$GOTO END2
$!W "AUTOMATIC BATCH PROCESSING NOT IMPLEMENTED AT THIS TIME"
$!W "USER MUST SUBMIT DIAL SOLVE PROCESSOR IN FILE TAPE12E.COM MANUALLY."
$!W "THE DIAL DATA BASE GENERATED HERE MUST BE USED OR REGENERATED ON"
$!W "WHATEVER MACHINE IS CHOSEN FOR BATCH PROCESSING."
$!W ""
$W "Due you wish to proceed into post processing to view geometry or loads?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$@PPMOD2.COM
$GOTO END2

$END1:
$W ""
$W "Due you wish to proceed into post processing of analysis results?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$@PPMOD2.COM

$END2:
$W ""
$INQUIRE P "Do you want to end this session?(Y/N) >> "
$IF P .EQS. "Y" THEN GOTO CLOSE
$IF P .EQS. "N" THEN GOTO STP2B

$CLOSE:
$PUR *.OUT
$CLOSEA:
$FILE=F$SEARCH("TAPE10.DAT")
$IF FILE .EQS. "" THEN GOTO FD10
$DELETE TAPE10.DAT;*
$FD10:
$FILE=F$SEARCH("TAPE11.DAT")
$IF FILE .EQS. "" THEN GOTO FD11
```

```
$DELETE TAPE11.DAT;*
$FD11:
$FILE=F$SEARCH("TAPE10.")
$IF FILE .EQS. "" THEN GOTO FD12
$DELETE TAPE10.;*
$FD12:
$FILE=F$SEARCH("TAPE11.")
$IF FILE .EQS. "" THEN GOTO FD13
$DELETE TAPE11.;*
$FD13:
$PUR 'GEOM'
$PUR 'MAT'
$PUR 'NG'
$PUR 'NM'
$PUR 'CNAM'
$DEASSIGN FIL002.
$W ""
$INQUIRE CR "Do you want to write a CRAY COM file?(Y/N) >> "
$IF CR .EQS. "Y" THEN GOTO CRAY
$IF CR .EQS. "N" THEN GOTO NOCR
$CRAY:
$W ""
$INQUIRE CNAM "What name would you give the CRAY runstream? >> "
$RUN CONVERT.EXE
$COPY  CRAY.COM 'CNAM'
$W ""
$NOCR:
$W ""
$W "END OF THIS SESSION FOR CURVED STRINGER STIFFENED PANEL.   Bye!!!"
$W ""
```

A3.2.3 TUBULAR TRUSS CORE PANEL MODULE # 3 DRIVER

```
$!                  Modified by JTH, LASC,
$!                  from MOD2.COM (8/23/91) by AYW/WMB, LMSC
$!
$!                  Calls TUB212.EXE and P2F20X.COM
$!
$!  ------------------------------------------------
$!               DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO ACT AS THE
$!  BASIC RUN DRIVER FOR THE TUBULAR STIFFENED
$!  FLAT SANDWITCH GENERIC MODEL AS PART OF
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.3   2-28-92
$!  AUTHOR: A. Y. WEI AND W. M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!  ------------------------------------------------
$START:
$  W := WRITE SYS$OUTPUT
$W ""
$W "        Generic Model for Flat Rectangular Tubular Panel (TUB) "
$W ""
$W "                              MODULE 3      "
$W "                           VAX VERSION 1.3  "
$W "                              February 1992 "
$W ""
$W "BASIC ANALYSIS INFORMATION:"
$W ""

$STPO:
$INQUIRE DB "Does a DIAL database file exist?(Y/N) >> "
$IF DB .EQS. "Y" THEN GOTO STP1
$IF DB .EQS. "N" THEN GOTO EXE1

$STP1:
$W ""
$INQUIRE FN "Is FILOO2. the name of the DIAL database file? (Y/N) >> "
$IF      FN .EQS. "Y" THEN GOTO STP1A
$W ""
$INQUIRE D1 "What is the name of the DIAL database? >> "
$COPY 'D1' FILOO2.
$STP1A:

$W ""
$INQUIRE P3 "Is the DIAL solution run done already ? (Y/N)>> "
$IF P3 .EQS. "Y" THEN GOTO END1

$W ""
$INQUIRE P2 "Have the DIAL data and geometry check run been done? (Y/N)>> "
$IF P2 .EQS. "Y" THEN GOTO STP6A1

$EXE1:
$W ""
$INQUIRE P1 "Have the DIAL runstreams been generated? (Y/N)>> "
$IF P1 .EQS. "Y" THEN GOTO LAST

$W ""
$FILE=F$SEARCH("TMAT.DAT")
$IF FILE .EQS. "" THEN GOTO STP21
$RENAME TMAT.DAT TMATK.DAT
```

```
$W "TMAT.dat was renamed as TMATK.dat"

$STP21:
$W ""
$FILE=F$SEARCH("TGEOM.DAT")
$IF FILE .EQS. "" THEN GOTO STP22
$RENAME TGEOM.DAT TGEOMK.DAT
$W "TGEOM.dat was renamed as TGEOMK.dat"

$STP22:
$W ""
$INQUIRE D2 "Does a geometry input file exist for this analysis?(Y/N) >> "
$IF D2 .EQS. "Y" THEN GOTO STP3
$IF D2 .EQS. "N" THEN GOTO STP4

$STP3:
$W ""
$W "What is the name of the existing geometry file? "
$INQUIRE GEOM "Do not use TGEOM or TGEOM.DAT >> "
$COPY 'GEOM' TGEOM.DAT
$GOTO STP4A

$STP4:
$W ""
$W "Input a name for the geometry file."
$INQUIRE GEOM "Do not use TGEOM or TGEOM.DAT >> "

$STP4A:
$W ""
$INQUIRE D3 "Does a material properties input file exist? (Y/N) >> "
$IF D3 .EQS. "Y" THEN GOTO STP5
$IF D3 .EQS. "N" THEN GOTO STP6

$STP5:
$W ""
$W "What is the name of the existing material properties file? "
$INQUIRE MAT "Do not use TMAT or TMAT.DAT >> "
$COPY 'MAT' TMAT.DAT
$GOTO EXE2

$STP6:
$W ""
$W "Input a name for the material properties file.
$INQUIRE MAT "Do not use TMAT or TMAT.DAT >> "

$EXE2:
$define/user sys$input sys$command
$W ""
$W " Starting to run TUB.EXE ..."
$RUN  UD11:[SCRATCH.E586018.TUB]TUB212.EXE
$W ""
$W " The DIAL runstreams were generated - end of TUB.EXE. "
$!  ------

$FILE=F$SEARCH("TMAT.DAT")
$IF FILE .EQS. "" THEN GOTO GE01
$W ""

$INQUIRE P "Do you want to assign a new material data file name? (Y/N) >> "
$IF P .EQS. "N" THEN GOTO MAT0
$INQUIRE MAT "The new material data file name is :>> "
$MAT0:
```

```
$INQUIRE P2 "Did you change material data? (Y/N) >> "
$IF P2 .EQS. "N" THEN GOTO MAT1
$COPY TMAT.DAT 'MAT'
$GOTO MAT2
$MAT1:
$IF P .EQS. "Y" THEN COPY TMAT.DAT. 'MAT'
$MAT2:

$W ""
$DEL TMAT.dat;*
$W "The TMAT.dat file was deleted. "
$W ""
$W "Your designated material file was not purged. "
$! -------

$GEO1:
$FILE=F$SEARCH("TGEOM.DAT")
$IF FILE .EQS. "" THEN GOTO LAST
$W ""

$INQUIRE P "Do you want to assign a new geometry data file name? (Y/N) >> "
$IF P .EQS. "N" THEN GOTO GEOO
$INQUIRE GEOM "The new geometry data file name is: >> "
$GEOO:

$INQUIRE P4 "Did you change geometry data? (Y/N) >> "
$IF P4 .EQS. "N" THEN GOTO GE01A
$COPY TGEOM.DAT 'GEOM'
$GOTO GE02
$GE01A:
$IF P .EQS. "Y" THEN COPY TGEOM.DAT. 'GEOM'
$GE02:

$W ""
$DEL TGEOM.dat;*
$W "The TGEOM.dat file was deleted. "
$W ""
$W "Your designated geometry file was not purged. "
$! -----------------

$LAST:
$W ""
$INQUIRE P "Have the DIAL runstreams been executed? (Y/N) >> "
$IF P .EQS. "Y" THEN GOTO STP6A1

$W ""
$INQUIRE P "Do you want to execute the DIAL runstreams? (Y/N) >> "
$IF P .EQS. "N" THEN GOTO END2
$!   ------------------------------

$W ""
$W "The DIAL runstreams are being executed. "
$W ""
$@TMESH.COM
$W "MESH GENERATION COMPLETED."

$@TBSET.COM
$W "BANDWIDTH OPTIMIZATION COMPLETED."

$@TMAT.COM
$W "MATERIAL PROCESSOR COMPLETED."
```

```
$@TLOAD.COM
$W "LOAD PROCESSOR COMPLETED."

$STP6A1:
$W ""
$W "Do you want to REVIEW the DIAL INPUTs and PLOTs "
$INQUIRE P "before the SOLUTION run? (Y/N) >> "
$IF P .EQS. "N" THEN GOTO STP6A2
$@UD11:[SCRATCH.E586018.TUB]p2f20x.COM

$STP6A2:
$W ""
$INQUIRE P "Do you want to execute DIAL for a solution? (Y/N) >> "
$IF P .EQS. "N" THEN GOTO END2
$! ----------------------------------

$EXE4:
$W ""
$W "Do you wish to call the SOLVE processor now or submit as a batch job ? "
$INQUIRE SOL "Type C to solve now; type B to submit as a Batch job >> "
$IF SOL .EQS. "B" THEN GOTO BATCH
$@TSOLVE.COM
$GOTO END1

$BATCH:
$W ""
$W "AUTOMATIC BATCH PROCESSING NOT IMPLEMENTED AT THIS TIME"
$W "USER MUST SUBMIT DIAL SOLVE PROCESSOR IN FILE TLOAD.COM MANUALLY."
$W "THE DIAL DATA BASE GENERATED MUST BE USED OR REGENERATED ON"
$W "WHATEVER MACHINE IS CHOSEN FOR BATCH PROCESSING."

$! ------------------------------------------
$END1:
$W ""
$W "DO YOU WANT TO PROCESS the ANALYSIS RESULTS?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$@UD11:[SCRATCH.E586018.TUB]p2f20x.com

$! ------------------------------------------
$END2:
$W ""
$W         " Do you want to give the DIAL database file FIL002."
$INQUIRE P " a new file name? (Y/N) >> "
$IF P .EQS. "N" THEN GOTO END2A
$INQUIRE D1 "Input the new DIAL database file name : >> "
$RENAME FIL002. 'D1'
$PUR 'D1'
$W "The DIAL database file FIL002. has been renamed as you wanted."
$GOTO END2B
$W ""
$END2A:
$W   "The DIAL database is still stored in file FIL002."
$END2B:

$W ""
$INQUIRE P "Do you want to end this session?(Y/N) >> "
$IF P .EQS. "Y" THEN GOTO STP7
$W ""
$W "OK, you want to preprocess another tubular panel problem."
$IF P .EQS. "N" THEN GOTO STP0
```

```
$STP7:
$W ""
$W "END OF THIS SESSION. Bye!!!"
```

A.3.2.4 GEODESIC CURVED STIFFENED PANEL MODULE # 4 DRIVER #####

```
$!
$!                  DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO ACT AS THE
$!  BASIC RUN DRIVER FOR THE GEODESICALLY STIFFENED
$!  CURVED GENERIC MODEL AS PART OF
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.1   12-4-90
$!  AUTHOR: A. Y. WEI AND W. M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#########################################################################
$START:
$  W := WRITE SYS$OUTPUT
$W ""
$W "
$W "       Generic Model for Geodesically Reinforced Cylindrical Panel"
$W ""
$W "                              MODULE 4
$W "                           VAX VERSION 1.1"
$W ""
$W ""
$W "BASIC ANALYSIS INFORMATION:"
$W ""
$INQUIRE DB "Does a DIAL database file exist?(Y/N) Default is FIL002. >> "
$IF DB .EQS. "Y" THEN GOTO STP1
$IF DB .EQS. "N" THEN GOTO STP2

$STP1:
$W ""
$INQUIRE D1 "What is the name of the DIAL database? >> "
$ASSIGN 'D1' FIL002.
$GOTO END1

$STP2:
$W ""
$INQUIRE D2 "Does a geometry file already exist for this analysis?(Y/N) >> "
$IF D2 .EQS. "Y" THEN GOTO STP3
$IF D2 .EQS. "N" THEN GOTO STP4

$STP3:
$W ""
$INQUIRE GEOM "What is the name of the existing geometry file? >> "
$ASSIGN  'GEOM' GEOM
$GOTO STP4A

$STP4:
$W ""
$W "Input a name for the geometry file.
$INQUIRE GEOM "Do not use GEOM or GEOM.DAT >> "
$ASSIGN  'GEOM' GEOM
$STP4A:
$INQUIRE MAT "Does a material properties file exist?(Y/N) >> "
$IF MAT .EQS. "Y" THEN GOTO STP5
$IF MAT .EQS. "N" THEN GOTO STP6

$STP5:
$W ""
$INQUIRE MAT "What is the name of the existing material properties file? >> "
```

```
$ASSIGN  'MAT' MAT
$GOTO STP6A

$STP6:
$W "Input a name for the material properties file.
$INQUIRE MAT "Do not use MAT or MAT.DAT >> "
$ASSIGN  'MAT' MAT

$STP6A:
$define/user sys$input sys$command
$RUN geo.exe
$W ""
$W "CREATION OF DIAL RUNSTREAMS COMPLETED, MESH GENERATION STARTED."
$@FMESH.COM
$W "MESH GENERATION COMPLETED."
$@FBSET.COM
$W "BANDWIDTH OPTIMIZATION COMPLETED."
$@FMAT.COM
$W "MATERIAL PROCESSOR COMPLETED."
$@FLOAD.COM
$W "LOAD PROCESSOR COMPLETED."
$W ""
$W "Do you wish to enter the solve processor now or submit as a"
$INQUIRE SOL "batch job? (Continue=C, Batch job=B) >> "
$IF SOL .EQS. "B" THEN GOTO BATCH
$@FSOLVE.COM
$GOTO END1

$BATCH:
$W ""
$W "AUTOMATIC BATCH PROCESSING NOT IMPLEMENTED AT THIS TIME"
$W "USER MUST SUBMIT DIAL SOLVE PROCESSOR IN FILE FLOAD.COM MANUALLY."
$W "THE DIAL DATA BASE GENERATED MUST BE USED OR REGENERATED ON"
$W "WHATEVER MACHINE IS CHOSEN FOR BATCH PROCESSING."
$W ""
$W "DUE YOU WISH TO PROCEED INTO POST PROCESSING TO VIEW GEOMETRY OR LOADS?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$@PPMOD3.COM
$GOTO END2

$END1:
$W ""
$W "DUE YOU WISH TO PROCEED INTO POST PROCESSING OF ANALYSIS RESULTS?"
$INQUIRE PP "(Y/N) >> "
$IF PP .EQS. "N" THEN GOTO END2
$@PPMOD3.COM

$END2:
$W ""
$INQUIRE P "Do you want to end this session?(Y/N) >> "
$IF P .EQS. "Y" THEN GOTO STP7
$IF P .EQS. "N" THEN GOTO STP6

$STP7:
$FILE=F$SEARCH("MAT.DAT")
$IF FILE .EQS. "" THEN GOTO GEOM
$DELETE MAT.DAT;*
$GEOM:
$FILE=F$SEARCH("GEOM.DAT")
$IF FILE .EQS. "" THEN GOTO LAST
```

```
$DELETE GEOM.DAT;*
$LAST:
$DEASSIGN FIL002.
$DEASSIGN MAT
$DEASSIGN GEOM
$W ""
$W "END OF THIS SESSION.   Bye!!!"
```

A3.3.1 FLAT STIFFENED PANEL MODULE # 1 PROGRAM

```
C 9/11/91 9/10/91, 3671+4+10 lines, SP910.FOR, from Tony Wei, 9/09/91
C
      PROGRAM GBINP
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
c ...1T1(10,5),SXT(10,5),SXC(10,5),SYT(10,5),SYC(10,5),SS(10,5),NT(10)
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ... ***************************************************
c     *  Advanced Compostie Structural Program Input     *
c     *  PROBLEM 1A: Laminated flat panel w/ SECAUT Beam *
c     *                                                  *
c     *  variables definition ...                        *
c     *                                                  *
c     *   [ Geometry data ]                              *
c     *   M = numbers of axial stiffeners                *
c     *   N = numbers of lateral ribs                    *
c     *   MI= numbers of sub-grid between stiffeners     *
c     *   NI= numbers of sub-grid between ribs           *
c     *   A = axial stiffener distances (M+2)            *
c     *   B = lateral rib distances (N+2)                *
c     *                                                  *
c     *   MSET  1: STF1                                  *
c     *   MSET  2: STF2                                  *
c     *   MSET  n: STFn                                  *
c     *   MSET 10: BASE                                  *
c     *                                                  *
c     *    yal0: read tape10                             *
c     *    yal1: read tape11                             *
c     *    nal0: input data for the new tape10           *
c     *    nal1: input data for the new tape11           *
c     *    flis0: list tape10                            *
c     *    flis1: list tape11                            *
c     *    fmody: modify input data                      *
c     *    fsave: update tape10, tape11                  *
c     *    fquit: quit and end                           *
c     *                                                  *
c     *    PROGRAMMED by Lloyd Chou    81-12 MSD         *
c     *    DATED December 5, 1990      ext.6-3075        *
c     *    UPDATED by Tony Wei         81-12 MSD         *
c     *     6/28/91                    ext.6-1137        *
c     *                                                  *
c     ***************************************************
      LOGICAL here10,here11,here12
      CHARACTER NAME*80
      dimension NAME(10)
c ...
c . checking input data for DIAL F.E. structural analysis
c ...
c . initialize geometry(tape10) and material(tape11) files
c ...
      inquire( file='tape10', exist=here10)
      inquire( file='tape11', exist=here11)
      if (here10) then
        call yal0
```

```
                else
        call nal0
      endif
      if (herell) then
        call yall
                else
        call nall
      endif
c .....
      call flist
c .....
      close(unit=10)
      close(unit=11)
c .....
      open(unit=10,status='unknown',file='tape10')
      open(unit=11,status='unknown',file='tape11')
      call fsave
c .....
c ###
c # end of main program
c ###
      STOP
      end
      SUBROUTINE NEWPAGE
      do 1 I=1,24
  1   write(6,'('' '')')
      write(6,2)
  2   format(/3x,'*************************************************',
     1       /3x,'* Advanced Composite Structural Program by LMSC  *',
     2       /3x,'* Flat Stiffener Panel : Revision 1.2, 6/28/91   *',
     3       /3x,'*************************************************')
      return
      end
      SUBROUTINE QUIT
      write(6,1)
  1   format(//5x,'************************************',
     1        /5x,'*  No Modification, Exiting Module!   *',
     2        /5x,'************************************')
      STOP
      end
      SUBROUTINE yal0
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . read tape10
c ...
      character NAME*80
      dimension NAME(10)
      open(unit=10,status='old',file='tape10')
  11  format(4I5,2F10.0)
  12  format(8F10.0)
  13  format(6I5)
  14  format(4F12.0)
  15  format(4i3,f10.5)
      read(10,11) M,N,MI,NI,AX,BY
```

```
      M2 = M+2
      N2 = N+2
      read(10,12) (A(I),I=1,M2)
      read(10,12) (B(I),I=1,N2)
c ...
      read(10,13) LEND,LRIB,LRBC,LEBC,LSBC,LTOT
      read(10,14) (FL(I),I=1,4)
      read(10,15) NBUC,NDF,ND,MO,TAR
      return
      end
c ...
c ...
      SUBROUTINE yall
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . read tape11
c ...
      character NAME*80
      dimension NAME(10)
      open(unit=11,status='old',file='tape11')
  11  format(9I5)
  12  format(6F12.0)
  13  format(I5,2F10.0)
      read(11,11) NMAT
      do 110 I=1,NMAT
       J = 1
      read(11,12) (E1(K,I,J),K=1,3),(V1(K,I,J),K=1,3)
      read(11,12) (G1(K,I,J),K=1,3),(A1(K,I,J),K=1,3)
      read(11,12) (SS(K,I,J),K=1,6)
 110  read(11,12) (SS(K,I,J),K=7,12)
c ...
      read(11,11) NLAY
      do 120 I=1,NLAY
 120  read(11,13) MW(I),TW(I),PW(I)
      read(11,11) MTYP
      read(11,12) XH,XB,XC,XT
      if (MTYP.eq.1.or.MTYP.eq.5) then
         IP=2
      elseif(MTYP.eq.2) then
         IP=4
      else
         IP=3
      endif
      do 130 I=1,IP
      read(11,11) MLAY(I)
      do 130 J=1,MLAY(I)
 130  read(11,13) MS(J,I),TS(J,I),PS(J,I)
      return
      end
c ...
c ...
      SUBROUTINE na10
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
```

```
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
c ...
c . create input for tape10
c ...
c ...
   11  format(2x,'Input Panel Width and Length >>(',2F8.3,')>'$)
   12  format(2x,'Input No. of Stiffeners and Ribs >>(',2I3,')>'$)
   13  format(2x,'Input No. of Sub-Grid between Stif/Rib >>(',2I2,')>'$)
   14  format(2x,'Input',I2,' Axial Stiffener locations >>'$)
   15  format(2x,'Input',I2,' Lateral Rib locations >>'$)
   20  format(2x,'----- Boundary Condition Definition -----')
   21  format(2x,'Should it represent a part of a much large',
     1' panel?(Y/N)>>(N)>'$)
   22  format(2x,'Are there ribs at Ends ?(Y/N)>>(Y)>'$)
   23  format(2x,'Ribs on:(1)Skin (2)STIF (3)Both ?>>(1)>'$)
   24  format(2x,'Choose a B.C. Type no. at Ribs:'
     1/10x,'(1) Simply Supported; (2) S.S. and Rotation; ',
     2     ' ?>>(1)>'$)
   25  format(2x,'Choose a B.C. type no. at Ends:',
     */10x,'(1) Part of a Larger Panel',
     */10x,'(2) Free',
     1/10x,'(3) Simply Supported',
     3/10x,'(4) Semi-Clamped-1 (axial & lateral free)',
     4/10x,'(5) Semi-Clamped-2 (axial only free)',
     5/10x,'(6) Fully Clamped (all DOF fixed) --->>(1)>'$)
   26  format(2x,'Choose a B.C. type no. at Sides:',
     */10x,'(1) Part of a Larger Panel',
     */10x,'(2) Free',
     1/10x,'(3) Simply Supported',
     3/10x,'(4) Semi-Clamped-1 (axial & lateral free)',
     4/10x,'(5) Semi-Clamped-2 (axial only free)',
     5/10x,'(6) Fully Clamped (all DOF fixed) --->>(2)>'$)
   30  format(2x,'----- Loading Condition Definition -----')
   31  format(2x,'Choose input loads type (Total=1,Line=2)',
     c '?>>(2)>'$)
   32  format(5x,'following total load unit : lb')
   33  format(5x,'following line load unit: lb/in')
   34  format(7x,'Input axial load >>(',F8.1,')>'$)
   35  format(7x,'Input lateral load >>(',F8.1,')>'$)
   36  format(7x,'Input shear load along ends >>(',F8.1,')>'$)
   37  format(7x,'Input lateral pressure in psi >>',F8.2,')>'$)
   38  format(2x,'----- Type of Analysis ----')
c ...
      call NEWPAGE
      ra(1)=10.
      ra(2)=10.
      write(6,11) ra(1),ra(2)
      call NREAL(2,ra)
      AX=ra(1)
      BY=ra(2)
      ia(1)=2
      ia(2)=1
      write(6,12) ia(1),ia(2)
      call NINTG(2,ia)
      M=ia(1)
```

```
      N=ia(2)
      ia(1)=2
      ia(2)=2
      write(6,'(2x,''Does the user want to specify Panel Sub-divisions?
     *(Y/N)'')')
      write(6,'(2x,''Enter >>''$)')
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') then
         write(6,*)' '
         write(6,'(2x,''Sub-Grid Between Stiffeners are 2.'')')
         write(6,'(2x,''Sub-Grid Between Ribs are 3.'')')
         MI=2
         NI=3
      elseif (ans .eq. 'Y' .or. ans .eq. 'y') then
         write(6,13) ia(1),ia(2)
         call NINTG(2,ia)
         MI=ia(1)
         NI=ia(2)
      endif
      A(1)=0.
      B(1)=0.
      DA = AX/(M+1)
      DB = BY/(N+1)
      do 1 I=1, M+1
    1    A(I+1) = DA * I
      do 2 I=1, N+1
    2    B(I+1) = DB * I
      write(6,'(5x,''Stiffeners equally spaced (Y/N)?  >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 4
      write(6,14) M
      call NREAL(M,ra)
      do 3 I=1, M
    3    A(I+1) = ra(I)
    4 A(M+2) = AX
      write(6,'(5x,''Ribs equally spaced (Y/N)? >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 6
      write(6,15) N
      call NREAL(N,ra)
      do 5 I=1, N
    5    B(I+1) = ra(I)
    6 B(N+2) = BY
c ...
      write(6,20)
   50 write(6,22)
      ans='Y'
      LEND=1
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 55
      LEND=2
   55 write(6,23)
      ia(1)=1
      call NINTG(1,ia)
      LRIB=ia(1)
      write(6,24)
      ia(1)=1
      call NINTG(1,ia)
```

```
      LRBC=ia(1)
      write(6,25)
      ia(1)=1
      call NINTG(1,ia)
      LEBC=ia(1)
      write(6,26)
      ia(1)=2
      call NINTG(1,ia)
      LSBC=ia(1)
c ...
      write(6,30)
      write(6,31)
      ia(1)=2
      call NINTG(1,ia)
      LTOT=ia(1)
      if (LTOT .eq. 1) write(6,32)
      if (LTOT .ne. 1) write(6,33)
      ra(1)=1000.
      write(6,34) ra(1)
      call NREAL(1,ra)
      FL(1)=ra(1)
      ra(1)=0.
      write(6,35) ra(1)
      call NREAL(1,ra)
      FL(2)=ra(1)
      ra(1)=0.
      write(6,36) ra(1)
      call NREAL(1,ra)
      FL(3)=ra(1)
      ra(1)=0.
      write(6,37) ra(1)
      call NREAL(1,ra)
      FL(4)=ra(1)
c.............Solve Strategy...........
      write(6,38)
      write(6,'('' Choose ONE of the following type of analysis>>'')')
      write(6,'('' 1> Linear Analysis.'')')
      write(6,'('' 2> Bifurcation Buckling Analysis.'')')
      write(6,'('' 3> Post-Buckling Analysis.'')')
      write(6,'('' Enter >>''$)')
      call NINTG(1,NBUC)
      NDF=0
      ND=0
      TAR=0.
      MO=10
      if (NBUC.eq.3) then
c.....Caution message about the Post Buckling Technique
         write(6,'('' Type of Target Point ?>>'')')
         write(6,'('' 0> Load Factor.'')')
         write(6,'('' 1-6> Displacement DOF.'')')
         write(6,'('' Enter>>''$)')
         call NINTG(1,NDF)
         if (NDF.gt.0) then
            write(6,'('' Enter Node Number >>''$)')
            call NINTG(1,ND)
         endif
         write(6,'('' Target Point Value =''$)')
         call NREAL(1,TAR)
      elseif (NBUC .eq. 2) then
         write(6,'('' How many MODE shape(s) do you want (Max. 50)? >>'
```

```
      *'$)')
          call NINTG(1,MO)
          write(6,*)' '
       endif
       return
       end
c ...
c ...
       SUBROUTINE nall
       common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),GRID
       common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
      1              SS(12,10,5),NT(10),NAME
       common /wall/ NLAY,MW(30),TW(30),PW(30)
       common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
       common /solve/ NBUC,TAR,NDF,ND,MO
       dimension ra(10),ia(10),NAME(10)
       character ans*1,NAME*80
c ...
c . create input for tapell
c ...
  11   format(2x,'Input No. of Materials >>(',I2,')>'$)
  13   format(2x,'Input Matl#',I2,' Elastic & Shear Modulus,',
     1 ' Poison #, AML +/- Allowables')
  14   format(5x,'E1,E2,E3    >>(',3E9.2,')>'$)
  15   format(5x,'G12,G23,G31 >>(',3E9.2,')>'$)
  16   format(5x,'V12,V23,V31 >>(',3F9.4,')>'$)
  17   format(5x,'+ Allow.>>(',3(F5.1,F7.3),')>'$)
  18   format(5x,'- Allow.>>(',3(F5.1,F7.3),')>'$)
  19   format(2x,'-----------------------------------------------')
c ...
  20   format(2x,'Input No. of plies for the basic panel >>(',I2,')>'$)
  21   format(5x,'Input Matl#, Thickness, Angle')
  22   format(5x,'Ply#',I2,'>>(',I2,F7.4,F7.2,')>'$)
  28   format(2x,'Stiffener Type: 1)Blade, 2)Close Hat, 3)I-shape, 4
     1)J-shape,',/18x,'5)Angle, 6)Zee, 7)Bead, 8)Open Hat.')
  29   format(5x,' Select axial stiffener type >>(',I1,')>'$)
  30   format(5x,'Input XH,XB,XC,XT >>(',4F6.3,')>'$)
  31   format(2x,'Input No. of plies for stiffener ELEM #',I1,'>>(',
     1 I2,')>'$)
  32   format(5x,'Input XH,XB,XC >>(',3F6.3,')>'$)
  33   format(5x,'Input XH,XB >>('2F6.3,')>'$)
  34   format(5x,'Input XH,XB,XT >>('3F6.3,')>'$)
c ...
       write(6,19)
       ia(1)=1
       write(6,11) ia(1)
       call NINTG(1,ia)
       NMAT=ia(1)
       J=1
       do 110 I=1,NMAT
       write(6,13) I
       ra(1)=30.E6
       ra(2)=ra(1)
       ra(3)=ra(1)
       write(6,14) (ra(II),II=1,3)
       call NREAL(3,ra)
       do 101 K=1,3
 101   E1(K,I,J) = ra(K)
c ...
       ra(1)=11.5E6
```

```
      ra(2)=ra(1)
      ra(3)=ra(1)
      write(6,15) (ra(II),II=1,3)
      call NREAL(3,ra)
      do 102 K=1,3
 102  G1(K,I,J) = ra(K)
c ...
      ra(1)=0.30435
      ra(2)=ra(1)
      ra(3)=ra(1)
      write(6,16) (ra(II),II=1,3)
      call NREAL(3,ra)
      do 103 K=1,3
 103  V1(K,I,J) = ra(K)
c ...
      ra(1)=-60.
      ra(2)=0.001
      ra(3)=0.
      ra(4)=0.002
      ra(5)=60.
      ra(6)=0.003
      write(6,'('' Do you want to input AML Failure Parameters (Y/N)?''$
     *)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,17) (ra(II),II=1,6)
         call NREAL(6,ra)
      endif
      do 104 K=1,6
 104  SS(K,I,J) = ra(K)
c ...
      ra(1)=-60.
      ra(2)=-0.002
      ra(3)=0.
      ra(4)=-0.003
      ra(5)=60.
      ra(6)=-0.004
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,18) (ra(II),II=1,6)
         call NREAL(6,ra)
      endif
      do 105 K=7,12
      KK=K-6
 105  SS(K,I,J) = ra(KK)
 110  continue
      write(6,19)
c ...
      ia(1)=2
      write(6,20) ia(1)
      call NINTG(1,ia)
      NLAY=ia(1)
      write(6,21) .
      write(6,*)' '
      write(6,'(2x,''Are the information the same for ALL plies (Y/N)?''
     *$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         ia(1) = 1
         ra(2) = 0.1
         ra(3) = 0.0
```

```
          I = 1
          write(6,22) I, ia(1), ra(2), ra(3)
          call NREAL(3,ra)
          ia(1) = int(ra(1))
          MW(1) = ia(1)
          TW(1) = ra(2)
          PW(1) = ra(3)
          do 119 i=2,NLAY
             MW(i)=mw(1)
             tw(i)=tw(1)
             pw(i)=pw(1)
 119      continue
       else
       do 120 I=1,NLAY
       ia(1)=1
       ra(2)=0.1
       ra(3)=0.0
       if (I .eq. 1) ra(3)=-45.
       if (I .eq. 2) ra(3)= 45.
       write(6,22) I,ia(1),ra(2),ra(3)
       call NREAL(3,ra)
       ia(1) = int(ra(1))
       MW(I) = ia(1)
       TW(I) = ra(2)
 120   PW(I) = ra(3)
       endif
c ...
       write(6,19)
       ia(1)=1
       write(6,28)
       write(6,29) ia(1)
       call NINTG(1,ia)
       MTYP=ia(1)
       ra(1)=0.4
       ra(2)=0.5
       ra(3)=0.35
       ra(4)=0.25
       if (MTYP.eq.1) then
          write(6,32) ra(1),ra(2),ra(3)
          call NREAL(3,ra)
          XH=ra(1)
          XB=ra(2)
          XC=ra(3)
       elseif (MTYP.eq.5.or.MTYP.eq.7) then
          write(6,33) ra(1),ra(2)
          call NREAL(2,ra)
          XH=ra(1)
          XB=ra(2)
       elseif (MTYP.eq.6) then
          write(6,34) XH,XB,XT
          call NREAL(3,ra)
          XH=ra(1)
          XB=ra(2)
          XT=ra(3)
       else
          write(6,30) ra(1),ra(2),ra(3),ra(4)
          call NREAL(4,ra)
          XH=ra(1)
          XB=ra(2)
          XC=ra(3)
```

```
         XT=ra(4)
       endif
c ...
       if (MTYP.eq.1.or.MTYP.eq.5) then
         IP=2
       elseif (MTYP.eq.2) then
         IP=4
       else
         IP=3
       endif
       do 130 I1=1,IP
       write(6,19)
       ia(1)=2
       write(6,31) I1,ia(1)
       call NINTG(1,ia)
       MLAY(I1)=ia(1)
       write(6,21)
       write(6,*)' '
       write(6,'(2x,''Are the information the save for ALL plies (Y/N)?''
      *$)')
       call STITLE(ans)
       if ( ans .eq. 'Y' .or. ans .eq. 'y') then
          ia(1)=1
          ra(2)=0.1
          ra(3)=0.0
          i=1
          write(6,22) i,ia(1),ra(2),ra(3)
          call NREAL(3,ra)
          ia(1) = int(ra(1))
          MS(1,I1) = ia(1)
          TS(1,I1) = ra(2)
          PS(1,I1) = ra(3)
          do 131 i=2,mlay(I1)
             ms(i,il)=ms(1,il)
             ts(i,il)=ts(1,il)
             ps(i,il)=ps(1,il)
 131      continue
       else
       do 132 I=1,MLAY(I1)
       ia(1)=1
       ra(2)=0.1
       ra(3)=0.0
       if (I .eq. 1) ra(3)=-45.
       if (I .eq. 2) ra(3)= 45.
       ia(1)=ra(1)
       write(6,22) I,ia(1),ra(2),ra(3)
       call NREAL(3,ra)
       ia(1) = int(ra(1))
       MS(I,I1) = ia(1)
       TS(I,I1) = ra(2)
 132   PS(I,I1) = ra(3)
       endif
 130   continue
c ...
       return
       end
c ...
       SUBROUTINE menu0
       common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
       common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
```

```
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character NAME*80
    9 format(3x,'MESH MENU (MENU0):')
   10 format(4x,'0,0) Geometry & General Data')
   11 format(4x,'0,1) Base Panel Width and Length (AX,BY:)',2F10.3)
   12 format(4x,'0,2) No. of Stiffeners and Ribs (M,N):',2I5)
   13 format(4x,'0,3) No. of GRID Between Stiffeners and Ribs(MI,NI):',2
     *I5)
   14 format(4x,'0,4) Stiffener_locations (A):',6F7.3)
   15 format(4x,'0,5) Rib_locations (B):',6F7.3)
   16 format(3x,'-----------------------------------------')
      call NEWPAGE
      write(6,9)
      ia(1)=0
      ia(2)=0
      write(6,10)
      write(6,11) AX,BY
      write(6,12) M,N
      write(6,13) MI,NI
      write(6,14) (A(I),I=2,M+1)
      write(6,15) (B(I),I=2,N+1)
      write(6,16)
c ...
      return
      end
c ...
      SUBROUTINE menu1(IMAT)
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
    9 format(3x,'MATL MENU (MENU1):')
   10 format(4x,'1,0) Total No. of Materials (NMAT):',I3)
   11 format(4x,'     Properties for MATL No. :',I2)
   14 format(10x,'Elastic Modulus  (E1,E2,E3):',3E9.2)
   15 format(10x,'Shear Modulus (G12,G23,G31):',3E9.2)
   16 format(10x,'Poisson Ratio (V12,V23,V13):',3F9.4)
   17 format(10x,'AML + Allowables:',3(F5.1,F7.5))
   18 format(10x,'AML - Allowables:',3(F5.1,F7.5))
   19 format(3x,'-----------------------------------------------')
   20 format(10x,'Name of the Material: ',80A)
   21 format(4x,'1,',i1,') Material Number ',i1)
c ...
      call NEWPAGE
      write(6,9)
      write(6,10) NMAT
      do 1 i=1,IMAT
          write(6,21) i,i
 1    continue
c ...
      return
```

```
      end
      SUBROUTINE menu2
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
   9  format(3x,'Base Ply MENU (MENU2):')
  10  format(4x,'2,0) No. of Plies for The Base Panel (NLAY):',I3)
  11  format(4x,'     Ply #  Material #  Thickness  Angle')
  12  format(8x,I4,8x,I2,7x,f7.4,2x,f7.1)
  19  format(3x,'-----------------------------------------------')
c ...
      call NEWPAGE
      write(6,9)
      write(6,10) NLAY
      write(6,11)
      do 1 I=1,NLAY
   1  write(6,12) I,MW(I),TW(I),PW(I)
      write(6,19)
c ...
      return
      end
c ...
      SUBROUTINE menu3(ILAY)
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
  10  format(3x,'STIF MENU (MENU3):')
  11  format(5x,'1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,',
     1 /,5x,' 5)Angle, 6)Zee, 7)Bead, 8)Open Hat.')
  12  format(4x,'3,0) Type of Stiffener :',I3)
  13  format(10x,'XH=',F6.3,' XB=',F6.3)
  14  format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3)
  15  format(10x,'XH=',F6.3,' XB=',F6.3,' XT=',F6.3)
  16  format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3,' XT=',F6.3)
  17  format(4x,'3,',il,') No. of Plies for STIF,ELEM(',I1,'):',I3)
  18  format(4x,'     Ply #  Material #  Thickness  Angle')
  19  format(8x,I4,8x,I2,7x,f7.4,2x,f7.1)
  20  format(3x,'-----------------------------------------------')
c.....
      call NEWPAGE
      write(6,10)
      write(6,11)
      write(6,*)' '
c.. indicate which type of stiffener
      write(6,12) MTYP
c.. print dimensions of the stiffener on screen
      if (MTYP .eq. 5 .or. MTYP .eq. 7) then
         write(6,13) XH,XB
      elseif (MTYP .eq. 6) then
         write(6,15) XH,XB,XT
```

```
      elseif (MTYP .eq. 1) then
         write(6,14) XH,XB,XC
      else
         write(6,16) XH,XB,XC,XT
      endif
c....  print nominal stiffener element info.
      do 1 i=1,ILAY
        write(6,17) i,i,MLAY(i)
   1  continue
c..   back to fmod3 for choosing items to modify ...
      write(6,20)
      return
      end
c ...
      SUBROUTINE menu4
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
c ...
   9  format(3x,'BCLC MENU (MENU4 Boundary Cond. & Loading Cond.):')
  19  format(3x,'-----------------------------------------------')
  20  format(2x,'   ----- Boundary Condition Definition -----')
  21  format(4x,'The model is a part of a much large panel ? --->',A1)
  22  format(4x,'There are ribs at Ends ?   ---------------->',A1)
  23  format(4x,'Ribs on:(1)Skin (2)STIF (3)Both ?  ------------>',I1)
  24  format(4x,'B.C. type @ Ribs: 1) Simply Supp.; 2) S.S. and Rot.;',
     1' -->',I1)
  25  format(4x,'B.C. type no. at Ends:',
     */6x,'1) Part of Larger Panel;',
     */6x,'2) Free;                          3) Simply Supp.;',
     2/6x,'4) Clamped-1/axial+lateral free; 5) Clamped-2/axial free',
     3/6x,'6) Fully Clamped (all DOF fixed) ------>',I1)
  26  format(2x,'B.C. type no. at Sides:',
     */6x,'1) Part of Larger Panel;',
     1/6x,'2) Free;                          3) Simply Supp.;',
     2/6x,'4) Clamped-1/axial+lateral free; 5) Clamped-2/axial free',
     3/6x,'6) Fully Clamped (all DOF fixed) ------>',I1)
  30  format(2x,'   ----- Loading Condition Definition -----')
  31  format(4x,'Loads type (1:Total lb; 2:Line lb/in)  ---->',I1)
  32  format(7x,'axial load    ---------------->',F8.1)
  33  format(7x,'lateral load    -------------->',F8.1)
  34  format(7x,'shear load along ends    ----->',F8.1)
  35  format(7x,'lateral pressure in psi  ----->',F8.2)
c ...
c ...
      call NEWPAGE
      write(6,9)
      write(6,20)
      ans='N'
      if (LEND .eq. 1) ans='Y'
      write(6,22) ans
      write(6,23) LRIB
      write(6,24) LRBC
      write(6,25) LEBC
```

```
      write(6,26) LSBC
      write(6,30)
      write(6,31) LTOT
      write(6,32) FL(1)
      write(6,33) FL(2)
      write(6,34) FL(3)
      write(6,35) FL(4)
c ...
      return
      end
c ...
      SUBROUTINE flist
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character NAME*80
c
c       1         2         3         4         5         6         7 2
c ...
c . list input
c ...
   9  format(4x,'m,n) >>> MAIN MENU <<< ',
     1 '(general MESH data)')
  10  format(4x,'0,0) Geometry & General Data')
  11  format(4x,'0,1) Base Panel Width and Length (AX,BY:)',2F10.3)
  12  format(4x,'0,2) No. of Stiffeners and Ribs (M,N):',2I5)
  13  format(4x,'0,3) No. of GRID for Stiffeners and Ribs(MI,NI):',2I5)
  14  format(4x,'0,4) Stiffener_locations (A):',6F7.3)
  15  format(4x,'0,5) Rib_locations (B):',6F7.3)
c ...
  16  format(4x,'1,0) Total No. of Materials (NMAT):',I2)
  17  format(4x,'-------------------------------------------------')
  18  format(4x,'     >>> OTHER MENUS <<< ',
     1 '(MATL, BASE Ply, STIF Ply, BC & LC)')
c ...
  21  format(4x,'2,0) No. of Plies for The Base Panel(NLAY):',I2)
  29  format(4x,'3,0) Axial stiffener type No.(MTYP):',I2,
     1      /4x,'Blade=1,Close Hat=2,"I"=3,"J"=4,Angle=5,"Z"=6,Bead=7,
     1Open Hat=8.')
  30  format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3,' XT=',F6.3)
  31  format(4x,'3,1) No. of Plies for STIF, Element 1(MLAY1):',I2)
  32  format(4x,'3,2) No. of Plies for STIF, Element 2(MLAY2):',I2)
  33  format(4x,'3,3) No. of Plies for STIF, Element 3(MLAY3):',I2)
  34  format(4x,'3,4) No. of Plies for STIF, Element 4(MLAY4):',I2)
  35  format(4x,'4,0) Boundary & Loading Conditions')
  36  format(10x,"XH=',F6.3,' XB=',F6.3,' XC=',F6.3)
  37  format(10x,'XH=',F6.3,' XB=',F6.3)
  38  format(10x,'XH=',F6.3,' XB=',F6.3,' XT=',F6.3)
c ...
   1  call NEWPAGE
      write(6,9)
      ia(1)=0
      ia(2)=0
      write(6,10)
      write(6,11) AX,BY
```

```
      write(6,12)  M, N
      write(6,13) MI,NI
      write(6,14) (A(I),I=2,M+1)
      write(6,15) (B(I),I=2,N+1)
c ...
      write(6,17)
      write(6,18)
      write(6,16)NMAT
      write(6,21)NLAY
      write(6,29)MTYP
      if (MTYP.eq.1) then
         write(6,36)XH,XB,XC
      elseif (MTYP.eq.5.or.MTYP.eq.7) then
         write(6,37) XH,XB
      elseif (MTYP.eq.6) then
         write(6,38) XH,XB,XT
      else
         write(6,30) XH,XB,XC,XT
      endif
      write(6,35)
c ...
      write(6,'('' Select m,n=> -1:END; 9:QUIT; m,n:UPDATE >>''$)')
      ia(1)=0
      ia(2)=0
      call NINTG(2,ia)
      ISEL1= ia(1)
      ISEL2= ia(2)
50    if (ISEL1 .eq. -1) go to 90
      if (ISEL1 .eq.  9) call QUIT
      if (ISEL1 .eq. 0 .and. ISEL2 .ne. 0) call fmod0(ISEL2)
      if (ISEL1 .eq. 0 .and. ISEL2 .eq. 0) go to 1
      if (ISEL1 .eq. 1 ) call fmod1(ISEL1,ISEL2)
      if (ISEL1 .eq. 2 ) call fmod2(ISEL1,ISEL2)
      if (ISEL1 .eq. 3 ) call fmod3(ISEL1,ISEL2)
      if (ISEL1 .eq. 4 ) call fmod4(ISEL1,ISEL2)
      go to 50
c ...
  90  continue
      return
      end
c ...
c ...
      SUBROUTINE fmod0(IS2)
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1            SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c       1         2         3         4         5         6         7 2
c ...
c . modify geometry input
c ...
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
  11  format(5x,'Modify Panel Width and Length>>(',2F8.3,')>'$)
  12  format(5x,'Modify No. of Stiffeners and Ribs >>(',2I3,')>'$)
  13  format(5x,'Modify No. of Sub-Grid between Stif/Rib >>(',2I2,')>'$)
  14  format(5x,'Modify',I2,' Axial Stiffener locations >>'$)
  15  format(5x,'Modify',I2,' Lateral Rib locations >>'$)
```

```
c ...
      call menu0
      if (IS2 .le. 1) then
         ra(1)=AX
         ra(2)=BY
         write(6,11)ra(1),ra(2)
         call NREAL(2,ra)
         AX=ra(1)
         BY=ra(2)
      endif
      if (IS2 .le. 2) then
         ia(1)=M
         ia(2)=N
         write(6,12) ia(1),ia(2)
         call NINTG(2,ia)
         M=ia(1)
         N=ia(2)
      endif
      if (IS2 .le. 3) then
         ia(1)=MI
         ia(2)=NI
         write(6,13) ia(1),ia(2)
         write(6,*)' '
         write(6,'(5x,''Does the user want to specify Sub-Divisions? (Y/
     *N)'')')
         write(6,'(5x,''Enter >>''$)')
         call STITLE(ans)
         if (ans .eq. 'N' .or. ans .eq. 'n') then
            MI=2
            NI=3
            write(6,'(5x,''Sub-Grids between Stiffeners are 2.'')')
            write(6,'(5x,''Sub-Grids between Ribs are 3.'')')
         elseif (ans .eq. 'Y' .or. ans .eq. 'y') then
            write(6,13) ia(1),ia(2)
            call NINTG(2,ia)
            MI=ia(1)
            NI=ia(2)
         endif
      endif
      if (IS2 .le. 4) then
      A(1)=0.
      DA = AX/(M+1)
      do 1 I=1, M+1
    1 A(I+1) = DA * I
      write(6,'(5x,''Stiffeners equally spaced (Y/N)?  >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 4
      write(6,14) M
      call NREAL(M,ra)
      do 3 I=1, M
    3 A(I+1) = ra(I)
    4 A(M+2) = AX
      endif
      if (IS2 .eq. 5 .or. IS2 .le. 3) then
      B(1)=0.
      DB = BY/(N+1)
      do 2 I=1, N+1
    2 B(I+1) = DB * I
      write(6,'(5x,''Ribs equally spaced (Y/N)? >>(Y)>''$)')
```

```
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 6
      write(6,15) N
      call NREAL(N,ra)
      do 5 I=1, N
    5 B(I+1) = ra(I)
    6 B(N+2) = BY
      endif
c ...
  110 write(6,112)
      ia(1)=1
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
  111 continue
  112 format(3x,' Selection=> 1,n: modify MATL #n;',
     1 ' -1:End; 9:Quit;  m,0: other MENUS >>'$)
      return
      end
c ...
c
      SUBROUTINE fmod1(IS1,IS2)
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . modify geometry input
c ...
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
   11 format(5x,'Enter E1,E2,E3    > '$)
   12 format(5x,'Enter G12,G23,G31 > '$)
   13 format(5x,'Enter V12,V23,V31 > '$)
   14 format(5x,'Enter AML + Allowables (3 pairs)> '$)
   15 format(5x,'Enter AML - Allowables (3 pairs)> '$)
   16 format(5x,'Update Elastic Modulus (Y/N)? >>(N)>'$)
   17 format(5x,'Update Shear Modulus (Y/N)? >>(N)>'$)
   18 format(5x,'Update Poisson Ratio (Y/N)? >>(N)>'$)
   19 format(5x,'Update AML +Allowables(Y/N)? >>(N)>'$)
   20 format(5x,'Update AML -Allowables(Y/N)? >>(N)>'$)
   21 format(5x,'Modify Above Material Properties (Y/N)? >>(N)>'$)
   22 format(5x,'Add MATL ID',I2,' (Current number of MATL=',I2,')')
   23 format(5x,'Add New Material No. ',i2,' (Y/N)? >>(Y)>'$)
   24 format(5x,'Name of the Material > '$)
   25 format(5x,'Update Name of the Material (Y/N)? >>(N)>'$)
   26 format(5x,'Add Material (Y/N)? >>(N)>'$)
  112 format(3x,'Selection=> 1,n: modify MATL #n;',
     1/,5x,'-1: Exit; 9: Quit; m,0: other MENUS >>'$)
  113 format(4x,'     Properties for MATL No. :',I2)
  114 format(5x,'1>>Elastic Modulus  (E1,E2,E3):',3E9.2)
  115 format(5x,'2>>Shear Modulus (G12,G23,G31):',3E9.2)
  116 format(5x,'3>>Poisson Ratio (V12,V23,V31):',3F9.4)
  117 format(5x,'4>>AML + Allowables:',3(F5.1,2x,F7.4))
  118 format(5x,'5>>AML - Allowables:',3(F5.1,2x,F7.4))
  119 format(3x,'-----------------------------------------------')
```

```
c ...
 110  IMAT=NMAT
      call menu1(IMAT)
      write(6,112)
      ia(1)=1
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
      if (IS1 .ne. 1) go to 111
      if (IS2 .eq. 0) go to 100
c.....................................................
      call NEWPAGE
      write(6,113) IS2
      write(6,114) (E1(j,Is2,1),j=1,3)
      write(6,115) (G1(j,Is2,1),j=1,3)
      write(6,116) (V1(j,Is2,1),j=1,3)
      write(6,117) (SS(j,Is2,1),j=1,6)
      write(6,118) (SS(j,Is2,1),j=7,12)
      write(6,119)
c ...
      write(6,21)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,'(4x,''Select Item to be modified >>''$)')
      call NINTG(1,kk)
      go to (1,2,3,4,7)kk
  1   continue
      write(6,16)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      continue
      write(6,11)
      ra(1)=E1(1,IS2,1)
      ra(2)=E1(2,IS2,1)
      ra(3)=E1(3,IS2,1)
      call NREAL(3,ra)
      E1(1,IS2,1)=ra(1)
      E1(2,IS2,1)=ra(2)
      E1(3,IS2,1)=ra(3)
      go to 10
  2   continue
c ...
      write(6,17)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,12)
      ra(1)=G1(1,IS2,1)
      ra(2)=G1(2,IS2,1)
      ra(3)=G1(3,IS2,1)
      call NREAL(3,ra)
      G1(1,IS2,1)=ra(1)
      G1(2,IS2,1)=ra(2)
      G1(3,IS2,1)=ra(3)
      go to 10
  3   continue
c ...
```

```
      write(6,18)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,13)
      ra(1)=V1(1,IS2,1)
      ra(2)=V1(2,IS2,1)
      ra(3)=V1(3,IS2,1)
      call NREAL(3,ra)
      V1(1,IS2,1)=ra(1)
      V1(2,IS2,1)=ra(2)
      V1(3,IS2,1)=ra(3)
      go to 10
   4  continue
c ...
      write(6,19)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,14)
      do 5 I=1,6
   5  ra(I)=SS(I,IS2,1)
      call NREAL(6,ra)
      do 6 I=1,6
   6  SS(I,IS2,1)=ra(I)
      go to 10
   7  continue
c ...
      write(6,20)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,15)
      do 8 I=1,6
      II=I+6
   8  ra(I)=SS(II,IS2,1)
      call NREAL(6,ra)
      do 9 I=1,6
      II=I+6
   9  SS(II,IS2,1)=ra(I)
  10  go to 110
c.....................................................
 100  itmp=IMAT+1
      write(6,23) itmp
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 110
c.....
      call NEWPAGE
      NMAT=itmp
      write(6,11)
      ra(1)=E1(1,IMAT,1)
      ra(2)=E1(2,IMAT,1)
      ra(3)=E1(3,IMAT,1)
      call NREAL(3,ra)
      E1(1,itmp,1)=ra(1)
      E1(2,itmp,1)=ra(2)
      E1(3,itmp,1)=ra(3)
c ...
      write(6,12)
```

```
      ra(1)=G1(1,IMAT,1)
      ra(2)=G1(2,IMAT,1)
      ra(3)=G1(3,IMAT,1)
      call NREAL(3,ra)
      G1(1,itmp,1)=ra(1)
      G1(2,itmp,1)=ra(2)
      G1(3,itmp,1)=ra(3)
c ...
      write(6,13)
      ra(1)=V1(1,IMAT,1)
      ra(2)=V1(2,IMAT,1)
      ra(3)=V1(3,IMAT,1)
      call NREAL(3,ra)
      V1(1,itmp,1)=ra(1)
      V1(2,itmp,1)=ra(2)
      V1(3,itmp,1)=ra(3)
c ...
c ...   AML input ...
c ...
      ra(1) = -60.
      ra(2) = 0.001
      ra(3) = 0.
      ra(4) = 0.002
      ra(5) = 60.
      ra(6) = 0.001
      write(6,'(''Do you want to input AML Failure Parameters (Y/N)?''$)
     *')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y') then
         write(6,14)
         do 105 I=1,6
 105     ra(I)=SS(I,IMAT,1)
         call NREAL(6,ra)
      endif
      do 106 I=1,6
 106  SS(I,itmp,1)=ra(I)
c ...
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,15)
         do 108 I=1,6
         II=I+6
 108     ra(I)=SS(II,IMAT,1)
         call NREAL(6,ra)
      endif
      do 109 I=1,6
      II=I+6
 109  SS(II,itmp,1)=ra(I)
c ...
c ...
 111  continue
      return
      end
c ...
      SUBROUTINE fmod2(IS1,IS2)
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
```

```
c ...
c . modify basic panel plies
c ...
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
  11  format(5x,'Modify the no. of plies @ the panel(Y/N)? >>(N)>'$)
  12  format(5x,'Enter MATL#, THK & Angle @ PLY #',I2,'>'$)
  13  format(5x,'Enter the number of plies >>(',I2,')>'$)
  14  format(3x,' Selection => 2,n: modify panel plies;',
     1/,' -1:End; 9:Quit; m,0: other MENUS >>'$)
c ...
   1  if (IS1 .ne. 2) go to 9
      call menu2
      write(6,11)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      ia(1)=NLAY
      write(6,13) ia(1)
      call NINTG(1,ia)
      NLAY=ia(1)
      write(6,*)' '
      write(6,'(5x,''Enter MATL#, THK & Angle @ PLY #>'')')
      write(6,'(2x,''Are all information the same for ALL plies (Y/N)?''
     *$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y') then
         ia(1)=mw(1)
         ra(2)=tw(1)
         ra(3)=pw(1)
         ii=1
         write(6,12)ii
         call NREAL(3,ra)
         ia(1) = int(ra(1))
         mw(1)=ia(1)
         tw(1)=ra(2)
         pw(1)=ra(3)
         do 20 i=2,NLAY
         mw(i)=mw(1)
         tw(i)=tw(1)
         pw(i)=pw(1)
 20      continue
         go to 30
      else
      do 2 I=1,NLAY
      ia(1)=MW(I)
      ra(2)=TW(I)
      ra(3)=PW(I)
      write(6,12) I
      call NREAL(3,ra)
      ia(1) = int(ra(1))
      MW(I)=ia(1)
      TW(I)=ra(2)
   2  PW(I)=ra(3)
      go to 30
      endif
c ...
  10  write(6,'('' Choose PLY NUMBER to modify (Y/N)?''$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
```

```
         write(6,'('' Enter PLY NUMBER ='')')
         call NINTG(1,kp)
         write(6,12)kp
         call NREAL(3,ra)
         ia(1) = int(ra(1))
         MW(kp)=ia(1)
         TW(kp)=ra(2)
         PW(kp)=ra(3)
      endif
 30   write(6,14)
      ia(1)=2
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
      go to 1
c ...
   9  continue
      return
      end
c ...
c ...
      SUBROUTINE fmod3(IS1,IS2)
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
c ...
c . modify MENU3: STIF, ELEMENT Ply Layup
c ...
  11  format(3x,' Selection => 3,n: modify panel plies;',
     1/,16x,' -1:End; 9:Quit; m,0: other MENUS; -1: exit. >>'$)
  12  format(5x,'Change Dimensions of the Stiffeners (Y/N)? >>(N)>'$)
  13  format(5x,'Enter XH,XB,XC,XT >>(',4F6.3,')>'$)
  14  format(5x,'Enter XH,XB,XC >>(',3F6.3,')>'$)
  15  format(5x,'Enter XH,XB >>(',2F6.3,')>'$)
  16  format(5x,'Enter XH,XB,XT >>(',3F6.3,')>'$)
  17  format(5x,'Change the Stiffener Type (Y/N)? >>(N)>'$)
  18  format(5x,'Change Axial Stiffener from Type ',il,' to >> '$)
  19  format(5x,'1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,',
     1 /,5x,' 5)Angle, 6)Zee, 7)Bead, 8)Open Hat.')
  20  format(5x,'Enter MATL#, THK & Angle @ PLY #',I2,'>'$)
  22  format(5x,'Enter the number of plies >>(',I2,')>'$)
  23  format(5x,'Enter NUMBER OF PLIES for STIFF ELEM ',il,' >>'$)
  24  format(5x,'Modify the number of above plies (Y/N)? >>(N)>'$)
  25  format(4x,'3,',il,') No. of Plies for STIF,ELEM(',I1,'):',I3)
  26  format(4x,'     Ply #  Material #  Thickness  Angle')
  27  format(8x,I4,8x,I2,7x,f7.4,2x,f7.1)
c.....
c...  identify how many element for this type of stiffener ...
   1  if (MTYP .EQ. 1 .OR. MTYP .EQ. 5) then
         IP=2
      elseif (MTYP.eq.2) then
         IP=4
      else
         IP=3
```

```
      endif
c..
      call menu3(IP)
c..   choose items to modify or exit this menu ..
 10   write(6,11)
      ia(1)=3
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
      if (IS1 .ne. 3) go to 9
      if (IS2 .lt. 0 .or. IS2 .gt. IP) then
         write(6,'('' Select Again, Please.'')')
         go to 10
      endif
c.
c ..  change stiffener geometry ..
c.
      if (IS2.eq.0) then
c.
c ..     change stiffener type ? ..
c.
         write(6,17)
         ans='N'
         call STITLE(ans)
         if (ans .eq. 'Y' .or. ans .eq. 'y') go to 500
c.
c ..     change stiffener dimensions ? ..
c.
         write(6,12)
         ans='N'
         call STITLE(ans)
         if (ans .eq. 'N' .or. ans .eq. 'n') then
            go to 1
         else
            if (MTYP .EQ. 5 .or. MTYP .eq. 7) THEN
               write(6,15) XH,XB
               call NREAL(2,XH,XB)
            elseif (MTYP .eq. 6) then
               write(6,16) XH,XB,XT
               call NREAL(3,XH,XB,XT)
            elseif (MTYP .eq. 1) then
               write(6,14) XH,XB,XC
               call NREAL(3,XH,XB,XC)
            else
               write(6,13) XH,XB,XC,XT
               call NREAL(4,XH,XB,XC,XT)
            endif
            go to 1
         endif
c..
c...     change stiffener type ..
c..
500      call NEWPAGE
         ia(1)=MTYP
         write(6,19)
         write(6,18) ia(1)
         call NINTG(1,ia)
         MTYP=ia(1)
         ra(1)=XH
```

```
          ra(2)=XB
          ra(3)=XC
          ra(4)=XT
          if (MTYP.eq.1) then
             write(6,'('' Enter dimensions XH, XB, XC >>''$)')
             call NREAL(3,ra)
             XH=ra(1)
             XB=ra(2)
             XC=ra(3)
          elseif (MTYP.eq.5.or.MTYP.eq.7) then
             write(6,'('' Enter dimensions XH, XB >>''$)')
             call NREAL(2,ra)
             XH=ra(1)
             XB=ra(2)
          elseif (MTYP.eq.6) then
             write(6,'('' Enter dimensions XH, XB, XT >>''$)')
             call NREAL(3,ra)
             XH=ra(1)
             XB=ra(2)
             XT=ra(3)
          else
             write(6,'('' Enter dimensions XH, XB, XC, XT >>''$)')
             call NREAL(4,ra)
             XH=ra(1)
             XB=ra(2)
             XC=ra(3)
             XT=ra(4)
          endif
c ...
          if (MTYP .eq. 1 .or. MTYP .eq. 5) then
             ip=2
          elseif (mtyp .eq. 2) then
             ip=4
          else
             ip=3
          endif
c ...
          do 501 i=1,IP
             write(6,23)i
             call NINTG(1,ia)
             MLAY(i)=ia(1)
c ...
             write(6,'(2x,''Are all information the same for ALL plies (Y
     *N)?''$)')
             call STITLE(ans)
             if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
                j=1
                write(6,20) J
                call NREAL(3,ra)
                ia(1) = int(ra(1))
                ms(1,i)=ia(1)
                ts(1,i)=ra(2)
                ps(1,i)=ra(3)
                do 502 k=2,MLAY(i)
                ms(k,i)=ms(1,i)
                ts(k,i)=ts(1,i)
                ps(k,i)=ps(1,i)
 502            continue
             else
             do 503 j=1,MLAY(i)
```

```
                ia(1)=MS(j,i)
                ra(2)=TS(j,i)
                ra(3)=PS(j,i)
                write(6,20)J
                call NREAL(3,ra)
                ia(1) = int(ra(1))
                MS(j,i)=ia(1)
                TS(j,i)=ra(2)
                PS(j,i)=ra(3)
  503         continue
              endif
  501      continue
           go to 1
        endif
c...      modify STIFF ELEM only ...
        call NEWPAGE
        write(6,25) is2,is2,MLAY(is2)
        write(6,26)
        do 601 j=1,MLAY(is2)
           write(6,27) j,MS(j,is2),TS(j,is2),PS(j,is2)
  601   continue
        write(6,'(8x,''00  No Change'')')
        write(6,*)' '
        write(6,24)
        ans='N'
        call STITLE(ans)
        write(6,*)' '
        if (ans .eq. 'N' .or. ans .eq. 'n') then
           write(6,'(5x,''Choose the PLY NUMBER you wish to change >>''$)
      *')
           call NINTG(1,k)
           if ( k .eq. 0 ) go to 1
           write(6,20) k
           call NREAL(3,ra)
cccp
cccp           write(6,*)ra(1),ra(2),ra(3)
cccp
           ia(1) = int(ra(1))
           MS(k,is2)=ia(1)
           TS(k,is2)=ra(2)
           PS(k,is2)=ra(3)
cccp
cccp           write(6,*)ps(k,is2)
cccp
           go to 1
        else
           write(6,23)is2
           call NINTG(1,MLAY(is2))
           write(6,*)' '
           write(6,'(2x,''Are all information the same for ALL plies (Y/N)
      *?''$)')
           call STITLE(ans)
           if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
              j=1
              write(6,20)J
              call NREAL(3,ra)
cccp
cccp              write(6,*)ra(1),ra(2),ra(3)
cccp
              ia(1) = int(ra(1))
```

```
            MS(1,is2)=ia(1)
            ts(1,is2)=ra(2)
            ps(1,is2)=ra(3)
cccp
cccp            write(6,*)ps(1,is2)
cccp
            do 602 kg=2,MLAY(is2)
            ms(kg,is2)=ms(1,is2)
            ts(kg,is2)=ts(1,is2)
            ps(kg,is2)=ps(1,is2)
  602       continue
          else
          do 603 I=1,MLAY(is2)
             write(6,20) I
             call NREAL(3,ra)
cccp
cccp            write(6,*)ra(1),ra(2),ra(3)
cccp
             ia(1) = int(ra(1))
             MS(i,is2)=ia(1)
             TS(i,is2)=ra(2)
             PS(i,is2)=ra(3)
cccp
cccp            write(i,is2)
cccp
  603     continue
          endif
          go to 1
        endif
c.....................................................
    9   continue
        return
        end
c ...
c ...
        SUBROUTINE fmod4(IS1,IS2)
        common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
        common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
       1              SS(12,10,5),NT(10),NAME
        common /wall/ NLAY,MW(30),TW(30),PW(30)
        common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
        common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
        common /solve/ NBUC,TAR,NDF,ND,MO
        dimension ra(10),ia(10),NAME(10)
        character ans*1,NAME*80,type*15
c*****************************
c  FORMAT STATEMENTS FOR LISTING
c*****************************
  21  format(2x,'Should it represent a part of a much larger',
     1' panel?(Y/N)>>(N)>'$)
  22  format(2x,'Are there ribs at Ends ?(Y/N)>>(Y)>'$)
  23  format(2x,'Ribs on:(1)Skin (2)STIF (3)Both ?>>(1)>'$)
  24  format(2x,'Choose a B.C. Type no. at Ribs:'
     1/10x,'(1) Simply Supported; (2) S. S. with Rotation;',
     2     ' ?>>(1)>'$)
  25  format(2x,'Choose a B.C. type no. at Ends:',
     */10x,'(1) Part of Larger Panel',
     */10x,'(2) Free',
     1/10x,'(3) Simply Supported',
     3/10x,'(4) Semi-Clamped-1 (axial & lateral free)',
```

```
     4/10x,'(5) Semi-Clamped-2 (axial only free)',
     5/10x,'(6) Fully Clamped (all DOF fixed) --->>(1)>'$)
  26  format(2x,'Choose a B.C. type no. at Sides:',
     */10x,'(1) Part of a Larger Panel',
     c/10x,'(2) Free',
     1/10x,'(3) Simply Supported',
     3/10x,'(4) Semi-Clamped-1 (axial & lateral free)',
     4/10x,'(5) Semi-Clamped-2 (axial only free)',
     5/10x,'(6) Fully Clamped (all DOF fixed) --->>(2)>'$)
  30  format(2x,'----- Loading Condition Definition -----')
  31  format(2x,'Choose input loads type (Total=1,Line=2)',
     c '?>>(2)>'$)
  32  format(5x,'following total load unit : lb')
  33  format(5x,'following line load unit: lb/in')
  34  format(7x,'Input axial load >>(',F8.1,')>'$)
  35  format(7x,'Input lateral load >>(',F8.1,')>'$)
  36  format(7x,'Input shear load along ends >>(',F8.1,')>'$)
  37  format(7x,'Input lateral pressure in psi >>',F8.1,')>'$)
  38  format(7x,' 4,7)Axial = ',f8.1)
  39  format(7x,' 4,8)Lateral = ',f8.1)
  40  format(7x,' 4,9)Shear Along the Ends = ',f8.1)
  41  format(7x,' 4,10)Lateral Pressure (psi.)= ',f8.1)
  42  format(2x,' --- Solution Technique ---')
  43  format(5x,'TYPE OF TARGET >',a15,i2,1x,'NODE=',i3,1x,
     *'TARGET=',f10.5)
  44  format(5x,'TYPE OF TARGET >',a15,1x,'TARGET=',f10.5)
c*************
c*************
50    call NEWPAGE
10    write(6,'('' ----- Boundary Condition Definition -----'')')
      if (LEND.eq.1) then
         write(6,'('' 4,1)There ARE ribs on the ends.'')')
            else
         write(6,'('' 4,1)There are NO ribs on the ends.'')')
      endif
      if (LRIB.eq.1) then
         write(6,'('' 4,2)Ribs on the Skin.'')')
            elseif (LRIB.eq.2) then
         write(6,'('' 4,2)Ribs on the Stiffener.'')')
            elseif (LRIB.eq.3) then
         write(6,'('' 4,2)Ribs on both the Skin and the Stiffener.'')')
      endif
      if (LRBC .eq. 1) then
         write(6,'('' 4,3)Ribs are simulated using Simple Support Constr
     *aints.'')')
      elseif (LRBC .eq. 2) then
         write(6,'('' 4,3)Ribs are simulated using Simple Support plus R
     *otational Constraints.'')')
      endif
      if (LEBC.eq.1) then
         write(6,'('' 4,4)Ends are Part of a Larger Panel.'')')
      elseif (LEBC.eq.2) then
         write(6,'('' 4,4)Free at the Ends.'')')
      elseif (LEBC.eq.3) then
         write(6,'('' 4,4)Simply Supported at the Ends. '')')
      elseif (LEBC.eq.4) then
         write(6,'('' 4,4)Semi-Clamped (axial & lateral free) at the End
     *s.'')')
      elseif (LEBC.eq.5) then
         write(6,'('' 4,4)Semi-Clamped (axial only free) at the Ends.'')
```

```
     *')
      elseif (LEBC.eq.6) then
         write(6,'('' 4,4)Fully Clamped (all DOF restricted) at the Ends
     *.'')')
      endif
      if (LSBC.eq.1) then
         write(6,'('' 4,5)Sides are Part of a Larger Panel.'')')
      elseif (LSBC.eq.2) then
         write(6,'('' 4,5)Free at the Sides.'')')
      elseif (LSBC.eq.3) then
         write(6,'('' 4,5)Simply Supported at the Sides. '')')
      elseif (LSBC.eq.4) then
         write(6,'('' 4,5)Semi-Clamped (axial & lateral free) at the Sid
     *es.'')')
      elseif (LSBC.eq.5) then
         write(6,'('' 4,5)Semi-Clamped (axial only free) at the Sides.''
     *)')
      elseif (LSBC.eq.6) then
         write(6,'('' 4,5)Fully Clamped (all DOF restricted) at the Side
     *s.'')')
      endif
         write(6,'(''-----------------------------------------------------
     *----------------------''')')
      write(6,30)
      if (LTOT.eq.1) then
         write(6,'(7x,'' 4,6)Total Loads (lbs.):'')')
            else
         write(6,'(7x,'' 4,6)Line Loads (lbs.-in.):'')')
      endif
      write(6,38)FL(1)
      write(6,39)FL(2)
      write(6,40)FL(3)
      write(6,41)FL(4)
      write(6,42)
      if (NBUC.eq.1) then
         write(6,'(7x,'' 4,11) Linear Analysis.'')')
      elseif (NBUC.eq.2) then
         write(6,'(7x,'' 4,11) Bifurcation Buckling Analysis.'')')
         write(6,'(16x,i3,'' Mode(s).'')')MO
      elseif (NBUC.eq.3) then
         write(6,'(7x,'' 4,11) Post-Buckling Analysis.'')')
         if (NDF.gt.0) then
            TYPE='DIS. DOF '
            write(6,43) TYPE,NDF,ND,TAR
         else
            TYPE='LOAD FACTOR '
            write(6,44) TYPE,TAR
         endif
      endif
c*****************************
c  FORMAT STATEMENTS FOR MODIFICATION
c*****************************
      write(6,'(''-----------------------------------------''')')
      write(6,'('' Select m,n=> -1:Exit; 9:Quit; m,n: Other Menu or Upda
     *te.>>''$)')
      ia(1)=0
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
```

```
      if (IS1.ne.4) then
         go to 999
      else
         if (IS2.eq.1) then
            write(6,22)
            call STITLE(ans)
            if (ans.eq.'N'.or.ans.eq.'n') then
               LEND=2
            else
               LEND=1
            endif
         elseif (IS2.eq.2) then
            write(6,23)
            call NINTG(1,LRIB)
         elseif (IS2 .eq. 3) then
            write(6,24)
            call NINTG(1,LRBC)
         elseif (IS2.eq.4) then
            write(6,25)
            call NINTG(1,LEBC)
         elseif (IS2.eq.5) then
            write(6,26)
            call NINTG(1,LSBC)
c ...
c . modify loading conditions
c ...
         elseif (IS2.eq.6) then
            write(6,31)
            call NINTG(1,LTOT)
         elseif (IS2.eq.7) then
            write(6,34)FL(1)
            call NREAL(1,ra)
            FL(1)=ra(1)
         elseif (IS2.eq.8) then
            write(6,35)FL(2)
            call NREAL(1,ra)
            FL(2)=ra(1)
         elseif (IS2.eq.9) then
            write(6,36)FL(3)
            call NREAL(1,ra)
            FL(3)=ra(1)
         elseif (IS2.eq.10) then
            write(6,37)FL(4)
            call NREAL(1,ra)
            FL(4)=ra(1)
         elseif (IS2.eq.11) then
            write(6,'('' Choose ONE of the following technique >>'')')
            write(6,'(''    1> Linear Solution.'')')
            write(6,'(''    2> Bifurcation Buckling Analysis.'')')
            write(6,'(''    3> Post-Buckling Analysis.'')')
            write(6,'(''       Enter>>''$)')
            call NINTG(1,NBUC)
            if (NBUC.eq.3) then
               write(6,'('' Type of Target -- 0) Load Factor.'')')
               write(6,'(''                   1-6) Displace. DOF.'')')
               write(6,'(''              Enter >>''$)')
               call NINTG(1,NDF)
               if (NDF.gt.0) then
                  write(6,'('' Node Number =''$)')
                  call NINTG(1,ND)
```

```
               endif
               write(6,'(''  Target Point = ''$)')
               call NREAL(1,TAR)
             elseif (NBUC .eq. 2) then
               write(6,'(''  How many MODE shape(s) do you want (Max. 50
     *)? >>''$)')
               call NINTG(1,MO)
             endif
           endif
           go to 50
 999    endif
        return
        end
c ...
c ...
        SUBROUTINE fsave
        common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
        common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
       1              SS(12,10,5),NT(10),NAME
        common /wall/ NLAY,MW(30),TW(30),PW(30)
        common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
        common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
        common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . save and update files
c ...
        character NAME*80
        dimension NAME(10)
   11   format(4I5,2F10.4)
   12   format(6E12.4)
   13   format(I5,2F10.4)
   14   format(8F10.4)
   15   format(6I5)
   16   format(4i3,f10.5)
        write(10,11) M,N,MI,NI,AX,BY
        M2 = M+2
        N2 = N+2
        write(10,14) (A(I),I=1,M2)
        write(10,14) (B(I),I=1,N2)
c ...
        write(10,15) LEND,LRIB,LRBC,LEBC,LSBC,LTOT
        write(10,12) (FL(I),I=1,4)
        write(10,16) NBUC,NDF,ND,MO,TAR
c ...
c ...
c ...
        write(11,11) NMAT
        do 110 I=1,NMAT
c...        write(11,11) NT(I)
c...          do 110 J=1,NT(I)
          J=1
          write(11,12) (E1(K,I,J),K=1,3),(V1(K,I,J),K=1,3)
          write(11,12) (G1(K,I,J),K=1,3),(A1(K,I,J),K=1,3)
          write(11,12) (SS(K,I,J),K=1,6)
  110     write(11,12) (SS(K,I,J),K=7,12)
c ...
        write(11,11) NLAY
        do 120 I=1,NLAY
  120   write(11,13) MW(I),TW(I),PW(I)
        write(11,11) MTYP
```

```
      write(11,12) XH,XB,XC,XT
      if (MTYP.eq.1.or.MTYP.eq.5) then
        IP=2
      elseif (MTYP.eq.2) then
        IP=4
      else
        IP=3
      endif
      do 130 I=1,IP
      write(11,11) MLAY(I)
      do 130 J=1,MLAY(I)
 130  write(11,13) MS(J,I),TS(J,I),PS(J,I)
c ...
c .   DIAL L3D2 Runstreams
c ...
      call fmesh
      call fband
      call fmatl
      call fload
      call fsolve
C ...      call fclus
C ...      call fscop
      return
      end
c ...
      SUBROUTINE fband
c ...
c .   fband : BAND and SETUP runstreams
c ...
         open(unit=12,status='unknown',file='tape12B.com')
      write(12,'('' $band'')')
      write(12,'(''open 6 "band.out"'')')
      write(12,'(''start -1 : regps : band : stop '')')
      write(12,'(''$setup'')')
      write(12,'(''open 6 "setup.out"'')')
      write(12,'(''start -1 : setup : stop '')')
      close(unit=12)
      return
      end
      SUBROUTINE fmatl
c ...
c .   MATL runstream
c ...
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      CHARACTER*40 fmt1,fmt2,fmt3,fmt4,fmt5,fmt6,fmt7,fmt8,fmt4a
      character NAME*80
      dimension NAME(10)
      fmt1='(''MATORT'',I3,3e11.4,3f10.4,'' >'')'
      fmt2='(9X,3e11.4,3f10.4)'
      fmt3='(''ORIENT'',I3,'' 0.,0.,0., 1'')'
      fmt7='(''MATALL'',I3,6f11.4,'' >'')'
      fmt8='(9x,6f11.4)'
      fmt4='(''WALLAM'',2I3,'' >'')'
      fmt4a='(I3,'' >'')'
      fmt5='(12X,  F8.4,'' >'')'
```

```
      fmt6='(12X,  F8.4)'
c ...
c .   fmatl : DIAL MATL runstream
c ...
         open(unit=12,status='unknown',file='tape12C.com')
      write(12,'('' $matl'')')
      write(12,'(''open 6 "matl.out"'')')
      write(12,'(''start -1'')')
      do 25 I=1,NMAT
      write(12,fmt1) I,(E1(K,I,1),K=1,3),(V1(K,I,1),K=1,3)
      write(12,fmt2)   (G1(K,I,1),K=1,3),(A1(K,I,1),K=1,3)
      write(12,fmt3) I
      write(12,fmt7) I,(SS(K,I,1),K=1,6)
   25 write(12,fmt8)   (SS(K,I,1),K=7,12)
c ...
      MM=10
      write(12,fmt4) MM,NLAY
      do 26 i=1,NLAY
         write(12,fmt4a)MW(I)
   26 continue
      do 27 i=1,NLAY
         write(12,fmt5)TW(I)
   27 continue
      do 28 i=1,NLAY-1
         write(12,fmt5)PW(I)
   28 continue
      write(12,fmt6)PW(NLAY)
c ...
      if (MTYP.eq.1.or.MTYP.eq.5) then
         IP=2
      elseif (MTYP.eq.2) then
         IP=4
      else
         IP=3
      endif
      do 30 I=1,IP
         MM= 10 + I
         write(12,fmt4) MM,MLAY(I)
         do 31 j=1,MLAY(I)
            write(12,fmt4a)MS(J,I)
   31    continue
         do 32 j=1,MLAY(I)
            write(12,fmt5)TS(J,I)
   32    continue
         do 33 j=1,MLAY(I)-1
            write(12,fmt5)PS(J,I)
   33    continue
         write(12,fmt6)PS(MLAY(I),I)
   30 continue
c ...
      write(12,'(''matl : stop'')')
c ...
      close(unit=12)
      return
      end
      SUBROUTINE fload
c ...
c .   LOAD runstream
c ...
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
```

```
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c .   fload : DIAL LOAD runstream
c ...
      dimension NAME(10)
      character NAME*80
      real temp(6)
         open(unit=12,status='unknown',file='tape12D.com')
      write(12,'('' $load'')')
      write(12,'(''open 6 "load.out"'')')
      write(12,'('' start -1'')')
c ...
      write(12,'('' mset 11 copy nset=11'')')
      write(12,'('' mset 11 mask type msh'')')
      write(12,'('' mset 11 mask volu -999. 999. -999. 999. -0.001 0.001
     *'')')
      write(12,'('' mset 12 copy nset=12'')')
      write(12,'('' mset 12 mask type msh'')')
      write(12,'('' mset 12 mask volu -999. 999. -999. 999. -0.001 0.001
     *'')')
      write(12,'('' mset 13 copy nset=13'')')
C 9/10/91, inserted 1 line as follows, per Tony Wei
      write(12,'('' mset 13 mask type msh'')')
      write(12,'('' mset 14 copy nset=14'')')
C 9/10/91, inserted 1 line as follows, per Tony Wei
      write(12,'('' mset 14 mask type msh'')')
      temp(1)=FL(1)
      temp(2)=FL(2)
      temp(3)=FL(3)
      temp(4)=FL(4)
      if (LTOT .eq. 1) then
        temp(1)=temp(1)/AX
        temp(2)=temp(2)/BY
        temp(3)=temp(3)/AX
        tmp=temp(3)/BY
      endif
      if (LTOT .eq. 2) then
        tmp=temp(3)
      endif
      write(12,'('' lcase 1'')')
      write(12,'('' pline'',f10.2,'',11, 0,4 mset=11'')') -temp(1)
      write(12,'('' pline'',f10.2,'',11, 0,2 mset=12'')')  temp(1)
      write(12,'('' pline'',f10.2,'',12, 0,1 mset=13'')') -temp(2)
      write(12,'('' pline'',f10.2,'',12, 0,3 mset=14'')')  temp(2)
      write(12,'('' pline'',f10.2,'',12, 0,4 mset=11'')')  temp(3)
      write(12,'('' pline'',f10.2,'',12, 0,2 mset=12'')') -temp(3)
      write(12,'('' pline'',f10.2,'',11, 0,1 mset=13'')')  tmp
      write(12,'('' pline'',f10.2,'',11, 0,3 mset=14'')') -tmp
      write(12,'('' psurf'',f10.2,'',13, 0,mesh=1'')') -FL(4)
c ...
      write(12,'('' load : stop'')')
c ...
      close(unit=12)
      return
      end
```

```
      SUBROUTINE fsolve
c ...
c .   SOLVE runstream
c ...
      common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension NAME(10)
      character NAME*80
c ...
c .   fsolve : DIAL SOLVE runstream
c ...
100   format(' loads ',i2,f12.4)
         open(unit=12,status='unknown',file='tape12E.com')
      if (NBUC.eq.2) then
         write(12,'(''$solvedp'')')
         write(12,'('' open 6 "solve.out"'')')
         write(12,'('' start -1'')')
         write(12,'('' ASSIGN IMPR=1'')')
         write(12,100)1,1.
         write(12,'('' SAVE,D'')')
         write(12,'('' SAVE,S'')')
         write(12,'('' nprint,D'')')
         write(12,'('' eprint,S'')')
         write(12,'('' EIGEN 2'')')
         write(12,'('' solve'')')
         write(12,'('' stop'')')
         write(12,'(''$clusterdp'')')
         write(12,'('' start -1'')')
         write(12,'('' assign mxit=50'')')
         write(12,'('' shift '',i3)')MO
         write(12,'('' save'')')
         write(12,'('' matrix:file,ki ki'')')
         write(12,'('' clust 2'')')
         write(12,'('' stop'')')
      elseif(NBUC.eq.1) then
         write(12,'(''$solvedp'')')
         write(12,'('' open 6 "solve.out"'')')
         write(12,'('' start -1'')')
         write(12,'('' ASSIGN IMPR=1'')')
         write(12,100)1,1.
         write(12,'('' SAVE,D'')')
         write(12,'('' SAVE,S'')')
         write(12,'('' nprint,D'')')
         write(12,'('' eprint,S'')')
         write(12,'('' solve'')')
         write(12,'('' stop'')')
      elseif(NBUC.eq.3)then
         write(12,'('' $solvedp'')')
         write(12,'('' open 6 "solve.out"'')')
         write(12,'('' start -1'')')
         write(12,'('' ASSIGN IMPR=1'')')
         write(12,'('' static 2'')')
         write(12,'('' branch 0 1 1 '')')
         write(12,'('' loads 1 1. 1. 1.'')')
         write(12,'('' limit load'')')
```

```
         if (NF.ge.1) then
            write(12,'('' Target '',f12.5,'' Node= '',2i4)')TAR,NDF,ND
         else
            write(12,'('' Target'',f12.5)')TAR
         endif
         write(12,'('' save,d'')')
         write(12,'('' save,s'')')
         write(12,'('' assign slf=0.5, relr=0.02, stol=0.25, gtol=0.25''
     *)')
         write(12,'('' solve 1'')')
         write(12,'('' stop'')')
       endif
       close(unit=12)
       return
       end
       SUBROUTINE fmesh
c ...
c .   MESH runstream
c ...
       common /mesh/ M,N,MI,NI,AX,BY,A(10),B(10),NGRID
       common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
      1              SS(12,10,5),NT(10),NAME
       common /wall/ NLAY,MW(30),TW(30),PW(30)
       common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
       common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4)
       common /solve/ NBUC,TAR,NDF,ND,MO
       CHARACTER*60 fmt1,fmt2,fmt3,fmt4,fmt5,fmt6,fmt6b
       CHARACTER*60 fmt7,fmt8,fmt9,fmt0,fmta,fmtb,fmtc,fmtz,fmty,fmtz2
       character NAME*80
       dimension NAME(10)
c.....................................................................
c   VARIABLE DEFINITION TABLE
c.....................................................................
       fmt1='(''IJPOINT'',3I3,3f10.4)'
       fmt2='(''       '',3I3,3f10.4)'
       fmt3='(''SLINE'',6I3)'
       fmta='(''CIRCLE'',3I3)'
       fmt4='(''KNAME 0 0 '',2I3,''  kp'',I1)'
       fmt5='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-1,0,0,BASE'')'
       fmtb='(''MSET 10, COPY,NAME=BASE'')'
       fmt6='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-1,0,0'')'
       fmt6b='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-2,0,0'')'
       fmt7='(''PRISM'',I3,3f10.4)'
       fmt8='(''TRANS'',f10.4,'' 2'')'
       fmt9='(''KNAME '',4I3,''  kb'',I1)'
       fmt0='(''ELTNAME STF'',I1,'',STIF,,,,MESH=%IPP(9)'')'
       fmtc='(''MSET'',I3,'', COPY,ANAME=STF'',I1)'
       fmtz='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-1,0,0,part'')'
       fmtz2='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-2,0,0,part'')'
       fmty='(''BCSYS'',3I3,f10.4,'' 1.e+9,nset= '')'
c ...
c .   fmesh : DIAL MESH runstream
c ...
          open(unit=12,status='unknown',file='tape12A.com')
       write(12,'(''$mesh '')')
       write(12,'(''open 6 "mesh.out"'')')
       write(12,'(''clear -1'')')
       write(12,'(''eltype 4,2,6'')')
       write(12,'(''set syntax on'')')
       write(12,'(''assign IPNO=0 IPLC=0 IPEL=0 IPCO=0 '')')
```

```
      write(12,'(''# MESH Number 1, base panel'')')
c ...
      J0=1
      IP1=1
      JA1=1
      XA1=0.
      write(12,fmt1) IP1,JA1,J0,XA1,XA1
      IP=2+5*M
      JA=1+8*M+2*MI*(M+1)
      XA=AX
      write(12,fmt2) IP,JA,J0,XA1,XA
      do 10 L=1,M
      IP6=5*L+1
      IP5=5*L
      IP4=5*L-1
      IP3=5*L-2
      IP2=5*L-3
      JA6=1+2*(MI+4)*L
      JA5=1+2*(MI+4)*L-2
      JA4=1+2*(MI+4)*L-4
      JA3=1+2*(MI+4)*L-6
      JA2=1+2*(MI+4)*L-8
      if (MTYP.eq.2.or.MTYP.eq.8) then
c...   Global Y coor. for stiff. 2, or 8 .....
        XA6=A(1+L)+XB/2.
        XA5=A(1+L)+XC/2.
        XA4=A(1+L)
        XA3=A(1+L)-XC/2.
        XA2=A(1+L)-XB/2.
c...   Global Y coor. for stiff. 5, or 6 .....
      elseif (MTYP.eq.7) then
c...   Global Y coor. for stiff. 7 .....
        XA6=A(1+L)+XB/2.
        XA5=A(1+L)+XH
        XA4=A(1+L)
        XA3=A(1+L)-XH
        XA2=A(1+L)-XB/2.
      else
c...   Global Y coor. for stiff. 1,2, or 3, 4,6 .....
        XA6=A(1+L)+XB-XC
        XA5=A(1+L)+(XB-XC)/2.
        XA4=A(1+L)
        XA3=A(1+L)-XC/2.
        XA2=A(1+L)-XC
      endif
      write(12,fmt2) IP2,JA2,J0,XA1,XA2
      write(12,fmt2) IP3,JA3,J0,XA1,XA3
      write(12,fmt2) IP4,JA4,J0,XA1,XA4
      write(12,fmt2) IP5,JA5,J0,XA1,XA5
  10  write(12,fmt2) IP6,JA6,J0,XA1,XA6
c ...
      L1=1
      L2=2
      L3=3
      L4=4
      write(12,fmt3) L1,L2,L3,L4
      L1=M*5-1
      L2=M*5
      L3=M*5+1
      L4=M*5+2
```

```
      write(12,fmt3) L1,L2,L3,L4
      do 20 L=1,M-1
      L1=L*5-1
      L2=L*5
      L3=L*5+1
      L4=L*5+2
      L5=L*5+3
      L6=L*5+4
   20 write(12,fmt3) L1,L2,L3,L4,L5,L6
      L0=1
      L9=M*5+2
      T9=0
      do 25 L=1,NLAY
   25 T9=T9+TW(L)
      W9=0.
      M9=10
c ...
         i1=1
         i2=2
      do 26 i=1,M+1
         write(12,fmt5) i1,i2,T9,T9,W9,W9,M9
         i1=i1+5
         i2=i2+5
   26 continue
c ...
      do 30 L=1,N+1
      KB=L*2*NI+1
      XTT=0.
      L1=L+1
      YB=B(L1)
   30 write(12,fmt7) KB,YB
      do 33 L=1,N+2
      KB=L*2*NI-NI*2+1
   33 write(12,fmt4) KB,KB,L
      write(12,'('' msave'')')
      if (NBUC .eq. 2 .or. NBUC .eq. 3) then
         write(12,'('' refine 0 0 2 2 '')')
      endif
      write(12,'(''mesh'')')
c ...
c . Stiffener Type I.D.
c .  1: Blade   2: Hat
c .  3: I-shape 4: J-shape
c .  5: Angle   6: Zee
c .  7: Bead    8: Open Hat
c ...
   35 continue
      goto (110,120,130,140,150,160,170,180) mtyp
c ... Blade ...
  110 IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
```

```
      JA6=JA3
      JB1=1
      JB6=3
      XY =0.
      XA1=A(2)-XC
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=XA3
      YA6=0.
      ZA6=XH
      write(12,fmt1) IP1,JA1,JB1,XY,XA1
      write(12,fmt2) IP2,JA2,JB1,XY,XA2
      write(12,fmt2) IP3,JA3,JB1,XY,XA3
      write(12,fmt2) IP4,JA4,JB1,XY,XA4
      write(12,fmt2) IP5,JA5,JB1,XY,XA5
      write(12,fmt2) IP6,JA6,JB6,XY,XA6,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      M11=11
      M12=12
      YA7=0.
c 51   continue
      T11=0.
      T12=0.
      do 51 i=1,MLAY(1)
         T11=T11+ts(i,1)
  51  continue
      do 52 i=1,MLAY(2)
         T12=T12+ts(i,2)
  52  continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      go to 190
c ... Close Hat ...
 120  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      IP8=8
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA2
      JA7=JA3
      JA8=JA4
      JB1=1
      JB6=3
      XA1=A(2)-XB/2.
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+XC/2.
      XA5=A(2)+XB/2.
      XA6=A(2)-XT/2.
```

```
      XA7=A(2)
      XA8=A(2)+XT/2.
      YA6=0.
      ZA6=XH
      write(12,fmt1) IP1,JA1,JB1,YA6,XA1
      write(12,fmt2) IP2,JA2,JB1,YA6,XA2
      write(12,fmt2) IP3,JA3,JB1,YA6,XA3
      write(12,fmt2) IP4,JA4,JB1,YA6,XA4
      write(12,fmt2) IP5,JA5,JB1,YA6,XA5
      write(12,fmt2) IP6,JA6,JB6,YA6,XA6,ZA6
      write(12,fmt2) IP7,JA7,JB6,YA6,XA7,ZA6
      write(12,fmt2) IP8,JA8,JB6,YA6,XA8,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP6,IP7,IP8
      write(12,fmt3) IP2,IP6
      write(12,fmt3) IP4,IP8
      M11=11
      M12=12
      M13=13
      M14=14
      YA7=0.
      YA8=0.
      T11=0.
      T14=0.
      T12=0.
      T13=0.
      do 53 i = 1,MLAY(1)
         t11=t11+ts(i,1)
53    continue
      do 54 i = 1,MLAY(2)
         t12=t12+ts(i,2)
54    continue
      do 55 i = 1,MLAY(3)
         t13=t13+ts(i,3)
55    continue
      do 56 i = 1,MLAY(4)
         t14=t14+ts(i,2)
56    continue
      write(12,fmt6) IP1,IP2,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP2,IP4,T14,T14,YA8,YA8,M14
      write(12,fmt6) IP4,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6b) IP2,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP8,T13,T13,YA6,YA6,M13
      write(12,fmtz) IP8,IP4,T12,T12,YA6,YA6,M12
      go to 190
c ... I-shape ...
 130  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      IP8=8
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA3
```

```
      JA7=JA2
      JA8=JA4
      JB1=1
      JB6=3
      XA1=A(2)-XC
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=A(2)
      XA7=A(2)-XT/2.
      XA8=A(2)+XT/2.
      ZA6=XH
      YA6=0.
      M11=11
      M12=12
      M13=13
      write(12,fmt1) IP1,JA1,JB1,YA6,XA1
      write(12,fmt2) IP2,JA2,JB1,YA6,XA2
      write(12,fmt2) IP3,JA3,JB1,YA6,XA3
      write(12,fmt2) IP4,JA4,JB1,YA6,XA4
      write(12,fmt2) IP5,JA5,JB1,YA6,XA5
      write(12,fmt2) IP6,JA6,JB6,YA6,XA6,ZA6
      write(12,fmt2) IP7,JA7,JB6,YA6,XA7,ZA6
      write(12,fmt2) IP8,JA8,JB6,YA6,XA8,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      write(12,fmt3) IP7,IP6
      write(12,fmt3) IP6,IP8
      YA7=0.
c      do 55 i=1,MLAY(1)
      T11=0.
      T12=0.
      T13=0.
      do 57 i=1,MLAY(1)
         t11=t11+ts(i,1)
57    continue
      do 58 i=1,MLAY(2)
         t12=t12+ts(i,2)
58    continue
      do 59 i=1,MLAY(3)
         t13=t13+ts(i,3)
59    continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP7,IP6,T13,T13,YA6,YA6,M13
      write(12,fmt6) IP6,IP8,T13,T13,YA6,YA6,M13
      go to 190
c ... J-shape ...
 140  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      JA1=1
      JA2=3
      JA3=5
      JA4=7
```

```
      JA5=9
      JA6=JA3
      JA7=JA3
      JB1=1
      JB6=3
      JB7=5
      XA1=A(2)-XC
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=A(2)
      XA7=A(2)-XT
      ZA6=XH
      YA6=0.
      M11=11
      M12=12
      M13=13
      write(12,fmt1) IP1,JA1,JB1,YA6,XA1
      write(12,fmt2) IP2,JA2,JB1,YA6,XA2
      write(12,fmt2) IP3,JA3,JB1,YA6,XA3
      write(12,fmt2) IP4,JA4,JB1,YA6,XA4
      write(12,fmt2) IP5,JA5,JB1,YA6,XA5
      write(12,fmt2) IP6,JA6,JB6,YA6,XA6,ZA6
      write(12,fmt2) IP7,JA7,JB7,YA6,XA7,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      write(12,fmt3) IP6,IP7
      YA7=0.
      T11=0.
      T12=0.
      T13=0.
      do 60 i=1,MLAY(1)
         t11=t11+ts(i,1)
60    continue
      do 61 i=1,MLAY(2)
         t12=t12+ts(i,2)
61    continue
      do 62 i=1,MLAY(3)
         t13=t13+ts(i,3)
62    continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP7,T13,T13,YA6,YA6,M13
      go to 190
c ... Angle ...
 150  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA3
      JB1=1
      JB6=3
```

```
      XY =0.
      XA1=A(2)-XC
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=XA3
      YA6=0.
      ZA6=XH
      write(12,fmt1) IP1,JA1,JB1,XY,XA1
      write(12,fmt2) IP2,JA2,JB1,XY,XA2
      write(12,fmt2) IP3,JA3,JB1,XY,XA3
      write(12,fmt2) IP4,JA4,JB1,XY,XA4
      write(12,fmt2) IP5,JA5,JB1,XY,XA5
      write(12,fmt2) IP6,JA6,JB6,XY,XA6,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      M11=11
      M12=12
      YA7=0.
      T11=0.
      T12=0.
      do 63 i= 1,MLAY(1)
         t11=t11+ts(i,1)
63    continue
      do 64 i= 1,MLAY(2)
         t12=t12+ts(i,2)
64    continue
      write(12,fmt6) IP1,IP3,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP3,IP5,T9,T9,YA7,YA7,M9
      go to 190
c ... Zee-shape ...
 160  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA3
      JA7=JA3
      JB1=1
      JB6=3
      JB7=5
      XA1=A(2)-XB/2.
      XA2=A(2)-XB/4.
      XA3=A(2)
      XA4=A(2)+XB/4.
      XA5=A(2)+XB/2.
      XA6=A(2)
      XA7=A(2)-XT
      ZA6=XH
      YA6=0.
      M11=11
```

```
      M12=12
      M13=13
      write(12,fmt1) IP1,JA1,JB1,YA6,XA1
      write(12,fmt2) IP2,JA2,JB1,YA6,XA2
      write(12,fmt2) IP3,JA3,JB1,YA6,XA3
      write(12,fmt2) IP4,JA4,JB1,YA6,XA4
      write(12,fmt2) IP5,JA5,JB1,YA6,XA5
      write(12,fmt2) IP6,JA6,JB6,YA6,XA6,ZA6
      write(12,fmt2) IP7,JA7,JB7,YA6,XA7,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      write(12,fmt3) IP6,IP7
      YA7=0.
      T11=0.
      T12=0.
      T13=0.
      do 65 i=1,MLAY(1)
         t11=t11+ts(i,1)
65    continue
      do 66 i=1,MLAY(2)
         t12=t12+ts(i,2)
66    continue
      do 67 i=1,MLAY(3)
         t13=t13+ts(i,3)
67    continue
      write(12,fmt6) IP1,IP3,T9,T9,YA7,YA7,M9
      write(12,fmt6) IP3,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP7,T13,T13,YA6,YA6,M13
      go to 190
c ... Bead ...
 170  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      IP8=8
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA2
      JA7=JA3
      JA8=JA4
      JB1=1
      JB6=3
      XA1=A(2)-XB/2.
      XA2=A(2)-XH
      XA3=A(2)
      XA4=A(2)+XH
      XA5=A(2)+XB/2.
      XA6=A(2)-0.707*XH
      XA7=A(2)
      XA8=A(2)+0.707*XH
      YA6=0.
      ZA6=0.707*XH
      ZA7=XH
```

```
      ZA8=0.707*XH
      write(12,fmt1) IP1,JA1,JB1,YA6,XA1
      write(12,fmt2) IP2,JA2,JB1,YA6,XA2
      write(12,fmt2) IP3,JA3,JB1,YA6,XA3
      write(12,fmt2) IP4,JA4,JB1,YA6,XA4
      write(12,fmt2) IP5,JA5,JB1,YA6,XA5
      write(12,fmt2) IP6,JA6,JB6,YA6,XA6,ZA6
      write(12,fmt2) IP7,JA7,JB6,YA6,XA7,ZA7
      write(12,fmt2) IP8,JA8,JB6,YA6,XA8,ZA8
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmta) IP2,IP6,IP3
      write(12,fmta) IP6,IP7,IP3
      write(12,fmta) IP7,IP8,IP3
      write(12,fmta) IP8,IP4,IP3
      M11=11
      M12=12
      M13=13
      YA7=0.
      YA8=0.
      T11=0.
      T12=0.
      T13=0.
      do 68 i=1,MLAY(1)
         t11=t11+ts(i,1)
68    continue
      do 69 i=1,MLAY(2)
         t12=t12+ts(i,2)
69    continue
      do 70 i=1,MLAY(3)
         t13=t13+ts(i,3)
70    continue
      write(12,fmt6) IP1,IP2,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP2,IP4,T13,T13,YA8,YA8,M13
      write(12,fmt6) IP4,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6b) IP2,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP8,T12,T12,YA6,YA6,M12
      write(12,fmtz) IP8,IP4,T12,T12,YA6,YA6,M12
      go to 190
c..... Open Hat ....
 180  IP1 = 1
      IP2 = 2
      IP3 = 3
      IP4 = 4
      IP5 = 5
      IP6 = 6
      IP7 = 7
      IP8 = 8
      IP9 = 9
      JA1 = 1
      JA2 = 3
      JA3 = 5
      JA4 = 7
      JA5 = 9
      JA6 = JA2
      JA7 = JA1
      JA8 = JA4
      JA9 = JA5
      JB1 = 1
      JB6 = 3
      XA1 = A(2) - XB/2.
```

```
      XA2 = A(2) - XC/2.
      XA3 = A(2)
      XA4 = A(2) + XC/2.
      XA5 = A(2) + XB/2.
      XA6 = A(2) - XC/2.
      XA7 = A(2) - XT/2.
      XA8 = A(2) + XC/2.
      XA9 = A(2) + XT/2.
      YA6 = 0.
      ZA6 = XH
      write(12,fmt1) IP1,JA1,JB1,YA6,XA1
      write(12,fmt2) IP2,JA2,JB1,YA6,XA2
      write(12,fmt2) IP3,JA3,JB1,YA6,XA3
      write(12,fmt2) IP4,JA4,JB1,YA6,XA4
      write(12,fmt2) IP5,JA5,JB1,YA6,XA5
      write(12,fmt2) IP6,JA6,JB6,YA6,XA6,ZA6
      write(12,fmt2) IP7,JA7,JB6,YA6,XA7,ZA6
      write(12,fmt2) IP8,JA8,JB6,YA6,XA8,ZA6
      write(12,fmt2) IP9,JA9,JB6,YA6,XA9,ZA6
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP2,IP6
      write(12,fmt3) IP6,IP7
      write(12,fmt3) IP4,IP8
      write(12,fmt3) IP8,IP9
      M11 = 11
      M12 = 12
      M13 = 13
      YA7=0.
      T11=0.
      T12=0.
      T13=0.
      do 71 i=1,MLAY(1)
         t11=t11+ts(i,1)
71    continue
      do 72 i=1,MLAY(2)
         t12=t12+ts(i,2)
72    continue
      do 73 i=1,MLAY(3)
         t13=t13+ts(i,3)
73    continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6b) IP2,IP6,T12,T12,YA6,YA6,M12
      write(12,fmtz) IP8,IP4,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP7,IP6,T13,T13,YA6,YA6,M13
      write(12,fmt6) IP8,IP9,T13,T13,YA6,YA6,M13
      go to 190
 190  continue
      YA7=0.
      YA8=0.
c ...
c . ditto 1st stiffener
c ...
      do 192 L=1,N+1
        KB=L*2*NI+1
        XTT=0.
        L1=L+1
      YB=B(L1)
 192  write(12,fmt7) KB,YB
      do 193 L=1,N+2
         KB=L*2*NI-2*NI+1
```

```
          if (MTYP .eq. 1 .or. MTYP .eq. 5) then
             NP1=6
             NP2=6
             write(12,fmt9) NP1,NP2,KB,KB,L
             NP1=1
             NP2=5
             write(12,fmt9) NP1,NP2,KB,KB,L
          elseif (MTYP .eq. 2) then
             NP1=6
             NP2=8
             write(12,fmt9) NP1,NP2,KB,KB,L
             NP1=1
             NP2=2
             write(12,fmt9) NP1,NP2,KB,KB,L
             NP1=4
             NP2=5
             write(12,fmt9) NP1,NP2,KB,KB,L
          elseif (MTYP .eq. 3) then
             NP1=6
             NP2=7
             NP3=8
             write(12,fmt9) NP1,NP2,KB,KB,L
             write(12,fmt9) NP1,NP3,KB,KB,L
             NP1=1
             NP2=5
             write(12,fmt9) NP1,NP2,KB,KB,L
          elseif (MTYP .eq. 4 .or. 6) then
             NP1=6
             NP2=7
             write(12,fmt9) NP1,NP2,KB,KB,L
             NP1=1
             NP2=5
             write(12,fmt9) NP1,NP2,KB,KB,L
          elseif (MTYP .eq. 7) then
             NP1=7
             NP2=7
             write(12,fmt9) NP1,NP2,KB,KB,L
             NP1=1
             NP2=2
             write(12,fmt9) NP1,NP2,KB,KB,L
             NP1=4
             NP2=5
             write(12,fmt9) NP1,NP2,KB,KB,L
          elseif (MTYP .eq. 8) then
             NP1=6
             NP2=7
             NP3=8
             NP4=9
             write(12,fmt9) NP1,NP2,KB,KB,L
             write(12,fmt9) NP3,NP4,KB,KB,L
             NP1=1
             NP2=5
             write(12,fmt9) NP1,NP2,KB,KB,L
          endif
c 193     write(12,fmt9) KB,KB,L
 193      continue
          write(12,'('' msave'')')
          if (NBUC .eq. 2 .or. NBUC .eq. 3) then
             write(12,'('' refine 0 0 2 2'')')
          endif
```

```
      write(12,'(''mesh'')')
      write(12,'('' eltsen 0 1 0 '')')
      if (MTYP.eq.2.or.MTYP.eq.7.or.MTYP.eq.8) then
         write(12,'('' eltsen 0 1 0 aname=part'')')
      endif
      L=1
      write(12,fmt0) L
      write(12,fmtc) L,L
      write(12,'(''merge'')')
      if (M .le. 1) go to 199
      do 195 L=1,M
      if (L .le. 1) go to 195
      write(12,'(''ditto  MESH=2'')')
      da=A(L+1)-A(2)
      write(12,fmt8) da
      write(12,fmt0) L
      write(12,fmtc) L,L
      write(12,'(''merge'')')
 195  continue
 199  continue
c ...
c  CONO SUBROUTINE
c ...
      write(12,'(''mset 10 insert type msh'')')
      write(12,'(''mset 10 del name=base'')')
      write(12,'(''set syntax on'')')
      write(12,'(''####################'')')
      write(12,'(''# CONODE subroutine'')')
      write(12,'(''####################'')')
      write(12,'('' sub CONO &lst1 &lst2 &lst3'')')
      write(12,'('' set echo on'')')
      write(12,'('' let &nnod=%ipp(1)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst2)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n1=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst3)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n2=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst1)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' ;f2 format ''''(1x,i5)'')')
      write(12,'('' ;f3 format ''''(''''Print Nodal Coordinates'''',1x,'
     *''''Node'''')'')')
      write(12,'('' write 6 ;f2 &n1'')')
      write(12,'('' write 6 ;f2 &n2'')')
      write(12,'('' write 6 ;f3'')')
      write(12,'('' do ;20 &n=1,&nnod'')')
      write(12,'('' if %ibcl(&ln,&n) ;20,;20,1'')')
      write(12,'('' let &nn=%ibcl(&ln,&n)'')')
      write(12,'('' write 6 ;f2 &nn'')')
      write(12,'('' conode &nn &n1 &n2 1 1.e+9'')')
      write(12,'('' ;20 continue'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' return'')')
      write(12,'('' end'')')
c ...
c . boundary conditions definition
c ...
```

```
        x1= -0.001
        x2=  0.001
        y1=  -999.
        y2=   999.
        z1=  -999.
        z2=   999.
      write(12,'(''nset l1 copy volu '',6f10.3)') x1,x2,y1,y2,z1,z2
        x1= -0.001+BY
        x2=  0.001+BY
        y1=  -999.
        y2=   999.
        z1=  -999.
        z2=   999.
      write(12,'(''nset l2 copy volu '',6f10.3)') x1,x2,y1,y2,z1,z2
        y1= -0.001
        y2=  0.001
        x1=  -999.
        x2=   999.
        z1=  -999.
        z2=   999.
      write(12,'(''nset l3 copy volu '',6f10.3)') x1,x2,y1,y2,z1,z2
        y1= -0.001+AX
        y2=  0.001+AX
        x1=  -999.
        x2=   999.
        z1=  -999.
        z2=   999.
      write(12,'(''nset l4 copy volu '',6f10.3)') x1,x2,y1,y2,z1,z2
c...
c...   Boundary conditions on Ribs (Frames).
       NB1=1
       NB2=N+2
       if (LEND .ne. 1) NB1=2
       if (LEND .ne. 1) NB2=N+1
       do 210 IB=NB1,NB2
      if (LRIB .eq. 3) then
        if (LRBC .eq. 1) then
           write(12,'(''dofsup/dnf  3,aname=kp'',I1)') IB
           write(12,'(''dofsup/dnf  3,aname=kb'',I1)') IB
        elseif (LRBC .eq. 2) then
           write(12,'(''dofsup/dnf 53,aname=kp'',I1)') IB
           write(12,'(''dofsup/dnf 53,aname=kb'',I1)') IB
        endif
      elseif (LRIB .eq. 1) then
        if (LRBC .eq. 1) then
           write(12,'(''dofsup/dnf 3,aname=kp'',I1)') IB
        elseif (LRBC .eq. 2) then
           write(12,'(''dofsup/dnf  53,aname=kp'',I1)') IB
        endif
      elseif (LRIB .eq. 2) then
        if (LRBC .eq. 1) then
           write(12,'(''dofsup/dnf 3,aname=kb'',I1)') IB
        elseif (LRBC .eq.2) then
           write(12,'(''dofsup/dnf  53,aname=kb'',I1)') IB
        endif
      endif
  210 continue
c...  left front corner
      x1=-0.001+BY
      x2=0.001+BY
```

```
      yl=-0.001
      y2=0.001
      zl=-999.999
      z2=999.999
      write(12,'('' nlist 11 insert volu '',6f9.3)')x1,x2,yl,y2,zl,z2
c...  right front corner
      xl=-0.001+BY
      x2=0.001+BY
      yl=-0.001+AX
      y2=0.001+AX
      zl=-999.999
      z2=999.999
      write(12,'('' nlist 12 insert volu '',6f9.3)')x1,x2,yl,y2,zl,z2
c...  left rear corner
      xl=-0.001
      x2=0.001
      yl=-0.001
      y2=0.001
      zl=-999.999
      z2=999.999
      write(12,'('' nlist 13 insert volu '',6f9.3)')x1,x2,yl,y2,zl,z2
c...  right rear corner
      xl=-0.001
      x2=0.001
      yl=-0.001+AX
      y2=0.001+AX
      zl=-999.999
      z2=999.999
      write(12,'('' nlist 14 insert volu '',6f9.3)')x1,x2,yl,y2,zl,z2
      write(12,'('' nset 11 del nlist 11 12 13 14'')')
      write(12,'('' nset 12 del nlist 11 12 13 14'')')
      write(12,'('' nset 13 del nlist 11 12 13 14'')')
      write(12,'('' nset 14 del nlist 11 12 13 14'')')
      write(12,'('' nlist 21 insert nset 11 '')')
      write(12,'('' nlist 22 insert nset 12 '')')
      write(12,'('' nlist 23 insert nset 13 '')')
      write(12,'('' nlist 24 insert nset 14 '')')
      NSET1=11
      NSET2=12
      NSET3=13
      NSET4=14
c...
      write(12,'('' call cono 22 11 12 '')')
      write(12,'('' call cono 21 13 14 '')')
      write(12,'('' nset 11 insert nlist 13 14'')')
      write(12,'('' nset 12 insert nlist 11 12'')')
      write(12,'('' nset 13 insert nlist 11 13'')')
      write(12,'('' nset 14 insert nlist 12 14'')')
c...
c... BOUNDARY CONDITIONS SETUP
c...
c... LARG = 1 yes. LARG not = 1 , NO
c    one - part of the larger panel
c.........................
c     part of a larger panel
c.........................
      if (LSBC.eq.1) then
         write(12,'('' dofsup/dnf 4 nset 13'')')
         write(12,'('' dofsup/dnf 4 nset 14'')')
      elseif (LSBC.eq.3) then
```

```
c.........................
c     sides simply supported
c.........................
         write(12,'('' dofsup/dnf 34 nset 13'')')
         write(12,'('' dofsup/dnf 34 nset 14'')')
      elseif (LSBC.eq.4) then
c.........................
c     sides clamped (axial and lateral free)
c.........................
         write(12,'('' dofsup/dnf 2456 nset 13'')')
         write(12,'('' dofsup/dnf 2456 nset 14'')')
      elseif (LSBC.eq.5) then
c.........................
c     sides clamped (axial only free)
c.........................
         write(12,'('' dofsup/dnf 23456 nset 13'')')
         write(12,'('' dofsup/dnf 23456 nset 14'')')
      elseif (LSBC.eq.6) then
c.........................
c     sides fully clamped
c.........................
         write(12,'('' dofsup 0 nset 13'')')
         write(12,'('' dofsup 0 nset 14'')')
      endif
      if (LEBC.eq.1) then
c.........................
c     part of a larger panel
c.........................
         write(12,'('' dofsup/dnf 5 nset 11'')')
         write(12,'('' dofsup/dnf 5 nset 12'')')
      elseif (LEBC.eq.3) then
c.........................
c     ends simply supported
c.........................
         write(12,'('' dofsup/dnf 35 nset 11'')')
         write(12,'('' dofsup/dnf 35 nset 12'')')
      elseif (LEBC.eq.4) then
c.........................
c     ends clamped (axial and lateral free)
c.........................
         write(12,'('' dofsup/dnf 2456 nset 11'')')
         write(12,'('' dofsup/dnf 2456 nset 12'')')
      elseif (LEBC.eq.5) then
c.........................
c     ends clamped (axial only free)
c.........................
         write(12,'('' dofsup/dnf 32456 nset 11'')')
         write(12,'('' dofsup/dnf 32456 nset 11'')')
      elseif (LEBC.eq.6) then
c.........................
c     ends fully clamped
c.........................
         write(12,'('' dofsup 0 nset 11'')')
         write(12,'('' dofsup 0 nset 12'')')
      endif
c...
c     Corner node constraints to stablize the panel.
c...
999   write(12,'('' dofsup/dnf 123 nlist 11'')')
      write(12,'('' dofsup/dnf 13 nlist 12'')')
```

```
      write(12,'('' dofsup 3 nlist 13'')')
      write(12,'(''finish : stop'')')
c...
      close(unit=12)
      return
      end
c###################
c NTOKEN -- utility I/O subroutine
c  written by Dan Rickhoff LMSC 81-12;
c  with modification by L. L. Chou LMSC 81-12.
c###################
      subroutine nreal(ntk,ra)
      parameter ( maxtkn = 25 )
      character a*127, tkns(maxtkn)*10
      dimension lentk(maxtkn),ra(maxtkn)
c
  100   ierr=0
      call getlin( 5, a, irdata, *910 )
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
c       - - - - -
        do 200 i=1,ntk
        r = ra(i)
        call rintrp( tkns(i), lentk(i), r, *991 )
        ra(i) = r
        go to 200
  991     ierr=ierr+1
          write(6,'( 19x, ''Not a real '')')
  200 continue
      if (ierr .ge. 1) go to 100
      return
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
c
c
      subroutine nintg(ntk,ia)
      parameter ( maxtkn = 25 )
      character a*127, tkns(maxtkn)*10
      dimension lentk(maxtkn),ia(maxtkn)
c     ---------
  100 ierr=0
      call getlin( 5, a, irdata, *910 )
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
c
      do 200 i=1,ntk
        intgr = ia(i)
        call iintrp( tkns(i), lentk(i), intgr, *992 )
        ia(i) = intgr
        go to 200
  992     ierr=ierr+1
          write(6,'( 19x, ''Not an integer '')')
  200 continue
      if ( ierr .ge. 1) go to 100
      return
c     ---------
  999 continue
      write(6,'('' ERROR return from getkns'')')
```

```
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
c
c
      subroutine stitle(sline)
c
      character a*127, tkns(1)*60, sline*(*)
      dimension lentk(1)
c     ---------
      call getlin( 5, a, irdata, *910 )
      ntk = 1
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
      sline=tkns(1)(1:lentk(1))
      return
c
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
c
c ***************************************************************
c
      subroutine rintrp( tkn, lentk, r, * )
c
c#  Read a real number, R, from a character string, TKN, of length LENTK.
c#  (Note that blanks in the strings are ignored. Leading and/or trailing
c#   blanks are ignored, which is OK, but interior blanks are squeezed out,
c#   which is not OK; try to fix this sometime.)
c
      character tkn*(*)
c
      if ( lentk .eq. 0 ) return
c
      read( tkn(1:lentk), '(bn,f80.0)', err=990 ) r
c
      return
c
c -----------
  990 continue
      return 1
c
      end
c
c ***************************************************************
c
      subroutine iintrp( tkn, lentk, i, * )
c
c#  Read a integer number, I, from a character string, TKN, of length LENTK.
c#  (Note that blanks in the strings are ignored. Leading and/or trailing
c#   blanks are ignored, which is OK, but interior blanks are squeezed out,
c#   which is not OK; try to fix this sometime.)
c
      character tkn*(*)
c
      if ( lentk .eq. 0 ) return
c
```

```
      read( tkn(1:lentk), '(bn,i80)', err=990 ) i
c
      return
c
c -----------
  990 continue
      return 1
c
      end
c
c ****************************************************************
c
      subroutine getkns( a, irdata, tkns, lentk, ntk, * )
c
c ---------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     a ....... The given character string containing the input line.
c#                     If this string is blank (input IRDATA = 0) then all
c#                     NTK returned tokens will be blank with zero length.
c
c#    in     irdata .. The input line in string A extends to this character
c#                     position.
c
c#    out    tkns .... The character array of tokens.  The character lengths
c#                     of these array elements limit the lengths of input
c#                     tokens; tokens are correctly parsed but then truncated
c#                     to this length.
c
c#    out    lentk ... An integer array with the string lengths of the tokens
c#                     (after any necessary truncation).
c
c#    in     ntk ..... The number of tokens to be parsed from the line.
c
c#    -      * ....... Alternate RETURN in the event that a string token
c#                     initiated by an apostrophy had no trailing apostrophy, or
c#                     the next character after the trailing apostrophy was not
c#                     a delimiter (or end-of-line).
c#                     NOTE: The arguments that had been interpreted up to the
c#                           point that the error occured are returned; the
c#                           will be blank.
c
c ---------------------------------------------------------------
c#  PURPOSE:
c
c#  Parses an "input line" contained in a given string to find the first NTK
c#  tokens on the line.
c
c#  Tokens are delimited by beginning-of-line, commas, spaces, tabs, or
c#  end-of-line.  Tabs are equivalent to (single) spaces.
c
c#  A single asterisk (*) input token is recognized as a "null" token, for
c#  which the retuned string is blank and returned length is zero.  Some users
c#  may find entry of the null token preferable to that of a comma with no
c#  preceeding token, or two adjacent apostrophies (see the next
c#  paragraph).
c
c#  If a token is enclosed within apostrophies then it may contain commas,
c#  spaces, or tabs.  The returned length of such a token is the number
c#  number of character positions between the pair of apostrophies.
```

```
c#  Note that, for this length calculation, tabs are a single character.
c#  Contigous alphanumeric strings (no embedded delimiters) do not need to be
c#  enclosed within apostrophies.
c
c#  There is no provision for including apostrophies (') or the comment
c#  character (the hash symbol, #; see routine GETLIN) within a token,
c#  i.e., they may not be "escaped".  Strings enclosed within apostrophies
c#  may contain asterisks; these will not be misinterpreted as
c#  null tokens.
c
c#  A line with only "null" data will minimally contain (before an optional
c#  comment) either a null character, a comma or two adjacent apostrophies.
c
c#  Tokens are placed, left justified, into the character array TKNS.
c#  The tokens may be truncated in order to fit into the given TKNS array.
c
c#  The lengths of the returned tokens are placed in the LENTK array.
c
c -------------------------------------------------------------
c
c
c -------------------------------------------------------------
c#       Arrays for the returned tokens and their correcponding
c#       character lengths.
c -------------------------------------------------------------
c
      character tkns(*)*(*)
      dimension lentk(*)
c
c -------------------------------------------------------------
c#       Strings for the input line, the "curent" character on the
c#       line and delimiter characters (space, tab, and comma).
c -------------------------------------------------------------
c
      character a*(*), c*1, tab*1, space*1, comma*1, apos*1, null*1
c
      save  space      , comma      , apos       , null
      data  space /' '/, comma /','/, apos /''''/, null /'*'/
c
      tab = char(9)
c
      maxlen = len( tkns(1) )
c
c -------------------------------------------------------------
c#                 Initialize ntk tokens to blanks.
c -------------------------------------------------------------
c
      do 50 i=1,ntk
        tkns(i) = ' '
        lentk(i) = 0
   50 continue
c
      if ( irdata .eq. 0 ) return
c
c -------------------------------------------------------------
c#        Something is on the line (before any comment).
c#        Parse the tokens (up to NTKN of them).
c -------------------------------------------------------------
c
      is = 0
```

```
      itkn = 1
c
  200 continue
c       # Move along the line looking for a token or a comma (null token).
      is = is + 1
      c = a(is:is)
c
c     ---------
      if ( c.eq.tab .or. c.eq.space ) then
c         # Move to next character; to continue looking for token.
        go to 200
c     ---------
      else if ( c.eq.comma ) then
c         # No token found before this comma indicates null token.
        if ( itkn .eq. ntkn ) return
        itkn = itkn + 1
c
        if ( is .eq. irdata ) return
c         # Move to next character; look for next token.
        go to 200
c     ---------
      else
  300   continue
c         # We are at the left-most character of a non-null token.
        il = is
c
c       - - - - -
        if ( c .eq. apos ) then
c         # We look for the trailing apostrophy for this "quoted string".
          if ( is .eq. irdata ) return 1
  450     continue
          is = is + 1
          c = a(is:is)
          if ( c .eq. apos ) then
            length = is - il - 1
            if ( length .gt. maxlen ) then
              length = maxlen
            endif
            lentk( itkn ) = length
            if ( length .gt. 0 ) tkns( itkn ) = a( il+1: il+length )
            if ( is .eq. irdata ) return
            is = is + 1
            c = a(is:is)
            if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) )
     *        return 1
          else
            if ( is .eq. irdata ) return 1
            go to 450
          endif
c       - - - - -
        else
  400     continue
          if ( is .eq. irdata ) then
c           # The token extends to the end of the input data.
            if ( a(il:is) .eq. null ) return
            tkns( itkn ) = a(il: is)
c             # The input token length may exceed the length of the variable.
            length = is - il + 1
            if ( length .le. maxlen ) then
              lentk( itkn ) = length
```

```
            else
              lentk( itkn ) = maxlen
            endif
            return
          endif
c
c           # Move along the line looking for the end of the token.
          is = is + 1
          c = a(is:is)
c
          if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) ) then
            if ( c .eq. apos ) return 1
c             # Move to next character; continue looking for end of token.
            go to 400
          endif
c
c           # The previous character was the right-most character of the token
          if ( a(il: is-1) .ne. null ) then
            tkns( itkn ) = a(il: is-1)
c             # The input token length may exceed the length of the variable.
            length = is - il
            if ( length .le. maxlen ) then
              lentk( itkn ) = length
            else
              lentk( itkn ) = maxlen
            endif
          endif
c       - - - - -
        endif
c
        if ( itkn .eq. ntkn ) return
c         # Find the next location at which we should begin looking for the
c         # next token, i.e., after a following comma or at the next token.
        itkn = itkn + 1
        if ( is .eq. irdata ) return
        if ( c .eq. comma ) then
c           # Go back and look for a token following this comma.
          go to 200
        else
c           # Curent character was a space or a tab.
c           # Look for a comma, the next token, or end of line.
  500     continue
          is = is + 1
          c = a(is:is)
          if ( c .eq. comma ) then
            if ( is .eq. irdata ) return
            go to 200
          else if ( c.eq.space .or. c.eq.tab ) then
            if ( is .eq. irdata ) return
            go to 500
          else
            go to 300
          endif
        endif
c     ---------
      endif
c
      end
c
c ***************************************************************
```

```
c
      subroutine getlin( nunit, a, irdata, * )
c
c ---------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     nunit ... The Fortran logical unit number of the file from which
c#                     the input line(s) should be read.
c
c#    out    a ....... A character string into which the input line will be
c#                     read.  The length of this given string determines how
c#                     long the data line may be; input past this length is
c#                     ignored.
c
c#    out    irdata .. The input "data" in string A extends to this character
c#                     position.  The data portion does not include any
c#                     trailing comment (if present).
c
c#    -      * ....... Alternare RETURN in the event that End-of-File is
c#                     encountered on attempt to read a line from unit NUNIT.
c
c ---------------------------------------------------------------
c#  PURPOSE:
c
c#  Reads line(s) from unit NUNIT and determines the location, IRDATA, in the
c#  string beyond which there may be only spaces, tabs, or a comment.  Comments
c#  begin with a hash symbol (#).  Blank lines and lines that contain only a
c#  comment line are ignored and the next line is read until a line with some
c#  data or End-of-File is encountered.  ("Some data" might minimally be a
c#  single comma, an asterisk, or two adjacent apostrophies.)
c
c ---------------------------------------------------------------
c
      character a*(*), comnt*1
      save comnt
      data comnt /'#'/
c
c ---------------------------------------------------------------
c#                Read the input "line" from disk.
c ---------------------------------------------------------------
c
  100 continue
      read( nunit, '(a)', end=910 ) a
c
c#          We want to ignore any trailing comment and lines
c#          that are blank except for a comment (if any).
c
      icom = index( a, comnt )
c
c  # Location of right-most non-blank before comment (if any).
      if ( icom .gt. 1 ) then
        irdata = iright( a(1: icom-1) )
      else if ( icom .eq. 1 ) then
        go to 100
      else
        irdata = iright( a )
      endif
c
c        # If only blanks before comment (if present) then read next line.
      if (irdata .eq. 0) go to 100
```

```
c
      return
c
c -------- RETURN on End-of-File -------------------------
  910 continue
      return 1
c
      end
c
c ****************************************************************
c
      function ileft( string )
c
c  WRITTEN by:  D. T. Rickhoff
c
c ------------------------------------------------
c  Find the position of the leftmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c                                                                    ---
c ------------------------------------------------
c
      character*(*) string
      character*1 tab
c
      tab = char(9)
c
c ---------------------------------
c  Find the location of the first nonblank character.
c
      ls = len( string )
c
      do 100 i=1,ls
        if(      string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          ileft = i
          return
        endif
  100 continue
c
c ---------------------------------
c  String was blank.  Set ileft to zero (as a flag).
      ileft = 0
c
      return
      end
c
c ****************************************************************
c
      function iright( string )
c
c  WRITTEN by:  D. T. Rickhoff
c
c ------------------------------------------------
c  Find the position of the rightmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c                                                                    ---
c ------------------------------------------------
c
```

```
      character*(*) string
      character*1 tab
c
      tab = char(9)
c
c --------------------------------
c  Find the location of the last nonblank character.
c
      ls = len( string )
c
      do 100 i=ls,1,-1
        if(      string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          iright = i
          return
        endif
  100 continue
c
c --------------------------------
c  String was blank.  Set iright to zero (as a flag).
      iright = 0
c
      return
      end
C -----------  End of SP910.FOR
```

A3.3.2 CURVED STIFFENED PANEL MODULE # 2 PROGRAM

```
C
      PROGRAM GBINP
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
c ...1T1(10,5),SXT(10,5),SXC(10,5),SYT(10,5),SYC(10,5),SS(10,5),NT(10)
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
c ... ************************************************
c     *  Advanced Compostie Structural Program Input     *
c     *  PROBLEM 1A: Laminated flat panel w/ SECAUT Beam *
c     *                                                  *
c     *  variables definition ...                        *
c     *                                                  *
c     *   [ Geometry data ]                              *
c     *   M = numbers of axial stiffeners                *
c     *   N = numbers of lateral ribs                    *
c     *   MI= numbers of sub-grid between stiffeners     *
c     *   NI= numbers of sub-grid between ribs           *
c     *   A = axial stiffener distances (M+2)            *
c     *   B = lateral rib distances (N+2)                *
c     *                                                  *
c     *   MSET  1: STF1                                  *
c     *   MSET  2: STF2                                  *
c     *   MSET  n: STFn                                  *
c     *   MSET 10: BASE                                  *
c     *                                                  *
c     *    yal0: read tape10                             *
c     *    yal1: read tape11                             *
c     *    nal0: input data for the new tape10           *
c     *    nal1: input data for the new tape11           *
c     *    flis0: list tape10                            *
c     *    flis1: list tape11                            *
c     *    fmody: modify input data                      *
c     *    fsave: update tape10, tape11                  *
c     *    fquit: quit and end                           *
c     *                                                  *
c     *    PROGRAMMED by Lloyd Chou    81-12 MSD         *
c     *    DATED December 5, 1990      ext.6-3075        *
c     *    UPDATED by Tony Wei         81-12 MSD         *
c     *     6/28/91                    ext.6-1137        *
c     *                                                  *
c     ************************************************
      LOGICAL here10,here11,here12
      CHARACTER NAME*80
      dimension NAME(10)
c ...
c . checking input data for DIAL F.E. structural analysis
c ...
c . initialize geometry(tape10) and material(tape11) files
c ...
      inquire( file='tape10', exist=here10)
      inquire( file='tape11', exist=here11)
      if (here10) then
```

```
call yal0
```

```
                    else
          call nal0
        endif
        if (herell) then
          call yall
                    else
          call nall
        endif
c .....
        call flist
c .....
        close(unit=10)
        close(unit=11)
c .....
        open(unit=10,status='unknown',file='tape10')
        open(unit=11,status='unknown',file='tape11')
        call fsave
c .....
c ###
c # end of main program
c ###
        STOP
        end
        SUBROUTINE NEWPAGE
        do 1 I=1,24
  1     write(6,'('' '')')
        write(6,2)
  2     format(/3x,'*****************************************************',
       1        /3x,'* Advanced Composite Structural Program by LMSC  *',
       2        /3x,'* Flat Stiffener Panel : Revision 1.2, 6/28/91   *',
       3        /3x,'*****************************************************')
        return
        end
        SUBROUTINE QUIT
        write(6,1)
  1     format(//5x,'************************************',
       1         /5x,'*  No Modification, Exiting Module!   *',
       2         /5x,'************************************')
        STOP
        end
        SUBROUTINE ya10
        common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
        common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
       1              SS(12,10,5),NT(10),NAME
        common /wall/ NLAY,MW(30),TW(30),PW(30)
        common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
        common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
       1              LBC11,LBC12,LBC13,LBC14
        common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . read tape10
c ...
        character NAME*80
        dimension NAME(10)
        open(unit=10,status='old',file='tape10')
  11    format(4I5,3F10.0)
  12    format(8F10.0)
  13    format(8I5)
c  13   format(6I5)
  14    format(4F12.0)
```

```
  15  format(4i3,f10.5)
      read(10,11) M,N,MI,NI,AX,BY,R
      M2 = M+2
      N2 = N+2
      read(10,12) (A(I),I=1,M2)
      read(10,12) (B(I),I=1,N2)
c ...
      read(10,13) LEND,LRIB,LRBC,LBC11,LBC12,LBC13,LBC14,LTOT
c      read(10,13) LEND,LRIB,LEBC,LSBC,LTOT
      read(10,14) (FL(I),I=1,4)
      read(10,15) NBUC,NDF,ND,MO,TAR
      return
      end
c ...
c ...
      SUBROUTINE yall
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . read tape11
c ...
      character NAME*80
      dimension NAME(10)
      open(unit=11,status='old',file='tape11')
  11  format(9I5)
  12  format(6F12.0)
  13  format(I5,2F10.0)
      read(11,11) NMAT
      do 110 I=1,NMAT
       J = 1
      read(11,12) (E1(K,I,J),K=1,3),(V1(K,I,J),K=1,3)
      read(11,12) (G1(K,I,J),K=1,3),(A1(K,I,J),K=1,3)
      read(11,12) (SS(K,I,J),K=1,6)
 110  read(11,12) (SS(K,I,J),K=7,12)
c ...
      read(11,11) NLAY
      do 120 I=1,NLAY
 120  read(11,13) MW(I),TW(I),PW(I)
      read(11,11) MTYP
      read(11,12) XH,XB,XC,XT
      if (MTYP.eq.1.or.MTYP.eq.5) then
         IP=2
      elseif(MTYP.eq.2) then
         IP=4
      else
         IP=3
      endif
      do 130 I=1,IP
      read(11,11) MLAY(I)
      do 130 J=1,MLAY(I)
 130  read(11,13) MS(J,I),TS(J,I),PS(J,I)
      return
      end
c ...
c ...
      SUBROUTINE na10
```

```
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
c ...
c . create input for tape10
c ...
c ...
  11  format(2x,'Input Panel Width, Length and Radius >>(',3F8.3,')>'$)
  12  format(2x,'Input No. of Stiffeners and Ribs >>(',2I3,')>'$)
  13  format(2x,'Input No. of Sub-Grid between Stif/Rib >>(',2I2,')>'$)
  14  format(2x,'Input',I2,' Axial Stiffener locations >>'$)
  15  format(2x,'Input',I2,' Lateral Rib locations >>'$)
  20  format(2x,'----- Boundary Condition Definition -----')
  21  format(2x,'Should it represent a part of a much large',
     1' panel?(Y/N)>>(N)>'$)
  22  format(2x,'Are there ribs at Ends ?(Y/N)>>(Y)>'$)
  23  format(2x,'Ribs on:(1)Skin (2)STIF (3)Both ?>>(1)>'$)
  24  format(2x,'Choose a B.C. Type no. at Ribs:'
     1/10x,'(1) Simply Supported; (2) S.S. and Rotation; ',
     2     ' ?>>(1)>'$)
  25  format(2x,'Choose a B.C. type no. at z = 0:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation',
     1/10x,'(3) Free --->>(1)>'$)
  26  format(2x,'Choose a B.C. type no. at z = L:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation',
     1/10x,'(3) Free --->>(1)>'$)
  27  format(2x,'Choose a B.C. type no. at theta = 0:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation',
     1/10x,'(3) Free --->>(1)>'$)
  28  format(2x,'Choose a B.C. type no. at theta = theta_max:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation',
     1/10x,'(3) Free --->>(1)>'$)
  30  format(2x,'----- Loading Condition Definition -----')
  31  format(2x,'Choose input loads type (Total=1,Line=2)',
     c '?>>(2)>'$)
  32  format(5x,'following total load unit : lb')
  33  format(5x,'following line load unit: lb/in')
  34  format(7x,'Input axial load >>(',F10.1,')>'$)
  35  format(7x,'Input lateral load >>(',F10.1,')>'$)
  36  format(7x,'Input shear load along ends >>(',F10.1,')>'$)
  37  format(7x,'Input lateral pressure in psi >>',F10.2,')>'$)
  38  format(2x,'----- Type of Analysis ----')
c     */2x,' Post-Buckling Analysis Should Be Used Only
c ...
      call NEWPAGE
      ra(1)=10.
      ra(2)=10.
      ra(3)=122.
      write(6,11) ra(1),ra(2),ra(3)
```

```
      call NREAL(3,ra)
      AX=ra(1)
      BY=ra(2)
       R=ra(3)
      ia(1)=2
      ia(2)=1
      write(6,12) ia(1),ia(2)
      call NINTG(2,ia)
      M=ia(1)
      N=ia(2)
      ia(1)=2
      ia(2)=2
      write(6,'(2x,''Does the user want to specify Panel Sub-divisions?
     *(Y/N)'')')
      write(6,'(2x,''Enter >>''$)')
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') then
         write(6,*)' '
         write(6,'(2x,''Sub-Grid Between Stiffeners are 2.'')')
         write(6,'(2x,''Sub-Grid Between Ribs are 3.'')')
         MI=2
         NI=3
      elseif (ans .eq. 'Y' .or. ans .eq. 'y') then
         write(6,13) ia(1),ia(2)
         call NINTG(2,ia)
         MI=ia(1)
         NI=ia(2)
      endif
       B(1)=0.
       DB = BY/(N+1)
       do 2 I=1, N+1
    2     B(I+1) = DB * I
c
      DA = AX/M
      A(1) = -DA/2.
      do 1 I = 1,M
    1 A(I+1) = A(I) + DA
      A(1) = 0.
      A(M+2) = AX
c
      write(6,'(5x,''Stiffeners equally spaced (Y/N)?  >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 4
      write(6,14) M
      call NREAL(M,ra)
      do 3 I=1, M
   3     A(I+1) = ra(I)
   4  A(M+2) = AX
      write(6,'(5x,''Ribs equally spaced (Y/N)? >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 6
      write(6,15) N
      call NREAL(N,ra)
      do 5 I=1, N
   5     B(I+1) = ra(I)
   6  B(N+2) = BY
c ...
      write(6,20)
```

```
   50  write(6,22)
       ans='Y'
       LEND=1
       call STITLE(ans)
       if (ans .eq. 'Y' .or. ans .eq. 'y') go to 55
       LEND=2
   55  write(6,23)
       ia(1)=1
       call NINTG(1,ia)
       LRIB=ia(1)
       write(6,24)
       ia(1)=1
       call NINTG(1,ia)
       LRBC=ia(1)
       write(6,25)
       ia(1)=1
       call NINTG(1,ia)
       LBC11=ia(1)
       write(6,26)
       ia(1)=1
       call NINTG(1,ia)
       LBC12=ia(1)
       write(6,27)
       ia(1)=1
       call NINTG(1,ia)
       LBC13=ia(1)
       write(6,28)
       ia(1)=1
       call NINTG(1,ia)
       LBC14=ia(1)
       write(*,*) lbc11,lbc12,lbc13,lbc14
c ...
       write(6,30)
       write(6,31)
       ia(1)=2
       call NINTG(1,ia)
       LTOT=ia(1)
       if (LTOT .eq. 1) write(6,32)
       if (LTOT .ne. 1) write(6,33)
       ra(1)=1000.
       write(6,34) ra(1)
       call NREAL(1,ra)
       FL(1)=ra(1)
       ra(1)=0.
       write(6,35) ra(1)
       call NREAL(1,ra)
       FL(2)=ra(1)
       ra(1)=0.
       write(6,36) ra(1)
       call NREAL(1,ra)
       FL(3)=ra(1)
       ra(1)=0.
       write(6,37) ra(1)
       call NREAL(1,ra)
       FL(4)=ra(1)
c.............Solve Strategy...........
       write(6,38)
       write(6,'('' Choose ONE of the following type of analysis>>'')')
       write(6,'(''      1> Linear Analysis.'')')
       write(6,'(''      2> Bifurcation Buckling Analysis.'')')
```

```
      write(6,'('' 3> Post-Buckling Analysis.'')')
      write(6,'('' Enter >>''$)')
      call NINTG(1,NBUC)
      NDF=0
      ND=0
      TAR=0.
      MO=10
      if (NBUC.eq.3) then
c.....Caution message about the Post Buckling Technique
c       write(6,39)
        write(6,'('' Type of Target Point ?>>'')')
        write(6,'('' 0> Load Factor.'')')
        write(6,'('' 1-6> Displacement DOF.'')')
        write(6,'('' Enter>>''$)')
        call NINTG(1,NDF)
        if (NDF.gt.0) then
          write(6,'('' Enter Node Number >>''$)')
          call NINTG(1,ND)
        endif
        write(6,'('' Target Point Value =''$)')
        call NREAL(1,TAR)
      elseif (NBUC .eq. 2) then
        write(6,'('' How many MODE shape(s) do you want (Max. 50)? >>'
     *'$)')
        call NINTG(1,MO)
        write(6,*)' '
      endif
      return
      end
c ...
c ...
      SUBROUTINE nall
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
c ...
c . create input for tapell
c ...
  11  format(2x,'Input No. of Materials >>(',I2,')>'$)
  13  format(2x,'Input Matl#',I2,' Elastic & Shear Modulus,',
     1 ' Poison #, AML +/- Allowables')
c  14  format(5x,'E1,E2,E3    >>(',3E9.2,')>'$)
  14  format(5x,'E1,E2       >>(',2E9.2,')>'$)
  15  format(5x,'G12,G23,G31 >>(',3E9.2,')>'$)
c  16  format(5x,'V12,V23,V31 >>(',3F9.4,')>'$)
  16  format(5x,'V12         >>(',F9.4,')>'$)
  17  format(5x,'+ Allow.>>(',3(F5.1,F7.5),')>'$)
  18  format(5x,'- Allow.>>(',3(F5.1,F7.5),')>'$)
c  17  format(5x,'+ Allow. Point ',il,'>>(',F5.1,F7.3,')>'$)
c  18  format(5x,'- Allow. Point ',il,'>>(',F5.1,F7.3,')>'$)
  19  format(2x,'-----------------------------------------------')
c ...
  20  format(2x,'Input No. of plies for the basic panel >>(',I2,')>'$)
  21  format(5x,'Input Matl#, Thickness, Angle')
  22  format(5x,'Ply#',I2,'>>(',I2,F7.4,F7.2,')>'$)
```

```
   28  format(2x,'Stiffener Type: 1)Blade, 2)Close Hat, 3)I-shape, 4
      1)J-shape,',/18x,'5)Angle, 6)Zee, 7)Bead, 8)Open Hat.')
   29  format(5x,' Select axial stiffener type >>(',I1,')>'$)
   30  format(5x,'Input XH,XB,XC,XT >>(',4F6.3,')>'$)
   31  format(2x,'Input No. of plies for stiffener ELEM #',I1,'>>(',
      1 I2,')>'$)
   32  format(5x,'Input XH,XB,XC >>(',3F6.3,')>'$)
   33  format(5x,'Input XH,XB >>('2F6.3,')>'$)
   34  format(5x,'Input XH,XB,XT >>('3F6.3,')>'$)
c ...
      write(6,19)
      ia(1)=1
      write(6,11) ia(1)
      call NINTG(1,ia)
      NMAT=ia(1)
      J=1
      do 110 I=1,NMAT
      write(6,13) I
      ra(1)=30.E6
      ra(2)=ra(1)
      ra(3)=100.
      write(6,14) (ra(II),II=1,2)
      call NREAL(2,ra)
      E1(1,I,J) = ra(1)
      E1(2,I,J) = ra(2)
c ...
      ra(1)=11.5E6
      ra(2)=ra(1)
      ra(3)=ra(1)
      write(6,15) (ra(II),II=1,3)
      call NREAL(3,ra)
      do 102 K=1,3
 102  G1(K,I,J) = ra(K)
c ...
      ra(1)=0.30435
      ra(2)=0.
      ra(3)=0.
      write(6,16) ra(1)
      call NREAL(1,ra)
      V1(1,I,J) = ra(1)
c ...
      ra(1)=-60.
      ra(2)=0.001
      ra(3)=0.
      ra(4)=0.002
      ra(5)=60.
      ra(6)=0.003
      write(6,'('' Do you want to input AML Failure Parameters (Y/N)?''$
     *)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,17) (ra(II),II=1,6)
         call NREAL(6,ra)
      endif
      do 104 K=1,6
 104  SS(K,I,J) = ra(K)
c ...
      ra(1)=-60.
      ra(2)=-0.002
      ra(3)=0.
```

```
      ra(4)=-0.003
      ra(5)=60.
      ra(6)=-0.004
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,18) (ra(II),II=1,6)
         call NREAL(6,ra)
      endif
      do 105 K=7,12
      KK=K-6
 105  SS(K,I,J) = ra(KK)
 110  continue
      write(6,19)
c ...
      ia(1)=2
      write(6,20) ia(1)
      call NINTG(1,ia)
      NLAY=ia(1)
      write(6,21)
      write(6,*)' '
      write(6,'(2x,''Are the information the same for ALL plies (Y/N)?''
     *$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         ia(1) = 1
         ra(2) = 0.1
         ra(3) = 0.0
         I = 1
         write(6,22) I, ia(1), ra(2), ra(3)
         call NREAL(3,ra)
         ia(1) = int(ra(1))
         MW(1) = ia(1)
         TW(1) = ra(2)
         PW(1) = ra(3)
         do 119 i=2,NLAY
            MW(i)=mw(1)
            tw(i)=tw(1)
            pw(i)=pw(1)
 119     continue
      else
      do 120 I=1,NLAY
      ia(1)=1
      ra(2)=0.1
      ra(3)=0.0
      if (I .eq. 1) ra(3)=-45.
      if (I .eq. 2) ra(3)= 45.
      write(6,22) I,ia(1),ra(2),ra(3)
      call NREAL(3,ra)
      ia(1) = int(ra(1))
      MW(I) = ia(1)
      TW(I) = ra(2)
 120  PW(I) = ra(3)
      endif
c ...
      write(6,19)
      ia(1)=1
      write(6,28)
      write(6,29) ia(1)
      call NINTG(1,ia)
      MTYP=ia(1)
      ra(1)=0.4
```

```
      ra(2)=0.5
      ra(3)=0.35
      ra(4)=0.25
      if (MTYP.eq.1) then
         write(6,32) ra(1),ra(2),ra(3)
         call NREAL(3,ra)
         XH=ra(1)
         XB=ra(2)
         XC=ra(3)
      elseif (MTYP.eq.5.or.MTYP.eq.7) then
         write(6,33) ra(1),ra(2)
         call NREAL(2,ra)
         XH=ra(1)
         XB=ra(2)
      elseif (MTYP.eq.6) then
         write(6,34) XH,XB,XT
         call NREAL(3,ra)
         XH=ra(1)
         XB=ra(2)
         XT=ra(3)
      else
         write(6,30) ra(1),ra(2),ra(3),ra(4)
         call NREAL(4,ra)
         XH=ra(1)
         XB=ra(2)
         XC=ra(3)
         XT=ra(4)
      endif
c ...
      if (MTYP.eq.1.or.MTYP.eq.5) then
        IP=2
      elseif (MTYP.eq.2) then
        IP=4
      else
        IP=3
      endif
      do 130 I1=1,IP
      write(6,19)
      ia(1)=2
      write(6,31) I1,ia(1)
      call NINTG(1,ia)
      MLAY(I1)=ia(1)
      write(6,21)
      write(6,*)' '
      write(6,'(2x,''Are the information the save for ALL plies (Y/N)?''
     *$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y') then
         ia(1)=1
         ra(2)=0.1
         ra(3)=0.0
         i=1
         write(6,22) i,ia(1),ra(2),ra(3)
         call NREAL(3,ra)
         ia(1) = int(ra(1))
         MS(1,I1) = ia(1)
         TS(1,I1) = ra(2)
         PS(1,I1) = ra(3)
         do 131 i=2,mlay(I1)
            ms(i,i1)=ms(1,i1)
```

```
            ts(i,il)=ts(1,il)
            ps(i,il)=ps(1,il)
 131      continue
      else
      do 132 I=1,MLAY(I1)
      ia(1)=1
      ra(2)=0.1
      ra(3)=0.0
      if (I .eq. 1) ra(3)=-45.
      if (I .eq. 2) ra(3)= 45.
      ia(1)=ra(1)
      write(6,22) I,ia(1),ra(2),ra(3)
      call NREAL(3,ra)
      ia(1) = int(ra(1))
      MS(I,I1) = ia(1)
      TS(I,I1) = ra(2)
 132  PS(I,I1) = ra(3)
      endif
 130  continue
c ...
      return
      end
c ...
      SUBROUTINE menu0
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character NAME*80
   9  format(3x,'MESH MENU (MENU0):')
  10  format(4x,'0,0) Geometry & General Data')
  11  format(4x,'0,1) Base Panel Width, Length and Radius ',3F10.3)
  12  format(4x,'0,2) No. of Stiffeners and Ribs (M,N):',2I5)
  13  format(4x,'0,3) No. of GRID Between Stiffeners and Ribs(MI,NI):',2
     *I5)
  14  format(4x,'0,4) Stiffener_locations (A):',6F7.3)
  15  format(4x,'0,5) Rib_locations (B):',6F7.3)
  16  format(3x,'-------------------------------------------------')
      call NEWPAGE
      write(6,9)
      ia(1)=0
      ia(2)=0
      write(6,10)
      write(6,11) AX,BY,R
      write(6,12) M,N
      write(6,13) MI,NI
      write(6,14) (A(I),I=2,M+1)
      write(6,15) (B(I),I=2,N+1)
      write(6,16)
c ...
      return
      end
c ...
      SUBROUTINE menu1(IMAT)
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
```

```
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
    9 format(3x,'MATL MENU (MENU1):')
   10 format(4x,'1,0) Total No. of Materials (NMAT):',I3)
   11 format(4x,'     Properties for MATL No. :',I2)
c  14  format(10x,'Elastic Modulus  (E1,E2,E3):',3E9.2)
   14 format(10x,'Elastic Modulus  (E1,E2):',2E9.2)
   15 format(10x,'Shear Modulus (G12,G23,G31):',3E9.2)
c  16  format(10x,'Poisson Ratio (V12,V23,V13):'3,F9.4)
   16 format(10x,'Poisson Ratio (V12):',F9.4)
   17 format(10x,'AML + Allowables:',3(F5.1,F7.5))
   18 format(10x,'AML - Allowables:',3(F5.1,F7.5))
   19 format(3x,'-----------------------------------------------')
   20 format(10x,'Name of the Material: ',80A)
c  21  format(10x,'1,',1i,') ',a40)
c  21  format(4x,'1,',il,') ',a40)
   21 format(4x,'1,',il,') Material Number ',il)
c ...
      call NEWPAGE
      write(6,9)
      write(6,10) NMAT
      do 1 i=1,IMAT
          write(6,21) i,i
 1    continue
c ...
      return
      end
      SUBROUTINE menu2
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
    9 format(3x,'Base Ply MENU (MENU2):')
   10 format(4x,'2,0) No. of Plies for The Base Panel (NLAY):',I3)
   11 format(4x,'     Ply #  Material #  Thickness  Angle')
   12 format(8x,I4,8x,I2,7x,f7.4,2x,f7.1)
   19 format(3x,'-----------------------------------------------')
c ...
      call NEWPAGE
      write(6,9)
      write(6,10) NLAY
      write(6,11)
      do 1 I=1,NLAY
    1 write(6,12) I,MW(I),TW(I),PW(I)
      write(6,19)
c ...
      return
      end
c ...
      SUBROUTINE menu3(ILAY)
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
```

```
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
   10 format(3x,'STIF MENU (MENU3):')
   11 format(5x,'1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,',
     1 /,5x,' 5)Angle, 6)Zee, 7)Bead, 8)Open Hat.')
   12 format(4x,'3,0) Type of Stiffener :',I3)
   13 format(10x,'XH=',F6.3,' XB=',F6.3)
   14 format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3)
   15 format(10x,'XH=',F6.3,' XB=',F6.3,' XT=',F6.3)
   16 format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3,' XT=',F6.3)
   17 format(4x,'3,',il,') No. of Plies for STIF,ELEM(',I1,'):',I3)
   18 format(4x,'     Ply #  Material #  Thickness  Angle')
   19 format(8x,I4,8x,I2,7x,f7.4,2x,f7.1)
   20 format(3x,'-----------------------------------------------')
c.....................................................................
      call NEWPAGE
      write(6,10)
      write(6,11)
      write(6,*)' '
c.. indicate which type of stiffener
      write(6,12) MTYP
c.. print dimensions of the stiffener on screen
      if (MTYP .eq. 5 .or. MTYP .eq. 7) then
         write(6,13) XH,XB
      elseif (MTYP .eq. 6) then
         write(6,15) XH,XB,XT
      elseif (MTYP .eq. 1) then
         write(6,14) XH,XB,XC
      else
         write(6,16) XH,XB,XC,XT
      endif
c....  print nominal stiffener element info.
      do 1 i=1,ILAY
        write(6,17) i,i,MLAY(i)
    1 continue
c..   back to fmod3 for choosing items to modify ...
      write(6,20)
      return
      end
c ...
      SUBROUTINE menu4
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character ans*1,NAME*80
c ...
    9 format(3x,'BCLC MENU (MENU4 Boundary Cond. & Loading Cond.):')
   19 format(3x,'-----------------------------------------------')
   20 format(2x,'   ----- Boundary Condition Definition -----')
```

```
  21  format(4x,'The model is a part of a much large panel ? --->',A1)
  22  format(4x,'There are ribs at Ends ?   ---------------->',A1)
  23  format(4x,'Ribs on:(1)Skin (2)STIF (3)Both ?  ------------>',I1)
  24  format(4x,'B.C. type @ Ribs: 1) Simply Supp.; 2) S.S. and Rot.;',
     1' -->',I1)
  25  format(4x,'B.C. type no. at Ends:',
     */6x,'1) Part of Larger Panel;',
     */6x,'2) Free;                              3) Simply Supp.;',
     2/6x,'4) Clamped-1/axial+lateral free; 5) Clamped-2/axial free',
     3/6x,'6) Fully Clamped (all DOF fixed) ------>',I1)
  26  format(2x,'B.C. type no. at Sides:',
     */6x,'1) Part of Larger Panel;',
     1/6x,'2) Free;                              3) Simply Supp.;',
     2/6x,'4) Clamped-1/axial+lateral free; 5) Clamped-2/axial free',
     3/6x,'6) Fully Clamped (all DOF fixed) ------>',I1)
  30  format(2x,'   ----- Loading Condition Definition -----')
  31  format(4x,'Loads type (1:Total lb; 2:Line lb/in)  ---->',I1)
  32  format(7x,'axial load    ---------------->',F10.1)
  33  format(7x,'lateral load    -------------->',F10.1)
  34  format(7x,'shear load along ends    ----->',F10.1)
  35  format(7x,'lateral pressure in psi  ----->',F10.2)
c ...
c ...
      call NEWPAGE
      write(6,9)
      write(6,20)
      ans='N'
c      if (LARG .eq. 1) ans='Y'
c      write(6,21) ans
      ans='N'
      if (LEND .eq. 1) ans='Y'
      write(6,22) ans
      write(6,23) LRIB
      write(6,24) LRBC
      write(6,25) LEBC
      write(6,26) LSBC
      write(6,30)
      write(6,31) LTOT
      write(6,32) FL(1)
      write(6,33) FL(2)
      write(6,34) FL(3)
      write(6,35) FL(4)
c ...
      return
      end
c ...
      SUBROUTINE flist
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ia(10),ra(10),NAME(10)
      character NAME*80
c
c       1         2         3         4         5         6         7 2
c ...
```

```
c . list input
c ...
   9  format(4x,'m,n) >>> MAIN MENU <<< ',
     1 '(general MESH data)')
  10  format(4x,'0,0) Geometry & General Data')
  11  format(4x,'0,1) Base Panel Width, Length and Radius ',3F10.3)
  12  format(4x,'0,2) No. of Stiffeners and Ribs (M,N):',2I5)
  13  format(4x,'0,3) No. of GRID for Stiffeners and Ribs(MI,NI):',2I5)
  14  format(4x,'0,4) Stiffener_locations (A):',6F7.3)
  15  format(4x,'0,5) Rib_locations (B):',6F7.3)
c ...
  16  format(4x,'1,0) Total No. of Materials (NMAT):',I2)
  17  format(4x,'-------------------------------------------------')
  18  format(4x,'     >>> OTHER MENUS <<< ',
     1 '(MATL, BASE Ply, STIF Ply, BC & LC)')
c ...
  21  format(4x,'2,0) No. of Plies for The Base Panel(NLAY):',I2)
  29  format(4x,'3,0) Axial stiffener type No.(MTYP):',I2,
     1      /4x,'Blade=1,Close Hat=2,"I"=3,"J"=4,Angle=5,"Z"=6,Bead=7,
     1Open Hat=8.')
  30  format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3,' XT=',F6.3)
  31  format(4x,'3,1) No. of Plies for STIF, Element 1(MLAY1):',I2)
  32  format(4x,'3,2) No. of Plies for STIF, Element 2(MLAY2):',I2)
  33  format(4x,'3,3) No. of Plies for STIF, Element 3(MLAY3):',I2)
  34  format(4x,'3,4) No. of Plies for STIF, Element 4(MLAY4):',I2)
  35  format(4x,'4,0) Boundary & Loading Conditions')
  36  format(10x,'XH=',F6.3,' XB=',F6.3,' XC=',F6.3)
  37  format(10x,'XH=',F6.3,' XB=',F6.3)
  38  format(10x,'XH=',F6.3,' XB=',F6.3,' XT=',F6.3)
c ...
   1  call NEWPAGE
      write(6,9)
      ia(1)=0
      ia(2)=0
      write(6,10)
      write(6,11) AX,BY,R
      write(6,12)  M, N
      write(6,13) MI,NI
      write(6,14)'(A(I),I=2,M+1)
      write(6,15)'(B(I),I=2,N+1)
c ...
      write(6,17)
      write(6,18)
      write(6,16)NMAT
      write(6,21)NLAY
      write(6,29)MTYP
      if (MTYP.eq.1) then
         write(6,36)XH,XB,XC
      elseif (MTYP.eq.5.or.MTYP.eq.7) then
         write(6,37) XH,XB
      elseif (MTYP.eq.6) then
         write(6,38) XH,XB,XT
      else
         write(6,30) XH,XB,XC,XT
      endif
      write(6,35)
c ...
      write(6,'('' Select m,n=> -1:END; 9:QUIT; m,n:UPDATE >>''$)')
      ia(1)=0
      ia(2)=0
```

```
      call NINTG(2,ia)
      ISEL1= ia(1)
      ISEL2= ia(2)
 50   if (ISEL1 .eq. -1) go to 90
      if (ISEL1 .eq.  9) call QUIT
      if (ISEL1 .eq. 0 .and. ISEL2 .ne. 0) call fmod0(ISEL2)
      if (ISEL1 .eq. 0 .and. ISEL2 .eq. 0) go to 1
      if (ISEL1 .eq. 1 ) call fmod1(ISEL1,ISEL2)
      if (ISEL1 .eq. 2 ) call fmod2(ISEL1,ISEL2)
      if (ISEL1 .eq. 3 ) call fmod3(ISEL1,ISEL2)
      if (ISEL1 .eq. 4 ) call fmod4(ISEL1,ISEL2)
      go to 50
c ...
  90  continue
      return
      end
c ...
c ...
      SUBROUTINE fmod0(IS2)
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c       1         2         3         4         5         6         7 2
c ...
c . modify geometry input
c ...
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
  11  format(5x,'Modify Panel Width, Length and Radius >>(',3F8.3,')>'$)
  12  format(5x,'Modify No. of Stiffeners and Ribs >>(',2I3,')>'$)
  13  format(5x,'Modify No. of Sub-Grid between Stif/Rib >>(',2I2,')>'$)
  14  format(5x,'Modify',I2,' Axial Stiffener locations >>'$)
  15  format(5x,'Modify',I2,' Lateral Rib locations >>'$)
c ...
      call menu0
      if (IS2 .le. 1) then
         ra(1)=AX
         ra(2)=BY
         ra(3)=R
         write(6,11)ra(1),ra(2),ra(3)
         call NREAL(3,ra)
         AX=ra(1)
         BY=ra(2)
          R=ra(3)
      endif
      if (IS2 .le. 2) then
         ia(1)=M
         ia(2)=N
         write(6,12) ia(1),ia(2)
         call NINTG(2,ia)
         M=ia(1)
         N=ia(2)
      endif
      if (IS2 .le. 3) then
         ia(1)=MI
         ia(2)=NI
         write(6,13) ia(1),ia(2)
```

```
        write(6,*)' '
        write(6,'(5x,''Does the user want to specify Sub-Divisions? (Y/
     *N)'')')
        write(6,'(5x,''Enter >>''$)')
        call STITLE(ans)
        if (ans .eq. 'N' .or. ans .eq. 'n') then
           MI=2
           NI=3
           write(6,'(5x,''Sub-Grids between Stiffeners are 2.'')')
           write(6,'(5x,''Sub-Grids between Ribs are 3.'')')
        elseif (ans .eq. 'Y' .or. ans .eq. 'y') then
           write(6,13) ia(1),ia(2)
           call NINTG(2,ia)
           MI=ia(1)
           NI=ia(2)
        endif
      endif
      if (IS2 .le. 4) then
c
      DA = AX/M
      A(1) = -DA/2.
      do 1 I = 1,M
    1 A(I+1) = A(I) + DA
      A(1) = 0.
      A(M+2) = AX
c
c  end of correction 1-8-92
      write(6,'(5x,''Stiffeners equally spaced (Y/N)?  >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 4
      write(6,14) M
      call NREAL(M,ra)
      do 3 I=1, M
   3  A(I+1) = ra(I)
   4  A(M+2) = AX
      endif
      if (IS2 .eq. 5 .or. IS2 .le. 3) then
      B(1)=0.
      DB = BY/(N+1)
      do 2 I=1, N+1
   2  B(I+1) = DB * I
      write(6,'(5x,''Ribs equally spaced (Y/N)? >>(Y)>''$)')
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'Y' .or. ans .eq. 'y') go to 6
      write(6,15) N
      call NREAL(N,ra)
      do 5 I=1, N
   5  B(I+1) = ra(I)
   6  B(N+2) = BY
      endif
c ...
 110  write(6,112)
      ia(1)=1
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
 111  continue
```

```
  112  format(3x,' Selection=> 1,n: modify MATL #n;',
     1 ' -1:End; 9:Quit;  m,0: other MENUS >>'$)
       return
       end
c ...
c
       SUBROUTINE fmod1(IS1,IS2)
       common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
       common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
      1              SS(12,10,5),NT(10),NAME
       common /wall/ NLAY,MW(30),TW(30),PW(30)
       common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
       common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . modify geometry input
c ...
       dimension ra(10),ia(10),NAME(10)
       character ans*1,NAME*80
   11  format(5x,'Enter E1,E2        > '$)
   12  format(5x,'Enter G12,G23,G31 > '$)
   13  format(5x,'Enter V12         > '$)
   14  format(5x,'Enter AML + Allowables (3 pairs)> '$)
   15  format(5x,'Enter AML - Allowables (3 pairs)> '$)
   16  format(5x,'Update Elastic Modulus (Y/N)? >>(N)>'$)
   17  format(5x,'Update Shear Modulus (Y/N)? >>(N)>'$)
   18  format(5x,'Update Poisson Ratio (Y/N)? >>(N)>'$)
   19  format(5x,'Update AML +Allowables(Y/N)? >>(N)>'$)
   20  format(5x,'Update AML -Allowables(Y/N)? >>(N)>'$)
   21  format(5x,'Modify Above Material Properties (Y/N)? >>(N)>'$)
   22  format(5x,'Add MATL ID',I2,' (Current number of MATL=',I2,')')
   23  format(5x,'Add New Material No. ',i2,' (Y/N)? >>(Y)>'$)
   24  format(5x,'Name of the Material > '$)
   25  format(5x,'Update Name of the Material (Y/N)? >>(N)>'$)
   26  format(5x,'Add Material (Y/N)? >>(N)>'$)
  112  format(3x,'Selection=> 1,n: modify MATL #n;',
     1/,5x,'-1: Exit; 9: Quit; m,0: other MENUS >>'$)
  113  format(4x,'     Properties for MATL No. :',I2)
  114  format(5x,'1>>Elastic Modulus  (E1,E2):',2E9.2)
  115  format(5x,'2>>Shear Modulus (G12,G23,G31):',3E9.2)
  116  format(5x,'3>>Poisson Ratio (V12):',F9.4)
  117  format(5x,'4>>AML + Allowables:',3(F5.1,2x,F7.5))
  118  format(5x,'5>>AML - Allowables:',3(F5.1,2x,F7.5))
  119  format(3x,'-----------------------------------------------')
c ...
  110  IMAT=NMAT
       call menu1(IMAT)
       write(6,112)
       ia(1)=1
       ia(2)=0
       call NINTG(2,ia)
       IS1=ia(1)
       IS2=ia(2)
       if (IS1 .ne. 1) go to 111
       if (IS2 .eq. 0) go to 100
c.....................................................
       call NEWPAGE
       write(6,113) IS2
       write(6,114) (E1(j,Is2,1),j=1,2)
       write(6,115) (G1(j,Is2,1),j=1,3)
       write(6,116) (V1(j,Is2,1),j=1,1)
```

```
      write(6,117) (SS(j,Is2,1),j=1,6)
      write(6,118) (SS(j,Is2,1),j=7,12)
      write(6,119)
c ...
      write(6,21)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,'(4x,''Select Item to be modified >>''$)')
      call NINTG(1,kk)
      go to (1,2,3,4,7)kk
  1   continue
      write(6,16)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      continue
      write(6,11)
      ra(1)=E1(1,IS2,1)
      ra(2)=E1(2,IS2,1)
      ra(3)=100.
      call NREAL(2,ra)
      E1(1,IS2,1)=ra(1)
      E1(2,IS2,1)=ra(2)
      E1(3,IS2,1)=ra(3)
      go to 10
  2   continue
c ...
      write(6,17)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,12)
      ra(1)=G1(1,IS2,1)
      ra(2)=G1(2,IS2,1)
      ra(3)=G1(3,IS2,1)
      call NREAL(3,ra)
      G1(1,IS2,1)=ra(1)
      G1(2,IS2,1)=ra(2)
      G1(3,IS2,1)=ra(3)
      go to 10
  3   continue
c ...
      write(6,18)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,13)
      ra(1)=V1(1,IS2,1)
      ra(2)=0.
      ra(3)=0.
      call NREAL(1,ra)
      V1(1,IS2,1)=ra(1)
      V1(2,IS2,1)=ra(2)
      V1(3,IS2,1)=ra(3)
      go to 10
    4 continue
c ...
      write(6,19)
      ans='N'
```

```
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,14)
      do 5 I=1,6
    5 ra(I)=SS(I,IS2,1)
      call NREAL(6,ra)
      do 6 I=1,6
    6 SS(I,IS2,1)=ra(I)
      go to 10
    7 continue
c ...
      write(6,20)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
      write(6,15)
      do 8 I=1,6
      II=I+6
    8 ra(I)=SS(II,IS2,1)
      call NREAL(6,ra)
      do 9 I=1,6
      II=I+6
    9 SS(II,IS2,1)=ra(I)
   10 go to 110
c.....................................................
  100 itmp=IMAT+1
      write(6,23) itmp
      ans='Y'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 110
c.....
      call NEWPAGE
      NMAT=itmp
      write(6,11)
      ra(1)=E1(1,IMAT,1)
      ra(2)=E1(2,IMAT,1)
      ra(3)=E1(3,IMAT,1)
      call NREAL(3,ra)
      E1(1,itmp,1)=ra(1)
      E1(2,itmp,1)=ra(2)
      E1(3,itmp,1)=ra(3)
c ...
      write(6,12)
      ra(1)=G1(1,IMAT,1)
      ra(2)=G1(2,IMAT,1)
      ra(3)=G1(3,IMAT,1)
      call NREAL(3,ra)
      G1(1,itmp,1)=ra(1)
      G1(2,itmp,1)=ra(2)
      G1(3,itmp,1)=ra(3)
c ...
      write(6,13)
      ra(1)=V1(1,IMAT,1)
      ra(2)=V1(2,IMAT,1)
      ra(3)=V1(3,IMAT,1)
      call NREAL(3,ra)
      V1(1,itmp,1)=ra(1)
      V1(2,itmp,1)=ra(2)
      V1(3,itmp,1)=ra(3)
c ...
```

```
c ...   AML input ...
c ...
      ra(1) = -60.
      ra(2) = 0.001
      ra(3) = 0.
      ra(4) = 0.002
      ra(5) = 60.
      ra(6) = 0.001
      write(6,'(''Do you want to input AML Failure Parameters (Y/N)?''$)
     *')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y') then
         write(6,14)
         do 105 I=1,6
 105     ra(I)=SS(I,IMAT,1)
         call NREAL(6,ra)
      endif
      do 106 I=1,6
 106  SS(I,itmp,1)=ra(I)
c ...
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,15)
         do 108 I=1,6
         II=I+6
 108     ra(I)=SS(II,IMAT,1)
         call NREAL(6,ra)
      endif
      do 109 I=1,6
      II=I+6
 109  SS(II,itmp,1)=ra(I)
c ...
c ...
 111  continue
      return
      end
c ...
      SUBROUTINE fmod2(IS1,IS2)
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . modify basic panel plies
c ...
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
  11  format(5x,'Modify the no. of plies @ the panel(Y/N)? >>(N)>'$)
  12  format(5x,'Enter MATL#, THK & Angle @ PLY #',I2,'>'$)
  13  format(5x,'Enter the number of plies >>(',I2,')>'$)
  14  format(3x,' Selection => 2,n: modify panel plies;',
     1/,' -1:End; 9:Quit; m,0: other MENUS >>'$)
c ...
   1  if (IS1 .ne. 2) go to 9
      call menu2
      write(6,11)
      ans='N'
      call STITLE(ans)
      if (ans .eq. 'N' .or. ans .eq. 'n') go to 10
```

```
      ia(1)=NLAY
      write(6,13) ia(1)
      call NINTG(1,ia)
      NLAY=ia(1)
      write(6,*)' '
      write(6,'(5x,''Enter MATL#, THK & Angle @ PLY #>'')')
      write(6,'(2x,''Are all information the same for ALL plies (Y/N)?''
     *$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y') then
         ia(1)=mw(1)
         ra(2)=tw(1)
         ra(3)=pw(1)
         ii=1
         write(6,12)ii
         call NREAL(3,ra)
         ia(1) = int(ra(1))
         mw(1)=ia(1)
         tw(1)=ra(2)
         pw(1)=ra(3)
         do 20 i=2,NLAY
         mw(i)=mw(1)
         tw(i)=tw(1)
         pw(i)=pw(1)
 20      continue
         go to 30
      else
      do 2 I=1,NLAY
      ia(1)=MW(I)
      ra(2)=TW(I)
      ra(3)=PW(I)
      write(6,12) I
      call NREAL(3,ra)
      ia(1) = int(ra(1))
      MW(I)=ia(1)
      TW(I)=ra(2)
   2  PW(I)=ra(3)
      go to 30
      endif
c ...
  10  write(6,'('' Choose PLY NUMBER to modify (Y/N)?''$)')
      call STITLE(ans)
      if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
         write(6,'('' Enter PLY NUMBER =''$)')
         call NINTG(1,kp)
         write(6,12)kp
         call NREAL(3,ra)
         ia(1) = int(ra(1))
         MW(kp)=ia(1)
         TW(kp)=ra(2)
         PW(kp)=ra(3)
      endif
 30   write(6,14)
      ia(1)=2
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
      go to 1
c ...
```

```
   9  continue
      return
      end
c ...
c ...
      SUBROUTINE fmod3(IS1,IS2)
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80
c ...
c . modify MENU3: STIF, ELEMENT Ply Layup
c ...
  11  format(3x,' Selection => 3,n: modify panel plies;',
     1/,16x,' -1:End; 9:Quit; m,0: other MENUS; -1: exit. >>'$)
  12  format(5x,'Change Dimensions of the Stiffeners (Y/N)? >>(N)>'$)
  13  format(5x,'Enter XH,XB,XC,XT >>(',4F6.3,')>'$)
  14  format(5x,'Enter XH,XB,XC >>(',3F6.3,')>'$)
  15  format(5x,'Enter XH,XB >>(',2F6.3,')>'$)
  16  format(5x,'Enter XH,XB,XT >>(',3F6.3,')>'$)
  17  format(5x,'Change the Stiffener Type (Y/N)? >>(N)>'$)
  18  format(5x,'Change Axial Stiffener from Type ',il,' to >> '$)
  19  format(5x,'1)Blade, 2)Close Hat, 3)I-shape, 4)J-shape,',
     1 /,5x,' 5)Angle, 6)Zee, 7)Bead, 8)Open Hat.')
  20  format(5x,'Enter MATL#, THK & Angle @ PLY #',I2,'>'$)
  22  format(5x,'Enter the number of plies >>(',I2,')>'$)
  23  format(5x,'Enter NUMBER OF PLIES for STIFF ELEM ',il,' >>'$)
  24  format(5x,'Modify the number of above plies (Y/N)? >>(N)>'$)
  25  format(4x,'3,',il,') No. of Plies for STIF,ELEM(',I1,'):',I3)
  26  format(4x,'     Ply #  Material #  Thickness  Angle')
  27  format(8x,I4,8x,I2,7x,f7.4,2x,f7.1)
c.............................................................
c...  identify how many element for this type of stiffener ...
   1  if (MTYP .EQ. 1 .OR. MTYP .EQ. 5) then
         IP=2
      elseif (MTYP.eq.2) then
         IP=4
      else
         IP=3
      endif
c..
      call menu3(IP)
c..   choose items to modify or exit this menu ..
  10  write(6,11)
      ia(1)=3
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
      if (IS1 .ne. 3) go to 9
      if (IS2 .lt. 0 .or. IS2 .gt. IP) then
         write(6,'('' Select Again, Please.'')')
         go to 10
      endif
c.
c ..  change stiffener geometry ..
```

```
c.
      if (IS2.eq.0) then
c.
c ..     change stiffener type ? ..
c.
         write(6,17)
         ans='N'
         call STITLE(ans)
         if (ans .eq. 'Y' .or. ans .eq. 'y') go to 500
c.
c ..     change stiffener dimensions ? ..
c.
         write(6,12)
         ans='N'
         call STITLE(ans)
         if (ans .eq. 'N' .or. ans .eq. 'n') then
            go to 1
         else
            if (MTYP .EQ. 5 .or. MTYP .eq. 7) THEN
               write(6,15) XH,XB
               call NREAL(2,XH,XB)
            elseif (MTYP .eq. 6) then
               write(6,16) XH,XB,XT
               call NREAL(3,XH,XB,XT)
            elseif (MTYP .eq. 1) then
               write(6,14) XH,XB,XC
               call NREAL(3,XH,XB,XC)
            else
               write(6,13) XH,XB,XC,XT
               call NREAL(4,XH,XB,XC,XT)
            endif
            go to 1
         endif
c..
c...     change stiffener type ..
c..
500      call NEWPAGE
         ia(1)=MTYP
         write(6,19)
         write(6,18) ia(1)
         call NINTG(1,ia)
         MTYP=ia(1)
         ra(1)=XH
         ra(2)=XB
         ra(3)=XC
         ra(4)=XT
         if (MTYP.eq.1) then
            write(6,'('' Enter dimensions XH, XB, XC >>''$)')
            call NREAL(3,ra)
            XH=ra(1)
            XB=ra(2)
            XC=ra(3)
         elseif (MTYP.eq.5.or.MTYP.eq.7) then
            write(6,'('' Enter dimensions XH, XB >>''$)')
            call NREAL(2,ra)
            XH=ra(1)
            XB=ra(2)
         elseif (MTYP.eq.6) then
            write(6,'('' Enter dimensions XH, XB, XT >>''$)')
            call NREAL(3,ra)
```

```
               XH=ra(1)
               XB=ra(2)
               XT=ra(3)
            else
               write(6,'('' Enter dimensions XH, XB, XC, XT >>''$)')
               call NREAL(4,ra)
               XH=ra(1)
               XB=ra(2)
               XC=ra(3)
               XT=ra(4)
            endif
c ...
            if (MTYP .eq. 1 .or. MTYP .eq. 5) then
               ip=2
            elseif (mtyp .eq. 2) then
               ip=4
            else
               ip=3
            endif
c ...
            do 501 i=1,IP
               write(6,23)i
               call NINTG(1,ia)
               MLAY(i)=ia(1)
c ...
               write(6,'(2x,''Are all information the same for ALL plies (Y
     *N)?''$)')
               call STITLE(ans)
               if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
                  j=1
                  write(6,20) J
                  call NREAL(3,ra)
                  ia(1) = int(ra(1))
                  ms(1,i)=ia(1)
                  ts(1,i)=ra(2)
                  ps(1,i)=ra(3)
                  do 502 k=2,MLAY(i)
                  ms(k,i)=ms(1,i)
                  ts(k,i)=ts(1,i)
                  ps(k,i)=ps(1,i)
502               continue
               else
               do 503 j=1,MLAY(i)
                  ia(1)=MS(j,i)
                  ra(2)=TS(j,i)
                  ra(3)=PS(j,i)
                  write(6,20)J
                  call NREAL(3,ra)
                  ia(1) = int(ra(1))
                  MS(j,i)=ia(1)
                  TS(j,i)=ra(2)
                  PS(j,i)=ra(3)
503            continue
               endif
501         continue
            go to 1
         endif
c...       modify STIFF ELEM only ...
         call NEWPAGE
         write(6,25) is2,is2,MLAY(is2)
```

```
      write(6,26)
      do 601 j=1,MLAY(is2)
         write(6,27) j,MS(j,is2),TS(j,is2),PS(j,is2)
601   continue
      write(6,'(8x,''00  No Change'')')
      write(6,*)' '
      write(6,24)
      ans='N'
      call STITLE(ans)
      write(6,*)' '
      if (ans .eq. 'N' .or. ans .eq. 'n') then
         write(6,'(5x,''Choose the PLY NUMBER you wish to change >>''$)
     *')
         call NINTG(1,k)
         if ( k .eq. 0 ) go to 1
         write(6,20) k
         call NREAL(3,ra)
c
         ia(1) = int(ra(1))
         MS(k,is2)=ia(1)
         TS(k,is2)=ra(2)
         PS(k,is2)=ra(3)
c
         go to 1
       else
         write(6,23)is2
         call NINTG(1,MLAY(is2))
         write(6,*)' '
         write(6,'(2x,''Are all information the same for ALL plies (Y/N)
     *?''$)')
         call STITLE(ans)
         if ( ans .eq. 'Y' .or. ans .eq. 'y' ) then
            j=1
            write(6,20)J
            call NREAL(3,ra)
c
            ia(1) = int(ra(1))
            MS(1,is2)=ia(1)
            ts(1,is2)=ra(2)
            ps(1,is2)=ra(3)
c
            do 602 kg=2,MLAY(is2)
            ms(kg,is2)=ms(1,is2)
            ts(kg,is2)=ts(1,is2)
            ps(kg,is2)=ps(1,is2)
  602       continue
         else
         do 603 I=1,MLAY(is2)
            write(6,20) I
            call NREAL(3,ra)
c
            ia(1) = int(ra(1))
            MS(i,is2)=ia(1)
            TS(i,is2)=ra(2)
            PS(i,is2)=ra(3)
c
  603    continue
         endif
         go to 1
       endif
```

```
c.....................................................
   9  continue
      return
      end
c ...
c ...
      SUBROUTINE fmod4(IS1,IS2)
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
      dimension ra(10),ia(10),NAME(10)
      character ans*1,NAME*80,type*15
c*******************************
c  FORMAT STATEMENTS FOR LISTING
c*******************************
  21  format(2x,'Should it represent a part of a much larger',
     1' panel?(Y/N)>>(N)>'$)
  22  format(2x,'Are there ribs at Ends ?(Y/N)>>(Y)>'$)
  23  format(2x,'Ribs on:(1)Skin (2)STIF (3)Both ?>>(1)>'$)
  24  format(2x,'Choose a B.C. Type no. at Ribs:'
     1/10x,'(1) Simply Supported; (2) S. S. with Rotation;',
     2     ' ?>>(1)>'$)
  25  format(2x,'Choose a B.C. type no. at z = 0:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation',
     1/10x,'(3) Free --->>(1)>'$)
  26  format(2x,'Choose a B.C. type no. at z = L:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation ',
     1/10x,'(3) Free --->>(1)>'$)
  27  format(2x,'Choose a B.C. type no. at theta = 0:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation ',
     1/10x,'(3) Free --->>(1)>'$)
  28  format(2x,'Choose a B.C. type no. at theta = theta_max:',
     */10x,'(1) Simply Supported',
     */10x,'(2) S.S. and Rotation ',
     1/10x,'(3) Free --->>(1)>'$)
  30  format(2x,'----- Loading Condition Definition -----')
  31  format(2x,'Choose input loads type (Total=1,Line=2)',
     c '?>>(2)>'$)
  32  format(5x,'following total load unit : lb')
  33  format(5x,'following line load unit: lb/in')
  34  format(7x,'Input axial load >>(',F10.1,')>'$)
  35  format(7x,'Input lateral load >>(',F10.1,')>'$)
  36  format(7x,'Input shear load along ends >>(',F10.1,')>'$)
  37  format(7x,'Input lateral pressure in psi >>',F10.1,')>'$)
  38  format(7x,' 4,9)Axial = ',f10.1)
  39  format(7x,' 4,10)Lateral = ',f10.1)
  40  format(7x,' 4,11)Shear Along the Ends = ',f10.1)
  41  format(7x,' 4,12)Lateral Pressure (psi.)= ',f10.1)
  42  format(2x,' --- Solution Technique ---')
  43  format(5x,'TYPE OF TARGET >',a15,i2,1x,'NODE=',i3,1x,
     *'TARGET=',f10.5)
  44  format(5x,'TYPE OF TARGET >',a15,1x,'TARGET=',f10.5)
```

```
c*************
c*************
50    call NEWPAGE
10    write(6,'('' ----- Boundary Condition Definition -----'')')
      if (LEND.eq.1) then
         write(6,'('' 4,1)There ARE ribs on the ends.'')')
            else
         write(6,'('' 4,1)There are NO ribs on the ends.'')')
      endif
      if (LRIB.eq.1) then
         write(6,'('' 4,2)Ribs on the Skin.'')')
            elseif (LRIB.eq.2) then
         write(6,'('' 4,2)Ribs on the Stiffener.'')')
            elseif (LRIB.eq.3) then
         write(6,'('' 4,2)Ribs on both the Skin and the Stiffener.'')')
      endif
      if (LRBC .eq. 1) then
         write(6,'('' 4,3)Ribs are simulated using Simple Support Constr
     *aints.'')')
      elseif (LRBC .eq. 2) then
         write(6,'('' 4,3)Ribs are simulated using Simple Support plus R
     *otational Constraints.'')')
      endif
c
      if(lbcl1.eq.1) then
         write(6,'('' 4,4) Simply Supported at z=0'')')
      elseif(lbcl1.eq.2) then
         write(6,'('' 4,4) Simply Supported and Rotation at z=0'')')
      elseif(lbcl1.eq.3) then
         write(6,'('' 4,4) free at z=0'')')
      endif
c
      if(lbcl2.eq.1) then
         write(6,'('' 4,5) Simply Supported at z=L'')')
      elseif(lbcl2.eq.2) then
         write(6,'('' 4,5) Simply Supported and Rotation at z=L'')')
      elseif(lbcl2.eq.3) then
         write(6,'('' 4,5) free at z=L'')')
      endif
c
      if(lbcl3.eq.1) then
         write(6,'('' 4,6) Simply Suported at theta=0'')')
      elseif(lbcl3.eq.2) then
         write(6,'('' 4,6) Simply Supported and Rotation
     & at theta=0'')')
      elseif(lbcl3.eq.3) then
         write(6,'('' 4,6) free at theta=0'')')
      endif
c
      if(lbcl4.eq.1) then
         write(6,'('' 4,7) Simply Supported theta=theta_max'')')
      elseif(lbcl4.eq.2) then
         write(6,'('' 4,7) Simply Supported and Rotation
     & at theta=theta_max'')')
      elseif(lbcl4.eq.3) then
         write(6,'('' 4,7) free at theta=theta_max'')')
      endif
c
         write(6,'(''-------------------------------------------------
     *----------------------'')')
```

```
      write(6,30)
      if (LTOT.eq.1) then
         write(6,'(7x,'' 4,8)Total Loads (lbs.):'')')
           else
         write(6,'(7x,'' 4,8)Line Loads (lbs.-in.):'')')
      endif
      write(6,38)FL(1)
      write(6,39)FL(2)
      write(6,40)FL(3)
      write(6,41)FL(4)
      write(6,42)
      if (NBUC.eq.1) then
         write(6,'(7x,'' 4,13) Linear Analysis.'')')
      elseif (NBUC.eq.2) then
         write(6,'(7x,'' 4,13) Bifurcation Buckling Analysis.'')')
         write(6,'(16x,i3,'' Mode(s).'')')MO
      elseif (NBUC.eq.3) then
         write(6,'(7x,'' 4,13) Post-Buckling Analysis.'')')
         if (NDF.gt.0) then
            TYPE='DIS. DOF '
            write(6,43) TYPE,NDF,ND,TAR
         else
            TYPE='LOAD FACTOR '
            write(6,44) TYPE,TAR
         endif
      endif
c*****************************
c  FORMAT STATEMENTS FOR MODIFICATION
c*****************************
      write(6,'(''-----------------------------------------------'')')
      write(6,'('' Select m,n=> -1:Exit; 9:Quit; m,n: Other Menu or Upda
     *te.>>''$)')
      ia(1)=0
      ia(2)=0
      call NINTG(2,ia)
      IS1=ia(1)
      IS2=ia(2)
      if (IS1.ne.4) then
         go to 999
      else
         if (IS2.eq.1) then
            write(6,22)
            call STITLE(ans)
            if (ans.eq.'N'.or.ans.eq.'n') then
               LEND=2
            else
               LEND=1
            endif
         elseif (IS2.eq.2) then
            write(6,23)
            call NINTG(1,LRIB)
         elseif (IS2 .eq. 3) then
            write(6,24)
            call NINTG(1,LRBC)
         elseif (IS2.eq.4) then
            write(6,25)
            call NINTG(1,LBC11)
         elseif (IS2.eq.5) then
            write(6,26)
            call NINTG(1,LBC12)
```

```
         elseif (IS2.eq.6) then
            write(6,27)
            call NINTG(1,LBC13)
         elseif (IS2.eq.7) then
            write(6,28)
            call NINTG(1,LBC14)
c ...
c . modify loading conditions
c ...
         elseif (IS2.eq.8) then
            write(6,31)
            call NINTG(1,LTOT)
         elseif (IS2.eq.9) then
            write(6,34)FL(1)
            call NREAL(1,ra)
            FL(1)=ra(1)
         elseif (IS2.eq.10) then
            write(6,35)FL(2)
            call NREAL(1,ra)
            FL(2)=ra(1)
         elseif (IS2.eq.11) then
            write(6,36)FL(3)
            call NREAL(1,ra)
            FL(3)=ra(1)
         elseif (IS2.eq.12) then
            write(6,37)FL(4)
            call NREAL(1,ra)
            FL(4)=ra(1)
         elseif (IS2.eq.13) then
            write(6,'('' Choose ONE of the following technique >>'')')
            write(6,'('' 1> Linear Solution.'')')
            write(6,'('' 2> Bifurcation Buckling Analysis.'')')
            write(6,'('' 3> Post-Buckling Analysis.'')')
            write(6,'('' Enter>>''$)')
            call NINTG(1,NBUC)
            if (NBUC.eq.3) then
               write(6,'('' Type of Target -- 0) Load Factor.'')')
               write(6,'('' 1-6) Displace. DOF.'')')
               write(6,'('' Enter >>''$)')
               call NINTG(1,NDF)
               if (NDF.gt.0) then
                  write(6,'('' Node Number =''$)')
                  call NINTG(1,ND)
               endif
               write(6,'('' Target Point = ''$)')
               call NREAL(1,TAR)
            elseif (NBUC .eq. 2) then
               write(6,'('' How many MODE shape(s) do you want (Max. 50
     *)? >>''$)')
               call NINTG(1,MO)
            endif
         endif
         go to 50
999   endif
      return
      end
c ...
c ...
      SUBROUTINE fsave
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
```

```
      common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1              LBC11,LBC12,LBC13,LBC14
      common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c . save and update files
c ...
      character NAME*80
      dimension NAME(10)
   11 format(4I5,3F10.4)
   12 format(6E12.4)
   13 format(I5,2F10.4)
   14 format(8F10.4)
   15 format(8I5)
   16 format(4i3,f10.5)
      write(10,11) M,N,MI,NI,AX,BY,R
      M2 = M+2
      N2 = N+2
      write(10,14) (A(I),I=1,M2)
      write(10,14) (B(I),I=1,N2)
c ...
      write(10,15) LEND,LRIB,LRBC,LBC11,LBC12,LBC13,LBC14,LTOT
      write(10,12) (FL(I),I=1,4)
      write(10,16) NBUC,NDF,ND,MO,TAR
c ...
      write(11,11) NMAT
      do 110 I=1,NMAT
        J=1
        write(11,12) (E1(K,I,J),K=1,3),(V1(K,I,J),K=1,3)
        write(11,12) (G1(K,I,J),K=1,3),(A1(K,I,J),K=1,3)
        write(11,12) (SS(K,I,J),K=1,6)
  110   write(11,12) (SS(K,I,J),K=7,12)
c ...
      write(11,11) NLAY
      do 120 I=1,NLAY
  120 write(11,13) MW(I),TW(I),PW(I)
      write(11,11) MTYP
      write(11,12) XH,XB,XC,XT
      if (MTYP.eq.1.or.MTYP.eq.5) then
        IP=2
      elseif (MTYP.eq.2) then
        IP=4
      else
        IP=3
      endif
      do 130 I=1,IP
      write(11,11) MLAY(I)
      do 130 J=1,MLAY(I)
  130 write(11,13) MS(J,I),TS(J,I),PS(J,I)
c ...
c .   DIAL L3D2 Runstreams
c ...
      call fmesh
      call fband
      call fmatl
      call fload
      call fsolve
```

```
      return
      end
c ...
      SUBROUTINE fband
c ...
c .   fband : BAND and SETUP runstreams
c ...
         open(unit=12,status='unknown',file='tape12B.com')
      write(12,'('' $band'')')
      write(12,'(''open 6 "band.out"'')')
      write(12,'(''start -1 : regps : band : stop '')')
      write(12,'(''$setup'')')
      write(12,'(''open 6 "setup.out"'')')
      write(12,'(''start : setup : stop '')')
      close(unit=12)
      return
      end
      SUBROUTINE fmat1
c ...
c .   MATL runstream
c ...
      common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
      common /mat1/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1              SS(12,10,5),NT(10),NAME
      common /wall/ NLAY,MW(30),TW(30),PW(30)
      common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
      common /solve/ NBUC,TAR,NDF,ND,MO
      CHARACTER*40 fmt1,fmt2,fmt3,fmt4,fmt5,fmt6,fmt7,fmt8,fmt4a
      character NAME*80
      dimension NAME(10)
      fmt1='(''MATORT'',I3,3e11.4,3f10.4,'' >'')'
      fmt2='(9X,3e11.4,3f10.4)'
      fmt3='(''ORIENT'',I3,'' 0.,0.,0., 1'')'
c           12 3456789 0123456
      fmt7='(''MATALL'',I3,6f11.4,'' >'')'
      fmt8='(9x,6f11.4)'
      fmt4='(''WALLAM'',2I3,'' >'')'
      fmt4a='(I3,'' >'')'
      fmt5='(12X,  F8.4,'' >'')'
      fmt6='(12X,  F8.4)'
c ...
c .   fmat1 : DIAL MATL runstream
c ...
C        open(unit=12,status='unknown',file='tape12C')
         open(unit=12,status='unknown',file='tape12C.com')
c      write(12,'(''#'')')
c      write(12,'(''source /sgi423/dial/init.com'')')
c      write(12,'(''mat1 << \!'')')
      write(12,'('' $mat1'')')
      write(12,'(''open 6 "mat1.out"'')')
      write(12,'(''start'')')
      do 25 I=1,NMAT
      write(12,fmt1) I,(E1(K,I,1),K=1,3),(V1(K,I,1),K=1,3)
      write(12,fmt2)   (G1(K,I,1),K=1,3),(A1(K,I,1),K=1,3)
      write(12,fmt3) I
      write(12,fmt7) I,(SS(K,I,1),K=1,6)
   25 write(12,fmt8)   (SS(K,I,1),K=7,12)
c ...
      MM=10
c      write( fmt4(15:16), '(i2)') NLAY
```

```
c       write( fmt5(6:7), '(i2)') NLAY
c       write( fmt6(6:7), '(i2)') NLAY
        write(12,fmt4) MM,NLAY
        do 26 i=1,NLAY
           write(12,fmt4a)MW(I)
   26   continue
        do 27 i=1,NLAY
           write(12,fmt5)TW(I)
   27   continue
        do 28 i=1,NLAY-1
           write(12,fmt5)PW(I)
   28   continue
        write(12,fmt6)PW(NLAY)
c ...
        if (MTYP.eq.1.or.MTYP.eq.5) then
           IP=2
        elseif (MTYP.eq.2) then
           IP=4
        else
           IP=3
        endif
        do 30 I=1,IP
           MM= 10 + I
           write(12,fmt4) MM,MLAY(I)
           do 31 j=1,MLAY(I)
              write(12,fmt4a)MS(J,I)
   31      continue
           do 32 j=1,MLAY(I)
              write(12,fmt5)TS(J,I)
   32      continue
           do 33 j=1,MLAY(I)-1
              write(12,fmt5)PS(J,I)
   33      continue
           write(12,fmt6)PS(MLAY(I),I)
   30   continue
c ...
        write(12,'(''matl : stop'')')
c ...
        close(unit=12)
        return
        end
        SUBROUTINE fload
c ...
c .   LOAD runstream
c ...
        common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
        common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
       1              SS(12,10,5),NT(10),NAME
        common /wall/ NLAY,MW(30),TW(30),PW(30)
        common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
        common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
       1              LBC11,LBC12,LBC13,LBC14
        common /solve/ NBUC,TAR,NDF,ND,MO
c ...
c .   fload : DIAL LOAD runstream
c ...
        dimension NAME(10)
        character NAME*80
        real temp(6)
C          open(unit=12,status='unknown',file='tape12D')
```

```
        open(unit=12,status='unknown',file='tape12D.com')
c      write(12,'(''#'')')
c      write(12,'(''source /sgi423/dial/init.com'')')
c      write(12,'(''load << \!'')')
      write(12,'('' $load'')')
      write(12,'(''open 6 "load.out"'')')
      write(12,'('' start'')')
c ...
      RPdel = R + .1
      RMdel = R - .1
      write(12,'('' mset 11 copy nset=11'')')
      write(12,'('' mset 11 mask type msh'')')
      write(12,'('' mset 11 mask volu '',f8.4,1x,f8.4'' -180. 180.
     & -999. +999. 1'')') RMdel,RPdel
      write(12,'('' mset 12 copy nset=12'')')
      write(12,'('' mset 12 mask type msh'')')
      write(12,'('' mset 12 mask volu '',f8.4,1x,f8.4'' -180. 180.
     & -999. +999. 1'')') RMdel,RPdel
      write(12,'('' mset 13 copy nset=13'')')
C 9/10/91, inserted 1 line as follows, per Tony Wei
      write(12,'('' mset 13 mask type msh'')')
      write(12,'('' mset 14 copy nset=14'')')
C 9/10/91, inserted 1 line as follows, per Tony Wei
      write(12,'('' mset 14 mask type msh'')')
      write(12,'('' mset 15 copy type=msh'')')
      write(12,'('' mset 15 mask volu '',f8.4,1x,f8.4'' -180. 180.
     & -999. +999. 1'')') RMdel,RPdel
      temp(1)=FL(1)
      temp(2)=FL(2)
      temp(3)=FL(3)
      temp(4)=FL(4)
c           temp(5)=fl(3)/2.
c           temp(6)=temp(5)*(by/ax)
      if (LTOT .eq. 1) then
        temp(1)=temp(1)/AX
        temp(2)=temp(2)/BY
        temp(3)=temp(3)/AX
        tmp=temp(3)/BY
c         temp(5)=fl(3)/2.
c         temp(6)=temp(5)*(by/ax)
      endif
      if (LTOT .eq. 2) then
        tmp=temp(3)
      endif
      write(12,'('' lcase 1'')')
      write(12,'('' pline '',f10.2,'',11, 0,4 mset=11'')') -temp(1)
      write(12,'('' pline '',f10.2,'',11, 0,2 mset=12'')')  temp(1)
      write(12,'('' pline '',f10.2,'',2, 0,1 mset=13'')')  temp(2)
      write(12,'('' pline '',f10.2,'',2, 0,3 mset=14'')')  temp(2)
      write(12,'('' pline '',f10.2,'',3, 0,4 mset=11'')')  temp(3)
      write(12,'('' pline '',f10.2,'',3, 0,2 mset=12'')')  temp(3)
      write(12,'('' pline '',f10.2,'',3, 0,1 mset=13'')')  -tmp
      write(12,'('' pline '',f10.2,'',3, 0,3 mset=14'')')  -tmp
      write(12,'('' psurf '',f10.2,'',1, 0,mset=15'')') -FL(4)
c
c     p force cards apply concentrated loads to the panel corners
c
c      write(12,'('' p'',f10.2,'',2, node=1,17'')') temp(5)
c      write(12,'('' p'',f10.2,'',1, node=1'')') temp(6)
c      write(12,'('' p'',f10.2,'',1, node=17'')') -temp(6)
```

```
c      write(12,'('' p'',f10.2,'',2, node=173'')') -temp(5)
c ...
       write(12,'('' load : stop'')')
c      write(12,'(''!'')')
c ...
       close(unit=12)
       return
       end
       SUBROUTINE fsolve
c ...
c .    SOLVE runstream
c ...
       common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
       common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
      1              SS(12,10,5),NT(10),NAME
       common /wall/ NLAY,MW(30),TW(30),PW(30)
       common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
       common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
      1              LBC11,LBC12,LBC13,LBC14
       common /solve/ NBUC,TAR,NDF,ND,MO
       dimension NAME(10)
       character NAME*80
c ...
c .    fsolve : DIAL SOLVE runstream
c ...
100    format(' loads ',i2,f12.4)
C         open(unit=12,status='unknown',file='tape12E')
          open(unit=12,status='unknown',file='tape12E.com')
c      write(12,'(''#'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
       if (NBUC.eq.2) then
c         write(12,'('' solve << \!'')')
          write(12,'(''$assign [e565678.curve]fil002 fil002'')')
          write(12,'(''$solvedp'')')
          write(12,'('' open 6 "solve.out"'')')
          write(12,'('' start'')')
          write(12,'('' ASSIGN IMPR=1'')')
          write(12,100)1,1.
          write(12,'('' SAVE,D'')')
          write(12,'('' SAVE,S'')')
          write(12,'('' nprint,D'')')
          write(12,'('' eprint,S'')')
          write(12,'('' EIGEN 2'')')
          write(12,'('' solve'')')
          write(12,'('' stop'')')
c         write(12,'(''!'')')
c         write(12,'('' cluster << \!'')')
          write(12,'(''$clusterdp'')')
          write(12,'('' open 6 "cluster.out"'')')
          write(12,'('' start'')')
          write(12,'('' assign mxit=50'')')
          write(12,'('' shift '',i3)')MO
          write(12,'('' save'')')
          write(12,'('' matrix:file,ki ki'')')
          write(12,'('' clust 2'')')
          write(12,'('' stop'')')
c         write(12,'(''!'')')
       elseif(NBUC.eq.1) then
c         write(12,'('' solve << \!'')')
          write(12,'(''$assign [e565678.curve]fil002 fil002'')')
```

```
            write(12,'(''$solvedp'')')
            write(12,'('' open 6 "solve.out"'')')
            write(12,'('' start'')')
            write(12,'('' ASSIGN IMPR=1'')')
            write(12,100)1,1.
            write(12,'('' SAVE,D'')')
            write(12,'('' SAVE,S'')')
            write(12,'('' nprint,D'')')
            write(12,'('' eprint,S'')')
            write(12,'('' solve'')')
            write(12,'('' stop'')')
c            write(12,'(''!'')')
         elseif(NBUC.eq.3)then
c            write(12,'('' solve << \!'')')
            write(12,'(''$assign [e565678.curve]fil002 fil002'')')
            write(12,'('' $solvedp'')')
            write(12,'('' open 6 "solve.out"'')')
            write(12,'('' start -1'')')
            write(12,'('' ASSIGN IMPR=1'')')
            write(12,'('' static 2'')')
            write(12,'('' branch 0 1 1 '')')
            write(12,'('' loads 1 1. 1. 1.'')')
            write(12,'('' limit load'')')
            if (NF.ge.1) then
               write(12,'('' Target '',f12.5,'' Node= '',2i4)')TAR,NDF,ND
            else
               write(12,'('' Target'',f12.5)')TAR
            endif
            write(12,'('' save,d'')')
            write(12,'('' save,s'')')
            write(12,'('' assign slf=0.5, relr=0.02, stol=0.25, gtol=0.25''
     *)')
            write(12,'('' solve 1'')')
            write(12,'('' stop'')')
c            write(12,'('' !'')')
         endif
         close(unit=12)
         return
         end
         SUBROUTINE fmesh
c ...
c .   MESH runstream
c ...
         common /mesh/ M,N,MI,NI,AX,BY,R,A(10),B(10),NGRID
         common /matl/ NMAT,E1(3,10,5),V1(3,10,5),G1(3,10,5),A1(3,10,5),
     1                 SS(12,10,5),NT(10),NAME
         common /wall/ NLAY,MW(30),TW(30),PW(30)
         common /secb/ MTYP,XH,XB,XC,XT,MLAY(4),MS(30,4),TS(30,4),PS(30,4)
         common /load/ LARG,LEND,LRIB,LRBC,LEBC,LSBC,LTOT,FL(4),
     1                 LBC11,LBC12,LBC13,LBC14
         common /solve/ NBUC,TAR,NDF,ND,MO
         CHARACTER*60 fmt1,fmt2,fmt3,fmt4,fmt5,fmt6,fmt6b
         CHARACTER*60 fmt7,fmt8,fmt9,fmt0,fmta,fmtb,fmtc,fmtz,fmty,fmtz2
         CHARACTER*60 fmtd
         character NAME*80
         dimension NAME(10)
c.....................................................................
c   VARIABLE DEFINITION TABLE
c.....................................................................
c     M =
```

```
      fmt1='(''IJPOINT'',3I3,3f10.4,'' 1'')'
      fmt2='(''       '',3I3,3f10.4,'' 1'')'
      fmt3='(''SLINE'',6I3)'
      fmta='(''CIRCLE'',3I3)'
      fmt4='(''KNAME 0 0 '',2I3,''  kp'',I1)'
      fmt5='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-1,0,0,BASE'')'
      fmtb='(''MSET 10, COPY,NAME=BASE'')'
      fmt6='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-1,0,0'')'
      fmt6b='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-2,0,0'')'
      fmt7='(''PRISM'',I3,3f10.4)'
      fmt8='(''ROTATE'',f10.4,'' 1'')'
c      fmt9='(''KNAME 0 0 '',2I3,''  kb'',I1)'
      fmt9='(''KNAME '',4I3,''  kb'',I1)'
      fmt0='(''ELTNAME STF'',I1,'',STIF,,,,MESH=%IPP(9)'')'
      fmtc='(''MSET'',I3,'', COPY,ANAME=STF'',I1)'
      fmtz='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-1,0,0,part'')'
      fmtz2='(''IJSHELL'',2I3,4f10.4,I3,'' MSH,1,-2,0,0,part'')'
      fmty='(''BCSYS'',3I3,f10.4,'' 1.e+9,nset= '')'
      fmtd='(''DEFSYS 1 1 0.,0.,0. 0.,0.,-1. 0.,1.,-1. '')'
c ...
c .   fmesh : DIAL MESH runstream
c ...
C      open(unit=12,status='unknown',file='tape12A.')
        open(unit=12,status='unknown',file='tape12A.com')
c      write(12,'(''#'')')
c      write(12,'(''source /sgi423/dial/init.com'')')
c      write(12,'(''mesh << \!'')')
      write(12,'(''$mesh '')')
      write(12,'(''open 6 "mesh.out"'')')
      write(12,'(''clear -1'')')
      write(12,'(''eltype 4,2,6'')')
      write(12,'(''set syntax on'')')
      write(12,'(''assign IPNO=0 IPLC=0 IPEL=0 IPCO=0 TOLX=.01 '')')
      write(12,'(''# MESH Number 1, base panel'')')
c ...
c
c     system definition for cylindrical coor system
c
      write(12,fmtd)
c
      J0=1
      IP1=1
      JA1=1
      XA1=0.
      write(12,fmt1) IP1,JA1,J0,R,XA1,0.
      IP=2+5*M
      JA=1+8*M+2*M*(M+1)
      XA=AX
      TH=(XA/R)*180./3.1415927
      write(12,fmt2) IP,JA,J0,R,TH,0.
      do 10 L=1,M
      IP6=5*L+1
      IP5=5*L
      IP4=5*L-1
      IP3=5*L-2
      IP2=5*L-3
      JA6=1+2*(MI+4)*L
      JA5=1+2*(MI+4)*L-2
      JA4=1+2*(MI+4)*L-4
      JA3=1+2*(MI+4)*L-6
```

```
      JA2=1+2*(MI+4)*L-8
      if (MTYP.eq.2.or.MTYP.eq.8) then
c...   Global Y coor. for stiff. 2, or 8 .....
         XA6=A(1+L)+XB/2.
         XA5=A(1+L)+XC/2.
         XA4=A(1+L)
         XA3=A(1+L)-XC/2.
         XA2=A(1+L)-XB/2.
       elseif (MTYP.eq.5.or.MTYP.eq.6) then
c...   Global Y coor. for stiff. 5, or 6 .....
         XA6=A(1+L)+XB/2.
         XA5=A(1+L)+XB/4.
         XA4=A(1+L)
         XA3=A(1+L)-XB/4.
         XA2=A(1+L)-XB/2.
      elseif (MTYP.eq.7) then
c...   Global Y coor. for stiff. 7 .....
         XA6=A(1+L)+XB/2.
         XA5=A(1+L)+XH
         XA4=A(1+L)
         XA3=A(1+L)-XH
         XA2=A(1+L)-XB/2.
      else
c...   Global Y coor. for stiff. 1,2, or 3, 4,6 .....
         XA6=A(1+L)+XB-XC
         XA5=A(1+L)+(XB-XC)/2.
         XA4=A(1+L)
         XA3=A(1+L)-XC/2.
         XA2=A(1+L)-XC
      endif
c
c     Conversion to cylindrical coordinates
c
      TH2 = (XA2/R)*180./3.1415927
      TH3 = (XA3/R)*180./3.1415927
      TH4 = (XA4/R)*180./3.1415927
      TH5 = (XA5/R)*180./3.1415927
      TH6 = (XA6/R)*180./3.1415927
      write(12,fmt2) IP2,JA2,J0,R,TH2,0.
      write(12,fmt2) IP3,JA3,J0,R,TH3,0.
       write(12,fmt2) IP4,JA4,J0,R,TH4,0.
      write(12,fmt2) IP5,JA5,J0,R,TH5,0.
 10   write(12,fmt2) IP6,JA6,J0,R,TH6,0.
      Icenter=IP+1
      write(12,fmt2) Icenter,0,0,0.,0.,0.
c ...
      L1=1
      L2=2
      L3=3
      L4=4
      write(12,fmta) L1,L2,Icenter
      L1=M*5-1
      L2=M*5
      L3=M*5+1
      L4=M*5+2
      write(12,fmta) L3,L4,Icenter
      do 20 L=1,M-1
      L1=L*5-1
      L2=L*5
      L3=L*5+1
```

```
      L4=L*5+2
      L5=L*5+3
      L6=L*5+4
   20 write(12,fmta) L3,L4,Icenter
      L0=1
      L9=M*5+2
      T9=0
      do 25 L=1,NLAY
   25 T9=T9+TW(L)
      W9=0.
      M9=10
c ...
         i1=1
         i2=2
      do 26 i=1,M+1
         write(12,fmt5) i1,i2,T9,T9,W9,W9,M9
         i1=i1+5
         i2=i2+5
   26 continue
c ...
      do 30 L=1,N+1
      KB=L*2*NI+1
      XTT=0.
      L1=L+1
      YB=B(L1)
   30 write(12,fmt7) KB,YB
      do 33 L=1,N+2
      KB=L*2*NI-NI*2+1
   33 write(12,fmt4) KB,KB,L
      write(12,'('' msave'')')
      if (NBUC .eq. 2 .or. NBUC .eq. 3) then
         write(12,'('' refine 0 0 2 2 '')')
      endif
      write(12,'(''mesh'')')
c ...
c . Stiffener Type I.D.
c .  1: Blade   2: Hat
c .  3: I-shape 4: J-shape
c .  5: Angle   6: Zee
c .  7: Bead    8: Open Hat
c ...
   35 continue
      goto (110,120,130,140,150,160,170,180) mtyp
c ... Blade ...
  110 IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA3
      JB1=1
      JB6=3
      XY =0.
      XA1=A(2)-XC
```

```
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=XA3
      YA6=0.
      ZA6=XH
      TH1 = (XA1/R)*180./3.1415927
      TH2 = (XA2/R)*180./3.1415927
      TH3 = (XA3/R)*180./3.1415927
      TH4 = (XA4/R)*180./3.1415927
      TH5 = (XA5/R)*180./3.1415927
      TH6 = (XA3/R)*180./3.1415927
       R6 = R - XH
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt2)  7,0,0,0.,0.,0.
      write(12,fmta) 1,2,7
      write(12,fmta) 2,3,7
      write(12,fmta) 3,4,7
      write(12,fmta) 4,5,7
      write(12,fmt3) IP3,IP6
      M11=11
      M12=12
      YA7=0.
      T11=0.
      T12=0.
      do 51 i=1,MLAY(1)
         T11=T11+ts(i,1)
 51   continue
      do 52 i=1,MLAY(2)
         T12=T12+ts(i,2)
 52   continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      go to 190
c ... Closed Hat ...
120   IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      IP8=8
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA2
      JA7=JA3
      JA8=JA4
      JB1=1
      JB6=3
      XA1=A(2)-XB/2.
```

```
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+XC/2.
      XA5=A(2)+XB/2.
      XA6=A(2)-XT/2.
      XA7=A(2)
      XA8=A(2)+XT/2.
      TH1=(XA1/R)*180./3.1415927
      TH2=(XA2/R)*180./3.1415927
      TH3=(XA3/R)*180./3.1415927
      TH4=(XA4/R)*180./3.1415927
      TH5=(XA5/R)*180./3.1415927
      TH6=TH3 - (XT/(2*(R-XH)))*180./3.1415927
      TH7=(XA7/R)*180./3.1415927
      TH8=TH3 + (XT/(2*(R-XH)))*180./3.1415927
      YA6=0.
      ZA6=XH
       R6=R-XH
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt2) IP7,JA7,JB6,R6,TH7,0.
      write(12,fmt2) IP8,JA8,JB6,R6,TH8,0.
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP6,IP7,IP8
      write(12,fmt3) IP2,IP6
      write(12,fmt3) IP4,IP8
      M11=11
      M12=12
      M13=13
      M14=14
      YA7=0.
      YA8=0.
      T11=0.
      T14=0.
      T12=0.
      T13=0.
      do 53 i = 1,MLAY(1)
         t11=t11+ts(i,1)
53    continue
      do 54 i = 1,MLAY(2)
         t12=t12+ts(i,2)
54    continue
      do 55 i = 1,MLAY(3)
         t13=t13+ts(i,3)
55    continue
      do 56 i = 1,MLAY(4)
         t14=t14+ts(i,2)
56    continue
      write(12,fmt6) IP1,IP2,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP2,IP4,T14,T14,YA8,YA8,M14
      write(12,fmt6) IP4,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6b) IP2,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP8,T13,T13,YA6,YA6,M13
      write(12,fmtz) IP8,IP4,T12,T12,YA6,YA6,M12
      go to 190
c ... I-shape ...
```

```
130   IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      IP8=8
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA3
      JA7=JA2
      JA8=JA4
      JB1=1
      JB6=3
      XA1=A(2)-XC
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=A(2)
      XA7=A(2)-XT/2.
      XA8=A(2)+XT/2.
      ZA6=XH
      YA6=0.
      TH1=(XA1/R)*180./3.1415927
      TH2=(XA2/R)*180./3.1415927
      TH3=(XA3/R)*180./3.1415927
      TH4=(XA4/R)*180./3.1415927
      TH5=(XA5/R)*180./3.1415927
      TH6=(XA6/(R))*180./3.1415927
      TH7=th6-(xt/(2*(R-XH)))*180./3.1415927
      TH8=th6+(xt/(2*(R-XH)))*180./3.1415927
      R6=R-XH
      M11=11
      M12=12
      M13=13
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt2) IP7,JA7,JB6,R6,TH7,0.
      write(12,fmt2) IP8,JA8,JB6,R6,TH8,0.
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      write(12,fmt3) IP7,IP6
      write(12,fmt3) IP6,IP8
      YA7=0.
      T11=0.
      T12=0.
      T13=0.
      do 57 i=1,MLAY(1)
         t11=t11+ts(i,1)
57    continue
      do 58 i=1,MLAY(2)
```

```
         t12=t12+ts(i,2)
58     continue
       do 59 i=1,MLAY(3)
         t13=t13+ts(i,3)
59     continue
       write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
       write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
       write(12,fmt6) IP7,IP6,T13,T13,YA6,YA6,M13
       write(12,fmt6) IP6,IP8,T13,T13,YA6,YA6,M13
       go to 190
c ... J-shape ...
 140   IP1=1
       IP2=2
       IP3=3
       IP4=4
       IP5=5
       IP6=6
       IP7=7
       JA1=1
       JA2=3
       JA3=5
       JA4=7
       JA5=9
       JA6=JA3
       JA7=JA3
       JB1=1
       JB6=3
       JB7=5
       XA1=A(2)-XC
       XA2=A(2)-XC/2.
       XA3=A(2)
       XA4=A(2)+(XB-XC)/2.
       XA5=A(2)+XB-XC
       XA6=A(2)
       XA7=A(2)-XT
       ZA6=XH
       YA6=0.
       TH1=(XA1/R)*180./3.1415927
       TH2=(XA2/R)*180./3.1415927
       TH3=(XA3/R)*180./3.1415927
       TH4=(XA4/R)*180./3.1415927
       TH5=(XA5/R)*180./3.1415927
       TH6=(XA6/R)*180./3.1415927
       TH7=th6-(Xt/(2*(R-XH)))*180./3.1415927
        R6=R-XH
       M11=11
       M12=13
       M13=13
       write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
       write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
       write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
       write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
       write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
       write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
       write(12,fmt2) IP7,JA7,JB7,R6,TH7,0.
       write(12,fmt3) IP1,IP2,IP3,IP4,IP5
       write(12,fmt3) IP3,IP6
       write(12,fmt3) IP6,IP7
       YA7=0.
       T11=0.
```

```
      T12=0.
      T13=0.
      do 60 i=1,MLAY(1)
         t11=t11+ts(i,1)
60    continue
      do 61 i=1,MLAY(2)
         t12=t12+ts(i,2)
61    continue
      do 62 i=1,MLAY(3)
         t13=t13+ts(i,3)
62    continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP7,T13,T13,YA6,YA6,M13
      go to 190
c ... Angle ...
 150  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA3
      JB1=1
      JB6=3
      XY =0.
      XA1=A(2)-XC
      XA2=A(2)-XC/2.
      XA3=A(2)
      XA4=A(2)+(XB-XC)/2.
      XA5=A(2)+XB-XC
      XA6=XA3
      YA6=0.
      ZA6=XH
      TH1=(XA1/R)*180./3.1415927
      TH2=(XA2/R)*180./3.1415927
      TH3=(XA3/R)*180./3.1415927
      TH4=(XA4/R)*180./3.1415927
      TH5=(XA5/R)*180./3.1415927
      TH6=(XA6/R)*180./3.1415927
       R6=R-XH
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      M11=11
      M12=12
      YA7=0.
      T11=0.
      T12=0.
      do 63 i= 1,MLAY(1)
```


C-3

```
         t11=t11+ts(i,1)
 63    continue
       do 64 i= 1,MLAY(2)
         t12=t12+ts(i,2)
 64    continue
       write(12,fmt6) IP1,IP3,T11,T11,YA7,YA7,M11
       write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
       write(12,fmt6) IP3,IP5,T9,T9,YA7,YA7,M9
       go to 190
c ... Zee-shape ...
 160   IP1=1
       IP2=2
       IP3=3
       IP4=4
       IP5=5
       IP6=6
       IP7=7
       JA1=1
       JA2=3
       JA3=5
       JA4=7
       JA5=9
       JA6=JA3
       JA7=JA3
       JB1=1
       JB6=3
       JB7=5
       XA1=A(2)-XB/2.
       XA2=A(2)-XB/4.
       XA3=A(2)
       XA4=A(2)+XB/4.
       XA5=A(2)+XB/2.
       XA6=A(2)
       XA7=A(2)-XT
       ZA6=XH
       YA6=0.
       TH1=(XA1/R)*180./3.1415927
       TH2=(XA2/R)*180./3.1415927
       TH3=(XA3/R)*180./3.1415927
       TH4=(XA4/R)*180./3.1415927
       TH5=(XA5/R)*180./3.1415927
       TH6=(XA6/R)*180./3.1415927
       TH7=(XA7/(R-XH))*180./3.1415927
        R6=R-XH
      M11=11
      M12=12
      M13=13
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt2) IP7,JA7,JB7,R,TH7,0.
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP3,IP6
      write(12,fmt3) IP6,IP7
      YA7=0.
      T11=0.
      T12=0.
```

```
      T13=0.
      do 65 i=1,MLAY(1)
         t11=t11+ts(i,1)
65    continue
      do 66 i=1,MLAY(2)
         t12=t12+ts(i,2)
66    continue
      do 67 i=1,MLAY(3)
         t13=t13+ts(i,3)
67    continue
      write(12,fmt6) IP1,IP3,T9,T9,YA7,YA7,M9
      write(12,fmt6) IP3,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP3,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP7,T13,T13,YA6,YA6,M13
      go to 190
c ... Bead ...
 170  IP1=1
      IP2=2
      IP3=3
      IP4=4
      IP5=5
      IP6=6
      IP7=7
      IP8=8
      JA1=1
      JA2=3
      JA3=5
      JA4=7
      JA5=9
      JA6=JA2
      JA7=JA3
      JA8=JA4
      JB1=1
      JB6=3
      XA1=A(2)-XB/2.
      XA2=A(2)-XH
      XA3=A(2)
      XA4=A(2)+XH
      XA5=A(2)+XB/2.
      XA6=A(2)-0.707*XH
      XA7=A(2)
      XA8=A(2)+0.707*XH
      YA6=0.
      ZA6=0.707*XH
      ZA7=XH
      ZA8=0.707*XH
       R6=R -ZA6
       R7=R -ZA7
       R8=R -ZA8
      TH1=(XA1/R)*180./3.1415927
      TH2=(XA2/R)*180./3.1415927
      TH3=(XA3/R)*180./3.1415927
      TH4=(XA4/R)*180./3.1415927
      TH5=(XA5/R)*180./3.1415927
      TH6=th3-(zA6/R6)*180./3.1415927
      TH7=(XA7/R)*180./3.1415927
      TH8=th3+(zA8/R8)*180./3.1415927
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
```

```
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt2) IP7,JA7,JB6,R7,TH7,0.
      write(12,fmt2) IP8,JA8,JB6,R8,TH8,0.
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmta) IP2,IP6,IP3
      write(12,fmta) IP6,IP7,IP3
      write(12,fmta) IP7,IP8,IP3
      write(12,fmta) IP8,IP4,IP3
      M11=11
      M12=12
      M13=13
      YA7=0.
      YA8=0.
      T11=0.
      T12=0.
      T13=0.
      do 68 i=1,MLAY(1)
         t11=t11+ts(i,1)
68    continue
      do 69 i=1,MLAY(2)
         t12=t12+ts(i,2)
69    continue
      do 70 i=1,MLAY(3)
         t13=t13+ts(i,3)
70    continue
      write(12,fmt6) IP1,IP2,T11,T11,YA7,YA7,M11
      write(12,fmt6) IP2,IP4,T13,T13,YA8,YA8,M13
      write(12,fmt6) IP4,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6b) IP2,IP6,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP6,IP8,T12,T12,YA6,YA6,M12
      write(12,fmtz) IP8,IP4,T12,T12,YA6,YA6,M12
      go to 190
c..... Open Hat ....
 180  IP1 = 1
      IP2 = 2
      IP3 = 3
      IP4 = 4
      IP5 = 5
      IP6 = 6
      IP7 = 7
      IP8 = 8
      IP9 = 9
      JA1 = 1
      JA2 = 3
      JA3 = 5
      JA4 = 7
      JA5 = 9
      JA6 = JA2
      JA7 = JA1
      JA8 = JA4
      JA9 = JA5
      JB1 = 1
      JB6 = 3
      XA1 = A(2) - XB/2.
      XA2 = A(2) - XC/2.
      XA3 = A(2)
      XA4 = A(2) + XC/2.
      XA5 = A(2) + XB/2.
```

```
      XA6 = A(2) - XC/2.
      XA7 = A(2) - XT/2.
      XA8 = A(2) + XC/2.
      XA9 = A(2) + XT/2.
      YA6 = 0.
      ZA6 = XH
      R6=R-XH
      TH1=(XA1/R)*180./3.1415927
      TH2=(XA2/R)*180./3.1415927
      TH3=(XA3/R)*180./3.1415927
      TH4=(XA4/R)*180./3.1415927
      TH5=(XA5/R)*180./3.1415927
      TH6=th3-(Xc/(2*R6))*180./3.1415927
      TH7=th3-(Xt/(2*R6))*180./3.1415927
      TH8=th3+(Xc/(2*R6))*180./3.1415927
      TH9=th3+(Xt/(2*R6))*180./3.1415927
      write(12,fmt1) IP1,JA1,JB1,R,TH1,0.
      write(12,fmt2) IP2,JA2,JB1,R,TH2,0.
      write(12,fmt2) IP3,JA3,JB1,R,TH3,0.
      write(12,fmt2) IP4,JA4,JB1,R,TH4,0.
      write(12,fmt2) IP5,JA5,JB1,R,TH5,0.
      write(12,fmt2) IP6,JA6,JB6,R6,TH6,0.
      write(12,fmt2) IP7,JA7,JB6,R6,TH7,0.
      write(12,fmt2) IP8,JA8,JB6,R6,TH8,0.
      write(12,fmt2) IP9,JA9,JB6,R6,TH9,0.
      write(12,fmt3) IP1,IP2,IP3,IP4,IP5
      write(12,fmt3) IP2,IP6
      write(12,fmt3) IP6,IP7
      write(12,fmt3) IP4,IP8
      write(12,fmt3) IP8,IP9
      M11 = 11
      M12 = 12
      M13 = 13
      YA7=0.
      T11=0.
      T12=0.
      T13=0.
      do 71 i=1,MLAY(1)
         t11=t11+ts(i,1)
71    continue
      do 72 i=1,MLAY(2)
         t12=t12+ts(i,2)
72    continue
      do 73 i=1,MLAY(3)
         t13=t13+ts(i,3)
73    continue
      write(12,fmt6) IP1,IP5,T11,T11,YA7,YA7,M11
      write(12,fmt6b) IP2,IP6,T12,T12,YA6,YA6,M12
      write(12,fmtz) IP8,IP4,T12,T12,YA6,YA6,M12
      write(12,fmt6) IP7,IP6,T13,T13,YA6,YA6,M13
      write(12,fmt6) IP8,IP9,T13,T13,YA6,YA6,M13
      go to 190
 190  continue
      YA7=0.
      YA8=0.
c ...
c . ditto 1st stiffener
c ...
      do 192 L=1,N+1
        KB=L*2*NI+1
```

```
      XTT=0.
      L1=L+1
    YB=B(L1)
192 write(12,fmt7) KB,YB
    do 193 L=1,N+2
       KB=L*2*NI-2*NI+1
       if (MTYP .eq. 1 .or. MTYP .eq. 5) then
          NP1=6
          NP2=6
          write(12,fmt9) NP1,NP2,KB,KB,L
          NP1=1
          NP2=5
          write(12,fmt9) NP1,NP2,KB,KB,L
       elseif (MTYP .eq. 2) then
          NP1=6
          NP2=8
          write(12,fmt9) NP1,NP2,KB,KB,L
          NP1=1
          NP2=2
          write(12,fmt9) NP1,NP2,KB,KB,L
          NP1=4
          NP2=5
          write(12,fmt9) NP1,NP2,KB,KB,L
       elseif (MTYP .eq. 3) then
          NP1=6
          NP2=7
          NP3=8
          write(12,fmt9) NP1,NP2,KB,KB,L
          write(12,fmt9) NP1,NP3,KB,KB,L
          NP1=1
          NP2=5
          write(12,fmt9) NP1,NP2,KB,KB,L
       elseif (MTYP .eq. 4 .or. 6) then
          NP1=6
          NP2=7
          write(12,fmt9) NP1,NP2,KB,KB,L
          NP1=1
          NP2=5
          write(12,fmt9) NP1,NP2,KB,KB,L
       elseif (MTYP .eq. 7) then
          NP1=7
          NP2=7
          write(12,fmt9) NP1,NP2,KB,KB,L
          NP1=1
          NP2=2
          write(12,fmt9) NP1,NP2,KB,KB,L
          NP1=4
          NP2=5
          write(12,fmt9) NP1,NP2,KB,KB,L
       elseif (MTYP .eq. 8) then
          NP1=6
          NP2=7
          NP3=8
          NP4=9
          write(12,fmt9) NP1,NP2,KB,KB,L
          write(12,fmt9) NP3,NP4,KB,KB,L
          NP1=1
          NP2=5
          write(12,fmt9) NP1,NP2,KB,KB,L
       endif
```

```
 193  continue
      write(12,'('' msave'')')
      if (NBUC .eq. 2 .or. NBUC .eq. 3) then
         write(12,'('' refine 0 0 2 2'')')
      endif
      write(12,'(''mesh'')')
      write(12,'('' eltsen 0 1 0 '')')
      if (MTYP.eq.2.or.MTYP.eq.7.or.MTYP.eq.8) then
         write(12,'('' eltsen 0 1 0 aname=part'')')
      endif
      L=1
      write(12,fmt0) L
      write(12,fmtc) L,L
      write(12,'(''merge'')')
      if (M .le. 1) go to 199
      do 195 L=1,M
      if (L .le. 1) go to 195
      write(12,'(''ditto  MESH=2'')')
      DTH= ( A(L+1)/R - A(2)/R )*180/3.1415927
      WRITE(*,*) A(L+1),A(2),DTH
      write(12,fmt8) DTH
      write(12,fmt0) L
      write(12,fmtc) L,L
      write(12,'(''merge'')')
 195  continue
 199  continue
c ...
c  CONO SUBROUTINE
c ...
      write(12,'(''mset 10 insert type msh'')')
      write(12,'(''mset 10 del name=base'')')
      write(12,'(''set syntax on'')')
      write(12,'(''####################'')')
      write(12,'(''# CONODE subroutine'')')
      write(12,'(''####################'')')
      write(12,'('' sub CONO &lst1 &lst2 &lst3'')')
      write(12,'('' set echo on'')')
      write(12,'('' let &nnod=%ipp(1)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst2)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n1=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst3)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n2=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst1)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' ;f2 format ''''(1x,i5)'')')
      write(12,'('' ;f3 format ''''(''''Print Nodal Coordinates'''',1x,'
     *'''Node'''')'')')
      write(12,'('' write 6 ;f2 &n1'')')
      write(12,'('' write 6 ;f2 &n2'')')
      write(12,'('' write 6 ;f3'')')
      write(12,'('' do ;20 &n=1,&nnod'')')
      write(12,'('' if %ibcl(&ln,&n) ;20,;20,1'')')
      write(12,'('' let &nn=%ibcl(&ln,&n)'')')
      write(12,'('' write 6 ;f2 &nn'')')
      write(12,'('' conode/isys=1 &nn &n1 &n2 -2 1.e+9'')')
      write(12,'('' conode/isys=1 &nn &n1 &n2 -3 1.e+9'')')
```

```
      write(12,'('' ;20 continue'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' return'')')
      write(12,'('' end'')')
c ...
c . boundary conditions definition
c ...
        xl=  0.00
        x2=  9999.
        yl=  -180.
        y2=  +180.
        zl=  -.001
        z2=   .001
      write(12,'(''nset 11 copy volu '',6f10.3,'' 1'')')
     1           xl,x2,yl,y2,zl,z2
        xl=  0.0
        x2=  9999.
        yl=  -180.
        y2=  +180.
        zl=  -.001 + BY
        z2=  +.001 + BY
      write(12,'(''nset 12 copy volu '',6f10.3,'' 1'')')
     1            xl,x2,yl,y2,zl,z2
        xl=  0.0
        x2=  9999.
        yl=  -.001
        y2=  +.001
        zl=  -999.
        z2=   999.
      write(12,'(''nset 13 copy volu '',6f10.3,'' 1'')')
     1            xl,x2,yl,y2,zl,z2
C
        TH = AX/R*180./3.1415927
C
        xl=  0.0
        x2=  9999.
        yl=  -.001 + TH
        y2=  +.001 + TH
        zl=  -999.
        z2=   999.
      write(12,'(''nset 14 copy volu '',6f10.3,'' 1'')')
     1            xl,x2,yl,y2,zl,z2
c
      xl= 0.0
      x2= 9999.
      yl=-0.001
      y2=+0.001
      zl=-0.001 +BY
      z2=+0.001 +BY
      write(12,'('' nlist 11 insert volu '',6f9.3,'' 1'')')
     1             xl,x2,yl,y2,zl,z2
c...  right front corner
      xl= 0.0
      x2= 9999.
      yl= -.001 + TH
      y2= +.001 + TH
      zl=-0.001 +BY
      z2=+0.001 +BY
      write(12,'('' nlist 12 insert volu '',6f9.3,'' 1'')')
     1            xl,x2,yl,y2,zl,z2
```

```
c...  left rear corner
      x1=0.0
      x2=9999.
      y1= -.001
      y2= +.001
      z1= -.001
      z2= +.001
      write(12,'('' nlist 13 insert volu '',6f9.3,'' 1'')')
     1              x1,x2,y1,y2,z1,z2
c...  right rear corner
      x1=0.0
      x2=9999.
      y1=-.001 + TH
      y2=+.001 + TH
      z1=-.001
      z2=+.001
      write(12,'('' nlist 14 insert volu '',6f9.3,'' 1'')')
     1              x1,x2,y1,y2,z1,z2
      write(12,'('' nset 11 del nlist 11 12 13 14'')')
      write(12,'('' nset 12 del nlist 11 12 13 14'')')
      write(12,'('' nset 13 del nlist 11 12 13 14'')')
      write(12,'('' nset 14 del nlist 11 12 13 14'')')
      write(12,'('' nlist 21 insert nset 11 '')')
      write(12,'('' nlist 22 insert nset 12 '')')
      write(12,'('' nlist 23 insert nset 13 '')')
      write(12,'('' nlist 24 insert nset 14 '')')
      NSET1=11
      NSET2=12
      NSET3=13
      NSET4=14
c...
      write(12,'('' call cono 22 11 12 '')')
      write(12,'('' call cono 21 13 14 '')')
C
      write(12,'('' nset 11 insert nlist 13 14'')')
      write(12,'('' nset 12 insert nlist 11 12'')')
      write(12,'('' nset 13 insert nlist 11 13'')')
      write(12,'('' nset 14 insert nlist 12 14'')')
C
c...   Shell normals and skew local coordinate system
c
       del = .1
      RMdel = R - del
      RPdel = R + del
      write(12,'('' mset 1015 copy type=msh'')')
      write(12,'('' mset 1015 mask volu '',f8.4,1x,f8.4'' -180. 180.
     & -999. +999. 1'')') RMdel,RPdel
      write(12,'('' nset 1015 insert mset=1015'')')
c
       write(12,'('' shlnor'')')
       write(12,'('' nodske loca 0 nset=1015'')')
c
       write(12,'('' nset 1011 insert nset=11'')')
       write(12,'('' nset 1012 insert nset=12'')')
       write(12,'('' nset 1011 mask nset=1015'')')
       write(12,'('' nset 1012 mask nset=1015'')')
c
       if(lbcll.eq.1) then
         write(12,'('' dofsup 3 nset=1011'')')
       elseif(lbcll.eq.2) then
```

```
          write(12,'('' dofsup/dnf 35 nset=1011'')')
        endif
c
        if(lbc12.eq.1) then
          write(12,'('' dofsup 3 nset=1012'')')
        elseif(lbc12.eq.2) then
          write(12,'('' dofsup/dnf 35 nset=1012'')')
        endif
c
        if(lbc13.eq.1) then
          write(12,'('' dofsup 3 nset=13'')')
        elseif(lbc13.eq.2) then
          write(12,'('' dofsup/dnf 34 nset=13'')')
        endif
c
        if(lbc14.eq.1) then
          write(12,'('' dofsup 3 nset=14'')')
        elseif(lbc14.eq.2) then
          write(12,'('' dofsup/dnf 34 nset=14'')')
        endif
c
c Kinematic constraints
c
          write(12,'('' dofsup/dnf 123 nlist=13'')')
          write(12,'('' dofsup 1 nlist=14'')')
c          write(12,'('' bcsys 1  0 nlist=13'')')
c          write(12,'('' bcsys -1 0 nlist=14'')')
c
c Rib boundary conditions (Frame)
c
        NB1=1
        NB2=N+2
        if (LEND .ne. 1) NB1=2
        if (LEND .ne. 1) NB2=N+1
        do 210 IB=NB1,NB2
       if (LRIB .eq. 3) then
         if (LRBC .eq. 1) then
          write(12,'('' dofsup 3 aname=kp'',I1)') IB
          write(12,'('' dofsup 3 aname=kb'',I1)') IB
         elseif (LRBC .eq. 2) then
          write(12,'('' dofsup/dnf 35 aname=kp'',I1)') IB
          write(12,'('' dofsup/dnf 35 aname=kb'',I1)') IB
         endif
       elseif (LRIB .eq. 1) then
         if (LRBC .eq. 1) then
          write(12,'('' dofsup 3 aname=kp'',I1)') IB
         elseif (LRBC .eq. 2) then
          write(12,'('' dofsup/dnf 35 aname=kp'',I1)') IB
         endif
       elseif (LRIB .eq. 2) then
         if (LRBC .eq. 1) then
          write(12,'('' dofsup 3 aname=kb'',I1)') IB
         elseif (LRBC .eq.2) then
          write(12,'('' dofsup/dnf 35 aname=kb'',I1)') IB
           endif
       endif
  210 continue
c...
c     Corner node constraints to stablize the panel.
c...
```

```
 999   continue
      write(12,'(''finish : stop'')')
c...
      close(unit=12)
      return
      end
c####################
c NTOKEN -- utility I/O subroutine
c  written by Dan Rickhoff LMSC 81-12;
c  with modification by L. L. Chou LMSC 81-12.
c####################
      subroutine nreal(ntk,ra)
      parameter ( maxtkn = 25 )
      character a*127, tkns(maxtkn)*10
      dimension lentk(maxtkn),ra(maxtkn)
c
  100   ierr=0
      call getlin( 5, a, irdata, *910 )
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
c       - - - - -
        do 200 i=1,ntk
        r = ra(i)
        call rintrp( tkns(i), lentk(i), r, *991 )
        ra(i) = r
        go to 200
  991     ierr=ierr+1
          write(6,'( 19x, ''Not a real '')')
  200 continue
      if (ierr .ge. 1) go to 100
      return
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
c
      subroutine nintg(ntk,ia)
      parameter ( maxtkn = 25 )
      character a*127, tkns(maxtkn)*10
      dimension lentk(maxtkn),ia(maxtkn)
c     ---------
  100 ierr=0
      call getlin( 5, a, irdata, *910 )
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
c
      do 200 i=1,ntk
        intgr = ia(i)
        call iintrp( tkns(i), lentk(i), intgr, *992 )
c        write(6,'(9x, ''integer = '', i10)') intgr
        ia(i) = intgr
        go to 200
  992     ierr=ierr+1
          write(6,'( 19x, ''Not an integer '')')
  200 continue
      if ( ierr .ge. 1) go to 100
      return
c     ---------
  999 continue
      write(6,'('' ERROR return from getkns'')')
```

```
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
c
      subroutine stitle(sline)
c
      character a*127, tkns(1)*60, sline*(*)
      dimension lentk(1)
c     ---------
      call getlin( 5, a, irdata, *910 )
      ntk = 1
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
      sline=tkns(1)(1:lentk(1))
      return
c
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
c
c ********************************************************************
c
      subroutine rintrp( tkn, lentk, r, * )
c
c#  Read a real number, R, from a character string, TKN, of length LENTK.
c#  (Note that blanks in the strings are ignored. Leading and/or trailing
c#   blanks are ignored, which is OK, but interior blanks are squeezed out,
c#   which is not OK; try to fix this sometime.)
c
      character tkn*(*)
c
      if ( lentk .eq. 0 ) return
c
      read( tkn(1:lentk), '(bn,f80.0)', err=990 ) r
c
      return
c
c -----------
  990 continue
      return 1
c
      end
c
c ********************************************************************
c
      subroutine iintrp( tkn, lentk, i, * )
c
c#  Read a integer number, I, from a character string, TKN, of length LENTK.
c#  (Note that blanks in the strings are ignored. Leading and/or trailing
c#   blanks are ignored, which is OK, but interior blanks are squeezed out,
c#   which is not OK; try to fix this sometime.)
c
      character tkn*(*)
c
      if ( lentk .eq. 0 ) return
c
      read( tkn(1:lentk), '(bn,i80)', err=990 ) i
```

```
c
      return
c
c -----------
  990 continue
      return 1
c
      end
c
c ****************************************************************
c
      subroutine getkns( a, irdata, tkns, lentk, ntk, * )
c
c ---------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     a ....... The given character string containing the input line.
c#                     If this string is blank (input IRDATA = 0) then all
c#                     NTK returned tokens will be blank with zero length.
c
c#    in     irdata .. The input line in string A extends to this character
c#                     position.
c
c#    out    tkns .... The character array of tokens.  The character lengths
c#                     of these array elements limit the lengths of input
c#                     tokens; tokens are correctly parsed but then truncated
c#                     to this length.
c
c#    out    lentk ... An integer array with the string lengths of the tokens
c#                     (after any necessary truncation).
c
c#    in     ntk ..... The number of tokens to be parsed from the line.
c
c#    -      * ....... Alternate RETURN in the event that a string token
c#                     initiated by an apostrophy had no trailing apostrophy, or
c#                     the next character after the trailing apostrophy was not
c#                     a delimiter (or end-of-line).
c#                     NOTE: The arguments that had been interpreted up to the
c#                           point that the error occured are returned; the
c#                           will be blank.
c
c ---------------------------------------------------------------
c#  PURPOSE:
c
c#  Parses an "input line" contained in a given string to find the first NTK
c#  tokens on the line.
c
c#  Tokens are delimited by beginning-of-line, commas, spaces, tabs, or
c#  end-of-line.  Tabs are equivalent to (single) spaces.
c
c#  A single asterisk (*) input token is recognized as a "null" token, for
c#  which the retuned string is blank and returned length is zero.  Some users
c#  may find entry of the null token preferable to that of a comma with no
c#  preceeding token, or two adjacent apostrophies (see the next
c#  paragraph).
c
c#  If a token is enclosed within apostrophies then it may contain commas,
c#  spaces, or tabs.  The returned length of such a token is the number
c#  number of character positions between the pair of apostrophies.
c#  Note that, for this length calculation, tabs are a single character.
```

```
c#  Contigous alphanumeric strings (no embedded delimiters) do not need to be
c#  enclosed within apostrophies.
c
c#  There is no provision for including apostrophies (') or the comment
c#  character (the hash symbol, #; see routine GETLIN) within a token,
c#  i.e., they may not be "escaped".  Strings enclosed within apostrophies
c#  may contain asterisks; these will not be misinterpreted as
c#  null tokens.
c
c#  A line with only "null" data will minimally contain (before an optional
c#  comment) either a null character, a comma or two adjacent apostrophies.
c
c#  Tokens are placed, left justified, into the character array TKNS.
c#  The tokens may be truncated in order to fit into the given TKNS array.
c
c#  The lengths of the returned tokens are placed in the LENTK array.
c
c -------------------------------------------------------------
c
c
c -------------------------------------------------------------
c#       Arrays for the returned tokens and their correcponding
c#       character lengths.
c -------------------------------------------------------------
c
      character tkns(*)*(*)
      dimension lentk(*)
c
c -------------------------------------------------------------
c#       Strings for the input line, the "curent" character on the
c#       line and delimiter characters (space, tab, and comma).
c -------------------------------------------------------------
c
      character a*(*), c*1, tab*1, space*1, comma*1, apos*1, null*1
c
      save  space      , comma      , apos       , null
      data  space /' '/, comma /','/, apos /''''/, null /'*'/
c
      tab = char(9)
c
      maxlen = len( tkns(1) )
c
c -------------------------------------------------------------
c#                Initialize ntk tokens to blanks.
c -------------------------------------------------------------
c
      do 50 i=1,ntk
        tkns(i) = ' '
        lentk(i) = 0
   50 continue
c
      if ( irdata .eq. 0 ) return
c
c -------------------------------------------------------------
c#       Something is on the line (before any comment).
c#       Parse the tokens (up to NTKN of them).
c -------------------------------------------------------------
c
      is = 0
      itkn = 1
```

```
c
  200 continue
c       # Move along the line looking for a token or a comma (null token).
      is = is + 1
      c = a(is:is)
c
c     ---------
      if ( c.eq.tab .or. c.eq.space ) then
c         # Move to next character; to continue looking for token.
        go to 200
c     ---------
      else if ( c.eq.comma ) then
c         # No token found before this comma indicates null token.
        if ( itkn .eq. ntkn ) return
        itkn = itkn + 1
c
        if ( is .eq. irdata ) return
c         # Move to next character; look for next token.
        go to 200
c     ---------
      else
  300   continue
c         # We are at the left-most character of a non-null token.
        il = is
c
c       - - - - -
        if ( c .eq. apos ) then
c         # We look for the trailing apostrophy for this "quoted string".
          if ( is .eq. irdata ) return 1
  450     continue
          is = is + 1
          c = a(is:is)
          if ( c .eq. apos ) then
            length = is - il - 1
            if ( length .gt. maxlen ) then
              length = maxlen
            endif
            lentk( itkn ) = length
            if ( length .gt. 0 ) tkns( itkn ) = a( il+1: il+length )
            if ( is .eq. irdata ) return
            is = is + 1
            c = a(is:is)
            if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) )
     *        return 1
          else
            if ( is .eq. irdata ) return 1
            go to 450
          endif
c       - - - - -
        else
  400     continue
          if ( is .eq. irdata ) then
c           # The token extends to the end of the input data.
            if ( a(il:is) .eq. null ) return
            tkns( itkn ) = a(il: is)
c             # The input token length may exceed the length of the variable.
            length = is - il + 1
            if ( length .le. maxlen ) then
              lentk( itkn ) = length
            else
```

```
                lentk( itkn ) = maxlen
              endif
              return
            endif
c
c             # Move along the line looking for the end of the token.
            is = is + 1
            c = a(is:is)
c
            if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) ) then
              if ( c .eq. apos ) return 1
c               # Move to next character; continue looking for end of token.
              go to 400
            endif
c
c             # The previous character was the right-most character of the token
            if ( a(il: is-1) .ne. null ) then
              tkns( itkn ) = a(il: is-1)
c               # The input token length may exceed the length of the variable.
              length = is - il
              if ( length .le. maxlen ) then
                lentk( itkn ) = length
              else
                lentk( itkn ) = maxlen
              endif
            endif
c         - - - - - -
          endif
c
          if ( itkn .eq. ntkn ) return
c           # Find the next location at which we should begin looking for the
c           # next token, i.e., after a following comma or at the next token.
          itkn = itkn + 1
          if ( is .eq. irdata ) return
          if ( c .eq. comma ) then
c             # Go back and look for a token following this comma.
            go to 200
          else
c             # Curent character was a space or a tab.
c             # Look for a comma, the next token, or end of line.
  500       continue
            is = is + 1
            c = a(is:is)
            if ( c .eq. comma ) then
              if ( is .eq. irdata ) return
              go to 200
            else if ( c.eq.space .or. c.eq.tab ) then
              if ( is .eq. irdata ) return
              go to 500
            else
              go to 300
            endif
          endif
c       ---------
        endif
c
        end
c
c ***************************************************************************
c
```

```
      subroutine getlin( nunit, a, irdata, * )
c
c ---------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     nunit ... The Fortran logical unit number of the file from which
c#                     the input line(s) should be read.
c
c#    out    a ..... A character string into which the input line will be
c#                     read.  The length of this given string determines how
c#                     long the data line may be; input past this length is
c#                     ignored.
c
c#    out    irdata .. The input "data" in string A extends to this character
c#                     position.  The data portion does not include any
c#                     trailing comment (if present).
c
c#    -      * ..... Alternare RETURN in the event that End-of-File is
c#                     encountered on attempt to read a line from unit NUNIT.
c
c ---------------------------------------------------------------
c#  PURPOSE:
c
c#  Reads line(s) from unit NUNIT and determines the location, IRDATA, in the
c#  string beyond which there may be only spaces, tabs, or a comment.  Comments
c#  begin with a hash symbol (#).  Blank lines and lines that contain only a
c#  comment line are ignored and the next line is read until a line with some
c#  data or End-of-File is encountered.  ("Some data" might minimally be a
c#  single comma, an asterisk, or two adjacent apostrophies.)
c
c ---------------------------------------------------------------
c
      character a*(*), comnt*1
      save comnt
      data comnt /'#'/
c
c ---------------------------------------------------------------
c#                 Read the input "line" from disk.
c ---------------------------------------------------------------
c
  100 continue
      read( nunit, '(a)', end=910 ) a
c
c#          We want to ignore any trailing comment and lines
c#          that are blank except for a comment (if any).
c
      icom = index( a, comnt )
c
c  # Location of right-most non-blank before comment (if any).
      if ( icom .gt. 1 ) then
        irdata = iright( a(1: icom-1) )
      else if ( icom .eq. 1 ) then
        go to 100
      else
        irdata = iright( a )
      endif
c
c       # If only blanks before comment (if present) then read next line.
      if (irdata .eq. 0) go to 100
c
```

```
      return
c
c -------- RETURN on End-of-File ------------------------
  910 continue
      return 1
c
      end
c
c ********************************************************************
c
      function ileft( string )
c
c  WRITTEN by:  D. T. Rickhoff
c
c ------------------------------------------------
c  Find the position of the leftmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c                                                                    ---
c ------------------------------------------------
c
      character*(*) string
      character*1 tab
c
      tab = char(9)
c
c --------------------------------
c  Find the location of the first nonblank character.
c
      ls = len( string )
c
      do 100 i=1,ls
        if(      string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          ileft = i
          return
        endif
  100 continue
c
c --------------------------------
c  String was blank.  Set ileft to zero (as a flag).
      ileft = 0
c
      return
      end
c
c ********************************************************************
c
      function iright( string )
c
c  WRITTEN by:  D. T. Rickhoff
c
c ------------------------------------------------
c  Find the position of the rightmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c                                                                    ---
c ------------------------------------------------
c
      character*(*) string
```

```
      character*1 tab
c
      tab = char(9)
c
c --------------------------------
c  Find the location of the last nonblank character.
c
      ls = len( string )
c
      do 100 i=ls,1,-1
        if(      string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          iright = i
          return
        endif
  100 continue
c
c --------------------------------
c  String was blank.  Set iright to zero (as a flag).
      iright = 0
c
      return
      end
C -----------  End of SP910.FOR
```

A3.3.3 TUBULAR TRUSS CORE PANEL MODULE # 3 PROGRAM

```
C 2/28/92  TUB212.FOR
C
C        Modified by JTH, LASC,
C        from TUB.FOR (9/4/91) by AYW, LMSC
C
C        Changes made to subroutines TMESH, TMAT and TSOLVE, etc.
C
      PROGRAM TUB
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/  NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c     *              T(3,6),C(3,6),SH(3,6),F(3,6)
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *               CT(99),AT(99),AB(99),AC(99)
      common /load/  XL(4),SL(4),XE,XS,SE,SS,TMPD
      common /name/  nmt
c ... ***************************************************
c     *  Advanced Composite Structural Program Input     *
c     *  variables definition ...                        *
c     *                                                  *
c     *   [ Geometry data ]                              *
c     *   A= bottom width of the tube                    *
c     *   B= top width of the tube                       *
c     *   PH= hight of panel                             *
c     *   PL= length of panel                            *
c     *   NW = number of tubes (whole tubes)             *
c     *   NBC = boundary condition indication            *
c     *   NBUC = fine or coarse mesh indication          *
c     *                                                  *
c     *   [ Material properties ]                        *
c     *   NMAT=Number of materials.                      *
c     *   Young's modulus: E1(i,j)                       *
c     *   Possion ratio  : V1(i,j)                       *
c     *   Shear modulus  : G1(i,j)                       *
c     *   Filler Material ditinquisher: NF 1--Isotropic  *
c     *                                    2--Orthotropic*
c     *                                                  *
c     *   [ Composite Wall Layups ]                      *
c     *   number of plies for the top sheet = MWT        *
c     *   number of plies for the bot. sheet = MWB       *
c     *   number of plies for the tube wall = MWC        *
c     *   type of material for top sheet = NWT           *
c     *   type of material for bot. sheet = NWB          *
c     *   type of material for tube wall = NWC           *
c     *   ply thickness of the top sheet = TT            *
c     *   ply thickness of the bottom sheet = BT         *
c     *   ply thickness of the tube wall = CT            *
c     *   ply orientation of the top sheet = AT(MWT)     *
c     *   ply orientation of the bot. sheet = AB(MWB)    *
c     *   ply orientation of the tube wall = AC(MWC)     *
c     *                                                  *
c     *   [Loads]                                        *
c     *   axial load: XL(i)                              *
c     *   shear load: SL(i)                              *
c     *                                                  *
c     *   [AML Failure Parameter]                        *
c     *   aml angles: AMLA(6,6)                          *
c     *   aml strain allowables: AMLS(6,6)               *
```

```
c     *   aml strain angle curve temperature: AMLT(6)    *
c     *   aml invoking indicator: DA (1. yes, 0. no)     *
c     *                                                  *
c     *   SUBROUTINES:                                   *
c     *    yal0: read tgeom.dat(tape10)                  *
c     *    yal1: read tmat.dat(tape11)                   *
c     *    nal0: input data for the new tgeom.dat(tape10)*
c     *    nal1: input data for the new tmat.dat(tape11) *
c     *    tlod: loads input                             *
c     *    tlis: list data                               *
c     *    walis: list wall composition                  *
c     *    tmod: modify input data                       *
c     *    walmod: modify wall composition               *
c     *    tsave: update data                            *
c     *                                                  *
c     *     Jul . 1991                                   *
c     *    Original creator -- L. L. Chou,  LMSC 81-12   *
c     *    Updated by          A. Y. Wei,   LMSC 81-12   *
c     *    Updated by          J. T. Huang, LASC 73-C2   *
c     *                                                  *
c     ****************************************************
      LOGICAL here10,here11,here12
      character*1 tok
      character nmt*50
      dimension nmt(7)
100   format(a1)
      write(6,'('' Advanced Composite Structural Program '')')
      write(6,*)' '
      write(6,'('' TUBULAR TRUSS CORE PANEL MODULE '')')
      write(6,'('' Version 1.3, 2/28/92 '')')
      write(6,*)' '
      write(6,'('' Updated by Tony Wei. LMSC Bld. 157 Ext:6-1137'')
     *')
      write(6,*)' '
c
c        checking existance of input files for DIAL model
c        initialize geometry(tgeom.dat) and material(tmat.dat) files
c
      inquire( file='tgeom.dat', exist=here10)
      inquire( file='tmat.dat', exist=here11)
      if (here10) then
        write(6,'('' Geometric Database Already Exist.'')')
        call yal0
      else
        call nal0
      endif
         write(6,*)' '
      if (here11) then
        write(6,'('' Material Database Already Exist.'')')
        call yal1
        write(6,*)' '
      else
        call nal1
        call tlod
      endif
c
c        list input parameters
c
      call tlis
      call tmod
```

```
      write(6,*)' '
c
c #      end of interactive input.
c #      begin Dial runstream generation.
c #
      write(6,'(''     *** Begin Dial Runstream Generation ***'')')
      write(6,*)' '
      call tmesh
      call tbset
      call tmat
      call tload
      call tsolve
      write(6,'(''     *** Dial Runstream Generation Complete ***'')')
      write(6,*)' '
         stop
         end
c
c        Read geometry input from existing file tgeom.dat
c
      SUBROUTINE ya10
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c     *               T(3,6),C(3,6),SH(3,6),F(3,6)
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *CT(99),AT(99),AB(99),AC(99)
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
      character blk
c
c        read tgeom.dat (unit10)
c
      open(unit=10,status='old',file='tgeom.dat')
  101  format(3i3)
  102  format(3f19.9)
  103  format(4f10.5)
  104  format(4i5)
  105  format(1X,i2,1x,i2,1x,f7.5,1x,f7.2)
  106  format(a1)
  107  format(3f9.5)
  108  format(f12.5,3i3)
c
c        geom.dat (tape 10): geometry
c
      read(10,103)A,B,PH,PL
      read(10,104)NW,K,NBC,NBUC
      read(10,108)TAR,NDF,ND,MS
      read(10,107)TCT,BCT,CTT
c
c        wall layups
c
      read(10,101)MWT,MWB,MWC
      do 10 i=1,MWT
         read(10,105)j,NWT(j),TT(j),AT(j)
10    continue
      read(10,106)blk
      do 11 i=1,MWB
         read(10,105)j,NWB(j),BT(j),AB(j)
11    continue
      read(10,106)blk
      do 12 i=1,MWC
```

```
         read(10,105)j,NWC(j),CT(j),AC(j)
 12    continue
         return
         end
 c
 c        Read material input from exising file tmat.dat
 c
       SUBROUTINE yall
       common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
       common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
 c     *               T(3,6),C(3,6),SH(3,6),F(3,6)
       common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
       common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
      *CT(99),AT(99),AB(99),AC(99)
       common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
       common /name/ nmt
       character*1 tok
       character nmt*50
       dimension nmt(7)
 c
 C     TMAT.dat (unit 11)
 C
       open(unit=11,status='old',file='tmat.dat')
 c
   101  format(f19.3)
   102  format(i5)
   103  format(3f19.3)
   104  format(3i5)
   105  format(i5,2f19.3)
   106  format(a1)
   107  format(f9.5)
   108  format(f12.3)
   109  format(a50)
 c
       read(11,102)NMAT
       do 12 i=1,NMAT
          read(11,109)nmt(i)
          read(11,103)E1(1,i),E1(2,i),E1(3,i)
          read(11,103)G1(1,i),G1(2,i),G1(3,i)
          read(11,103)V1(1,i),V1(2,i),V1(3,i)
          read(11,106)tok
 12    continue
       if (NBC.eq.2.or.NBC.eq.5)then
          NTEMP=7
          read(11,'(i2)')NF
          read(11,109)nmt(7)
          read(11,103)E1(1,NTEMP),E1(2,NTEMP),E1(3,NTEMP)
          read(11,103)G1(1,NTEMP),G1(2,NTEMP),G1(3,NTEMP)
          read(11,103)V1(1,NTEMP),V1(2,NTEMP),V1(3,NTEMP)
          read(11,106)tok
       else
          continue
       endif
 c
 c        tmat.dat (unit11):load
 c
       do 13 i=1,4
          read(11,108)XL(i)
          read(11,108)SL(i)
 13    continue
```

```
      write(6,*)' '
      read(11,107)DA
c
c tmat.dat (unit11):AML parameters.
c
         do 14 j=1,NMAT
            read(11,107)(AMLA(i,j),i=1,6)
            read(11,107)(AMLS(i,j),i=1,6)
c            read(11,107)AMLT(j)
            read(11,106)tok
14       continue
            return
            end
c
c           Read geometry input interactively
c
      SUBROUTINE na10
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c     *               T(3,6),C(3,6),SH(3,6),F(3,6)
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *CT(99),AT(99),AB(99),AC(99)
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
      character*1 yn
c
c     *   A= bottom width of the tube                    *
c     *   B= top width of the tube                       *
c     *   PH= thickness of panel                         *
c     *   PL= length of panel                            *
c     *   NW = number of tubes (whole tubes)             *
c     *   NBC = boundary condition indicator             *
c     *   NBUC = fine or coarse mesh indicator           *
c     *                                                  *
c     *   [ Composite Wall Layups ]                      *
c     *   number of plies for the top sheet = MWT        *
c     *   number of plies for the bot. sheet = MWB       *
c     *   number of plies for the tube wall = MWC        *
c     *   type of material for top sheet = NWT(MWT)      *
c     *   type of material for bot. sheet = NWB(MWB)     *
c     *   type of material for tube wall = NWC(MWC)      *
c     *   ply thickness of the top sheet = TT(MWT)       *
c     *   ply thickness of the bottom sheet = BT(MWT)    *
c     *   ply thickness of the tube wall = CT(MWC)       *
c     *   ply orientation of the top sheet = AT(MWT)     *
c     *   ply orientation of the bot. sheet = AB(MWB)    *
c     *   ply orientation of the tube wall = AC(MWC)     *
c
c        create input for tgeom.dat (unit10)
c
c        default values
c
c        one inch wide at the tube bottom
      A=1.0
c        0.5 inch wide at the tube top
      B=0.5
c        0.5 inch thickness for the panel
      PH=0.5
c        5. inch panel length
      PL=5.
```

```
c        5 tubes in the width direction
      NW=5
c        5 elements in the length direction
      K=5
c        simple stress analysis
      NBUC=1
c        infinite panel b. c.
      NBC=1
c        number of plies for bottom face sheet
      MWB=4
c        number of plies for top face sheet
      MWT=4
c        number of plies for tube core
      MWC=4
C
      do 10 i=1,16
         AT(i)=0.
         AB(i)=0.
         AC(i)=0.
10    continue
      TCT=0.0
      BCT=0.0
      CTT=0.0
c
 101  format(' Input Top Width of the Tube (',f10.4,'in.) >>'$)
 102  format(' Input Bottom Width of the Tube (',f10.4,'in.) >>'$)
 103  format(' Input Height of the Tubes (',f10.4,'in.) >>'$)
 104  format(' Input Length of the Panel (',f10.4,'in.) >>'$)
 105  format(' Input Number of Tubes in the Width Direction (Y) ('
     *,i2,') >>'$)
c
      write(6,'('' ***Geometric Parameter Inputs***'')')
      write(6,*)' '
      write(6,'('' Input 1) for Stress Analysis. '')')
      write(6,'('' 2) for Buckling Analysis.'')')
      write(6,'('' 3) for Post Buckling Analysis.'')')
      write(6,'('' Enter >> ''$)')
C
      call NINTG(1,NBUC)
      TAR=0.
      ND=0
      NDF=0
      MS=10
C
      if (NBUC.eq.3) then
         write(6,*)' '
         write(6,'('' Input Target Point Info. >>'')')
         write(6,'('' Type of target 0> Load Factor.'')')
         write(6,'('' 1-6> Displacement 1 to 6.'')')
         write(6,'('' Enter>>''$)')
         call NINTG(1,NDF)
         write(6,*)' '
         if (NDF.ge.1) then
            write(6,'('' Node Number specification.'')')
            write(6,'('' Enter Node #>>''$)')
            call NINTG(1,ND)
         endif
         write(6,*)' '
         write(6,'('' Enter Target Point >>''$)')
         call NREAL(1,TAR)
```

```
       elseif (NBUC .eq. 2) then
          write(6,*)' '
          write(6,'('' How many MODE shapes do you want (Max. 50)? >>''$)
      *')
          call NINTG(1,MS)
       endif
C
       write(6,*)' '
       write(6,101)B
       call NREAL(1,B)
       write(6,102)A
       call NREAL(1,A)
       write(6,103)PH
       call NREAL(1,PH)
       write(6,104)PL
       call NREAL(1,PL)
       write(6,105)NW
       call nintg(6,NW)
       write(6,*)' '
       write(6,'('' Input Boundary Conditions for the Panel.'')')
       write(6,'('' 1-Infinite Panel.'')')
       write(6,'('' 2-Sides Simply Supported. (with Filler Material; ends
      * free)'')')
       write(6,'('' 3-Sides Simply Supported. (w/o Filler Material; ends
      *free)'')')
       write(6,'('' 4-Ends Clamped; w/o Filler at the Sides. (Sides Free)
      *'')')
       write(6,'('' 5-Ends Clamped; with Filler at the Sides. (Sides Free
      *)'')')
       write(6,'('' Select Your Choice >>''$)')
       call NINTG(1,NBC)
       write(6,*)' '
          return
          end
c
C         Read material input interactively
c
       SUBROUTINE nall
C
       common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
       common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c      *               T(3,6),C(3,6),SH(3,6),F(3,6)
       common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
       common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
      *CT(99),AT(99),AB(99),AC(99)
       common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
       common /name/ nmt
       character*1 tok,yn
       character nmt*50
       dimension nmt(7)
c
c      NMAT--number of materials used. (excludes filler material)
c      E(i,j)--elastic moduli.
c      V(i,j)--Poisson's ratio.
c      G(i,j)--shear moduli.
c      AMLS(i,j)--AML curve shear allowable.
c      AMLA(i,j)--AML curve angle.
c      AMLT(i)-- AML curve temperature.
c      XL(i)--axial loads.
c      SL(i)--shear laods.
```

```
C
c       Material Number >>7<< is reserved for the Filler Material ***
C
c       create input for tmat.dat (unit11)
c
c       default values for matl
c
c       Elastic & Shear Moduli.
C
      do 10 i=1,7
         E1(1,i)=1000.
         E1(2,i)=1000.
         E1(3,i)=1000.
         G1(1,i)=1000.
         G1(2,i)=1000.
         G1(3,i)=1000.
         V1(1,i)=0.300
         V1(2,i)=0.300
         V1(3,i)=0.300
10    continue
c
101   format(' Input Information for Mat.',i,' >>')
102   format(' Input Elastic Moduli: E11 (',f8.2,') E22 (',f8.2,
     *      ') >>'$)
103   format(' Input Shear Moduli: G12 (',f8.2,') G23 (',f8.2,
     *      ') G31 (',f8.2,') >>'$)
104   format(' Input Poisson''''s Ratio: V12 (',f8.6,') >>'$)
105   format(a1)
106   format(' Input AML Paramters for Material ',i2,' >>')
107   format(' Point ',il,'-- Angle (Degree) and Strain Allowable. ')
108   format(' Name (No More Than 50 Characters) >>'$)
109   format(a50)
c
      write(6,*)' '
      write(6,'('' **************REMINDER***************''')')
      write(6,'('' **Please Keep the Total Number of Materials''')')
      write(6,'('' (Excluding Filler Material) at Maximum of 6**''
     *)')
      write(6,*)' '
      write(6,'('' Input No. of Materials Used.(Exclude Filler Material)
     *>>''$)')
      call NINTG(1,NMAT)
      write(6,*)''
C
      do 20 i=1,NMAT
         write(6,101)i
         write(6,*)' '
         write(6,108)
         read(5,109)nmt(i)
         write(6,*)' '
         write(6,'('' Input Elastic Moduli: E11,E22 >>''$)')
         call NREAL(2,E1(1,i),E1(2,i))
         write(6,*)' '
         write(6,'('' Input Shear Moduli: G12,G23,G31 >>''$)')
         call NREAL(3,G1(1,i),G1(2,i),G1(3,i))
         write(6,*)' '
         write(6,'('' Input Poisson''''s Ratio: V12 >>''$)')
         call NREAL(1,V1(1,i))
         write(6,*)' '
20    continue
```

```
C
      if (NBC.eq.2.or.NBC.eq.5)then
         write(6,'('' ***Filler Material***'')')
         NTEMP=7
         write(6,101)NTEMP
         write(6,*)' '
         write(6,108)
         read(5,109)nmt(7)
         write(6,*)' '
         write(6,'('' 1)Isotropic. 2)Orthotropic.'')')
         write(6,'('' Select >>''$)')
         call nintg(1,NF)
         write(6,*)' '
C
         if (NF.eq.1) then
            write(6,'('' Input Elastic Modulus: E (psi). >>''$)')
            call nreal(1,E1(1,NTEMP))
            write(6,*)' '
            write(6,'('' Input Poisson''''s ratio. >>''$)')
            call nreal(1,V1(1,NTEMP))
            write(6,*)' '
         elseif (NF.eq.2) then
            write(6,'('' Input Elastic Moduli: E11,E22,E33 (psi)>>''$)')
            call NREAL(3,E1(1,NTEMP),E1(2,NTEMP),E1(3,NTEMP))
            write(6,*)' '
            write(6,'('' Input Shear Moduli: G12,G23,G31 (psi)>>''$)')
            call NREAL(3,G1(1,NTEMP),G1(2,NTEMP),G1(3,NTEMP))
            write(6,*)' '
            write(6,'('' Input Poisson''''s Ratios: V12,V23,V31 >>''$)')
            call NREAL(3,V1(1,NTEMP),V1(2,NTEMP),V1(3,NTEMP))
            write(6,*)' '
         endif
      else
         continue
      endif
C
      DA=0.
21    write(6,'('' Do You Want to use the AML Failure Criterion? [Y
     *or N] >>''$)')
      read(5,105)tok
      if ((tok.eq.'Y').or.(tok.eq.'y'))then
         DA=1.
         write(6,*)' '
         write(6,'('' Input AML Parameters :'')')
         write(6,'('' ***for further definition, please consult the user
     *''''s manual.***'')')
         write(6,*)' '
         do 22 k=1,NMAT
            write(6,106)k
            write(6,*)' '
            write(6,'('' *** Tension Strain Values ***'')')
               do 23 l=1,6
                  write(6,107)l
                  write(6,'('' Enter Angle. (deg.) >>''$)')
                  call nreal(1,AMLA(l,k))
                  write(6,'('' Enter Strain Allowable. >>''$)')
                  call nreal(1,AMLS(l,k))
                  write(6,*)' '
                  if (l.eq.3) then
                     write(6,'('' *** Compression Strain Values ***''
```

```
     *)')
                  endif
 23               continue
                write(6,*)' '
                write(6,*)' '
 22          continue
          elseif ((tok.eq.'N').or.(tok.eq.'n'))then
             DA=0.
             do 24 i=1,NMAT
                do 25 j=1,6
                   AMLS(i,j)=0.
                   AMLA(i,j)=0.
   25        continue
   24        continue
          else
             write(6,'('' Error: Wrong Character Inputted!!!'')')
             write(6,*)' '
             go to 21
          endif
c
  117   format(' Input No. of Plies for Top Face Sheet (',i3,
        *') >>'$)
  118   format(' Input No. of Plies for Bottom Face Sheet (',i3,
        *') >>'$)
  119   format(' Input No. of Plies for Tube Core (',i3,') >>'$)
  120   format(' Input Material Numbers for Top Face Sheet Lay-up >>')
  121   format(' Input Material Numbers for Bottom Face Sheet Lay-up >>')
  122   format(' Input Material Numbers Tube Core Lay-up>>')
  123   format(' Ply Number ',i2,' >>'$)
  124   format(' Input Thicknesses for Top Face Sheet Lay-up(in.) >>')
  125   format(' Input Thicknesses for Bottom Face Sheet Lay-up(in.) >>')
  126   format(' Input Thicknesses for Tube Core Lay-up(in.) >>')
  127   format(' Is it the Same for All Plies? [Y or N] >>'$)
  128   format(a1)
  129   format(' Input Parameter for the Lay-up: (Enter ',i2,' Numbers):')
  130   format('    Enter >>'$)
        write(6,'(''  ***Wall Layup Information Input***'')')
        write(6,*)' '
c
c          Input Number of Plies for Top, Bottom and Core Wall
c
        write(6,*)' '
        write(6,117)4
        call nintg(1,MWT)
        write(6,*)' '
        write(6,118)4
        call nintg(1,MWB)
        write(6,*)' '
        write(6,119)4
        call nintg(1,MWC)
        write(6,*)' '
c
c          Wall Composition Input
c          Top Face Sheet
c
c          Top Wall Ply Material Number
        write(6,120)
        write(6,*)' '
        write(6,127)
        read(5,128)yn
```

```
      write(6,*)' '
      if(yn.eq.'Y'.or.yn.eq.'y')then
         write(6,'('' Enter >>''$)')
         call nintg(1,NWT(1))
         do 35 i=2,MWT
            NWT(i)=NWT(1)
35       continue
         write(6,*)' '
      else
         write(6,129)MWT
         do 36 i=1,MWT
            write(6,130)
            call nintg(1,NWT(i))
36       continue
         write(6,*)' '
      endif
c        Top Wall Ply Thicknesses
      write(6,124)
      write(6,*)' '
      write(6,127)
      read(5,128)yn
      if (yn.eq.'Y'.or.yn.eq.'y') then
         write(6,'('' Enter >>''$)')
         call nreal(1,TT(1))
         do 37 i=2,MWT
            TT(i)=TT(1)
37       continue
         write(6,*)' '
      else
         write(6,129)MWT
         do 38 j=1,MWT
            write(6,130)
            call nreal(1,TT(j))
38       continue
         write(6,*)' '
      endif
c        Top Wall Ply Orientations
      write(6,'('' Input Orientations of Plies for Top Face Sheet (deg.)
     * >>'')')
      write(6,*)' '
      write(6,127)
      read(5,128)yn
      write(6,*)' '
      if (yn.eq.'Y'.or.yn.eq.'y') then
         write(6,'('' Enter >>''$)')
         call nreal(1,AT(1))
         do 39 i=2,MWT
            AT(i)=AT(1)
39       continue
         write(6,*)' '
      else
         write(6,129)MWT
         do 40 j=1,MWT
            write(6,130)
            call nreal(1,AT(j))
40       continue
         write(6,*)' '
      endif
c
c        Bottom Face Sheet
```

```
c         Bottom Wall Ply Material Numbers
      write(6,121)
      write(6,*)' '
      write(6,127)
      read(5,128)yn
      write(6,*)' '
      if (yn.eq.'Y'.or.yn.eq.'y') then
      write(6,'('' Enter >>''$)')
      call nintg(1,NWB(1))
         do 41 i=2,MWB
            NWB(i)=NWB(1)
41       continue
         write(6,*)' '
      else
         write(6,129)MWB
         do 42 j=1,MWB
            write(6,130)
            call nintg(1,NWB(j))
42       continue
         write(6,*)' '
      endif
c        Bottom Wall Ply Thicknesses
      write(6,125)
      write(6,*)' '
      write(6,127)
      read(5,128)yn
      write(6,*)' '
      if (yn.eq.'Y'.or.yn.eq.'y') then
         write(6,'('' Enter >>''$)')
         call nreal(1,BT(1))
         do 43 i=2,MWB
            BT(i)=BT(1)
43       continue
         write(6,*)' '
      else
         write(6,129)MWB
         do 44 j=1,MWB
            write(6,130)
            call nreal(1,BT(j))
44       continue
         write(6,*)' '
      endif
c        Bottom Wall Ply Orientations
      write(6,'('' Input Orientations of Plies for Bottom Face Sheet (de
     *g.) >>'')')
      write(6,*)' '
      write(6,127)
      read(5,128)yn
      write(6,*)' '
      if (yn.eq.'Y'.or.yn.eq.'y') then
         write(6,'('' Enter >>''$)')
         call nreal(1,AB(1))
         do 45 i=2,MWB
            AB(i)=AB(1)
45       continue
         write(6,*)' '
      else
         write(6,129)MWB
         do 46 j=1,MWB
            write(6,130)
```

```
            call nreal(1,AB(j))
 46       continue
          write(6,*)' '
       endif
 c
 c        Tube Core
 c
 c        Core Wall Ply Material Numbers
       write(6,122)
       write(6,*)' '
       write(6,127)
       read(5,128)yn
       write(6,*)' '
       if (yn.eq.'Y'.or.yn.eq.'y') then
       write(6,'('' Enter >>''$)')
       call nintg(1,NWC(1))
          do 47 i=2,MWC
             NWC(i)=NWC(1)
 47       continue
          write(6,*)' '
       else
          write(6,129)MWC
          do 48 k=1,MWC
             write(6,130)
             call nintg(1,NWC(k))
 48       continue
          write(6,*)' '
       endif
 c        Core Wall Ply Thicknesses
       write(6,126)
       write(6,*)' '
       write(6,127)
       read(5,128)yn
       write(6,*)' '
       if (yn.eq.'Y'.or.yn.eq.'y') then
          write(6,'('' Enter >>''$)')
          call nreal(1,CT(1))
          do 49 i=2,MWC
             CT(i)=CT(1)
 49       continue
          write(6,*)' '
       else
          write(6,129)MWC
          do 50 k=1,MWC
             write(6,130)
             call nreal(1,CT(k))
 50       continue
          write(6,*)' '
       endif
 c        Core Wall Ply Orientations
       write(6,'('' Input Orientations of Plies for Core Wall (deg.) >>''
      *)')
       write(6,*)' '
       write(6,127)
       read(5,128)yn
       write(6,*)' '
       if (yn.eq.'Y'.or.yn.eq.'y') then
          write(6,'('' Enter >>''$)')
          call nreal(1,AC(1))
          do 51 i=2,MWC
```

```
            AC(i)=AC(1)
51          continue
            write(6,*)' '
         else
            write(6,129)MWC
            do 52 i=1,MWC
               write(6,130)
               call nreal(1,AC(i))
52          continue
            write(6,*)' '
         endif
         write(6,*)' '
c
c           Top Face Sheet Thickness
c
         do 53 i=1,MWT
            TCT=TCT+TT(i)
53       continue
c
c           Bottom Face Sheet Thickness
c
         do 54 i=1,MWB
            BCT=BCT+BT(i)
54       continue
c...
c...        Core Wall Thickness
c...
         do 55 i=1,MWC
            CTT=CTT+CT(i)
55       continue
            return
            end
c
c           loads input
c
         SUBROUTINE TLOD
         common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
         common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c        *               T(3,6),C(3,6),SH(3,6),F(3,6)
         common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
         common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
        *CT(99),AT(99),AB(99),AC(99)
         common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
c..
c        XL(1),XL(3) -- Axial Loads at the Sides.
c        XL(2),XL(4) -- Axial Loads at the Ends.
c        XE -- Total Axial Load on the End.
c        XS -- Total Axial Load on the Side.
c        SE -- Total Shear Load on the End.
c        SS -- Total Shear Load on the Side.
c        TMPD -- length of the ends.
c        PL   -- length of the sides.
c..
         TMPD=(A+B)/2.*(NW+1)
         do 10 i=1,8
          write(6,'('' '')')
10       continue
         write(6,'('' Apply loads to the model:'')')
         write(6,'('' Load may be input as total load or as an average
        *line load.'')')
```

```
      write(6,'(''     How would you like to input loads?'')')
      write(6,'(''     1>Total Load.'')')
      write(6,'(''     2>Line Load.'')')
      write(6,'(''     Enter >>''$)')
      call nintg(1,N)
      write(6,*)' '
      if(N.eq.1)then
         write(6,'(''     Total Tensile (+) or Comppressive (-) Load (lb
     *.) on the Ends.'')')
         write(6,'(''     Enter >>''$)')
         call nreal(1,XL(2))
c
         XL(2)=XL(2)/TMPD
         XL(4)=XL(2)
         write(6,*)' '
         write(6,'(''     Total Tensile (+) or Comppressive (-) Load (lb
     *.) on the Sides.'')')
         write(6,'(''     Enter >>''$)')
         call nreal(1,XL(1))
c
         XL(1)=XL(1)/pl
         XL(3)=XL(1)
         write(6,*)' '
         write(6,'(''     Total Shear Load (+ CCW and - CW) (lb.)'')')
         write(6,'(''     1) On the Ends.'')')
         write(6,'(''     2) On the Sides.'')')
         write(6,'(''     Enter >>''$)')
         call nintg(1,I)
         write(6,*)' '
         if (I.eq.1) then
             write(6,'(''     Shear Load on the Ends = ''$)')
             call nreal(1,SL(2))
c
             SL(2)=SL(2)/TMPD
             SL(4)=SL(2)
             SL(1)=SL(2)
             SL(3)=SL(2)
c
             write(6,*)' '
         else
             write(6,'(''     Shear Load on the Sides = ''$)')
             call nreal(1,SL(1))
c
             SL(1)=SL(1)/pl
             SL(3)=SL(1)
             SL(2)=SL(1)
             SL(4)=SL(1)
c
             write(6,*)' '
         endif
      elseif(N.eq.2)then
         write(6,'(''     Input Tensile (+) or Compressive (-) Line Load
     * on the Ends. (lb./in) >>'')')
         write(6,'(''     Enter >>''$)')
         call nreal(1,XL(2))
c
         XL(4)=XL(2)
         write(6,*)' '
         write(6,'(''     Input Tensile (+) or Compressive (-) Line Load
     * on the Sides. (lb./in) >>'')')
```

```
         write(6,'('' Enter >>''$)')
         call nreal(1,XL(1))
c
         XL(3)=XL(1)
         write(6,*)' '
         write(6,'('' Input Shear Line Load (lb./in) >>'')')
         write(6,'('' Enter >>''$)')
         call nreal(1,SL(1))
c
         SL(2)=SL(1)
         SL(3)=SL(1)
         SL(4)=SL(1)
         write(6,*)' '
       endif
       write(6,*)' '
       write(6,*)' '
         return
         end
c
c        Record data in tgeom.dat(unit10) and tmat.dat(unit11)
c
       SUBROUTINE tsave
       common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
       common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c      *              T(3,6),C(3,6),SH(3,6),F(3,6)
       common /tam1/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
       common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
      *CT(99),AT(99),AB(99),AC(99)
       common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
       common /name/ nmt
       character blk
       character nmt*50
       dimension nmt(7)
c
c        OPEN tgeom.dat & tmat.dat
c
       open(unit=10,status='unknown',file='tgeom.dat')
       open(unit=11,status='unknown',file='tmat.dat')
       rewind(10)
       rewind(11)
       blk=' '
c
  101  format(3i3)
  102  format(3f19.9)
  103  format(4f10.5)
  104  format(4i5)
  105  format(1X,i2,1x,i2,1x,f7.5,1x,f7.2)
  106  format(a1)
  107  format(f9.5)
  108  format(I5)
  109  format(3f19.3)
  110  format(3i5)
  111  format(i5,2f19.3)
  112  format(3f9.5)
  113  format(f12.3)
  114  format(i2)
  115  format(a50)
  116  format(f12.5,3i3)
c
c        save and update files
```

```
c
c         tgeom.dat (unit10): geometry
c
        write(10,103)A,B,PH,PL
        write(10,104)NW,K,NBC,NBUC
        write(10,116)TAR,NDF,ND,MS
        write(10,112)TCT,BCT,CTT
c.......
c         wall layups
c.......
         write(10,101)MWT,MWB,MWC
        do 10 i=1,MWT
           write(10,105)i,NWT(i),TT(i),AT(i)
10      continue
        write(10,106)blk
        do 11 i=1,MWB
           write(10,105)i,NWB(i),BT(i),AB(i)
11      continue
        write(10,106)blk
        do 12 i=1,MWC
           write(10,105)i,NWC(i),CT(i),AC(i)
12      continue
c...........
c         tmat.dat (unit11): material
c...........
        write(11,108)NMAT
        do 13 i=1,NMAT
           write(11,115)nmt(i)
           write(11,109)E1(1,i),E1(2,i),E1(3,i)
           write(11,109)G1(1,i),G1(2,i),G1(3,i)
           write(11,109)V1(1,i),V1(2,i),V1(3,i)
           write(11,106)tok
13      continue
        if (NBC.eq.2.or.NBC.eq.5)then
           NTEMP=7
           write(11,114)NF
           write(11,115)nmt(7)
           write(11,109)E1(1,NTEMP),E1(2,NTEMP),E1(3,NTEMP)
           write(11,109)G1(1,NTEMP),G1(2,NTEMP),G1(3,NTEMP)
           write(11,109)V1(1,NTEMP),V1(2,NTEMP),V1(3,NTEMP)
           write(11,106)tok
        else
           continue
        endif
c...........
c         tmat.dat (unit11):load
c...........
        do 14 i=1,4
           write(11,113)XL(i)
           write(11,113)SL(i)
14      continue
        write(11,107)DA
        do 15 j=1,NMAT
           write(11,107)(AMLA(i,j),i=1,6)
           write(11,107)(AMLS(i,j),i=1,6)
c           write(11,107)AMLT(j)
           write(11,106)tok
15      continue
        close(unit=10)
        close(unit=11)
```

```
      return
      end
c
c     interactive listing of the paramerters
c
      SUBROUTINE tlis
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c     *               T(3,6),C(3,6),SH(3,6),F(3,6)
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *CT(99),AT(99),AB(99),AC(99)
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
      common /name/ nmt
      character tok
      character nmt*50,type*15
      dimension nmt(7)
c
 101  format('     Width at the Tube Bottom =',f8.3,'in.',/,
     *'     Width at the Tube Top=',f8.3,'in.')
 102  format('     Material Number:',i2)
 103  format('     Number of Tubes =',i2)
 104  format('     E11(psi)=',f13.1,2x,'E22(psi)=',f13.1)
 105  format('     G12(psi)=',f13.1,2x,'G23(psi)=',f13.1,2x,
     *'G31(psi)=',f13.1)
 106  format('     V12=',f7.5)
 107  format('     Ply Number',I2,/,'       Material Number:',i2,2x,
     */,'       Ply Thickness(in)=',f7.5,2x,/,
     *'       Ply Angle(degree)=',f7.2)
 108  format('     Panel Length, PL=',f9.3,'in.',/,
     *'     Total Height of the Panel=',f12.6,'in.',/,
     *'     Tube Height ,PH=',f9.3,'in.')
 109  format('     ------------------------')
 110  format('     Top Cover Sheet Thickness=',f9.6,'in.')
 111  format('     Bottom Cover Sheet Thickness=',f9.6,'in.')
 112  format('     Tube Wall Thickness=',f9.6,'in.')
 113  format('     Point ',i,' Angle(Degree)=',f6.2,' Strain Allowable='
     *,f9.5)
C
 115  format(a1)
 116  format('     Axial Line Load on the Ends. (lb./in)=',f12.3,/,
     *'      Total Axial Load on the Ends. (lb.)=',f12.3)
 117  format('     Axial Line Load on the Sides. (lb./in)=',f12.3,/,
     *'      Total Axial Load on the Sides. (lb.)=',f12.3)
 118  format('     Shear Line Load (lb./in)=',f12.3,/,
     *'      Total Shear Load on the Ends. (lb.)=',f12.3,/,
     *'      Total Shear Load on the Sides. (lb.)=',f12.3)
 119  format('     E11(psi)=',f13.1,2x,'E22(psi)=',f13.1,2x,'E33(psi)=',
     *f13.1)
 120  format('     V12=',f7.5,2x,'V23=',f7.5,2x,'V31=',f7.5)
 121  format('     Panel Width = ',f8.3,'in.')
 122  format('     E(psi)=',f13.1,'  V=',f7.5)
 123  format('     Material Name = ',a50)
 124  format('     Target Type >>',a15,' Target Point = ',f12.3)
 125  format('     Target Type >>',a15,i2,' Node=',i4,' Target Point = '
     *,f12.3)
c ...
c       list files
c ...
      wdth=(A+B)/2.*(NW+1)
```

```
10    write(6,'('' Choose the Info. to be Listed:'')')
      write(6,'('' 1>Panel Geometry.'')')
      write(6,'('' 2>Material Data.'')')
      write(6,'('' 3>Wall Layups.'')')
      write(6,'('' 4>Load.'')')
      write(6,'('' 5>Continue.'')')
      write(6,'('' Enter>>''$)')
      call nintg(1,N)
c.....
c       tgeom.dat (unit10): geometry
c.....
      if(N.eq.1)then
         write(6,*)' '
         write(6,101)A,B
         write(6,103)NW
         write(6,121)wdth
         tmp=PH+TCT+BCT
         write(6,108)ABS(PL),tmp,PH
         write(6,110)TCT
         write(6,111)BCT
         write(6,112)CTT
         if(NBC.eq.1)then
            write(6,'('' Boundary Condition: Infinite Panel.'')')
         elseif(NBC.eq.2)then
            write(6,'('' Boundary Condition: Sides Simply Supported;
     * with Filler Material.'')')
            write(6,'('' Ends Free.'')')
         elseif(NBC.eq.3)then
            write(6,'('' Boundary Condition: Sides Simply Supported;
     * w/o Filler Material.'')')
            write(6,'('' Ends Free.'')')
         elseif(NBC.eq.4)then
            write(6,'('' Boundary Condition: Ends Clamped; w/o Fille
     *r at the Sides.'')')
            write(6,'('' Sides Free.'')')
         elseif(NBC.eq.5)then
            write(6,'('' Boundary Condition: Ends Clamped; with Fill
     *er at the Sides.'')')
            write(6,'('' Sides Free.'')')
         endif
         if(NBUC.eq.1)then
            write(6,'('' Stress Analysis.'')')

         elseif(NBUC.eq.2)then
            write(6,'('' Buckling Analysis.'')')
            write(6,'('' '',i3,'' Mode(s).'')')MS
         elseif(NBUC.eq.3)then
            write(6,'('' Post Buckling Analysis.'')')
            if (NDF.ge.1) then
               TYPE='DISPLACEM. DEG.'
               write(6,125)type,NDF,ND,TAR
            else
               TYPE='LOAD FACTOR.   '
               write(6,124)type,TAR
            endif
         endif
         write(6,*)' '
         write(6,'('' ***Hit <<Return>> for Next Page.***''$)')
         read(5,115)tok
         write(6,*)' '
```

```
         write(6,*)' '
         write(6,*)' '
         go to 10
c..........
c           tmat.dat (unit11): material
c..........
      elseif(N.eq.2)then
         do 14 j=1,NMAT
            write(6,*)' '
            write(6,102)j
            write(6,123)nmt(j)
            write(6,'(''     Orthotropic Material'')')
            write(6,104)E1(1,j),E1(2,j)
            write(6,105)G1(1,j),G1(2,j),G1(3,j)
            write(6,106)V1(1,j)
            write(6,109)
            write(6,'(''  AML parameters for this material.'')')
            write(6,'(''  *** Tension Strain Values ***'')')
            do 21 i=1,3
               write(6,113)i,AMLA(i,j),AMLS(i,j)
21          continue
            write(6,'(''  *** Compression Strain Values ***'')')
            do 22 i=4,6
               write(6,113)i,AMLA(i,j),AMLS(i,j)
22          continue
               write(6,'(''     ***Hit <<Return>> for Next Page.***''$)
     *')
               read(5,115)tok
               write(6,*)' '
14       continue
         if (DA.eq.1) then
            write(6,'(''    AML Failure Calculation is invoked.'')')
            write(6,*)' '
         else
            write(6,'(''    AML Failure Calculation is not invoked.'')'
     *)
         endif
            write(6,*)' '
         write(6,*)' '
         if (NBC.eq.2.or.NBC.eq.5)then
            NTEMP=7
            write(6,'('' ***Filler Material***'')')
            write(6,*)' '
            write(6,102)NTEMP
            if (NF.eq.1) then
               write(6,123)nmt(7)
               write(6,'(''     Isotropic Material'')')
               write(6,122)E1(1,NTEMP),V1(1,NTEMP)
               write(6,*)' '
               write(6,'(''     ***Hit <<Return>> for Next Page.***''$)
     *')
               read(5,115)tok
            elseif (NF.eq.2) then
               write(6,123)nmt(7)
               write(6,'(''     Orthotropic Material'')')
               write(6,119)E1(1,NTEMP),E1(2,NTEMP),E1(3,NTEMP)
               write(6,105)G1(1,NTEMP),G1(2,NTEMP),G1(3,NTEMP)
               write(6,120)V1(1,NTEMP),V1(2,NTEMP),V1(3,NTEMP)
               write(6,*)' '
               write(6,'(''     ***Hit <<Return>> for Next Page.***''$)
```

```
      *')
                read(5,115)tok
              endif
           else
              continue
           endif
        go to 10
c...........
c        wall
c...........
       elseif(N.eq.3)then
          write(6,*)' '
          write(6,'(''      Top Face Sheet Layup.'')')
          call walis(MWT,NWT,TT,AT,TCT)
          write(6,*)' '
          write(6,'(''     Bottom Face Sheet Layup.'')')
          call walis(MWB,NWB,BT,AB,BCT)
          write(6,*)' '
          write(6,'(''      Tube Wall Layup.'')')
          call walis(MWC,NWC,CT,AC,CTT)
          write(6,*)' '
          go to 10
c........
c        load
c........
       elseif(N.eq.4)then
          write(6,*)' '
          write(6,*)' '
          XE=XL(2)*WDTH
          XS=XL(1)*PL
          SE=SL(1)*WDTH
          SS=SL(1)*PL
          write(6,116)XL(2),XE
          write(6,117)XL(1),XS
          write(6,118)SL(1),SE,SS
          write(6,*)' '
          write(6,'(''     ***Hit <<Return>> for Next Page.***''$)')
          read(5,115)tok
          write(6,*)' '
          go to 10
c
       elseif(N.eq.5)then
          go to 17
       else
          go to 10
       endif
17        return
          end
c
c
c        interactive listing of wall composition info.
c
      SUBROUTINE WALIS(NUM,MAT,THK,ANG,TK)
      dimension MAT(99),THK(99),ANG(99)
      character*1 tok
c.......
c   Num -- corresponds to MWT,MWB,MWC in main. Number of Plies.
c   Mat -- corresponds to NWT,NWB,NWC in main. Ply Material Number.
c   Thk -- corresponds to TT,BT,CT in main. Ply Thickness.
c   Ang -- corresponds to AT,BT,CT in main. Ply angle.
```

```
c   Tk  -- corresponds to TCT,BCT,CTT in main. Total Lay-up thickness.
c.......
101   format('     Ply Number',I2,/,'       Material Number:',i2,2x,
     */,'       Ply Thickness(in)=',f7.5,2x,/,
     *'       Ply Angle(degree)=',f7.2)
102   format('     Total Thickness=',f9.6,'in.')
103   format(a1)
104   format('     Bottom Face Sheet Thickness=',f9.6,'in.')
105   format('     Tube Wall Thickness=',f9.6,'in.')
C
      nlop=INT(NUM/4.)
      ibeg=1
      iend=4
      rnum=num*1.
      if (MOD(rnum,4.).ne.0.) then
         do 10 i=1,nlop
            do 11 j=ibeg,iend
               write(6,101)j,MAT(j),THK(j),ANG(j)
11          continue
            write(6,*)' '
            write(6,'('' ***Hit <<Return>> for Next Page.***''$)')
            read(5,103)tok
            write(6,*)' '
            ibeg=ibeg+4
            iend=iend+4
10       continue
         do 12 i=ibeg,NUM
            write(6,101)i,MAT(i),THK(i),ANG(i)
12       continue
         write(6,*)' '
         write(6,102)TK
         write(6,*)' '
         write(6,'('' ***Hit <<Return>> for Next Page.***''$)')
         read(5,103)tok
         write(6,*)' '
      else
         do 13 i=1,nlop
            do 14 j=ibeg,iend
               write(6,101)j,MAT(j),THK(j),ANG(j)
14          continue
         write(6,*)' '
         write(6,'('' ***Hit <<Return>> for Next Page.***''$)')
         read(5,103)tok
         write(6,*)' '
         ibeg=ibeg+4
         iend=iend+4
13    continue
         write(6,*)' '
         write(6,102)TK
         write(6,*)' '
         write(6,*)' '
         write(6,'('' ***Hit <<Return>> for Next Page.***''$)')
         read(5,103)tok
         write(6,*)' '
      endif
         return
         end
c#
c        interactive parameter modification.
c#
```

```
      SUBROUTINE tmod
C
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
c     *               T(3,6),C(3,6),SH(3,6),F(3,6)
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *CT(99),AT(99),AB(99),AC(99)
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
      common /name/ nmt
      character*1 IDC,tok,tok2
      character nmt*50
      dimension nmt(7)
C
 100  format(1x,i2,'>Material ',i2,'.')
 101  format(1x,i2,'>Return to Menu.')
 102  format('     Material Number ',i2,'.')
 103  format('     Ply Number ',i2,'.')
 104  format(a1)
 105  format(1x,'7>Filler Material (Number ',i2,').')
 106  format('     Enter Angle for Point ',i2,' ='$)
 107  format('     Enter Strain Allowable for Point ',i2,' ='$)
 108  format(5x,'Enter AML Curve Angle (Degree) & Strain Allowable >>'
     *$)
 109  format(' Name >> '$)
 110  format(a50)
c ...
c        modify files
c ...
10    write(6,*)' '
      write(6,'(''     Choose the Info. to be Modified:'')')
      write(6,'(''      1>Panel Geometry.'')')
      write(6,'(''      2>Material Data.'')')
      write(6,'(''      3>Wall Layups.'')')
      write(6,'(''      4>Load.'')')
      write(6,'(''     ------------------------------------'')')
      write(6,'(''      5>Return to List Menu.'')')
      write(6,'(''      6>Save the Changes and Exit.'')')
      write(6,'(''     Enter>>''$)')
      call nintg(1,N)
c.........
c        tgeom.dat (unit10): geometry
c.........
      if(N.eq.1)then
11       write(6,*)' '
         write(6,'(''     Choose the Following Geometrical Parameter to
     *be Modified:'')')
         write(6,'(''      1>Tube Height, PH.'')')
         write(6,'(''      2>Tube Width -- Bottom. A'')')
         write(6,'(''      3>Tube Width -- Top. B'')')
         write(6,'(''      4>Panel Length, PL.'')')
         write(6,'(''      5>Number of Tubes.'')')
         write(6,'(''      6>Boundary Condition.'')')
         write(6,'(''      7>Type of the Analysis.'')')
         write(6,'(''      8>Return to Menu.'')')
         write(6,'(''     Enter>>''$)')
         call nintg(1,M)
         write(6,*)' '
C
         if(M.eq.1)then
```

```
            write(6,'(''     Tube Height, PH(in)=''$)')
            call nreal(1,PH)
            write(6,*)' '
         elseif(M.eq.2)then
            write(6,'(''     Tube Bottom Width, A(in)=''$)')
            call nreal(1,A)
            write(6,*)' '
         elseif(M.eq.3)then
            write(6,'(''     Tube Top Width, B(in)=''$)')
            call nreal(1,B)
            write(6,*)' '
         elseif(M.eq.4)then
            write(6,'(''     Panel Length, PL(in)=''$)')
            call nreal(1,PL)
            write(6,*)' '
         elseif(M.eq.5)then
            write(6,'(''     Number of Tubes=''$)')
            call nintg(1,NW)
            write(6,*)' '
         elseif(M.eq.6)then
            write(6,'(''     Boundary Condition:'')')
            write(6,'(''      1>Infinite Panel.'')')
            write(6,'(''      2>Sides Simply Supported; with Filler Mate
     *rial. Ends Free.'')')
            write(6,'(''      3>Sides Simply Supported; w/o Filler Mater
     *ial. Ends Free.'')')
            write(6,'(''      4>Ends Clamped; w/o Filler at the Sides. S
     *ides Free.'')')
            write(6,'(''      5>Ends Clamped; with Filler at the Sides.
     * Sides Free.'')')
            write(6,'(''     Enter >>''$)')
            call nintg(1,NBC)
            write(6,*)' '
C
            if (NBC.eq.2.or.NBC.eq.5) then
               write(6,'(''   Do You Want to Add or Change the Filler Ma
     *terial? [Y or N] '')')
               write(6,'('' ****WARNING!!! IF YOU CHANGE BOUNDARY CONDIT
     *ION'')')
               write(6,'('' TO INCLUDE FILLER ON THE SIDE, YOU WOULD HAV
     *E TO ADD '')')
               write(6,'('' FILLER MATERIAL PROPERTIES.****>>''$)')
               read(5,104)tok
               write(6,*)' '
               if (tok.eq.'Y'.or.tok.eq.'y')then
                  write(6,'(''     Add Filler Material. >>'')')
                  write(6,109)
                  read(5,110)nmt(7)
                  write(6,*)' '
                  write(6,'(''   Select 1)Isotropic. 2)Orthotropic. >>''
     *$)')
                  call nintg(1,NF)
                  write(6,*)' '
                  write(6,'(''   Input Material Information:'')')
                  if (NF.eq.1) then
                     write(6,'(''     Enter E(psi)>>''$)')
                     call nreal(1,E1(1,7))
                     write(6,*)' '
                     write(6,'(''     Enter V>>''$)')
                     call nreal(1,V1(1,7))
```

```
                  write(6,*)' '
               elseif (NF.eq.2) then
                  write(6,'(''     Enter E11, E22, E33(psi)>>''$)')
                  call nreal(3,E1(1,7),E1(2,7),E1(3,7))
                  write(6,*)' '
                  write(6,'(''     Enter G12, G23, G31 (psi)>>''$)')
                  call nreal(3,G1(1,7),G1(2,7),G1(3,7))
                  write(6,*)' '
                  write(6,'(''     Enter V12, V23, V31>>''$)')
                  call nreal(3,V1(1,7),V1(2,7),V1(3,7))
                  write(6,*)' '
               endif
            endif
         else
            continue
         endif
      elseif(M.eq.7)then
C
         write(6,'(''     Type of the Analysis:1>Stress.'')')
         write(6,'(''                          2>Bifurcation Buckling
     *.'')')
         write(6,'(''                          3>Post Buckling.'')')
         write(6,'(''     Enter >>''$)')
         call nintg(1,NBUC)
         write(6,*)' '
         if (NBUC.eq.3) then
            write(6,'('' Type of target >> 0)Load Factor.'')')
            write(6,'(''                  1-6)Displce.'')')
            write(6,'(''              Enter >>''$)')
            call NINTG(1,NDF)
            write(6,*)' '
            if (NDF.ge.1) then
               write(6,'('' Node # =''$)')
               call NINTG(1,ND)
               write(6,*)' '
            endif
            write(6,'('' Target Point = ''$)')
            call NREAL(1,TAR)
            write(6,*)' '
         elseif (NBUC .eq. 2) then
            write(6,'('' How many MODE shape(s) do you want (Max. 50)
     *? >>''$)')
            call NINTG(1,MS)
            write(6,*)' '
         endif
      else
      go to 10
      endif
12    go to 11
c.....
c     tmat.dat (unit11): material
c.....
      elseif(N.eq.2)then
         write(6,*)' '
13       write(6,'(''     1> Add New Material.'')')
         write(6,'(''     2> Modified Existing Material(or Add Filler Ma
     *terial).'')')
         write(6,'(''     3> Y/N on AML Failure Calculation.'')')
         write(6,'(''     4> Return to Previous Menu.'')')
         write(6,'(''       Select >>''$)')
```

```
      call nintg(1,MP)
      write(6,*)' '
      if (MP.eq.1) then
         write(6,'(''     **Please Keep the Material Number'')')
         write(6,'(''         (Excludes Filler Material) at Maximum o
     *f 6**'')')
         write(6,*)' '
         NMAT=NMAT+1
         E1(3,NMAT)=100.
         write(6,'('' Material Number >>'',i2)')NMAT
         write(6,109)
         read(5,110)nmt(NMAT)
         write(6,*)' '
         write(6,'(''   Enter Material Information >>'')')
         write(6,'(''     Enter E11, E22 (psi)>>''$)')
         call nreal(2,E1(1,NMAT),E1(2,NMAT))
         write(6,*)' '
         write(6,'(''     Enter G12, G23, G31 (psi)>>''$)')
         call nreal(3,G1(1,NMAT),G1(2,NMAT),G1(3,NMAT))
         write(6,*)' '
         write(6,'(''     Enter V12 >>''$)')
         call nreal(1,V1(1,NMAT))
         write(6,*)' '
         if (DA.eq.1) then
            call amlinp(NMAT,AMLA,AMLS,AMLT,DA)
         else
            do 77 i=1,NMAT
               do 78 j=1,6
                  AMLS(i,j)=0.
                  AMLA(i,j)=0.
78             continue
c               AMLT(i)=0.
77             continue
         endif
C
         write(6,*)' '
         go to 13
      elseif (MP.eq.2) then
14       write(6,'(''     Choose the Material to be Modified.'')')
         if (NBC.eq.2.or.NBC.eq.5) then
            NTMP=7
            DO 80 i=1,8
               if(i.le.NMAT)then
                  write(6,100)i,i
               elseif(i.eq.7)then
                  write(6,105)7
               elseif(i.eq.8)then
                  write(6,101)i
               endif
80             continue
               write(6,'(''     Enter >>''$)')
               call nintg(1,J)
               write(6,*)' '
               if (J.le.NMAT) then
                  call matmod(J,E1,G1,V1,NF,nmt)
                  if (DA.EQ.1.) THEN
                     call amlinp(J,AMLA,AMLS,AMLT,DA)
                  endif
                  go to 14
               elseif (J.eq.7) then
```

```
                call matmod(J,El,Gl,Vl,NF,nmt)
                go to 14
             elseif (J.eq.NTMP+1)then
                go to 13
             elseif (J.gt.8) then
                write(6,'(''     Entered Number Exceeds Total Number o
     *f Materials.'')')
                write(6,*)' '
                go to 14
             endif
             write(6,*)' '
          else
             NTMP=NMAT+1
15           DO 81 i=1,NTMP
                if(i.le.NMAT)then
                   write(6,100)i,i
                else
                   write(6,101)i
                endif
81           continue
             write(6,'(''     Enter >>''$)')
             call nintg(1,J)
             write(6,*)' '
             if (J.le.NMAT) then
                call matmod(J,El,Gl,Vl,NF,nmt)
                if (DA.eq.1.) then
                   call amlinp(J,AMLA,AMLS,AMLT,DA)
                endif
                go to 15
             elseif (J.eq.NTMP)then
                go to 13
             else
                write(6,'(''     Entered Number Exceeds Total Number o
     *f Material.'')')
                write(6,*)' '
                go to 15
             endif
             write(6,*)' '
          endif
       elseif (MP.eq.3) then
          if (DA.eq.1) then
             write(6,*)' '
             write(6,'(''    AML Failure Calculation IS INVOKED.'')')
          else
             write(6,*)' '
             write(6,'(''    AML Failure Calculation IS NOT INVOKED.''
     *)')
          endif
          write(6,'(''    Do You Want to Change It? [Y or N]>>''$)')
          read(5,104)tok
          write(6,*)' '
          if (tok.eq.'Y'.or.tok.eq.'y') then
             if (DA.eq.1.) then
                DA=0.
             elseif (DA.eq.0.) then
                DA=1.
             endif
          endif
          go to 13
       elseif (MP.eq.4) then
```

```
            go to 10
         endif
c.....
c        wall
c.....
      elseif(N.eq.3)then
         write(6,*)' '
20       write(6,'(''     Choose One of the Following ---'')')
         write(6,'(''      1>Change the Top Cover Sheet Layup.'')')
         write(6,'(''      2>Change the Bottom Cover Sheet Layup.'')')
         write(6,'(''      3>Change the Core Layup.'')')
         write(6,'(''      4>Return to the Previous Menu.'')')
         write(6,'(''     Enter>>''$)')
         call nintg(1,M)
         write(6,*)' '
         if(M.eq.1)then
            call walmod(MWT,NWT,TT,AT)
               TCT=0.
               do 83 k=1,MWT
                  TCT=TCT+TT(k)
83             continue
               write(6,*)' '
               go to 20
         elseif(M.eq.2)then
            call walmod(MWB,NWB,BT,AB)
               BCT=0.
               do 85 k=1,MWB
                  BCT=BCT+BT(k)
85             continue
               write(6,*)' '
               go to 20
          elseif(M.eq.3)then
            call walmod(MWC,NWC,CT,AC)
               CTT=0.
               do 87 k=1,MWC
                  CTT=CTT+CT(k)
87             continue
               go to 20
          else
            go to 10
          endif
c.....
c        load
c.....
       elseif(N.eq.4)then
          TMPD=(A+B)/2.*(NW+1)
          write(6,*)' '
29        write(6,'(''     Which load do you want to change?'')')
          write(6,'(''      1>End loads.'')')
          write(6,'(''      2>Side loads.'')')
          write(6,'(''      3>Shear load.'')')
          write(6,'(''      4>Return to Menu.'')')
          write(6,'(''     Enter >>''$)')
          call nintg(1,M)
          write(6,*)' '
          if(M.eq.1)then
             write(6,'(''     Do you want to enter the 1>Total Load or 2>
     *Line Load? >>'')')
             write(6,'(''     Enter >>''$)')
             call nintg(1,ML)
```

```
        write(6,*)' '
        if(ML.eq.1)then
           write(6,'('' Input total tensile (+) or compressive (
 *-) load (lb.)=''$)')
           call nreal(1,XL(2))
           XL(2)=XL(2)/TMPD
           XL(4)=XL(2)
           write(6,*)' '
           go to 29
        else
           write(6,'('' Input tensile (+) or compressive (-) lin
 *e load (lb./in)=''$)')
           call nreal(1,XL(2))
           XL(4)=XL(2)
           write(6,*)' '
           go to 29
        endif
     elseif(M.eq.2)then
        write(6,'('' Do you want to enter the 1>Total Load or 2>
 *Line Load? >>'')')
        write(6,'('' Enter >>''$)')
        call nintg(1,ML)
        write(6,*)' '
        if(ML.eq.1)then
           write(6,'('' Input total tensile (+) or compressive (
 *-) load (lb.)=''$)')
           call nreal(1,XL(1))
           XL(1)=XL(1)/pl
           XL(3)=XL(1)
           write(6,*)' '
           go to 29
        else
           write(6,'('' Input tensile (+) or compressive (-) lin
 *e load (lb./in)=''$)')
           call nreal(1,XL(1))
           XL(3)=XL(1)
           write(6,*)' '
           go to 29
        endif
     elseif(M.eq.3)then
        write(6,'('' Do you want to enter the 1>Total Load or 2>
 *Line Load? >>'')')
        write(6,'('' Enter >>''$)')
        call nintg(1,ML)
        write(6,*)' '
        if(ML.eq.1)then
           write(6,'('' 1> On the Ends.'')')
           write(6,'('' 2> On the Sides.'')')
           write(6,'('' Enter >>''$)')
           call nintg(1,M)
           write(6,*)' '
           if (M.eq.1) then
              write(6,'('' Total Shear Load (lb.)=''$)')
              call nreal(1,SL(2))
              SL(2)=SL(2)/TMPD
              SL(4)=SL(2)
              SL(1)=SL(2)
              SL(3)=SL(2)
              write(6,*)' '
              go to 29
```

```
            else
               write(6,'('' Total Shear Load (lb.)=''$)')
               call nreal(1,SL(1))
               SL(1)=SL(1)/pl
               SL(3)=SL(1)
               SL(2)=SL(1)
               SL(4)=SL(1)
               write(6,*)' '
               go to 29
            endif
         else
            write(6,'('' Shear Line Load (lb./in)=''$)')
            call nreal(1,SL(2))
            SL(4)=SL(2)
            SL(1)=SL(2)
            SL(3)=SL(2)
            write(6,*)' '
            go to 29
         endif
      else
         go to 10
      endif
   elseif(N.eq.5)then
      write(6,*)' '
      call tlis
      go to 10
   else
      write(6,*)' '
      write(6,'('' Do you want to quit'')')
      write(6,'('' input listing and modification? [Y/N]'')')
      write(6,'('' Enter >>''$)')
      read(5,104)tok
      if(tok.eq.'Y'.or.tok.eq.'y')then
C
         call tsave
         write(6,'('' *****End of Data Input Session.*****'')')
         go to 30
      else
         go to 10
      endif
   endif
30    return
      end
c
C     Read AML inputs
C
      SUBROUTINE AMLINP(ma,ang,str,tem,da)
C
      dimension ang(6,6),str(6,6),tem(6)
      integer ma
      real da
      character*1 tok,tok1
104   format(a1)
c.....
      write(6,'('' Current AML Curve Parameter for Material '',i2)')ma
      write(6,'('' ** Tension Strain Values **'')')
      do 20 k=1,6
         write(6,'('' Point '',i2,'' : Angle(deg.) = '',f6.2,
     *'' Strain Allowable = '',f9.5)')k,ang(k,ma),str(k,ma)
         if (k.eq.3) then
```

```
            write(6,'(''     ** Compression Strain Values **'')')
          endif
 20     continue
        write(6,*)' '
 900    write(6,'(''     Do You Want to Change (or Input) This AML Failure
       * Curve? [Y or N] >>''$)')
        read(5,104)tok
        write(6,*)' '
        if (tok.eq.'Y'.or.tok.eq.'y') then
           DA=1.
           write(6,'(''     Change the Entire Curve? [Y or N]''$)')
           read(5,104)tok1
           write(6,*)' '
           if (tok1.eq.'Y'.or.tok1.eq.'y') then
              write(6,*)' '
              write(6,'(''     ** Tension Strain Values **'')')
              do 25 k=1,6
                 write(6,'(''     Point '',i2)')k
                 write(6,'(''       Angle (deg.) = ''$)')
                 call nreal(1,ang(k,ma))
                 write(6,'(''       Strain Allowable = ''$)')
                 call nreal(1,str(k,ma))
                 write(6,*)' '
                 if (k.eq.3) then
                    write(6,'(''     ** Compression Strain Values **'')')
                 endif
 25           continue
              write(6,*)' '
           else
 950          write(6,'(''     Choose the Item to be Modified. >>'')')
              write(6,'(''     **  Tension Strain Values **'')')
              write(6,'(''     1> Point 1.'')')
              write(6,'(''     2> Point 2.'')')
              write(6,'(''     3> Point 3.'')')
              write(6,'(''     **  Compression Strain Values **'')')
              write(6,'(''     4> Point 4.'')')
              write(6,'(''     5> Point 5.'')')
              write(6,'(''     6> Point 6.'')')
              write(6,'(''     7> Return.'')')
              write(6,'(''     Enter >>''$)')
              call nintg(1,N2)
              write(6,*)' '
              if(N2.le.6)then
                 write(6,'(''     Point '',i2)')N2
                 write(6,'(''      Angle (deg.) = ''$)')
                 call nreal(1,ang(N2,ma))
                 write(6,'(''      Strain Allowable = ''$)')
                 call nreal(1,str(N2,ma))
                 write(6,*)' '
              elseif(N2.eq.7)then
                 go to 999
              endif
              go to 950
           endif
        else
           write(6,'(''     AML Curve for Material'',i2,'' Unchanged.'')')
           write(6,*)' '
        endif
 999    continue
           return
```

```
      end
c
c     MATMOD
c
      SUBROUTINE MATMOD(i,elas,rig,pois,nf,nmt)
      dimension elas(3,7),rig(3,7),pois(3,7)
      integer i
      character*1 tok
      character nmt*50
      dimension nmt(7)
400   format(a1)
401   format('   Enter New Material Name >> '$)
402   format('   Name >> ',a50)
403   format(a50)
c
c       i - material number >> J in TMOD
c       elas - elastic modulus >> El in main
c       rig - shear modulus >> Gl in main
c       pois - Poisson's ratio >> Vl in main
c
      if (i.eq.7) then
         go to 950
      endif
900   write(6,'(''     Do You Want to Change It to Another Material? [Y
     *or N] >>''$)')
      read(5,400)tok
      write(6,*)' '
      if (tok.eq.'Y'.or.tok.eq.'y') then
         write(6,401)
         read(5,403)nmt(i)
         write(6,*)' '
         write(6,'(''     Enter E11, E22 (psi)>>''$)')
         call nreal(2,elas(1,i),elas(2,i))
         write(6,*)' '
         write(6,'(''     Enter G12, G23, G31 (psi)>>''$)')
         call nreal(3,rig(1,i),rig(2,i),rig(3,i))
         write(6,*)' '
         write(6,'(''     Enter V12 >>''$)')
         call nreal(1,pois(1,i))
         write(6,*)' '
         go to 999
      elseif (tok.eq.'N'.or.tok.eq.'n') then
910      write(6,402)nmt(i)
         write(6,'(''     Choose the Item to be Modified. >>'')')
         write(6,'(''      1>Name'')')
         write(6,'(''      2>E11, E22'')')
         write(6,'(''      3>G12, G23, G31'')')
         write(6,'(''      4>V12'')')
         write(6,'(''      5>Continue'')')
         write(6,'(''     Enter >>''$)')
         call nintg(1,N1)
         write(6,*)' '
         if(N1.eq.1)then
            write(6,401)
            read(5,403)nmt(i)
            write(6,*)' '
            go to 910
         elseif(N1.eq.2)then
            write(6,'(''     E11, E22(psi)=''$)')
            call nreal(2,elas(1,i),elas(2,i))
```

```
            write(6,*)' '
            go to 910
         elseif(N1.eq.3)then
            write(6,'(''     G12, G23, G31(psi)=''$)')
            call nreal(3,rig(1,i),rig(2,i),rig(3,i))
            write(6,*)' '
            go to 910
         elseif(N1.eq.4)then
            write(6,'(''     V12=''$)')
            call nreal(1,pois(1,i))
            write(6,*)' '
            go to 910
         elseif(N1.eq.5)then
            go to 999
         endif
      endif
c......
c        Filler Material Modification.
c......
950   write(6,'(''     Do You Want to Add or Change the Filler Material?
     * [Y or N] >>''$)')
      read(5,400)tok
      write(6,*)' '
      if (tok.eq.'Y'.or.tok.eq.'y') then
         write(6,401)
         read(5,403)nmt(7)
         write(6,*)' '
         write(6,'(''     Select 1)Isotropic. 2)Orthotropic. >>''$)')
         call nintg(1,NF)
         write(6,*)' '
         if (NF.eq.1) then
            write(6,'(''     Enter E (psi) >>''$)')
            call nreal(1,elas(1,7))
            write(6,*)' '
            write(6,'(''     Enter V >>''$)')
            call nreal(1,pois(1,7))
            write(6,*)' '
         elseif (NF.eq.2) then
            write(6,'(''     Enter E11, E22 ,E33 (psi) >>''$)')
            call nreal(3,elas(1,7),elas(2,7),elas(3,7))
            write(6,*)' '
            write(6,'(''     Enter G12, G23, G31 (psi) >>''$)')
            call nreal(3,rig(1,7),rig(2,7),rig(3,7))
            write(6,*)' '
            write(6,'(''     Enter V12, V23, V31 >>''$)')
            call nreal(3,pois(1,7),pois(2,7),pois(3,7))
            write(6,*)' '
         endif
         go to 999
      elseif (tok.eq.'N'.or.tok.eq.'n') then
         write(6,*)' '
         if (NF.eq.1) then
            write(6,402)nmt(7)
            write(6,'(''   Isotropic.'')')
            write(6,'(''     Choose the Item to be Modified. >>'')')
            write(6,'(''      1>Elastic Modulus -- E'')')
            write(6,'(''      2>Poisson''''s Ratio -- V'')')
            write(6,'(''      Enter >>''$)')
            call nintg(1,N1)
            write(6,*)' '
```

```
            if (N1.eq.1) then
               write(6,'('' E (psi) =''$)')
               call nreal(1,elas(1,7))
               write(6,*)' '
            elseif (N1.eq.2) then
               write(6,'('' V =''$)')
               call nreal(1,pois(1,7))
               write(6,*)' '
            endif
         elseif (NF.eq.2) then
            write(6,402)nmt(7)
            write(6,'('' Orthotropic.'')')
            write(6,'('' Choose the Item to be Modified. >>'')')
            write(6,'('' 1>E11, E22, E33'')')
            write(6,'('' 2>G12, G23, G31'')')
            write(6,'('' 3>V12, V23, V31'')')
            write(6,'('' Enter >>''$)')
            call nintg(1,N1)
            write(6,*)' '
            if(N1.eq.1)then
               write(6,'('' E11, E22, E33 (psi)=''$)')
               call nreal(3,elas(1,7),elas(2,7),elas(3,7))
               write(6,*)' '
            elseif(N1.eq.2)then
               write(6,'('' G12, G23, G31(psi)=''$)')
               call nreal(3,rig(1,7),rig(2,7),rig(3,7))
               write(6,*)' '
            else
               write(6,'('' V12, V23, V31=''$)')
               call nreal(3,pois(1,7),pois(2,7),pois(3,7))
               write(6,*)' '
            endif
         endif
         go to 999
      endif
999   continue
         return
         end
c
c        wall layup modification
C
      SUBROUTINE WALMOD(NUM,MAT,THK,ANG)
      dimension MAT(99),THK(99),ANG(99)
      character*1 tok2,yn
c
c        Num -- corresponds to MWT,MWB,MWC in main. Number of Plies.
c        Mat -- corresponds to NWT,NWB,NWC in main. Ply Material Number.
c        Thk -- corresponds to TT,BT,CT in main. Ply Thickness.
c        Ang -- corresponds to AT,BT,CT in main. Ply angle.
c
100   format(a1)
101   format(' Is it the Same for All Plies? [Y or N] >>'$)
102   format(' Ply ',i2,':',/,' Material ',i2,' Thick.=',f7.5,' in.  Ori
     *entation=',f6.2,'Deg.')
40    write(6,*)' '
      write(6,'('' Do You Want to Change the Entire Lay-up? [Y or N]'
     *'$)')
      read(5,100)tok2
      write(6,*)' '
      if (tok2.eq.'Y'.or.tok2.eq.'y') then
```

```
          write(6,'('' Enter Total Number of Plies >>''$)')
          call nintg(1,NUM)
          write(6,*)' '
c.....
c         Wall Construction Input
c.....    Ply Material Number
C
          write(6,'('' Input Ply Material Number >>'')')
          write(6,101)
          read(5,100)yn
          write(6,*)' '
          if(yn.eq.'Y'.or.yn.eq.'y')then
             write(6,'('' Enter >>''$)')
             call nintg(1,MAT(1))
             do 15 i=2,NUM
             MAT(i)=MAT(1)
15           continue
             write(6,*)' '
          else
             write(6,'('' Input Material Number List for the Lay-up. ( To
     *tal of '',i2,'' Plies.) >>'')')NUM
             do 16 i=1,NUM
                write(6,'('' Enter >>''$)')
                call nintg(1,MAT(i))
16           continue
                write(6,*)' '
          endif
c            Ply Thicknesses
          write(6,'('' Input Ply Thickness (in.) >>'')')
          write(6,101)
          read(5,100)yn
          if (yn.eq.'Y'.or.yn.eq.'y') then
             write(6,'('' Enter >>''$)')
             call nreal(1,THK(1))
             do 17 i=2,NUM
                THK(i)=THK(1)
17           continue
             write(6,*)' '
          else
             write(6,'('' Input Ply Thickness List for the Lay-up. ( Tota
     *l of '',i2,'' Plies.) >>'')')NUM
             do 18 i=1,NUM
                write(6,'('' Enter >>''$)')
                call nreal(1,THK(i))
18           continue
             write(6,*)' '
          endif
c.....       Ply Orientations
          write(6,'('' Input Ply Orientation (deg.) >>'')')
          write(6,101)
          read(5,100)yn
          write(6,*)' '
          if (yn.eq.'Y'.or.yn.eq.'y') then
             write(6,'('' Enter >>''$)')
             call nreal(1,ANG(1))
             do 19 i=2,NUM
                ANG(i)=ANG(1)
19           continue
             write(6,*)' '
          else
```

```
            write(6,'('' Input Orientation List for the Lay-up. ( Total
     *of '',i2,'' Plies.) >>'')')NUM
            do 20 i=1,NUM
               write(6,'('' Enter >>''$)')
               call nreal(1,ANG(i))
20          continue
            write(6,*)' '
         endif
      else
c...
c.....   Individual Ply Changes
c...
50       write(6,'('' Current Layup Info ---'')')
         do 21 i=1,NUM
            write(6,102)i,MAT(i),THK(i),ANG(i)
21       continue
         write(6,*)' '
51       write(6,'('' Select the Ply to be Modified ---'')')
         write(6,'('' Enter >>''$)')
         call nintg(1,N)
         write(6,*)' '
         if (N.gt.NUM) then
            write(6,'('' Input Too Big. Try Again!!!'')')
            go to 51
         else
            write(6,'('' 1)Material Number 2)Thickness 3)Orientation
     *'')')
            write(6,'('' Enter >>''$)')
            call nintg(1,N2)
            write(6,*)' '
            if (N2.eq.1) then
               write(6,'('' Material Number for Ply '',i2,'' is >>''
     *$)')N
               call nintg(1,MAT(N))
               write(6,*)' '
            elseif (N2.eq.2) then
               write(6,'('' Thickness for Ply '',i2,'' is >>''$)')N
               call nreal(1,THK(N))
               write(6,*)' '
            else
               write(6,'('' Orientation for Ply '',i2,'' is >>''$)
     *')N
               call nreal(1,ANG(N))
               write(6,*)' '
            endif
         endif
      endif
      write(6,'('' Do You Want Further Changes? [Y or N] >>''$)')
      read(5,100)yn
      write(6,*)' '
      if (yn.eq.'Y'.or.yn.eq.'y') then
         go to 40
      endif
         return
         end
c...
c        MESH GENERATION
c...
      SUBROUTINE TMESH
C
```

```
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/  NMAT,E1(3,7),V1(3,7),G1(3,7),NF
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *               CT(99),AT(99),AB(99),AC(99)
      common /load/  XL(4),SL(4),XE,XS,SE,SS,TMPD
C
101   format('let &a=',f8.5,'       # &a = B = top width of the tube.')
102   format('    &b=',f8.5,'       # &b = A = bot width of the tube.')
103   format('    &ph0=',f9.4,'    # panel height,ts/t IF to bs/t IF.')
116   format('    &ph =',f9.4,'    # model height,ts/t MP to ts/t MP.')
104   format('    &pl=',f9.4,'     # length of the panel.')
106   format('    &nmat=',i2,'          #number of matl types used.')
107   format('    &tl=',f9.6,'     # thickness of the top cover ')
108   format('    &t2=',f9.6,'     # thickness of the tube.')
109   format('    &t3=',f9.6,'     # thickness of the bot. cover ')
110   format('let &nbuc=',i2,'   # = 1, static ana.;  =2, buckl ana.')
115   format('    &IFIN=',i2,'   # = 1, coarse mesh;  =2, fine mesh.')
111   format('    &nbc =',i2,'   # boundary condition indicator')
112   format('    &nw  =',i2,'   # number of the tubes.')
113   format('    &nev =',i2,'   # even number of tubes if nev=1')
114   format('    &k=',i2,5X,'        # no. of elements in x-dir.')
c___
c...     Warning!!
c...         The A-tube bottom width is &b in the mesh processor.
c...     and the B-tube top    width is &a in the mesh processor.
c___
         open (unit=12,status='unknown',file='tmesh.com')
C
      IFIN=1
      t=(A+B)/2.
      K=NINT(PL/(3.*t))
      KEV=K-K/2*2
      IF (KEV.GT.0) K=K+1
      if (K.le.6) K=6
C
      IF (NBUC.EQ.2) IFIN=2
      IF (IFIN.EQ.2) K=NINT(PL/(2.*t))
      KEV=K-K/2*2
      IF (KEV.GT.0) K=K+1
      IF (IFIN.EQ.2 .AND. K.LE.8 ) K=8
C
      write(12,'(''$mesh'')')
      write(12,'(''OPEN 6''''mesh.out'')')
      write(12,'(''CLEAR -1'')')
      write(12,'(''max 20000 20000'')')
      write(12,'(''ELTYPE 4,2,6'')')
      write(12,'(''assign IPNO=0 IPEQ=1 IPSU=1 iplc=0  '')')
      write(12,'(''set syntax on'')')
C
      write(12,'(''#'')')
      write(12,'(''#   CONODE subroutine'')')
      write(12,'(''#'')')
      write(12,'('' sub CONO &lst1 &lst2 &lst3  &Itype '')')
      write(12,'('' set echo on'')')
      write(12,'('' let &nnod=%ipp(1)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst2)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &nl=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
```

```
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst3)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n2=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst1)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'(''#'')')
      write(12,'('' ;f2 format ''''(1x,i5)'')')
      write(12,'('' ;f3 format ''''(''''Print Nodal Coordinates'''',1x,'
     *'''Node'''')'')')
      write(12,'('' write 6 ;f2 &n1'')')
      write(12,'('' write 6 ;f2 &n2'')')
      write(12,'('' write 6 ;f3'')')
      write(12,'(''#'')')
      write(12,'('' do ;20 &n=1,&nnod'')')
      write(12,'('' if %ibcl(&ln,&n) ;20,;20,1'')')
      write(12,'('' let &nn=%ibcl(&ln,&n)'')')
      write(12,'('' write 6 ;f2 &nn'')')
      write(12,'(''#                                    '')')
      write(12,'('' conode &nn &n1 &n2  &Itype  1.E+8   '')')
      write(12,'(''#                                    '')')
      write(12,'('' ;20 continue'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
      write(12,'('' return'')')
      write(12,'('' end'')')
C
      write(12,'(''#'')')
      write(12,'(''#   parameter input'')')
      write(12,'(''#'')')
C
      PH2=PH+0.5*(TCT+BCT)-CTT
      write(12,101)B
      write(12,102)A
      write(12,103)PH
      write(12,116)PH2
      write(12,104)PL
      write(12,114)K
      write(12,'(''#'')')
      write(12,106)NMAT
      write(12,107)TCT
      write(12,108)CTT
      write(12,109)BCT
      write(12,'(''#'')')
C
C       Save these for later use
C
C      write(12,'('' &of1=&t1/2.   #offset of the top skin.'')')
C      write(12,'('' &of3=&t3/2.'')')
C      write(12,'('' &of3=-1.*&of3 #offset of the bottom skin. '')')
C      write(12,'('' &of4=&t2/2.   #offset of the tube bottom'')')
C      write(12,'('' &of5=&of4*-1. #offset of the tube top'')')
C
      write(12,'('' &of1=0.        # offset of the top skin.   '')')
      write(12,'('' &of3=0.        # offset of the bot.skin.    '')')
      write(12,'('' &of4=0.        # offset of the tube bottom '')')
      write(12,'('' &of5=0.        # offset of the tube top    '')')
      write(12,'(''#'')')
C
      write(12,'('' &k=&k*2'')')
      write(12,'('' &k=&k+1'')')
```

```
      write(12,'('' &z1=&a/2.'')')
      write(12,'('' &z2=&b/2.'')')
      write(12,'('' &z=&z1+&z2'')')
      write(12,'('' &z2=&z*2.'')')
      write(12,'('' &z3=&z+&z'')')
C
      write(12,'(''#                    variables for fine mesh gen.'')')
      write(12,'('' &NE=4'')')
      write(12,'('' &NE1=&NE+1'')')
      write(12,'('' &NE3=&NE+3'')')
      write(12,'('' &NE5=&NE+5'')')
      write(12,'('' &NE21=&NE*2+1'')')
      write(12,'('' &NE23=&NE*2+3'')')
      write(12,'('' &NE25=&NE*2+5'')')
      write(12,'(''#'')')
      write(12,'('' ;f22 format ''''('''' * NE,NE1,...NE25 *'''') '')')
      write(12,'('' ;f2  format ''''(1x,4I5)'')')
      write(12,'('' write 6 ;f22     '')')
      write(12,'('' write 6 ;f2 &NE,&NE1,&NE3,&NE5  '')')
      write(12,'('' write 6 ;f2 &NE21,&NE23,&NE25   '')')
      write(12,'(''#'')')
C
      write(12,110)NBUC
      write(12,115)IFIN
      write(12,111)NBC
      write(12,112)NW
      write(12,'('' &nm=2      # mesh number counter.'')')
      write(12,'('' &nn=1      # indicator for auto-mesh gen. '')')
      write(12,'(''#             of filler material.'')')
C
      WN=NW*1.
      IF (MOD(WN,2.).GT.0.0)THEN
         NEV=2
      ELSE
         NEV=1
      ENDIF
      write(12,113)NEV
      write(12,'(''#               odd  number of tubes if nev=2'')')
C
      write(12,'(''#'')')
      write(12,'('' let &ntmp=&nmat            '')')
      write(12,'('' if &nbc-2 ;pok 1 ;pok '')')
      write(12,'('' let &ntmp=7                '')')
      write(12,'('' goto ;nxt                  '')')
      write(12,'('' ;pok if &nbc-5 ;nxt 1 ;nxt '')')
      write(12,'('' let &ntmp=7                '')')
      write(12,'('';nxt continue'')')
      write(12,'(''#'')')
      write(12,'(''if &IFIN-1 1 1 ;fin'')')
      write(12,'(''#  ------------------------------------------'')')
C
      write(12,'(''#   '')')
      write(12,'(''#   Coarse mesh generation   '')')
      write(12,'(''#   '')')
C
      write(12,'(''#   LEFT HALF MESH'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 1   1   1   0.   0.    0.'')')
      write(12,'('' 2   3   1   0.   &a/2. 0.'')')
      write(12,'('' 3   5   1   0.   &z    0.'')')
```

```
      write(12,'(''sline  1t3'')')
      write(12,'(''ijshell 1,3 &t3 &t3 &of3 &of3 &ntmp+5 msh 1 4 0 0. BO
     *T'')')
      write(12,'(''#'')')
C
      write(12,'(''ijpoint 4   1   3   0.   0.    0.'')')
      write(12,'(''        5   3   3   0.   &a/2. 0.'')')
      write(12,'(''        6   5   3   0.   &z    0.'')')
      write(12,'(''#'')')
      write(12,'(''        7   1   7   0.   0.    &ph'')')
      write(12,'(''        8   3   7   0.   &b/2. &ph'')')
      write(12,'(''        9   5   7   0.   &z    &ph'')')
      write(12,'(''sline   4t6:7t9:5,8'')')
      write(12,'(''#'')')
      write(12,'(''ijshell 4,6 &t2 &t2 &of4 &of4 &ntmp+4 msh 1 4 0 0. BO
     *XB'')')
      write(12,'(''ijshell 5,8 &t2 &t2 0.   0.   &ntmp+2 msh 1 4 0 0. BO
     *XW'')')
      write(12,'(''ijshell 7,9 &t2 &t2 &of5 &of5 &ntmp+3 msh 1 4 0 0. BO
     *XT'')')
      write(12,'(''#'')')
C
      write(12,'(''ijpoint 10   1  9  0.  0.    &ph'')')
      write(12,'(''        11   3  9  0.  &b/2. &ph'')')
      write(12,'(''        12   5  9  0.  &z    &ph'')')
      write(12,'(''sline   10t12'')')
      write(12,'(''ijshell 10 12 &t1 &t1 &of1 &of1 &ntmp+1 msh 1 4 0 0.
     *TOP '')')
      write(12,'(''#'')')
      write(12,'(''prism &k &pl'')')
      write(12,'(''mesh'')')
      write(12,'(''merge'')')
C
      write(12,'(''#'')')
      write(12,'(''#   RIGHT HALF MESH'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 1   1   1   0.   0.    0.'')')
      write(12,'(''        2   3   1   0.   &b/2. 0.'')')
      write(12,'(''        3   5   1   0.   &z    0.'')')
      write(12,'(''sline  1t3'')')
      write(12,'(''ijshell 1,3 &t3 &t3 &of3 &of3 &ntmp+5 msh 1 4 0 0. BO
     *T'')')
      write(12,'(''#'')')
C
      write(12,'(''ijpoint 4   1   3   0.   0.    0.'')')
      write(12,'(''        5   3   3   0.   &b/2. 0.'')')
      write(12,'(''        6   5   3   0.   &z    0.'')')
      write(12,'(''#'')')
      write(12,'(''        7   1   7   0.   0.    &ph'')')
      write(12,'(''        8   3   7   0.   &a/2. &ph'')')
      write(12,'(''        9   5   7   0.   &z    &ph'')')
      write(12,'(''sline   4t6:7t9:5,8'')')
      write(12,'(''#'')')
      write(12,'(''ijshell 4,6 &t2 &t2  &of4 &of4 &ntmp+4 msh 1 4 0 0. B
     *OXB '')')
      write(12,'(''ijshell 5,8 &t2 &t2  0.   0.   &ntmp+2 msh 1 4 0 0. B
     *OXW '')')
      write(12,'(''ijshell 7,9 &t2 &t2  &of5 &of5 &ntmp+3 msh 1 4 0 0. B
     *OXT '')')
      write(12,'(''#'')')
```

```
      write(12,'(''ijpoint 10   1  9  0.  0.    &ph'')')
      write(12,'(''        11   3  9  0.  &a/2. &ph'')')
      write(12,'(''        12   5  9  0.  &z    &ph'')')
      write(12,'(''sline   10t12'')')
      write(12,'(''ijshell 10 12 &t1 &t1 &of1 &of1 &ntmp+1 msh 1 4 0 0.
     *TOP '')')
      write(12,'(''#'')')
C
      write(12,'(''prism &k &p1'')')
      write(12,'(''mesh'')')
      write(12,'(''trans &z 2'')')
      write(12,'(''merge'')')
      write(12,'(''let &y1=&z3 #right edge nodes boundary indicator'')')
C
      write(12,'(''#'')')
      write(12,'(''#   AUTO PART GENERATION'')')
      write(12,'(''#'')')
      write(12,'(''set echo on'')')
      write(12,'(''if &nw-1 ;11 ;11 1'')')
      write(12,'(''#'')')
      write(12,'(''do ;10 &i=2,&nw'')')
      write(12,'(''    if &nn-1 1 1 ;13'')')
      write(12,'(''    ditto mesh 1'')')
      write(12,'(''    trans &z3 2'')')
      write(12,'(''    merge'')')
      write(12,'(''    let &nm=&nm+1'')')
      write(12,'(''        &nn=2'')')
      write(12,'(''        &y1=&y1+&z'')')
      write(12,'(''     goto ;1      '')')
      write(12,'('';13 ditto mesh 2  '')')
      write(12,'(''    trans &z3 2'')')
      write(12,'(''    merge '')')
      write(12,'(''    let &nm=&nm+1'')')
      write(12,'(''        &nn=1'')')
      write(12,'(''    let &z3=&z3+&z'')')
      write(12,'(''        &z3=&z3+&z'')')
      write(12,'(''        &y1=&y1+&z'')')
      write(12,'('';1 remark '''' Loop complete'')')
      write(12,'('';10 continue'')')
      write(12,'('';11 continue'')')
      write(12,'(''#'')')
C
      write(12,'(''set echo off'')')
      write(12,'(''mset  1 dele all'')')
      write(12,'(''mset  2 dele all'')')
      write(12,'(''mset  3 dele all'')')
      write(12,'(''mset  4 dele all'')')
      write(12,'(''mset  9 dele all'')')
      write(12,'(''mset 10 dele all'')')
      write(12,'(''mset 11 dele all'')')
      write(12,'(''#'')')
      write(12,'(''mset  1 insert name=TOP'')')
      write(12,'(''mset  2 insert name=BOXW'')')
      write(12,'(''mset  3 insert name=BOT'')')
      write(12,'(''mset  9 insert name=BOXT'')')
      write(12,'(''mset 10 insert name=BOXB'')')
      write(12,'(''mset 11 insert mset=2,9,10 '')')
      write(12,'(''#'')')
C
      write(12,'(''#   FILLER MATERIAL GENERATION'')')
```

```
      write(12,'(''#'')')
      write(12,'(''#  det. in FORTRAN pro. how many tubes are there'')')
      write(12,'(''#  use that as the basis for the following mesh'')')
      write(12,'(''#'')')
      write(12,'(''if &nbc-2 ;f 2 1'')')
      write(12,'(''if &nbc-5 ;f 1 ;f'')')
      write(12,'(''#'')')
      write(12,'('' ijpoint 1   1   1   0.   0.    0.'')')
      write(12,'('''          2   3   1   0.   &a/2. 0.'')')
      write(12,'('''          3   3   5   0.   &b/2. &ph'')')
      write(12,'('''          4   1   5   0.    0.   &ph'')')
      write(12,'('' sline  1t4,1'')')
      write(12,'('' ijsolid 1 3 7 so 0 FIL'')')
      write(12,'(''#'')')
      write(12,'('' prism &k &pl'')')
      write(12,'('' mesh'')')
      write(12,'('' merge mesh=1'')')
      write(12,'('' mset 4 insert name=FIL '')')
      write(12,'('' mset 45 insert mset 4'')')
      write(12,'('' ditto mset 4'')')
      write(12,'('' mirror 2 1'')')
      write(12,'(''#'')')
      write(12,'('' if &nev-1 1,1,;od'')')
      write(12,'('''    mirror 3 1'')')
      write(12,'('''    trans &z3+&z 2'')')
      write(12,'('''    trans &ph 3'')')
      write(12,'('''    merge'')')
      write(12,'('''    mset 4 insert name=FIL'')')
      write(12,'('''    eltsen 1 0 0 mset 4'')')
      write(12,'('''    goto ;f'')')
      write(12,'('';od  continue '')')
      write(12,'('''    trans &z3 2'')')
      write(12,'('''    merge'')')
      write(12,'('''    mset 4 insert name=FIL'')')
      write(12,'('''    eltsen 1 0 0 mset 45'')')
      write(12,'('';f  continue'')')
      write(12,'(''goto ;fi'')')
      write(12,'(''#   ------------------------------------------'')')
C
      write(12,'('';fin continue'')')
      write(12,'(''#  '')')
      write(12,'(''#   Fine mesh generation      '')')
      write(12,'(''#  '')')
C
      write(12,'(''#   LEFT HALF MESH'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 1   3   1   0.   0.   0.'')')
      write(12,'('''       2   5   1   0.  &a/2. 0.'')')
      write(12,'('''       3 &NE5  1   0.   &z   0.'')')
C     write(12,'('''      13   1   1   0.  -1.   0.'')')
      write(12,'(''sline  1t3'')')
      write(12,'(''ijshell 1,3 &t3 &t3 &of3 &of3 &ntmp+5 msh 1 4 0 0. BO
     *T'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 4   3   3   0.   0.   0.'')')
      write(12,'('''       5   5   3   0.  &a/2. 0.'')')
      write(12,'('''       6 &NE5  3   0.   &z   0.'')')
C     write(12,'('''      14   1   3   0.  -1.   0.'')')
      write(12,'(''#'')')
C
```

```
      write(12,'(''        7   1  &NE23   0.    0.   &ph'')')
      write(12,'(''        8 &NE1 &NE23   0.   &b/2. &ph'')')
      write(12,'(''        9 &NE3 &NE23   0.    &z   &ph'')')
      write(12,'(''sline   4t6:7t9:5,8'')')
      write(12,'(''#'')')
      write(12,'(''ijshell 4,6 &t2 &t2 &of4 &of4 &ntmp+4 msh 1 4 0 0. BO
     *XB'')')
      write(12,'(''ijshell 5,8 &t2 &t2 0.   0.   &ntmp+2 msh 1 4 0 0. BO
     *XW'')')
      write(12,'(''ijshell 7,9 &t2 &t2 &of5 &of5 &ntmp+3 msh 1 4 0 0. BO
     *XT'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 10   1  &NE25  0.  0.   &ph'')')
      write(12,'(''        11 &NE1 &NE25  0. &b/2. &ph'')')
      write(12,'(''        12 &NE3 &NE25  0.  &z   &ph'')')
      write(12,'(''sline   10t12'')')
      write(12,'(''ijshell 10 12 &t1 &t1 &of1 &of1 &ntmp+1 msh 1 4 0 0.
     *TOP '')')
      write(12,'(''#'')')
C
      write(12,'(''prism &k &p1'')')
      write(12,'(''mesh'')')
      write(12,'(''merge'')')
C
      write(12,'(''#'')')
      write(12,'(''#  RIGHT HALF MESH'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 1   1   1   0.   0.   0.'')')
      write(12,'(''        2 &NE1  1   0.  &b/2. 0.'')')
      write(12,'(''        3 &NE3  1   0.   &z   0.'')')
      write(12,'(''sline  1t3'')')
      write(12,'(''ijshell 1,3 &t3 &t3 &of3 &of3 &ntmp+5 msh 1 4 0 0. BO
     *T'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 4   1   3    0.  0.   0.'')')
      write(12,'(''        5 &NE1  3    0. &b/2. 0.'')')
      write(12,'(''        6 &NE3  3    0.  &z   0.'')')
      write(12,'(''#'')')
      write(12,'(''        7   3  &NE23 0.  0.   &ph'')')
      write(12,'(''        8   5  &NE23 0. &a/2. &ph'')')
      write(12,'(''        9 &NE5 &NE23 0.  &z   &ph'')')
      write(12,'(''sline   4t6:7t9:5,8'')')
      write(12,'(''#'')')
      write(12,'(''ijshell 4,6 &t2 &t2  &of4 &of4 &ntmp+4 msh 1 4 0 0. B
     *OXB'')')
      write(12,'(''ijshell 5,8 &t2 &t2  0.   0.   &ntmp+2 msh 1 4 0 0. B
     *OXW'')')
      write(12,'(''ijshell 7,9 &t2 &t2  &of5 &of5 &ntmp+3 msh 1 4 0 0. B
     *OXT'')')
      write(12,'(''#'')')
      write(12,'(''ijpoint 10   3  &NE25  0.  0.   &ph'')')
      write(12,'(''        11   5  &NE25  0. &a/2. &ph'')')
      write(12,'(''        12 &NE5 &NE25  0. &z    &ph'')')
      write(12,'(''sline   10t12'')')
      write(12,'(''ijshell 10 12 &t1 &t1 &of1 &of1 &ntmp+1 msh 1 4 0 0.
     *TOP '')')
      write(12,'(''#'')')
C
      write(12,'(''prism &k &p1'')')
      write(12,'(''mesh'')')
```

```
      write(12,'(''trans &z 2'')')
      write(12,'(''merge'')')
      write(12,'(''let &yl=&z3 #right edge nodes boundary indicator'')')
C
      write(12,'(''#'')')
      write(12,'(''#   AUTO PART GENERATION'')')
      write(12,'(''#'')')
      write(12,'(''set echo on'')')
      write(12,'(''    if &nw-1 ;21 ;21 1'')')
      write(12,'(''do ;22 &i=2,&nw'')')
      write(12,'(''    if &nn-1 1 1 ;23'')')
      write(12,'(''    ditto mesh 1'')')
      write(12,'(''    trans &z3 2'')')
      write(12,'(''    merge'')')
      write(12,'(''    let &nm=&nm+1'')')
      write(12,'(''        &nn=2'')')
      write(12,'(''        &yl=&yl+&z'')')
      write(12,'(''     goto ;11'')')
      write(12,'('';23 ditto mesh 2'')')
      write(12,'(''    trans &z3 2'')')
      write(12,'(''    merge '')')
      write(12,'(''    let &nm=&nm+1'')')
      write(12,'(''        &nn=1'')')
      write(12,'(''    let &z3=&z3+&z'')')
      write(12,'(''        &z3=&z3+&z'')')
      write(12,'(''        &yl=&yl+&z'')')
      write(12,'('';11 remark '''' Loop complete'')')
      write(12,'('';22 continue'')')
      write(12,'('';21 continue'')')
      write(12,'(''#'')')
C
      write(12,'(''set echo off'')')
      write(12,'(''mset  1 dele all'')')
      write(12,'(''mset  2 dele all'')')
      write(12,'(''mset  3 dele all'')')
      write(12,'(''mset  4 dele all'')')
      write(12,'(''mset  9 dele all'')')
      write(12,'(''mset 10 dele all'')')
      write(12,'(''mset 11 dele all'')')
      write(12,'(''#'')')
      write(12,'(''mset  1 insert name=TOP'')')
      write(12,'(''mset  2 insert name=BOXW'')')
      write(12,'(''mset  3 insert name=BOT'')')
      write(12,'(''mset  9 insert name=BOXT'')')
      write(12,'(''mset 10 insert name=BOXB'')')
      write(12,'(''mset 11 insert mset=2,9,10 '')')
C
      write(12,'(''#'')')
      write(12,'(''#   FILLER MATERIAL GENERATION'')')
      write(12,'(''#'')')
      write(12,'(''# det. in fortran prog. how many tubes are there'')')
      write(12,'(''# use that as the basis for the following mesh'')')
      write(12,'(''#'')')
      write(12,'(''if &nbc-2 ;fi 2 1'')')
      write(12,'(''if &nbc-5 ;fi 1 ;fi'')')
      write(12,'(''#'')')
      write(12,'(''   ijpoint 1   1   1    0.    0.   0.'')')
      write(12,'(''           2 &NE1  1    0. &a/2.   0.'')')
      write(12,'(''           3 &NE1 &NE21 0. &b/2.  &ph'')')
      write(12,'(''           4   1  &NE21 0.    0.  &ph'')')
```

```
      write(12,'(''   sline  1t4,1'')')
      write(12,'(''   ijsolid 1 3 7 so 0 FIL'')')
      write(12,'(''#'')')
C
      write(12,'(''   prism &k &p1'')')
      write(12,'(''   mesh'')')
      write(12,'(''   merge mesh=1'')')
      write(12,'(''   mset 4 insert name=FIL '')')
      write(12,'(''   mset 45 insert mset 4'')')
      write(12,'(''   ditto mset 4'')')
      write(12,'(''   mirror 2 1'')')
      write(12,'(''#'')')
      write(12,'(''   if &nev-1 1,1,;odd'')')
      write(12,'(''      mirror 3 1'')')
      write(12,'(''      trans &z3+&z 2'')')
      write(12,'(''      trans &ph 3'')')
      write(12,'(''      merge'')')
      write(12,'(''      mset 4 insert name=FIL'')')
      write(12,'(''      eltsen 1 0 0 mset 4'')')
      write(12,'(''      goto ;fi'')')
      write(12,'('';odd continue '')')
      write(12,'(''      trans &z3 2'')')
      write(12,'(''      merge'')')
      write(12,'(''      mset 4 insert name=FIL'')')
      write(12,'(''      eltsen 1 0 0 mset 45'')')
      write(12,'(''#   ------------------------------------------------'')')
      write(12,'('';fi continue'')')
C
      write(12,'(''#   ---------------   '')')
      write(12,'(''#          '')')
C     write(12,'(''#   CONODE '')')
C     write(12,'(''#          '')')
      write(12,'(''nset 1 insert mset 1'')')
      write(12,'(''nset 1 insert mset 3  #all nodes in top and bottom fa
     *ce sheets.'')')
      write(12,'(''nset 10 insert mset 3'')')
      write(12,'(''#   ---------------   '')')
C
      write(12,'(''#'')')
      write(12,'(''#   Boundary condition Setup'')')
      write(12,'(''#'')')
      write(12,'(''# &nbc=1 infinite panel'')')
      write(12,'(''#     =2 sides simply supported with filler material'
     *')')
      write(12,'(''#     =3 sides simply supported w/o filler material''
     *)')
      write(12,'(''#     =4 ends clamped; w/o filler at sides'')')
      write(12,'(''#     =5 ends clamped; with filler at sides'')')
      write(12,'(''#'')')
      write(12,'(''if &nbc-1 1,1,;p2'')')
C
      write(12,'(''#   case one, part of a larger panel'')')
      write(12,'(''      bcsys -5 0 1 0.  1.e+9 nset=1'')')
      write(12,'(''      bcsys -4 0 2 0.  1.e+9 nset=1'')')
      write(12,'(''      bcsys -5 0 1 &p1 1.e+9 nset=1'')')
      write(12,'(''      bcsys -4 0 2 &y1 1.e+9 nset=1'')')
      write(12,'(''      goto ;con'')')
      write(12,'(''#'')')
C
      write(12,'(''#   cases 2 and 3, sides are simply supported'')')
```

```
      write(12,'(''&p2   if &nbc-3 1 1 &p4'')')
      write(12,'(''      bcsys -5 0 2 0.  1.e+9 nset=10'')')
      write(12,'(''      bcsys -3 0 2 0.  1.e+9 nset=10'')')
      write(12,'(''      bcsys -5 0 2 &y1 1.e+9 nset=10'')')
      write(12,'(''      bcsys -3 0 2 &y1 1.e+9 nset=10'')')
      write(12,'(''      goto &con'')')
      write(12,'(''#'')')
C
      write(12,'(''#   cases four & five, ends are clamped'')')
      write(12,'(''&p4   bcsys -4 0 1 0. 1.e+9 nset=1'')')
      write(12,'(''      bcsys -5 0 1 0. 1.e+9 nset=1'')')
      write(12,'(''      bcsys -3 0 1 0. 1.e+9 nset=1'')')
      write(12,'(''      bcsys -2 0 1 0. 1.e+9 nset=1'')')
      write(12,'(''#'')')
      write(12,'(''      bcsys -4 0 1 &p1 1.e+9 nset=1'')')
      write(12,'(''      bcsys -5 0 1 &p1 1.e+9 nset=1'')')
      write(12,'(''      bcsys -3 0 1 &p1 1.e+9 nset=1'')')
      write(12,'(''      bcsys -2 0 1 &p1 1.e+9 nset=1'')')
      write(12,'(''&con  continue'')')
      write(12,'(''#   ---------------   '')')
      write(12,'(''#'')')
C
      write(12,'(''nset 2 insert nset 1'')')
      write(12,'(''nset 2 mask volu=-999. 999. -0.01 0.01 -999. 999.''
     *')
      write(12,'(''nset 2 list          #all nodes on left side.'')')
      write(12,'(''#'')')
      write(12,'(''nset 3 insert nset 1'')')
      write(12,'(''nset 3 mask volu=-999. 999. &y1-0.01 &y1+0.01 -999. 9
     *99.'')')
      write(12,'(''nset 3 list          #all nodes on right side.'')')
      write(12,'(''#'')')
      write(12,'(''nset 4 insert nset 1'')')
      write(12,'(''nset 4 mask volu=-0.01 0.01 -999. 999. -999. 999.''
     *')
      write(12,'(''nset 4 list          #all nodes on front end.'')')
      write(12,'(''#'')')
      write(12,'(''nset 5 insert nset 1'')')
      write(12,'(''nset 5 mask volu=&p1-0.01 &p1+0.01 -999. 999. -999. 9
     *99.'')')
      write(12,'(''nset 5 list          #all nodes on back end. '')')
C
      write(12,'(''#   ---------------   '')')
      write(12,'(''#'')')
      write(12,'(''nset 6 insert nset 2'')')
      write(12,'(''nset 6 mask nset 4'')')
      write(12,'(''nset 6 list          #front left corner nodes'')')
      write(12,'(''#'')')
      write(12,'(''nset 7 insert nset 4'')')
      write(12,'(''nset 7 mask nset 3'')')
      write(12,'(''nset 7 list          #front right corner nodes'')')
      write(12,'(''#'')')
      write(12,'(''nset 8 insert nset 3'')')
      write(12,'(''nset 8 mask nset 5'')')
      write(12,'(''nset 8 list          #back right corner nodes'')')
      write(12,'(''#'')')
      write(12,'(''nset 9 insert nset 5'')')
      write(12,'(''nset 9 mask nset 2'')')
      write(12,'(''nset 9 list          #back left corner nodes'')')
C
```

```
      write(12,'(''#                        '')')
      write(12,'(''nset 76 insert nset 9  '')')
      write(12,'(''nset 77 insert nset 8  '')')
      write(12,'(''nset 78 insert nset 6  '')')
      write(12,'(''nset 79 insert nset 7  '')')
C
      write(12,'(''#                                                '')')
      write(12,'(''#      All Nodes in the Above Sets Are in the''')')
      write(12,'(''#      Top and Bottom Face Sheets.               '')')
C
      write(12,'(''#   ---------------   '')')
      write(12,'(''nset  99 insert nset 7'')')
      write(12,'(''nset  99 mask mset 3  #front bottom-right corner node
     *'')')
      write(12,'(''nset  99 list'')')
      write(12,'(''nlist 99 insert nset 99'')')
      write(12,'(''#'')')
      write(12,'(''nlist 98 insert node=1 #front bottom-left corner node
     *'')')
      write(12,'(''nlist 98 list'')')
      write(12,'(''#'')')
      write(12,'(''nset  97 insert nset 8'')')
      write(12,'(''nset  97 mask mset 3  #back bottom-right corner node'
     *')')
      write(12,'(''nset  97 list'')')
      write(12,'(''nlist 97 insert nset 97'')')
      write(12,'(''#'')')
      write(12,'(''nset  96 insert nset 9'')')
      write(12,'(''nset  96 mask mset 3  #back bottom-left corner node''
     *)')
      write(12,'(''nset  96 list'')')
      write(12,'(''nlist 96 insert nset 96'')')
      write(12,'(''#   ---------------   '')')
C
      write(12,'(''nset 76 del nlist 96  '')')
      write(12,'(''nset 77 del nlist 97  '')')
      write(12,'(''nset 78 del nlist 98  '')')
      write(12,'(''nset 79 del nlist 99  '')')
      write(12,'(''#             '')')
      write(12,'(''nset 76 list '')')
      write(12,'(''nset 77 list '')')
      write(12,'(''nset 78 list '')')
      write(12,'(''nset 79 list '')')
C
      write(12,'(''#'')')
      write(12,'(''#   all nodes on left edge.'')')
      write(12,'(''nset 80 insert volu=-999. 999. -0.01 0.01 -999. 999.'
     *')')
      write(12,'(''mset 6 insert mset 1,3'')')
      write(12,'(''mset 6 mask nset 80'')')
      write(12,'(''#'')')
      write(12,'(''#   all nodes on right edge.'')')
      write(12,'(''nset 81 insert volu=-999. 999. &y1-0.01 &y1+0.01 -999
     *. 999.'')')
      write(12,'(''mset 5 insert mset 1,3'')')
      write(12,'(''mset 5 mask nset 81'')')
      write(12,'(''#'')')
      write(12,'(''#   all nodes on front edge.'')')
      write(12,'(''nset 82 insert volu=-0.01 0.01 -999. 999. -999. 999.'
     *')')
```

```
      write(12,'(''NSET 88 INSERT NSET 82 '')')
C
      write(12,'(''mset 7 insert mset 1,3'')')
      write(12,'(''mset 7 mask nset 82'')')
      write(12,'(''#'')')
      write(12,'(''#   all nodes on back edge.'')')
      write(12,'(''nset 83 insert volu=&pl-0.01 &pl+0.01 -999. 999. -999
     *. 999.'')')
      write(12,'(''mset 8 insert mset 1,3'')')
      write(12,'(''mset 8 mask nset 83'')')
      write(12,'(''#'')')
      write(12,'(''#   ---------------   '')')
C
      write(12,'(''nset 80 del nset 82'')')
      write(12,'(''nset 80 del nset 83'')')
      write(12,'(''nset 81 del nset 82'')')
      write(12,'(''nset 81 del nset 83'')')
      write(12,'(''nset 82 dele nset 6 7'')')
      write(12,'(''nset 83 dele nset 9 8'')')
      write(12,'(''#   '')')
C
      write(12,'(''nlist 80 insert nset 80'')')
      write(12,'(''nlist 81 insert nset 81'')')
      write(12,'(''nlist 82 insert nset 82 '')')
      write(12,'(''nlist 83 insert nset 83'')')
      write(12,'(''nlist 88 insert nset 88'')')
C
      write(12,'(''#'')')
      write(12,'(''nlist 80 list'')')
      write(12,'(''nlist 81 list'')')
      write(12,'(''nlist 82 list'')')
      write(12,'(''nlist 83 list'')')
      write(12,'(''nlist 88 list'')')
      write(12,'(''#   ---------------   '')')
C
      write(12,'(''#'')')
      write(12,'(''#   Conode'')')
      write(12,'(''#'')')
      write(12,'(''set echo on'')')
      write(12,'(''Let &Itype= -1                  '')')
      write(12,'(''    call cono 80 96 98 &Itype  '')')
      write(12,'(''    call cono 81 99 97 &Itype  '')')
      write(12,'(''    call cono 83 97 96 &Itype  '')')
      write(12,'(''Let &Itype= -2                  '')')
      write(12,'(''    call cono 80 96 98 &Itype  '')')
      write(12,'(''    call cono 81 99 97 &Itype  '')')
      write(12,'(''    call cono 82 98 99 &Itype  '')')
      write(12,'(''    call cono 83 97 96 &Itype  '')')
      write(12,'(''#'')')
      write(12,'(''DOFEQU 1 nlist=96,97, 0, nset=76,77  '')')
      write(12,'(''DOFEQU 2 nlist=96,97, 0, nset=76,77  '')')
      write(12,'(''DOFEQU 1 nlist=98,99, 0, nset=78,79  '')')
      write(12,'(''DOFEQU 2 nlist=98,99, 0, nset=78,79  '')')
C
      write(12,'(''#   Corner DOF sup. for rigid body stability'')')
      write(12,'(''#'')')
      write(12,'(''DOFSUP 1 NSET =88'')')
      write(12,'(''dofsup 2 nlist=98'')')
      write(12,'(''dofsup 3 nlist=98'')')
      write(12,'(''dofsup 3 nlist 99'')')
```

```
      write(12,'(''#'')')
      write(12,'(''finish'')')
      write(12,'(''stop'')')
         close(unit=12)
         return
         end
C
      SUBROUTINE tbset
C
c        fbset : DIAL band, setup generation.
         open(unit=12,status='unknown',file='tbset.com')
c
      write(12,'(''$band'')')
      write(12,'('' open 6 "band.out"'')')
      write(12,'('' START -1'')')
      write(12,'('' REGPS'')')
      write(12,'('' BAND'')')
      write(12,'('' STOP'')')
      write(12,'(''$setup'')')
      write(12,'('' open 6 "setup.out"'')')
      write(12,'('' START -1'')')
      write(12,'('' SETUP'')')
      write(12,'('' STOP'')')
      close(unit=12)
         return
         end
c
c        TMAT -- dial material runstream generation
c
      SUBROUTINE tmat
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /bmat/ NMAT,E1(3,7),V1(3,7),G1(3,7),NF
      common /taml/  AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /twall/ MWT,MWB,MWC,NWT(99),NWB(99),NWC(99),TT(99),BT(99),
     *CT(99),AT(99),AB(99),AC(99)
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
c
c          NMAT -- Number of Material.
c          E(i,j)    -- Elastic Modulus.
c          V(i,j)    -- Poissons' Ratio.
c          G(i,j)    -- Shear Modulus.
c          AMLS(i,j) -- AML Failure Curve Shear Allowables.
c          AMLA(i,j) -- AML Failure Curve Angles.
c          AMLT(j)   -- AML Failure Curve Temperature.
C
c          MWT  -- Number of Plies in Top Cover Sheet.
c          MWB  -- Number of Plies in Bottom Cover Shet.
c          MWC  -- Number of Plies in Core Tubes.
c          NWT  -- Material Number for Top Cover Sheet.
c          NWB  -- Material Number for Bottom Cover Sheet.
c          NWC  -- Material Number for Core Tubes.
C
c          TT   -- Top Face Sheet Ply Thickness.
c          AT(MWT)   -- Top Face Sheet Ply Angles.
c          TCT  -- Total Thickness of the Top Face Sheet.
c          BT   -- Bottom Face Sheet Ply Thickness.
c          BA(MWB)   -- Bottom Face Sheet Ply Angles.
c          BCT  -- Total Thickness of the Bottom Face Sheet.
c          CT   -- Core Tube Ply Thickness.
c          AC(MWC)   -- Core Tube Ply Angles.
```

```
c          CTT  -- Core Tube Thickness.
c          DA -- AML Invoking Indicator.
c
        open (unit=12,status='unknown',file='tmat.com')
c
      write(12,'(''$matl'')')
      write(12,'('' open 6 "matl.out"'')')
      write(12,'('' START -1'')')
      write(12,'('' set syntax on'')')
C
100   format(' MATISO ',i2,2x,e12.5,2x,e12.5)
101   format(' MATORT ',i2,2x,e12.5,2x,e12.5,2x,e12.5,',>')
102   format(1x,e12.5,2x,e12.5,2x,e12.5,',>')
103   format(1x,e12.5,2x,e12.5,2x,e12.5)
C
      do 10 i=1,NMAT
            write(12,101)i,
     *E1(1,i),E1(2,i),E1(3,i)
            write(12,102)V1(1,i),V1(2,i),V1(3,i)
            write(12,103)G1(1,i),G1(2,i),G1(3,i)
10    continue
C
      ntmp=nmat
      if (NBC.eq.2.or.NBC.eq.5) then
         ntmp=7
         if (NF.eq.1) then
            write(12,100)ntmp,E1(1,ntmp),V1(1,ntmp)
         elseif (NF.eq.2) then
            write(12,101)ntmp,E1(1,ntmp),E1(2,ntmp),E1(3,ntmp)
            write(12,102)V1(1,ntmp),V1(2,ntmp),V1(3,ntmp)
            write(12,103)G1(1,ntmp),G1(2,ntmp),G1(3,ntmp)
         endif
      endif
C
104   format(' WALLAM',2x,i2,2x,i2,',>')
105   format(1x,f16.6,',>')
106   format(1x,f16.6)
107   format(1x,i2,',>')
108   format(' MATALL',2x,i2,',>')
109   format(1x,f8.3,',',f8.5,',>')
110   format(1x,f8.3)
111   format(1x,f8.3,',',f8.5)
c
c        top face sheet layup
c
      ntmp=ntmp+1
      write(12,104)NTMP,MWT
      do 20 i=1,MWT
         write(12,107)NWT(i)
20    continue
      do 21 i=1,MWT
         write(12,105)TT(i)
21    continue
      do 22 i=1,MWT-1
         write(12,105)AT(i)
22    continue
      write(12,106)AT(MWT)
      ntmp=ntmp+1
c
c        core layup--1
```

```
c
      mt=mwc+mwc
      write(12,104)NTMP,MT
      do 23 i=1,MWC
         write(12,107)NWC(i)
23    continue
      do 24 i=MWC,1,-1
         write(12,107)NWC(i)
24    continue
      do 25 i=1,MWC
         write(12,105)CT(i)
25    continue
      do 26 i=MWC,1,-1
         write(12,105)CT(i)
26    continue
      do 27 i=1,MWC
         write(12,105)AC(i)
27    continue
      do 28 i=MWC,2,-1
         write(12,105)AC(i)
28    continue
      write(12,106)AC(1)
      ntmp=ntmp+1
c
c       core layup --- 2
c
      mt=mwc
      write(12,104)NTMP,MT
      do 29 i=1,MWC
         write(12,107)NWC(i)
29    continue
      do 30 i=1,MWC
         write(12,105)CT(i)
30    continue
      do 31 i=1,MWC-1
         write(12,105)AC(i)
31    continue
      write(12,106)AC(MWC)
      ntmp=ntmp+1
c
c       core layup --- 3
c
      mt=mwc
      write(12,104)NTMP,MT
      do 32 i=1,MWC
         write(12,107)NWC(i)
32    continue
      do 33 i=1,MWC
         write(12,105)CT(i)
33    continue
c
      do 34 i=1,MWC-1
         write(12,105)AC(i)
34    continue
      write(12,106)AC(MWC)
      ntmp=ntmp+1
c
c       bottom face sheet layup
c
      write(12,104)NTMP,MWB
```

```
      do 35 i=1,MWB
         write(12,107)NWB(i)
35    continue
      do 36 i=1,MWB
         write(12,105)BT(i)
36    continue
      do 37 i=1,MWB-1
         write(12,105)AB(i)
37    continue
      write(12,106)AB(MWB)
c
c        AML CURVE INVOKING
c
      if (DA.eq.1.) then
         do 38 i=1,NMAT
            write(12,108)i
            do 39 j=1,5
               write(12,109)AMLA(j,i),AMLS(j,i)
39          continue
            write(12,111)AMLA(6,i),AMLS(6,i)
38       continue
      endif
C
      write(12,'('' MATL'')')
      write(12,'('' STOP'')')
         close(unit=12)
         return
         end
c
c        TLOAD -- load runstream generation
c
      SUBROUTINE TLOAD
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
      open(unit=12,status='unknown',file='tload.com')
      xsr=-1.*XL(1)/2.
      xsl=XL(1)/2.
      xeb=-1.*XL(2)/2.
      xef=XL(2)/2.
      sl=SL(1)/2.
      s2=-1.*SL(1)/2.
c
      write(12,'(''$load'')')
      write(12,'('' open 6 "load.out"'')')
      write(12,'('' start'')')
      write(12,'('' lcase 1'')')
      write(12,'('' # axial line loads in front and back ends.'')')
      write(12,'('' pline '',f10.3,'' 11 0 1 mset=7'')')xeb
      write(12,'('' pline '',f10.3,'' 11 0 3 mset 8'')')xef
      write(12,'('' # axile line loads in right and left sides.'')')
      write(12,'('' pline '',f10.3,'' 12 0 2 mset 5'')')xsl
      write(12,'('' pline '',f10.3,'' 12 0 4 mset 6'')')xsr
      write(12,'('' # shear line loads.'')')
      write(12,'('' pline '',f10.3,'' 12 0 1 mset 7'')')s2
      write(12,'('' pline '',f10.3,'' 12 0 3 mset 8'')')sl
      write(12,'('' pline '',f10.3,'' 11 0 4 mset 6'')')s2
      write(12,'('' pline '',f10.3,'' 11 0 2 mset 5'')')sl
      write(12,'('' load'')')
      write(12,'('' stop'')')
c
        return
```

```
         end
c
c        TSOLVE --solve (and cluster) runstream generation
c
      SUBROUTINE tsolve
      common /tgeom/ A,B,PH,PL,NW,K,NBC,NBUC,TCT,BCT,CTT,TAR,NDF,ND,MS
      common /load/ XL(4),SL(4),XE,XS,SE,SS,TMPD
      open(unit=12,status='unknown',file='tsolve.com')
100   format(' loads ',i2,f12.4)
C
      xs=XL(1)/2.
      xe=XL(2)/2.
      s=SL(1)/2.
c
      if (NBUC.eq.2) then
         write(12,'(''$solveDP'')')
         write(12,'('' open 6 "solve.out"'')')
         write(12,'('' start'')')
         write(12,100) 1,  0.1
         write(12,'('' SAVE,D'')')
         write(12,'('' SAVE,S'')')
         write(12,'('' ASSIGN IMPR=1'')')
         write(12,'('' EIGEN 2'')')
         write(12,'('' solve'')')
         write(12,'('' stop'')')
c
         write(12,'(''$clusterDP'')')
         write(12,'('' open 6 "CLSTR.out"'')')
         write(12,'('' start'')')
         write(12,'('' assign mxit= 40 '')')
         write(12,'('' shift '',i3)')MS
         write(12,'('' save'')')
         write(12,'('' matrix:file,ki ki'')')
         write(12,'('' clust 2'')')
         write(12,'('' stop'')')
C
      elseif(NBUC.eq.1) then
         write(12,'(''$solveDP'')')
         write(12,'('' open 6 "solve.out"'')')
         write(12,'('' start'')')
         write(12,100) 1, 1.
         write(12,'('' SAVE,D    '')')
         write(12,'('' SAVE,S    '')')
C          write(12,'('' NPRINT,D  '')')
C          write(12,'('' EPRINT,S  '')')
         write(12,'('' solve'')')
         write(12,'('' stop'')')
      elseif(NBUC.eq.3)then
         write(12,'('' $solveDP'')')
         write(12,'('' open 6 "solve.out"'')')
         write(12,'('' start -1'')')
         write(12,'('' static 2'')')
         write(12,'('' branch 0 1 1 '')')
         write(12,'('' load 1 1. 1. 1.'')')
         write(12,'('' limit load'')')
         if (NF.ge.1) then
          write(12,'('' Target '',f12.5,'' Node= '',2i4)')TAR,NDF,ND
         else
            write(12,'('' Target'',f12.5)')TAR
         endif
```

```
         write(12,'('' save,d'')')
         write(12,'('' save,s'')')
         write(12,'('' assign slf=0.5, relr=0.02, stol=0.25, gtol=0.25''
     *)')
         write(12,'('' solve 1'')')
         write(12,'('' stop'')')
c         write(12,'('' !'')')
       endif
         return
         end
c  ------
c        NTOKEN -- utility I/O subroutine
c        by Dan Rickhoff LMSC 81-12;
c        modification by L. L. Chou LMSC 81-12.
c
      subroutine nreal(ntk,ra)
         parameter ( maxtkn = 25 )
         character a*127, tkns(maxtkn)*10
         dimension lentk(maxtkn),ra(maxtkn)
c
  100 ierr=0
         call getlin( 5, a, irdata, *910 )
         call getkns( a, irdata, tkns, lentk, ntk, *999 )
c       - - - - -
        do 200 i=1,ntk
        r = ra(i)
        call rintrp( tkns(i), lentk(i), r, *991 )
        ra(i) = r
        go to 200
  991     ierr=ierr+1
          write(6,'( 19x, ''Not a real '')')
  200 continue
      if (ierr .ge. 1) go to 100
        return
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
         return
         end
c ----
      subroutine nintg(ntk,ia)
         parameter ( maxtkn = 25 )
         character a*127, tkns(maxtkn)*10
         dimension lentk(maxtkn),ia(maxtkn)
c
  100 ierr=0
      call getlin( 5, a, irdata, *910 )
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
c
      do 200 i=1,ntk
        intgr = ia(i)
        call iintrp( tkns(i), lentk(i), intgr, *992 )
        ia(i) = intgr
        go to 200
  992     ierr=ierr+1
          write(6,'( 19x, ''Not an integer '')')
  200 continue
      if ( ierr .ge. 1) go to 100
         return
```

```
c
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
         return
         end
c ----
      subroutine stitle(sline)
         character a*127, tkns(1)*60, sline*(*)
         dimension lentk(1)
c
      call getlin( 5, a, irdata, *910 )
      ntk = 1
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
      sline=tkns(1)(1:lentk(1))
         return
c
  999 continue
      write(6,'('' ERROR return from getkns'')')
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
         return
         end
c ----
      subroutine rintrp( tkn, lentk, r, * )
c
c#      Read a real number, R, from a character string, TKN, of length LENTK.
c#      (Note that blanks in the strings are ignored. Leading and/or trailing
c#      blanks are ignored, which is OK, but interior blanks are squeezed out,
c#      which is not OK; try to fix this sometime.)
c
      character tkn*(*)
      if ( lentk .eq. 0 ) return
      read( tkn(1:lentk), '(bn,f80.0)', err=990 ) r
         return
c
  990 continue
         return 1
         end
c ----
      subroutine iintrp( tkn, lentk, i, * )
c
c#      Read a integer number, I, from a character string, TKN, of length LENTK.
c#     (Note that blanks in the strings are ignored. Leading and/or trailing
c#      blanks are ignored, which is OK, but interior blanks are squeezed out,
c#      which is not OK; try to fix this sometime.)
c
      character tkn*(*)
      if ( lentk .eq. 0 ) return
      read( tkn(1:lentk), '(bn,i80)', err=990 ) i
         return
c
  990 continue
         return 1
         end
c ----
      subroutine getkns( a, irdata, tkns, lentk, ntk, * )
c
c#       in     a ....... The given character string containing the input line.
```

```
c#                          If this string is blank (input IRDATA = 0) then all
c#                          NTK returned tokens will be blank with zero length.
c
c#       in     irdata .. The input line in string A extends to this character
c#                          position.
c
c#       out    tkns .... The character array of tokens.  The character lengths
c#                          of these array elements limit the lengths of input
c#                          tokens; tokens are correctly parsed but then truncated
c#                          to this length.
c
c#       out    lentk ... An integer array with the string lengths of the tokens
c#                          (after any necessary truncation).
c
c#       in     ntk ..... The number of tokens to be parsed from the line.
c
c#              * ....... Alternate RETURN in the event that a string token
c#                        initiated by an apostrophy had no trailing apostrophy, or
c#                        the next character after the trailing apostrophy was not
c#                        a delimiter (or end-of-line).
c#                        NOTE: The arguments that had been interpreted up to the
c#                              point that the error occured are returned; the
c#                              will be blank.
c  ----------
c#      Parses an "input line" contained in a given string to find the first NTK
c#      tokens on the line.
c
c#      Tokens are delimited by beginning-of-line, commas, spaces, tabs, or
c#      end-of-line.  Tabs are equivalent to (single) spaces.
c
c#  A single asterisk (*) input token is recognized as a "null" token, for
c#  which the retuned string is blank and returned length is zero.  Some users
c#  may find entry of the null token preferable to that of a comma with no
c#  preceeding token, or two adjacent apostrophies (see the next
c#  paragraph).
c
c#  If a token is enclosed within apostrophies then it may contain commas,
c#  spaces, or tabs.  The returned length of such a token is the number
c#  number of character positions between the pair of apostrophies.
c#  Note that, for this length calculation, tabs are a single character.
c#  Contigous alphanumeric strings (no embedded delimiters) do not need to be
c#  enclosed within apostrophies.
c
c#  There is no provision for including apostrophies (') or the comment
c#  character (the hash symbol, #; see routine GETLIN) within a token,
c#  i.e., they may not be "escaped".  Strings enclosed within apostrophies
c#  may contain asterisks; these will not be misinterpreted as
c#  null tokens.
c
c#  A line with only "null" data will minimally contain (before an optional
c#  comment) either a null character, a comma or two adjacent apostrophies.
c
c#  Tokens are placed, left justified, into the character array TKNS.
c#  The tokens may be truncated in order to fit into the given TKNS array.
c
c#  The lengths of the returned tokens are placed in the LENTK array.
c ------------------------------------------------------------------
c#       Arrays for the returned tokens and their correcponding
c#       character lengths.
c ------------------------------------------------------------------
```

```
      character tkns(*)*(*)
      dimension lentk(*)
c -------------------------------------------------------------
c#       Strings for the input line, the "curent" character on the
c#       line and delimiter characters (space, tab, and comma).
c -------------------------------------------------------------
      character a*(*), c*1, tab*1, space*1, comma*1, apos*1, null*1
      save  space      , comma      , apos       , null
      data  space /' '/, comma /','/, apos /''''/, null /'*'/
      tab = char(9)
      maxlen = len( tkns(1) )
c -------------------------------------------------------------
c#       Initialize ntk tokens to blanks.
c -------------------------------------------------------------
      do 50 i=1,ntk
        tkns(i) = ' '
        lentk(i) = 0
   50 continue
      if ( irdata .eq. 0 ) return
c -------------------------------------------------------------
c#        Something is on the line (before any comment).
c#        Parse the tokens (up to NTKN of them).
c -------------------------------------------------------------
      is = 0
      itkn = 1
  200 continue
c        # Move along the line looking for a token or a comma (null token).
      is = is + 1
      c = a(is:is)
c
      if ( c.eq.tab .or. c.eq.space ) then
c          # Move to next character; to continue looking for token.
      go to 200
      else if ( c.eq.comma ) then
c          # No token found before this comma indicates null token.
        if ( itkn .eq. ntkn ) return
        itkn = itkn + 1
c
        if ( is .eq. irdata ) return
c          # Move to next character; look for next token.
        go to 200
      else
  300   continue
c          # We are at the left-most character of a non-null token.
        il = is
        if ( c .eq. apos ) then
c          # We look for the trailing apostrophy for this "quoted string".
          if ( is .eq. irdata ) return 1
  450     continue
          is = is + 1
          c = a(is:is)
          if ( c .eq. apos ) then
            length = is - il - 1
            if ( length .gt. maxlen ) then
              length = maxlen
            endif
            lentk( itkn ) = length
            if ( length .gt. 0 ) tkns( itkn ) = a( il+1: il+length )
            if ( is .eq. irdata ) return
            is = is + 1
```

```
            c = a(is:is)
            if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) )
     *        return 1
          else
            if ( is .eq. irdata ) return 1
            go to 450
          endif
        else
  400     continue
          if ( is .eq. irdata ) then
c            # The token extends to the end of the input data.
            if ( a(il:is) .eq. null ) return
            tkns( itkn ) = a(il: is)
c            # The input token length may exceed the length of the variable.
            length = is - il + 1
            if ( length .le. maxlen ) then
              lentk( itkn ) = length
            else
              lentk( itkn ) = maxlen
            endif
            return
          endif
c           # Move along the line looking for the end of the token.
          is = is + 1
          c = a(is:is)
          if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) ) then
            if ( c .eq. apos ) return 1
c             # Move to next character; continue looking for end of token.
            go to 400
          endif
c           # The previous character was the right-most character of the token
          if ( a(il: is-1) .ne. null ) then
            tkns( itkn ) = a(il: is-1)
c             # The input token length may exceed the length of the variable.
            length = is - il
            if ( length .le. maxlen ) then
              lentk( itkn ) = length
            else
              lentk( itkn ) = maxlen
            endif
          endif
        endif
c
        if ( itkn .eq. ntkn ) return
c           # Find the next location at which we should begin looking for the
c           # next token, i.e., after a following comma or at the next token.
        itkn = itkn + 1
        if ( is .eq. irdata ) return
        if ( c .eq. comma ) then
c           # Go back and look for a token following this comma.
          go to 200
        else
c           # Curent character was a space or a tab.
c           # Look for a comma, the next token, or end of line.
  500     continue
          is = is + 1
          c = a(is:is)
          if ( c .eq. comma ) then
            if ( is .eq. irdata ) return
            go to 200
```

```
            else if ( c.eq.space .or. c.eq.tab ) then
              if ( is .eq. irdata ) return
              go to 500
            else
              go to 300
            endif
          endif
        endif
           end
c ----
      subroutine getlin( nunit, a, irdata, * )
c
c#    in      nunit ... The Fortran logical unit number of the file from which
c#                      the input line(s) should be read.
c
c#    out     a ....... A character string into which the input line will be
c#                      read.  The length of this given string determines how
c#                      long the data line may be; input past this length is
c#                      ignored.
c
c#    out     irdata .. The input "data" in string A extends to this character
c#                      position.  The data portion does not include any
c#                      trailing comment (if present).
c
c#            * ....... Alternare RETURN in the event that End-of-File is
c#                      encountered on attempt to read a line from unit NUNIT.
c
c#  Reads line(s) from unit NUNIT and determines the location, IRDATA, in the
c#  string beyond which there may be only spaces, tabs, or a comment.  Comments
c#  begin with a hash symbol (#).  Blank lines and lines that contain only a
c#  comment line are ignored and the next line is read until a line with some
c#  data or End-of-File is encountered.  ("Some data" might minimally be a
c#  single comma, an asterisk, or two adjacent apostrophies.)
c
c
      character a*(*), comnt*1
      save comnt
      data comnt /'#'/
c
c#       Read the input "line" from disk.
c
  100 continue
      read( nunit, '(a)', end=910 ) a
c
c#       We want to ignore any trailing comment and lines
c#       that are blank except for a comment (if any).
c
      icom = index( a, comnt )
c
c  #     Location of right-most non-blank before comment (if any).
      if ( icom .gt. 1 ) then
        irdata = iright( a(1: icom-1) )
      else if ( icom .eq. 1 ) then
        go to 100
      else
        irdata = iright( a )
      endif
c
c  #    If only blanks before comment (if present) then read next line.
      if (irdata .eq. 0) go to 100
```

```
        return
c
  910 continue
        return 1
        end
c ----
      function ileft( string )
c
c        by:  D. T. Rickhoff
c
c  Find the position of the leftmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c
      character*(*) string
      character*1 tab
      tab = char(9)
c
c        Find the location of the first nonblank character.
c
      ls = len( string )
      do 100 i=1,ls
        if(    string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          ileft = i
          return
        endif
  100 continue
c
c        String was blank.  Set ileft to zero (as a flag).
      ileft = 0
        return
        end
c ----
      function iright( string )
c
c        by:  D. T. Rickhoff
c
c  Find the position of the rightmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c
      character*(*) string
      character*1 tab
      tab = char(9)
c
c        Find the location of the last nonblank character.
c
      ls = len( string )
      do 100 i=ls,1,-1
        if(    string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          iright = i
          return
        endif
  100 continue
c
c        String was blank.  Set iright to zero (as a flag).
      iright = 0
        return
```

```
      end
C ------------  End of TUB212.FOR  ---
```

A.3.3.4 GEODESIC CURVED STIFFENED PANEL MODULE # 4 PROGRAM

```
      PROGRAM GEO
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
c      common /tam1/ AMLS(6,6),AMLA(6,6),AMLT(6),DA
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
      common /load/ XL(4),SL(4),PSURF,X,S,TEMP
c ... ****************************************************
c     *  Advanced Compostie Structural Program Input     *
c     *  PROBLEM 3: Geodesic Fuselage Panel              *
c     *                                                  *
c     *  variables definition ...                        *
c     *                                                  *
c     *   [ Geometry data ]                              *
c     *   NMAT = num. of materials used                  *
c     *   D = panel length                               *
c     *   ALPHA = stiffener angle                        *
c     *   R = panel radius                               *
c     *   T1 = panel thickness(calculated)               *
c     *   T2 = stiffener thickness(calculated)           *
c     *   B = height of stiffener                        *
c     *   NBY = number of bays in theta direction        *
c     *   NBZ = number of bays in z direction            *
c     *   NWAL = number of wall layups                   *
c     *                                                  *
c     *    yal0: read geom.dat (tape10)                  *
c     *    yal1: read mat.dat (tape11)                   *
c     *    nal0: input data for the new geom.dat (tape10)*
c     *    nal1: input data for the new mat.dat  (tape11)*
c     *    fmesh2: coarse mesh generation(stress,default)*
c     *    fmesh: fine mesh generation(buckling)         *
c     *    nlod: load module                             *
c     *    flis0: list data                              *
c     *    fmod : modify data                            *
c     *    fsave: update data                            *
c     *                                                  *
c     *    Composed by :                                 *
c     *          A. Y. Wei  LMSC/MSD/Bld. 157            *
c     *               Ext:6-1137                         *
c     *    With Assistance From:                         *
c     *          L. L. Chou LMSC/MSD/Bld. 157            *
c     *          W. M. Brown LMSC/MSD/Bld. 157           *
c     *          D. T. Rickhoff LMSC/MSD/Bld. 157        *
c     *    DATED Aug. 10, 1990                           *
c     *    Update: Nov. 21, 1990                         *
c     *    Version 1.3                                   *
c     *                                                  *
c     ****************************************************
      LOGICAL here10,here11,here12
      do 5 i=1,10
      write(6,*)(' ')
5     continue
      write(6,'(''      Advanced Composite Structural Program '')')
      write(6,'(''           Geodesic Fuselage Panel          '')')
      write(6,'(''            Updated Version - 1.3'')')
c ...
```

c . checking input data for DIAL F.E. structural analysis

```
c ...
c . initialize geom.dat(unit10) and mat.dat(unit11) files
c ...
      inquire( file='geom.dat', exist=here10)
      inquire( file='mat.dat', exist=here11)
      if (here10) then
        write(6,'(''     Geometric Database Already Exist.'')')
        call ya10
      else
        call na10
      endif
         write(6,*)(' ')
      if (here11) then
        write(6,'(''     Material Database Already Exist.'')')
        call ya11
                  else
        call na11
      endif
      call nlod
c.......................
      call fsave
c ...
c     list input parameters
c ...
       call flis0
       call fmod
c ...
c ...
c ###
c # end of user interactive input, begin
c # DIAL runstream generation.
c ###
      write(6,'(''     Begin DIAL Runstream Generation'')')
      if (NMESH .eq. 1) then
         call fmesh2
      else
         call fmesh
      endif
      call fbset
      call fmat1
      call fload
      call fsolve
      write(6,'(''     DIAL Runstream Generation Complete'')')
      stop
      end
c##############
cSUBROUTINE YA10
c  geom.dat-- geometric parameter input.
c##############
      SUBROUTINE ya10
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
      character blk
c ...
c . read geom.dta (unit10)
c ...
      open(unit=10,status='old',file='geom.dat')
```

```
  101  format(F19.3)
  102  format(I5)
  103  format(3f19.3)
  104  format(3i5)
  105  format(i5,f9.6,f7.2)
  106  format(a1)
  107  format(2f9.6,f9.3)
c.........
c geom.dat (unit10): geometry
c.........
      read(10,104)NBY,NBZ,NWAL
      read(10,103)D,ALPHA,R
      read(10,107)T1,T2,B
      read(10,102)NBC
      read(10,102)NMESH
c.......
c wall layups
c.......
      read(10,102)MWP
      read(10,102)MWS
      do 10 i=1,MWP
         read(10,105)NWP(i),PT(i),PA(i)
10    continue
      read(10,106)blk
      do 11 i=1,MWS
         read(10,105)NWS(i),ST(i),SA(i)
11    continue
      return
      end
c############
cSUBROUTINE YA11
c  mat.dat  -- material and load input
c############
      SUBROUTINE ya11
      common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
      common /load/ XL(4),SL(4),PSURF,X,S,TEMP
      character*1 tok
c ...
c . read mat.dat (unit11)
c ...
      open(unit=11,status='old',file='mat.dat')
  101  format(F19.3)
  102  format(I5)
  103  format(3f19.3)
  104  format(3i5)
  105  format(i5,2f19.3)
  106  format(a1)
c..........
c mat.dat (unit11): material
c..........
      read(11,102)NMAT
      do 12 i=1,NMAT
            read(11,102)MM(i)
         if (MM(i).eq.1)then
            read(11,101)E(i)
            read(11,101)V(i)
            read(11,106)tok
         else
            read(11,103)E1(1,i,1),E1(2,i,1),E1(3,i,1)
```

```
            read(11,103)G1(1,i,1),G1(2,i,1),G1(3,i,1)
            read(11,103)V1(1,i,1),V1(2,i,1),V1(3,i,1)
            read(11,106)tok
         endif
12    continue
c..........
c mat.dat (unit11):load
c..........
      do 14 i=1,4
         read(11,101)XL(i)
         read(11,101)SL(i)
14    continue
      read(11,101)PSURF
      return
      end
c############
cSUBROUTINE NA10
c interactive -- geometric parameter input
c############
      SUBROUTINE na10
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
c ...
c     NMAT--number of materials used.
c     NBY --number of bays in circumferential direction.
c     NBZ --number of 2-bays in axial direction.
c     NWAL--number of different composite wall lay-ups.
c     d   --bay length.
c     alpha-stiffener angle. (see manual)
c     r   --radius of the panel.
c     t1  --thickness of the panel.
c     t2  --thickness of the stiffeners.
c     b   --height of the stiffeners.
c     NBC--boundary condition of the panel.
c     NMESH-indicator for fine or coarse mesh.
c ...
      dimension ra(10),ira(10)
c ...
c . create input for geom.dat (unit10)
c ...
c***************
c default values
c***************
      NBY=1
      NBZ=1
      NWAL=2
      NMESH=1
      d=20.
      r=117.5
      alpha=48.
      t1=0.04
      t2=0.32
      b=2.0
      NBC=2
c ...
      write(6,'('' Input 1> for Stress Analysis, 2> for Buck. Analysis.>
     *>''$)')
      call NINTG(1,NMESH)
      write(6,*)' '
```

```
      write(6,'('' Input Bay Length (20. in.) >>''$)')
      call NREAL(1,d)
      write(6,*)' '
      write(6,'('' Input No. of Bays in Axial Direction. '')')
      write(6,'('' (Must be Even Number) >>''$)')
      call NINTG(1,NBZ)
      write(6,*)' '
      write(6,'('' Input No. of Bays in Circumferential Direction. >>''$
     *)')
      call NINTG(1,NBY)
      write(6,*)' '
      write(6,'('' Input Panel Radius. (in.)>>''$)')
      call NREAL(1,r)
      write(6,*)' '
      write(6,'('' Input Stiffener Angle (deg.)>>''$)')
      call NREAL(1,alpha)
      write(6,*)' '
      write(6,'('' Input Stiffener Height. (in.)>>''$)')
      call NREAL(1,b)
      write(6,*)' '
      write(6,'('' Input Boundary Conditions for the panel.'')')
      write(6,'('' 1-Infinite Panel.'')')
      write(6,'('' 2-Sides Clamped; Ends Free.'')')
      write(6,'('' 3-Ends Clamped; Sides Free.'')')
      write(6,'('' Select Your Choice >>''$)')
      call NINTG(1,NBC)
      write(6,*)' '
c ...
c ...
c ...
      write(6,'('' Input No. of Layers for Panel Wall. >>''$)')
      call NINTG(1,MWP)
      write(6,*)' '
      write(6,'('' Input No. of Layers for Stiffener Wall. >>''$)')
      call NINTG(1,MWS)
      write(6,*)' '
      write(6,'('' Input Panel Lay-up Material Num.'')')
100   format(' Layer Number ',I2,'. >>'$)
      do 90 i=1,MWP
         write(6,100)i
         call NINTG(1,NWP(i))
90    continue
      write(6,*)' '
      T1=0.
      write(6,'('' Input Panel Lay-up Thickness.'')')
      do 91 i=1,MWP
         write(6,100)i
         call NREAL(1,PT(i))
         T1=T1+PT(i)
91    continue
      write(6,*)' '
      write(6,'('' Input Panel Lay-up Orientation.'')')
      do 92 i=1,MWP
         write(6,100)i
         call NREAL(1,PA(i))
92    continue
      write(6,*)' '
      write(6,'('' *****Warning!! Stiffener Lay-up Must be Symmetric
     *.****'')')
      write(6,'('' Input Stiffener Lay-up Material Number.'')')
```

```
      do 93 i=1,MWS
         write(6,100)i
         call NINTG(1,NWS(i))
93    continue
      write(6,*)' '
      T2=0.
      write(6,'('' Input Stiffener Lay-up Thickness.'')')
      do 94 i=1,MWS
         write(6,100)i
         call NREAL(1,ST(i))
         T2=T2+ST(i)
94    continue
      write(6,*)' '
      write(6,'('' Input Stiffener Lay-up Orientation.'')')
      do 95 i=1,MWS
         write(6,100)i
         call NREAL(1,SA(i))
95    continue
      return
      end
c##############
cSUBROUTINE NA11
c interactive -- material parameter input
c##############
      SUBROUTINE na11
      common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
c ...
c     NMAT--number of materials used.
c     E   --elastic moduli.
c     V   --poisson's ratio.
c     G   --shear moduli.
c     MWP --number of layers for panel.
c     MWS --number of layers for stiffener.
c     NWP --matl composition of panel wall.
c     NWS --matl composition of stiffener wall.
c     PT  --panel layer thicknesses.
c     PA  --panel layer angles.
c     ST  --stiffener layer thicknesses.
c     SA  --stiffener layer angles.
c     MM  --indicator for iso--1, or ortho--2 matl.
c ...
c ...
c . create input for mat.dat (unit11)
c ...
c . default values for matl
c ...
      NMAT=2
c ...
101   format(' Input Properties for Matateriał Number ',i2,' >>')
      write(6,'('' Input No. of Materials Used. >>''$)')
      call NINTG(1,NMAT)
      write(6,*)' '
      do 20 i=1,NMAT
         write(6,101)i
         write(6,'('' Input 1> for Iso. Material. 2> for Ortho. Material
     *. >>'')')
         write(6,'(''   Enter >>''$)')
         call NINTG(1,MM(i))
```

```
         write(6,*)' '
         if (MM(i). eq. 1) then
            write(6,'('' Input Elastic Modulus. (psi)>>''$)')
            call NREAL(1,E(i))
            write(6,*)' '
            write(6,'('' Input Poisson''''s Ratio. >>''$)')
            call NREAL(1,V(i))
            write(6,*)' '
         else
            write(6,'('' Input Elastic Moduli: E11,E22 (psi)>>''$)')
            call NREAL(2,E1(1,i,1),E1(2,i,1))
            write(6,*)' '
            E1(3,i,1)=100.
            write(6,'('' Input Shear Moduli: G12,G23,G31 (psi)>>''$)')
            call NREAL(3,G1(1,i,1),G1(2,i,1),G1(3,i,1))
            write(6,*)' '
            write(6,'('' Input Poisson''''s Ratios: V12 >>''$)')
            call NREAL(1,V1(1,i,1))
            write(6,*)' '
            V1(2,i,1)=0.00
            V1(3,i,1)=0.00
         endif
20    continue
c ...
c ...
c ...
      return
      end
c################
cSUBROUTINE NLOD
c  load input
c################
      SUBROUTINE NLOD
      common /load/ XL(4),SL(4),PSURF,X,S,TEMP
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      TEMP=D
      A2=ALPHA/2.
      TEMP1=TEMP/COSD(A2)
      TEMP=TEMP1*SIND(A2)
      TEMP=TEMP*2.
      TEMP=TEMP*NBY
      do 10 i=1,8
      write(6,'('' '')')
10    continue
      write(6,'('' Apply Loads to the Model:'')')
      write(6,'('' How Would You Like to Input Loads?'')')
      write(6,'('' 1>as Total Load.'')')
      write(6,'('' 2>as Line Load.'')')
      write(6,'('' Enter >>''$)')
      call nintg(1,N)
      write(6,*)' '
      if(N.eq.1)then
         write(6,'('' Input Total Axial Load (lb.)=''$)')
         call nreal(1,X)
         XL(2)=X/TEMP
         XL(4)=X/TEMP
         write(6,*)' '
         write(6,'('' Input Total Shear Load on the 1)Ends 2)Sides''
     *)')
```

```
            write(6,'('' Enter >>''$)')
            call nintg(1,NN)
            write(6,*)' '
            if (NN.eq.1) then
               write(6,'('' Input Total Shear Load on the Ends (lb.)=''
     *$)')
               call nreal(1,S)
               write(6,*)' '
               SL(1)=S/TEMP
               SL(2)=S/TEMP
               SL(3)=S/TEMP
               SL(4)=S/TEMP
            else
               write(6,'('' Input Total Shear Load on the Sides (lb.)='
     *'$)')
               call nreal(1,S)
               write(6,*)' '
               TEMP2=D*NBZ
               SL(1)=S/TEMP2
               SL(2)=S/TEMP2
               SL(3)=S/TEMP2
               SL(4)=S/TEMP2
            endif
         elseif(N.eq.2)then
            write(6,'('' Input Axial Line Load (lb./in)=''$)')
            call nreal(1,X)
            XL(2)=X
            XL(4)=X
            write(6,*)' '
            write(6,'('' Input Shear Line Load (lb./in)=''$)')
            call nreal(1,S)
            write(6,*)' '
            SL(1)=S
            SL(2)=S
            SL(3)=S
            SL(4)=S
         endif
         write(6,'('' Input Pressure Load (psi)= ''$)')
         call NREAL(1,PSURF)
         write(6,*)' '
         write(6,*)' '
         return
         end
c
c#################
cSUBROUTINE FSAVE
c save data to geom.dat and mat.dat
c#################
c
         SUBROUTINE fsave
         common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
        *              NBC,NMESH
         common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
        *              E(6),V(6),MM(6)
         common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
         common /load/ XL(4),SL(4),PSURF,X,S,TEMP
         character blk
         open(unit=10,status='unknown',file='geom.dat')
         open(unit=11,status='unknown',file='mat.dat')
         rewind(10)
```

```
      rewind(11)
c ...
c . save and update files
c ...
  101  format(F19.3)
  102  format(I5)
  103  format(3f19.3)
  104  format(3i5)
  105  format(i5,f9.6,f7.2)
  106  format(a1)
  107  format(2f9.6,f9.3)
c.........
c geom.dat (unit10): geometry
c.........
      write(10,104)NBY,NBZ,NWAL
      write(10,103)D,ALPHA,R
      t1=0.
      t2=0.
      do 8 i=1,MWP
         t1=t1+PT(i)
8     continue
      do 9 i=1,MWS
         t2=t2+ST(i)
9     continue
      write(10,107)T1,T2,B
      write(10,102)NBC
      write(10,102)NMESH
c
c
      write(10,102)MWP
      write(10,102)MWS
      do 10 i=1,MWP
         write(10,105)NWP(i),PT(i),PA(i)
10    continue
      write(10,106)blk
      do 11 i=1,MWS
         write(10,105)NWS(i),ST(i),SA(i)
11    continue
c...........
c mat.dat (unit11): material
c...........
      write(11,102)NMAT
      do 12 i=1,NMAT
         if (MM(i).eq.1)then
            write(11,102)MM(i)
            write(11,101)E(i)
            write(11,101)V(i)
            write(11,106)blk
         else
            write(11,102)MM(i)
            write(11,103)E1(1,i,1),E1(2,i,1),E1(3,i,1)
            write(11,103)G1(1,i,1),G1(2,i,1),G1(3,i,1)
            write(11,103)V1(1,i,1),V1(2,i,1),V1(3,i,1)
            write(11,106)blk
         endif
12    continue
      do 14 i=1,4
         write(11,101)XL(i)
         write(11,101)SL(i)
14    continue
```

```
      write(11,101)PSURF
      close(unit=10)
      close(unit=11)
      return
      end
c
c######################
cSUBROUTINE FLIS0
c  list current parameters
c######################
c
      SUBROUTINE flis0
      common /geom/ NBY,NBZ;NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /matl/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
      common /load/ XL(4),SL(4),PSURF,X,S,TEMP
      character tok
c ...
c . list files
c ...
 101  format('     Number of Panels in: Axial Direction=',i3,2x,
     *'Hoop Direction=',i3)
 102  format('     Material Number:',i2)
 103  format('     E(psi)=',f13.1,2x,'V=',f9.3)
 104  format('     E11(psi)=',f13.1,2x,'E22(psi)=',f13.1)
 105  format('     G12(psi)=',f13.1,2x,'G23(psi)=',f13.1,2x,
     *'G31(psi)=',f13.1)
 106  format('     V12=',f9.3)
 107  format('     Ply Number',I2,/,'       Material Number:',i2,2x,
     */,'       Ply Thickness(in)=',f9.6,2x,/,
     *'       Ply Angle(deg.)=',f7.2)
 108  format('     Panel Radius(in)=',f9.3)
 109  format('     Panel Length, D(in)=',f9.3)
 110  format('     Stiffener Height(in)=',f9.3)
 111  format('     Stiffener Angle, ALPHA(degree)=',f5.2)
 112  format('     Panel Thickness(in)=',f9.6)
 113  format('     Stiffener Thickness(in)=',f9.6)
 115  format(a1)
 116  format('     Axial Line Load (lb./in)=',f9.3)
 117  format('     Shear Line Load (lb./in)=',f9.3)
 118  format('     Pressure Load (lb./in**2)=',f9.3)
 10   write(6,'(''     Choose the Info. to be Listed:'')')
      write(6,'(''     1>Panel and Stiffener Geometry.'')')
      write(6,'(''     2>Material Data.'')')
      write(6,'(''     3>Wall Lay-ups.'')')
      write(6,'(''     4>Load.'')')
      write(6,'(''     5>Exit.'')')
      write(6,'(''     Enter>>''$)')
      call nintg(1,N)
c.........
c gemo.dat (unit10): geometry
c.........
      if(N.eq.1)then
         write(6,*)' '
         write(6,108)R
         write(6,109)D
         write(6,110)B
         write(6,111)ALPHA
```

```
         write(6,101)NBZ,NBY
         write(6,112)T1
         write(6,113)T2
         if(NBC.eq.1)then
            write(6,'(''     Boundary Condition: Infinite Panel.'')')
         elseif(NBC.eq.2)then
            write(6,'(''     Boundary Condition: Sides Clamped; Ends Fre
     *e.'')')
         else
            write(6,'(''     Boundary Condition: Ends Clamped; Sides Fre
     *e.'')')
         endif
         if(NMESH.eq.1)then
            write(6,'(''     Stress Analysis.'')')
         else
            write(6,'(''     Buckling Analysis.'')')
         endif
         write(6,*)' '
         write(6,'(''     ***Hit <<Return>> for Next Page.***''$)')
         read(5,115)tok
         write(6,*)' '
         write(6,*)' '
         write(6,*)' '
         go to 10
c...........
c mat.dat (unit11): material
c...........
      elseif(N.eq.2)then
         do 11 i=1,NMAT
            write(6,*)' '
            write(6,102)i
            if(MM(i).eq.1)then
               write(6,'(''     Isotropic Material'')')
               write(6,103)E(i),V(i)
            else
               write(6,'(''     Orthotropic Material'')')
               write(6,104)E1(1,i,1),E1(2,i,1)
               write(6,105)G1(1,i,1),G1(2,i,1),G1(3,i,1)
               write(6,106)V1(1,i,1)
            endif
11       continue
         write(6,*)' '
         write(6,'(''     ***Hit <<Return>> for Next Page.***''$)')
         read(5,115)tok
         write(6,*)' '
         write(6,*)' '
         write(6,*)' '
         go to 10
c...........
c wall
c...........
      elseif(N.eq.3)then
         write(6,*)' '
         write(6,'(''     Panel Wall Lay-ups.'')')
            do 12 i=1,MWP
               write(6,107)i,NWP(i),PT(i),PA(i)
12          continue
         write(6,112)T1
         write(6,*)' '
         write(6,'(''     ***Hit <<Return>> for Next Page.***''$)')
```

```
         read(5,115)tok
         write(6,*)' '
         write(6,*)' '
         write(6,*)' '
         write(6,'(''     Stiffener Wall Lay-ups.'')')
            do 13 i=1,MWS
               write(6,107)i,NWS(i),ST(i),SA(i)
13          continue
         write(6,113)T2
         write(6,*)' '
         write(6,'(''     ***Hit <<Return>> for Next Page.***''$)')
         read(5,115)tok
         write(6,*)' '
         write(6,*)' '
         write(6,*)' '
         go to 10
c........
c load
c........
      elseif(N.eq.4)then
         write(6,116)XL(2)
         write(6,117)SL(1)
         write(6,118)PSURF
         write(6,*)' '
         write(6,'(''     ###Hit <<Return>> for Next Page.***''$)')
         read(5,115)tok
         write(6,*)' '
         go to 10
      elseif(N.eq.5)then
         go to 14
      else
         go to 10
      endif
14    return
      end
c
c#######################
cSUBROUTINE FMOD
c  modification of current database
c#######################
c
      SUBROUTINE fmod
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
      common /load/ XL(4),SL(4),PSURF,X,S,TEMP
      character*1 IDC,tok
 100  format(1x,i2,'>Material ',i2,'.')
 101  format(1x,i2,'>Return to Menu.')
 102  format('     Material Number ',i2,'.')
 103  format('     Ply Number ',i2,'.')
 104  format(a1)
c ...
c . modify files
c ...
10    write(6,*)' '
      write(6,'(''     Choose the Info. to be Modified:'')')
      write(6,'(''      1>Panel and Stiffener Geometry.'')')
```

```
      write(6,'(''      2>Material Data.'')')
      write(6,'(''      3>Wall Lay-ups.'')')
      write(6,'(''      4>Load.'')')
      write(6,'(''      5>Update Info. and List.'')')
      write(6,'(''      6>Exit.'')')
      write(6,'(''     Enter>>''$)')
      call nintg(1,N)
c.........
c geom.dat (unit10): geometry
c.........
      if(N.eq.1)then
11       write(6,*)' '
         write(6,'(''     Choose the Following Geometric Parameter to be
     * Modified:'')')
         write(6,'(''      1>Panel Radius.'')')
         write(6,'(''      2>Panel Length, D.'')')
         write(6,'(''      3>Stiffener Height.'')')
         write(6,'(''      4>Stiffener Angle, ALPHA.'')')
         write(6,'(''      5>Number of Panels in Axial & Hoop Direction.
     *'')')
         write(6,'(''      6>Boundary Condition.'')')
         write(6,'(''      7>Type of the Analysis.'')')
         write(6,'(''      8>Return to Menu.'')')
         write(6,'(''     Enter>>''$)')
         call nintg(1,M)
         write(6,*)' '
         if(M.eq.1)then
            write(6,'(''     Panel Radius(in)=''$)')
            call nreal(1,R)
            write(6,*)' '
         elseif(M.eq.2)then
            write(6,'(''     Panel Length, D(in)=''$)')
            call nreal(1,D)
            write(6,*)' '
         elseif(M.eq.3)then
            write(6,'(''     Stiffener Height(in)=''$)')
            call nreal(1,B)
            write(6,*)' '
         elseif(M.eq.4)then
            write(6,'(''     Stiffener Angle, ALPHA(degree)=''$)')
            call nreal(1,ALPHA)
            write(6,*)' '
         elseif(M.eq.5)then
            write(6,'(''     Number of Panels in Axial Direction=''$)')
            call nintg(1,NBZ)
            write(6,*)' '
            write(6,'(''     Number of Panels in Hoop Direction=''$)')
            call nintg(1,NBY)
            write(6,*)' '
         elseif(M.eq.6)then
            write(6,'(''     Boundary Condition:'')')
            write(6,'(''      1>Infinite Panel.'')')
            write(6,'(''      2>Sides Clamped; Ends Free.'')')
            write(6,'(''      3>Ends Clamped; Sides Free.'')')
            write(6,'(''     Enter >>''$)')
            call nintg(1,NBC)
            write(6,*)' '
         elseif(M.eq.7)then
            write(6,'(''     Type of the Analysis:1>Stress. 2>Buckling.
     *'')')
```

```
            write(6,'(''     Enter >>''$)')
            call nintg(1,NMESH)
            write(6,*)' '
         else
            go to 10
         endif
         go to 11
      elseif(N.eq.2)then
c...........
c mat.dat (unit11): material
c...........
12       write(6,'(''     Choose the Material to be Modified.'')')
         NTMP=NMAT+1
         do 13 i=1,NTMP
               if(i.le.NMAT)then
                  write(6,100)i,i
               else
                  write(6,101)i
               endif
13       continue
         write(6,'(''     Enter >>''$)')
         call nintg(1,J)
         write(6,*)' '
         if(J.le.NMAT)then
            write(6,102)J
c            write(6,*)J
            write(6,'(''     Do You Want to Change the Material? [Y or N
     *]''!)')
            write(6,'(''     Enter >>''$)')
            read(5,104)tok
            if(tok .eq. 'Y' .or. tok .eq. 'y')then
               write(6,'(''     Select 1>Isotropic, 2>Orthotropic Materi
     *al.'')')
               write(6,'(''     Enter >> ''$)')
               call nintg(1,MM(J))
               write(6,*)' '
               if(MM(J).eq.1)then
                  go to 16
               else
                  go to 15
               endif
            else
               if(MM(J).eq.1)then
                  go to 17
               else
                  go to 18
               endif
            endif
15          write(6,*)' '
            write(6,'(''     Enter E11, E22 (psi)>>''$)')
            call nreal(2,E1(1,J,1),E1(2,J,1))
            write(6,*)' '
            write(6,'(''     Enter G12, G23, G31 (psi)>>''$)')
            call nreal(3,G1(1,J,1),G1(2,J,1),G1(3,J,1))
            write(6,*)' '
            write(6,'(''     Enter V12 >>''$)')
            call nreal(1,V1(1,J,1))
            write(6,*)' '
            go to 12
16          write(6,*)' '
```

```
            write(6,'(''     Enter E (psi)>>''$)')
            call nreal(1,E(J))
            write(6,*)' '
            write(6,'(''     Enter V >>''$)')
            call nreal(1,V(J))
            write(6,*)' '
            go to 12
17          write(6,*)' '
            write(6,'(''     Choose the Item to be Modified. >>'')')
            write(6,'(''      1>E. 2>V'')')
            write(6,'(''     Enter >>''$)')
            call nintg(1,N1)
            write(6,*)' '
            if(N1.eq.1)then
               write(6,'(''     E(psi)=''$)')
               call nreal(1,E(J))
               write(6,*)' '
            else
               write(6,'(''     V=''$)')
               call nreal(1,V(J))
               write(6,*)' '
            endif
            go to 12
18          write(6,*)' '
            write(6,'(''     Choose the Item to be Modified. >>'')')
            write(6,'(''      1>E11, E22'')')
            write(6,'(''      2>G12, G23, G31'')')
            write(6,'(''      3>V12'')')
            write(6,'(''     Enter >>''$)')
            call nintg(1,N1)
            write(6,*)' '
            if(N1.eq.1)then
               write(6,'(''     E11, E22(psi)=''$)')
               call nreal(2,E1(1,J,1),E1(2,J,1))
               write(6,*)' '
            elseif(N1.eq.2)then
               write(6,'(''     G12, G23, G31(psi)=''$)')
               call nreal(3,G1(1,J,1),G1(2,J,1),G1(3,J,1))
               write(6,*)' '
            else
               write(6,'(''     V12=''$)')
               call nreal(1,V1(1,J,1))
               write(6,*)' '
            endif
            go to 12
         elseif(J.eq.(NMAT+1))then
            go to 10
         else
            write(6,'(''     Entered Number Exceeds Total Number of Mate
     *rial.'')')
            write(6,*)' '
            go to 12
         endif
c...........
c wall
c...........
      elseif(N.eq.3)then
      write(6,*)' '
19    write(6,'(''     Choose One of the Following ---'')')
      write(6,'(''      1>Change Panel Wall.'')')
```

```
      write(6,'(''      2>Change Stiffener Wall.'')')
      write(6,'(''      3>Return to Menu.'')')
      write(6,'(''     Enter>>''$)')
      call nintg(1,M)
      write(6,*)' '
         if(M.eq.1)then
            write(6,*)' '
            write(6,'(''     Do You Want to Change Total Number of Plies
     *? [Y or N]''$)')
            read(5,104)IDC
            write(6,*)' '
c            call stitle(IDC)
            if(IDC.eq.'Y'.or.IDC.eq.'y')then
               write(6,'(''     Number of Plies for Panel Wall=''$)')
               call nintg(1,MWP)
               write(6,*)' '
               do 20 i=1,MWP
                  write(6,*)' '
                  write(6,103)i
                  write(6,'(''     Enter Material Number. >>''$)')
                  call nintg(1,NWP(i))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Thickness. (in)>>''$)')
                  call nreal(1,PT(i))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Angle. (deg.)>>''$)')
                  call nreal(1,PA(i))
                  write(6,*)' '
20             continue
               T1=0.
               do 21 i=1,MWP
                  T1=T1+PT(i)
21             continue
            else
22             write(6,'(''     Enter Ply Number for Change. >>''$)')
               call nintg(1,N)
               write(6,*)' '
               if(N.gt.MWP)then
                  write(6,'(''     Number Entered Exceeds Total Panel Pl
     *ies.'')')
                  write(6,*)' '
                  go to 22
               else
                  write(6,'(''     Enter Material Number. >>''$)')
                  call nintg(1,NWP(N))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Thickness. (in)>>''$)')
                  call nreal(1,PT(N))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Angle. (deg.)>>''$)')
                  call nreal(1,PA(N))
                  write(6,*)' '
                  T1=0.
                  do 23 i=1,MWP
                     T1=T1+PT(i)
23                continue
               endif
            endif
            go to 19
         elseif(M.eq.2)then
```

```
            write(6,*)' '
            write(6,'(''     *****Warning!! Stiffener Lay-up Must be Sym
     *metric.****'')')
            write(6,'(''     Do You Want to Change Total Number of Plies
     *? [Y or N]''$)')
            read(5,104)IDC
            write(6,*)' '
c            call stitle(IDC)
            if(IDC.eq.'Y'.or.IDC.eq.'y')then
               write(6,'(''     Number of Plies for Stiffener=''$)')
               call nintg(1,MWS)
               write(6,*)' '
               do 24 i=1,MWS
                  write(6,*)' '
                  write(6,103)i
                  write(6,'(''     Enter Material Number. >>''$)')
                  call nintg(1,NWS(i))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Thickness. (in)>>''$)')
                  call nreal(1,ST(i))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Angle. (degree)>>''$)')
                  call nreal(1,SA(i))
                  write(6,*)' '
24             continue
               T2=0.
               do 25 i=1,MWS
                  T2=T2+ST(i)
25             continue
            else
26             write(6,'(''     Enter Ply Number for Change. >>''$)')
               call nintg(1,N)
               write(6,*)' '
               if(N.gt.MWS)then
                  write(6,'(''     Number Enter Exceeds Total Stiffener
     *Plies.'')')
                  write(6,*)' '
                  go to 26
               else
                  write(6,'(''     Enter Material Number. >>''$)')
                  call nintg(1,NWS(N))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Thickness. (in)>>''$)')
                  call nreal(1,ST(N))
                  write(6,*)' '
                  write(6,'(''     Enter Ply Angle. (deg.)>>''$)')
                  call nreal(1,SA(N))
                  write(6,*)' '
               endif
               T2=0.
               do 27 i=1,MWS
                  T2=T2+ST(i)
27             continue
            endif
            go to 19
         else
            write(6,*)' '
            go to 10
         endif
c.........
```

```
c load
c.........
      elseif(N.eq.4)then
         TEMP=D
         A2=ALPHA/2.
         TEMP1=TEMP/COSD(A2)
         TEMP=TEMP1*SIND(A2)
         TEMP=TEMP*2.
         TEMP=TEMP*NBY
28       write(6,'(''     Which Load do You Want to Change?'')')
         write(6,'(''      1>Axial Load.'')')
         write(6,'(''      2>Shear Load.'')')
         write(6,'(''      3>Pressure Load.'')')
         write(6,'(''      4>Return to Menu.'')')
         write(6,'(''     Enter >>''$)')
         call nintg(1,M)
         write(6,*)' '
         if(M.eq.1)then
            write(6,'(''     Do You Want to Enter the 1>Total Load or 2>
     *Line Load? >>'')')
            write(6,'(''     Enter >>''$)')
            call nintg(1,ML)
            write(6,*)' '
            if(ML.eq.1)then
               write(6,'(''     Total Axial Load (lb.)=''$)')
               call nreal(1,X)
               write(6,*)' '
               XL(2)=X/TEMP
               XL(4)=X/TEMP
            else
               write(6,'(''     Axial Line Load (lb./in)=''$)')
               call nreal(1,X)
               write(6,*)' '
               XL(2)=X
               XL(4)=X
            endif
            go to 28
         elseif(M.eq.2)then
            write(6,'(''     Do You Want to Enter the 1>Total Load or 2>
     *Line Load? >>'')')
            write(6,'(''     Select >>''$)')
            call nintg(1,ML)
            write(6,*)' '
            if(ML.eq.1)then
               write(6,'(''     On 1>Ends. 2>Sides. >>''$)')
               call nintg(1,NP)
               write(6,*)' '
               write(6,'(''     Total Shear Load (lb.)=''$)')
               call nreal(1,S)
               write(6,*)' '
               if (NP.eq.1) then
                  SL(1)=S/TEMP
                  SL(2)=S/TEMP
                  SL(3)=S/TEMP
                  SL(4)=S/TEMP
               elseif (NP.eq.2) then
                  SL(1)=S/D*NBZ
                  SL(2)=S/D*NBZ
                  SL(3)=S/D*NBZ
                  SL(4)=S/D*NBZ
```

```
               endif
            else
               write(6,'(''     Shear Line Load (lb./in)='',$)')
               call nreal(1,S)
               write(6,*)' '
               SL(1)=S
               SL(2)=S
               SL(3)=S
               SL(4)=S
            endif
            go to 28
         elseif(M.eq.3)then
            write(6,'(''     Pressure Load (lb./in**2)='',$)')
            call nreal(1,PSURF)
            write(6,*)' '
            go to 28
         elseif(M.eq.4)then
            go to 10
         else
            go to 28
         endif
       elseif(N.eq.5)then
         call fsave
         write(6,*)' '
         call flis0
         go to 10
       else
         write(6,*)' '
         write(6,'(''     Do You Really Want to Exit This Program? [Y or
      * N]'')')
         write(6,'(''    HAVE YOU SAVED YOUR DATABASE ???'')')
         write(6,'(''     Enter >>'',$)')
         read(5,104)tok
         if(tok.eq.'Y'.or.tok.eq.'y')then
            write(6,'(''     *****End of data input session.*****'')')
            go to 29
         else
            go to 10
         endif
       endif
29     return
       end
c
c-------------------
c  dial runstream generation subroutines.
c-------------------
c###############
cSUBROUTINE FMESH
c###############
       SUBROUTINE fmesh
       common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
      *              NBC,NMESH
       common /matl/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
      *              E(6),V(6),MM(6)
       common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
c ...
c ...
c .   fmesh : fine mesh generation
c ...
c ...
```

```
      character*40 fmt1,fmt2,fmt3,fmt4,fmt5,fmt6,fmt7,fmt8,fmt9
      character*40 frt1,frt2,frt3,frt4
      fmt1='('' &al2='',f10.4)'
      fmt2='('' &alpha='',f10.4)'
      fmt3='('' &r='',f10.4)'
      fmt4='('' &t1='',f9.6)'
      fmt5='('' &t2='',f9.6)'
      fmt6='('' &b='',f10.4)'
      fmt7='('' &nby='',i2)'
      fmt8='('' &nbz='',i2)'
      fmt9='('' let &nbc='',i2)'
100   format(' &mwp=',i2,'  #wallam num. for panel')
101   format(' &mws=',i2,'  #wallam num. for whole stif.')
102   format(' &mwh=',i2,'  #wallam num. for middle half stif.')
103   format(' &mwe=',i2,'  #wallam num. for top & bot. half stif.')
      open(unit=12,status='unknown',file='fmesh.com')
      t1=0.
      t2=0.
      do 11 i=1,MWP
         t1=t1+PT(i)
11    continue
      do 12 i=1,MWS
         t2=t2+ST(i)
12    continue
c      write(12,'(''#'')')
c      write(12,'('' ln -s act.f02 FIL002'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
c      write(12,'('' mesh << \!'')')
      write(12,'(''$MESH'')')
      write(12,'('' open 6 "mesh.out"'')')
      write(12,'('' CLEAR -1'')')
      write(12,'('' max 20000,20000'')')
      write(12,'('' ELTYPE 4,2,6'')')
c#########################
c#geodesic fuselage panel; updated version  8/8/90
c#mesh1---panel
c#########################
      write(12,'('' set syntax on'')')
      write(12,'('' sub CONO &lst1 &lst2 &lst3'')')
      write(12,'('' let &nnod=%ipp(1)'')')
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst2'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n1=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
c
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst3)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n2=%ibcl(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
c
      write(12,'('' let &ifn=%ifl(nlst.nv,0,&lst1)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
c
      write(12,' ('';f2 format ''''(1x,i5)'')')
      write(12,' ('';10 format ''''(''''Print Nodal Coordinates'''',1x,'
     *''''Node'''')'')')
      write(12,'('' write 6 ;f2 &n1'')')
      write(12,'('' write 6 ;f2 &n2'')')
      write(12,'('' write 6 ;10'')')
c
```

```
      write(12,'('' do ;20 &n=1,&nnod'')')
      write(12,'('' if %ibcl(&ln,&n) ;20,;20,1'')')
      write(12,'('' let &nn=%ibcl(&ln,&n)'')')
      write(12,'('' write 6 ;f2 &nn'')')
      write(12,'('' conode &nn &n1 &n2 1 1.e+9'')')
      write(12,'('' ;20 continue'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
c
      write(12,'('' return'')')
      write(12,'('' end'')')
c
c
      write(12,'('' let &all=120.'')')
      dtmp=D*2.
      write(12,fmt1) dtmp
      write(12,fmt2) ALPHA
      write(12,'('' &beta=66.                    #angle beta'')')
      write(12,fmt3) R
      write(12,fmt9)NBC
      write(12,'('' &dum1=&alpha/2.'')')
      write(12,'('' &dum1=&dum1/360.'')')
      write(12,'('' &dum1=&dum1*2.'')')
      write(12,'('' &dum1=&dum1*%pi'')')
      write(12,'('' &s2=&al2/2.'')')
      write(12,'('' &s1=&s2*%tan(&dum1)'')')
      write(12,'('' &dum2=2.*&r'')')
      write(12,'('' &dum2=&dum2*%pi'')')
      write(12,'('' &dum3=&s1/&dum2'')')
      write(12,'('' &dum3=&dum3*360.'')')
      write(12,'('' &dum3=%abs(&dum3)'')')
      write(12,'('' &y(1)=&dum3/3. #mpt coord. in theta dir.'')')
      write(12,'('' &y(2)=&y(1)*2.'')')
      write(12,'('' &y(3)=&dum3'')')
      write(12,'('' &y(4)=&dum3+&y(1)'')')
      write(12,'('' &y(5)=&dum3+&y(2)'')')
      write(12,'('' &y(6)=&dum3*2.'')')
      write(12,'('' &z(1)=&s2/3.   #mpt coord. in z dir.'')')
      write(12,'('' &z(2)=&z(1)*2.'')')
      write(12,'('' &z(3)=&s2'')')
      write(12,'('' &z(4)=&s2+&z(1)'')')
      write(12,'('' &z(5)=&s2+&z(2)'')')
      write(12,'('' &z(6)=&s2*2.'')')
      write(12,fmt4) T1
      write(12,fmt5) T2
      write(12,'('' &t3=&t2/2.     #half thick. of stif.'')')
      write(12,'('' &of1=&t1/2.    #panel offset'')')
      write(12,'('' &of2=&t3/2.    #bottom stif. offset'')')
      write(12,'('' &of3=&t3/-2.   #top stif. offset'')')
      write(12,fmt6) B
      write(12,'('' &n=3'')')
      write(12,fmt7) NBY
      rnbz=NBZ*1./2.
      inbz=INT(rnbz)
      write(12,fmt8) inbz
      write(12,fmt9) NMAT
      write(12,'('' &mw=2   #num. of wall layup'')')
      write(12,100)NMAT+1
      write(12,101)NMAT+2
      write(12,102)NMAT+3
      write(12,103)NMAT+4
```

```
write(12,'('' DEFSYS  1 1  0.,0.,0., 1.,0.,0., 1.,1.,0.'')')
write(12,'('' IJPOINT   1   1   1   &r  0.  0.  1'')')
write(12,'('' 2   3   1    &r      0.      0.      1'')')
write(12,'('' 3   7   1    &r      0.      0.      1'')')
write(12,'('' 4   9   1    &r   &y(1)      0.      1'')')
write(12,'('' 5  11   1    &r   &y(2)      0.      1'')')
write(12,'('' 6  13   1    &r   &y(3)      0.      1'')')
write(12,'('' 7  15   1    &r   &y(4)      0.      1'')')
write(12,'('' 8  17   1    &r   &y(5)      0.      1'')')
write(12,'('' 9  19   1    &r   &y(6)      0.      1'')')
write(12,'('' 10  23   1    &r   &y(6)      0.      1'')')
write(12,'('' 11  25   1    &r   &y(6)      0.      1'')')
write(12,'('' 12  25   3    &r   &y(6)   &z(1)      1'')')
write(12,'('' 13  25   5    &r   &y(6)   &z(2)      1'')')
write(12,'('' 14  25   7    &r   &y(6)   &z(3)      1'')')
write(12,'('' 15  25   9    &r   &y(6)   &z(4)      1'')')
write(12,'('' 16  25  11    &r   &y(6)   &z(5)      1'')')
write(12,'('' 17  25  13    &r   &y(6)   &z(6)      1'')')
write(12,'('' 18  23  13    &r   &y(6)   &z(6)      1'')')
write(12,'('' 19  19  13    &r   &y(6)   &z(6)      1'')')
write(12,'('' 20  17  13    &r   &y(5)   &z(6)      1'')')
write(12,'('' 21  15  13    &r   &y(4)   &z(6)      1'')')
write(12,'('' 22  13  13    &r   &y(3)   &z(6)      1'')')
write(12,'('' 23  11  13    &r   &y(2)   &z(6)      1'')')
write(12,'('' 24   9  13    &r   &y(1)   &z(6)      1'')')
write(12,'('' 25   7  13    &r      0.   &z(6)      1'')')
write(12,'('' 26   3  13    &r      0.   &z(6)      1'')')
write(12,'('' 27   1  13    &r      0.   &z(6)      1'')')
write(12,'('' 28   1  11    &r      0.   &z(5)      1'')')
write(12,'('' 29   1   9    &r      0.   &z(4)      1'')')
write(12,'('' 30   1   7    &r      0.   &z(3)      1'')')
write(12,'('' 31   1   5    &r      0.   &z(2)      1'')')
write(12,'('' 32   1   3    &r      0.   &z(1)      1'')')
write(12,'('' 33   7   3    &r   &y(1)   &z(1)      1'')')
write(12,'('' 34   7   5    &r   &y(2)   &z(2)      1'')')
write(12,'('' 35   7   7    &r   &y(3)   &z(3)      1'')')
write(12,'('' 36   7   9    &r   &y(2)   &z(4)      1'')')
write(12,'('' 37   7  11    &r   &y(1)   &z(5)      1'')')
write(12,'('' 38  13   3    &r   &y(3)   &z(1)      1'')')
write(12,'('' 39  13   5    &r   &y(3)   &z(2)      1'')')
write(12,'('' 40  13   7    &r   &y(3)   &z(3)      1'')')
write(12,'('' 41  13   9    &r   &y(3)   &z(4)      1'')')
write(12,'('' 42  13  11    &r   &y(3)   &z(5)      1'')')
write(12,'('' 43  19   3    &r   &y(5)   &z(1)      1'')')
write(12,'('' 44  19   5    &r   &y(4)   &z(2)      1'')')
write(12,'('' 45  19   7    &r   &y(3)   &z(3)      1'')')
write(12,'('' 46  19   9    &r   &y(4)   &z(4)      1'')')
write(12,'('' 47  19  11    &r   &y(5)   &z(5)      1'')')
write(12,'('' 48   3   7    &r   &y(1)   &z(3)      1'')')
write(12,'('' 49   5   7    &r   &y(2)   &z(3)      1'')')
write(12,'('' 50  21   7    &r   &y(4)   &z(3)      1'')')
write(12,'('' 51  23   7    &r   &y(5)   &z(3)      1'')')
write(12,'('' 52   0   0    0.      0.      0.      1'')')
write(12,'('' 53   0   0    0.      0.   &z(1)      1'')')
write(12,'('' 54   0   0    0.      0.   &z(2)      1'')')
write(12,'('' 55   0   0    0.      0.   &z(3)      1'')')
write(12,'('' 56   0   0    0.      0.   &z(4)      1'')')
write(12,'('' 57   0   0    0.      0.   &z(5)      1'')')
write(12,'('' 58   0   0    0.      0.   &z(6)      1'')')
write(12,'('' slines     1t3:9t19:25t32,1'')')
```

```
      write(12,'('' 6,38:38t42:42,22'')')
      write(12,'('' 35,40:40,45'')')
      write(12,'('' circles  3,4,52'')')
      write(12,'('' 4,5,52:5,6,52:6,7,52:7,8,52:8,9,52'')')
      write(12,'('' 32,33,53:33,38,53:38,43,53:43,12,53'')')
      write(12,'('' 31,34,54:34,39,54:39,44,54:44,13,54'')')
      write(12,'('' 30,48,55:48,49,55:49,35,55'')')
      write(12,'('' 45,50,55:50,51,55:51,14,55'')')
      write(12,'('' 29,36,56:36,41,56:41,46,56:46,15,56'')')
      write(12,'('' 28,37,57:37,42,57:42,47,57:47,16,57'')')
      write(12,'('' 25,24,58:24,23,58:23,22,58'')')
      write(12,'('' 22,21,58:21,20,58:20,19,58'')')
      write(12,'('' splines  3,35,1.,3,33,34,35'')')
      write(12,'('' 35,25,1.,35,36,37,25'')')
      write(12,'('' 9,45,1.,9,43,44,45'')')
      write(12,'('' 45,19,1.,45,46,47,19'')')
      write(12,'('' ksshell  1,26,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,LEFT'')')
      write(12,'('' ksshell  3,43,1,&tl,&tl,&tl,&tl,&ofl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,DOWN'')')
      write(12,'('' ksshell  10,17,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,RIGHT'')')
      write(12,'('' ksshell  37,19,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,UP'')')
      write(12,'('' ksshell  2,25,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1'')')
      write(12,'('' ksshell  33,47,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1'')')
      write(12,'('' ksshell  9,18,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1'')')
      write(12,'('' ijname 1 3 BTL'')')
      write(12,'('' 9 11 BTR'')')
      write(12,'('' 19 17 TOL'')')
      write(12,'('' 27 25 TOR'')')
      write(12,'('' mesh'')')
      write(12,'('' merge'')')
c...#######
c...#mesh2,3--stiffener
c...#######
      write(12,'('' ijpoint 1  1  1   &r   0.  &z(3)   1'')')
      write(12,'('' 2   3   1  &r    &y(1)    &z(3)   1'')')
      write(12,'('' 3   5   1  &r    &y(2)    &z(3)   1'')')
      write(12,'('' 4   7   1  &r    &y(3)    &z(3)   1'')')
      write(12,'('' 5   9   1  &r    &y(2)    &z(2)   1'')')
      write(12,'('' 6  11   1  &r    &y(1)    &z(1)   1'')')
      write(12,'('' 7  13   1  &r       0.       0.   1'')')
      write(12,'('' 8  15   1  &r    &y(1)       0.   1'')')
      write(12,'('' 9  17   1  &r    &y(2)       0.   1'')')
      write(12,'('' 10 19   1  &r    &y(3)       0.   1'')')
      write(12,'('' 11 21   1  &r    &y(4)       0.   1'')')
      write(12,'('' 12 23   1  &r    &y(5)       0.   1'')')
      write(12,'('' 13 25   1  &r    &y(6)       0.   1'')')
      write(12,'('' 14 27   1  &r    &y(5)    &z(1)   1'')')
      write(12,'('' 15 29   1  &r    &y(4)    &z(2)   1'')')
      write(12,'('' 16 31   1  &r    &y(3)    &z(3)   1'')')
      write(12,'('' 17 33   1  &r    &y(4)    &z(3)   1'')')
      write(12,'('' 18 35   1  &r    &y(5)    &z(3)   1'')')
      write(12,'('' 19 37   1  &r    &y(6)    &z(3)   1'')')
      write(12,'('' 20 37   3  &r-&b &y(6)    &z(3)   1'')')
      write(12,'('' 21 35   3  &r-&b &y(5)    &z(3)   1'')')
```

```
      write(12,'('' 22  33   3  &r-&b &y(4)    &z(3)   1'')')
      write(12,'('' 23  31   3  &r-&b &y(3)    &z(3)   1'')')
      write(12,'('' 24  29   3  &r-&b &y(4)    &z(2)   1'')')
      write(12,'('' 25  27   3  &r-&b &y(5)    &z(1)   1'')')
      write(12,'('' 26  25   3  &r-&b &y(6)       0.   1'')')
      write(12,'('' 27  23   3  &r-&b &y(5)       0.   1'')')
      write(12,'('' 28  21   3  &r-&b &y(4)       0.   1'')')
      write(12,'('' 29  19   3  &r-&b &y(3)       0.   1'')')
      write(12,'('' 30  17   3  &r-&b &y(2)       0.   1'')')
      write(12,'('' 31  15   3  &r-&b &y(1)       0.   1'')')
      write(12,'('' 32  13   3  &r-&b    0.       0.   1'')')
      write(12,'('' 33  11   3  &r-&b &y(1)    &z(1)   1'')')
      write(12,'('' 34   9   3  &r-&b &y(2)    &z(2)   1'')')
      write(12,'('' 35   7   3  &r-&b &y(3)    &z(3)   1'')')
      write(12,'('' 36   5   3  &r-&b &y(2)    &z(3)   1'')')
      write(12,'('' 37   3   3  &r-&b &y(1)    &z(3)   1'')')
      write(12,'('' 38   1   3  &r-&b    0.    &z(3)   1'')')
      write(12,'('' 39   0   0     0.    0.       0.   1'')')
      write(12,'('' 40   0   0     0.    0.    &z(1)   1'')')
      write(12,'('' 41   0   0     0.    0.    &z(2)   1'')')
      write(12,'('' 42   0   0     0.    0.    &z(3)   1'')')
      write(12,'('' slines    1,38:4,35:7,32:13,26:16,23:19,20'')')
      write(12,'('' circles   7,8,39:8,9,39:9,10,39:10,11,39'')')
      write(12,'('' 11,12,39:12,13,39'')')
      write(12,'('' 32,31,39:31,30,39'')')
      write(12,'('' 30,29,39:29,28,39:28,27,39:27,26,39'')')
      write(12,'('' 1,2,42:2,3,42:3,4,42'')')
      write(12,'('' 16,17,42:17,18,42:18,19,42'')')
      write(12,'('' 38,37,42:37,36,42:36,35,42'')')
      write(12,'('' 23,22,42:22,21,42:21,20,42'')')
      write(12,'('' splines   7,4,1.,7,6,5,4'')')
      write(12,'(''    32,35,1.,32,33,34,35'')')
      write(12,'(''    26,23,1.,26,25,24,23'')')
      write(12,'(''    13,16,1.,13,14,15,16'')')
      write(12,'('' ksshell    1,35,1,&t3,&t3,&t3,&t3,>'')')
      write(12,'('' &of3,&of3,&of3,&of3,&mwh,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    4,32,1,&t2,&t2,&t2,&t2,4m0.,>'')')
      write(12,'('' &mws,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    7,26,1,&t3,&t3,&t3,&t3,>'')')
      write(12,'('' &of2,&of2,&of2,&of2,&mwe,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    13,23,1,&t2,&t2,&t2,&t2,4m0.,>'')')
      write(12,'('' &mws,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    16,20,1,&t3,&t3,&t3,&t3,>'')')
      write(12,'('' &of3,&of3,&of3,&of3,&mwh,msh,1,-1,0,0.,STIF'')')
      write(12,'('' mesh'')')
      write(12,'('' merge mesh=1'')')
      write(12,'('' ditto mesh=2'')')
      write(12,'('' mirror 3 0 1'')')
      write(12,'('' trans  2.*&s2 3'')')
      write(12,'('' merge mesh=1,2'')')
c...##########
c...#auto bay generation
c...##########
      write(12,'('' let &nm=3  #mesh number counter'')')
      write(12,'('' let &dz=&z(6)  #axial position'')')
      write(12,'('' let &dy=&y(6)  #theta position'')')
      write(12,'('' if &nbz-1,8,8,1'')')
      write(12,'('' do ;10 &i=1,&nbz-1,1'')')
      write(12,'('' ditto mesh=1t3'')')
      write(12,'('' trans &dz 3'')')
```

```
      write(12,'('' merge'')')
      write(12,'('' let &dz=&dz+&z(6)'')')
      write(12,'('' let &nm=&nm+1'')')
      write(12,'('' ;10 nop'')')
      write(12,'('' continue'')')
      write(12,'('' if &nby-1,10,10,1'')')
      write(12,'('' do ;11 &i=1,&nby-1,1'')')
      write(12,'('' let &nd=1 #ditto mesh number counter'')')
      write(12,'('' ditto mesh=&nd'')')
      write(12,'('' rotate &dy 3'')')
      write(12,'('' merge'')')
      write(12,'('' let &nd=&nd+1'')')
      write(12,'('' if &nd-&nm,-4,-4,1'')')
      write(12,'('' let &dy=&dy+&y(6)'')')
      write(12,'('' ;11 nop'')')
      write(12,'('' continue'')')
      write(12,'('' let &br1=0.'')')
      write(12,'('' &br2=&r+0.01'')')
      write(12,'('' &bly1=-0.1'')')
      write(12,'('' &bly2=0.1+&y(1)'')')
      write(12,'('' &bry1=&dy-&y(1)'')')
      write(12,'('' &bry1=&bry1-0.1'')')
      write(12,'('' &bry2=&dy+0.1'')')
      write(12,'('' &bdz1=-0.1'')')
      write(12,'('' &bdz2=0.1+&z(1)'')')
      write(12,'('' &buz1=&dz-&z(1)'')')
      write(12,'('' &buz1=&buz1-0.1'')')
      write(12,'('' &buz2=&dz+0.1'')')
      write(12,'('' mset 1 insert name=left  #edge 4'')')
c      write(12,'('' mset 1 insert name=stif'')')
      write(12,'('' mset 1 mask volu >'')')
      write(12,'('' &br1,&br2 &bly1,&bly2 &bdz1,&buz2 1'')')
      write(12,'('' mset 2 insert name=down  #edge 1'')')
      write(12,'('' mset 2 mask volu >'')')
      write(12,'('' &br1,&br2 &bly1,&bry2 &bdz1,&bdz2 1'')')
      write(12,'('' mset 3 insert name=right #edge 2'')')
      write(12,'('' mset 3 mask volu >'')')
      write(12,'('' &br1,&br2 &bry1,&bry2 &bdz1,&buz2 1'')')
      write(12,'('' mset 4 insert name=up    #edge 3'')')
      write(12,'('' mset 4 mask volu >'')')
      write(12,'('' &br1,&br2 &bly1,&bry2 &buz1,&buz2 1'')')
      write(12,'('' mset 5 insert name=stif  #stiffener'')')
      write(12,'('' mset 6 insert mat1=&mwp  #panel'')')
c...# b c conditions setup
c...# nbc=1 infinite panel.
c...#    =2 sides clamped.
c...#    =3 ends clamped.
c...########
      write(12,'('' nset 1 insert mset=6'')')
      write(12,'('' nset 1 insert mset=5'')')
      N=1
      write(12,'('' if &nbc-2  1,;ca2,;ca3'')')
c#case one, part of a larger panel
      write(12,'('' bcsys -4 1 2 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -4 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' goto ;con'')')
c#case two, sides are clamped
      write(12,'('' ;ca2 bcsys -4 1 2 0. 1.e+9 nset=1'')')
```

```
      write(12,'('' bcsys -5 1 2 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -6 1 2 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -1 1 2 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -3 1 2 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -4 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' bcsys -6 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' bcsys -1 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' bcsys -3 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' goto ;con'')')
c#case three, ends are clamped
      write(12,'('' ;ca3 bcsys -4 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -6 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -3 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -2 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -4 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' bcsys -6 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' bcsys -3 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' bcsys -2 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' ;con continue'')')
      write(12,'('' nset 13 insert mset 3'')')
      write(12,'('' nset 14 insert mset 4'')')
      write(12,'('' nset 99 insert name=BTL'')')
      write(12,'('' nset 98 insert name=BTR'')')
      write(12,'('' nset 97 insert name=TOL'')')
      write(12,'('' nset 96 insert name=TOR'')')
      write(12,'('' nset 99 list'')')
      write(12,'('' nset 98 list'')')
      write(12,'('' nset 97 list'')')
      write(12,'('' nset 96 list'')')
      write(12,'('' nset 98 mask nset 13'')')
      write(12,'('' nset 97 mask nset 13'')')
      write(12,'('' nset 97 mask volu -999.,999.,-999.,999.,&buzl,&buz2
     *'')')
      write(12,'('' nset 96 mask nset 14'')')
      write(12,'('' nset 98 list'')')
      write(12,'('' nset 97 list'')')
      write(12,'('' nset 96 list'')')
c#####bottom left corner node
      write(12,'('' nlist 99 insert nset 99'')')
c#####bottom right corner node
      write(12,'('' nlist 98 insert nset 98'')')
c#####top right corner node
      write(12,'('' nlist 97 insert nset 97'')')
c#####top left corner node
      write(12,'('' nlist 96 insert nset 96'')')
      write(12,'('' dofsup 1 nset=99'')')
      write(12,'('' dofsup 0 nset=98'')')
      write(12,'('' dofsup 1 nset=97'')')
      write(12,'('' dofsup 2 nset=97'')')
      write(12,'('' set echo on'')')
c#############
c#all nodes on bottom edge
      write(12,'('' nset 95 insert volu -999.,999.,-999.,999.,0.,0.'')')
      write(12,'('' nset 95 list'')')
c#all nodes on top edge
      write(12,'('' nset 94 insert volu -999.,999.,-999.,999.,&dz,&dz
     *'')')
```

```
      write(12,'('' nset 94 list'')')
c#all nodes on left edge
      write(12,'('' nset 93 insert volu -999.,999.,0.,0.,-999.,999.'')')
c#all nodes on right edge
      write(12,'('' nset 92 insert volu 0.,999.,&dy,&dy,-999.,999.,1,0.0
     *01'')')
      write(12,'('' nset 93 dele nset 95'')')
      write(12,'('' nset 93 dele nset 94'')')
      write(12,'('' nset 92 dele nset 95'')')
      write(12,'('' nset 92 dele nset 94'')')
      write(12,'('' nset 93 list'')')
      write(12,'('' nset 92 list'')')
      write(12,'('' nlist 95 insert nset 95'')')
      write(12,'('' nlist 94 insert nset 94'')')
      write(12,'('' nlist 93 insert nset 93'')')
      write(12,'('' nlist 92 insert nset 92'')')
      write(12,'('' show var dy'')')
      write(12,'('' call cono 95 99 98'')')
      write(12,'('' call cono 94 96 97'')')
      write(12,'('' call cono 93 99 96'')')
      write(12,'('' call cono 92 98 97'')')
      write(12,'('' FINISH'')')
      write(12,'('' STOP'')')
c      write(12,'(''!'')')
      close(unit=12)
      return
      end
c#############
cSUBROUTINE FMESH2
c#############
      SUBROUTINE fmesh2
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /mat1/ NMAT,E1(3,6,5),V1(3,6,5),G1(3,6,5),
     *              E(6),V(6),MM(6)
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
c ...
c ...
c .   fmesh : coarse mesh generation
c ...
c ...
      character*40 fmt1,fmt2,fmt3,fmt4,fmt5,fmt6,fmt7,fmt8,fmt9
      character*40 frt1,frt2,frt3,frt4
      fmt1='('' &al2='',f10.4)'
      fmt2='('' &alpha='',f10.4)'
      fmt3='('' &r='',f10.4)'
      fmt4='('' &t1='',f9.6)'
      fmt5='('' &t2='',f9.6)'
      fmt6='('' &b='',f10.4)'
      fmt7='('' &nby='',i2)'
      fmt8='('' &nbz='',i2)'
      fmt9='('' let &nbc='',i2)'
100   format(' &mwp=',i2,'  #wallam num. for panel')
101   format(' &mws=',i2,'  #wallam num. for whole stif.')
102   format(' &mwh=',i2,'  #wallam num. for middle half stif.')
103   format(' &mwe=',i2,'  #wallam num. for top & bot. half stif.')
      t1=0.
      t2=0.
      do 11 i=1,MWP
         t1=t1+PT(i)
```

```
11    continue
      do 12 i=1,MWS
         t2=t2+ST(i)
12    continue
      open(unit=12,status='unknown',file='fmesh.com')
c      write(12,'(''#'')')
c      write(12,'('' ln -s act.f02 FIL002'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
c      write(12,'('' mesh << \!'')')
      write(12,'(''$MESH '')')
      write(12,'('' open 6 "mesh.out"'')')
      write(12,'('' CLEAR -1'')')
      write(12,'('' max 20000,20000'')')
      write(12,'('' ELTYPE 4,2,6'')')
c########################
c#geodesic fuselage panel; updated version  8/21/90
c#mesh1---panel
c#########################
      write(12,'('' set syntax on'')')
      write(12,'('' sub CONO &lst1 &lst2 &lst3'')')
c#lst1-- node list to be constrained.
c#lst2,lst3--corner nodes.
      write(12,'('' let &nnod=%ipp(1)'')')
c#
      write(12,'('' let &ifn=%if1(nlst.nv,0,&lst2)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n1=%ibc1(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
c#
      write(12,'('' let &ifn=%if1(nlst.nv,0,&lst3)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
      write(12,'('' let &n2=%ibc1(&ln,1)'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
c#
      write(12,'('' let &ifn=%if1(nlst.nv,0,&lst1)'')')
      write(12,'('' let &ln=%lfm(&ifn,1)'')')
c#
      write(12,'('' ;f2 format ''''(1x,i5)'')')
      write(12,'('' ;10 format ''''(''''Print Nodal Coordinates'''',1x,'
     *'''Node'''')'')')
      write(12,'('' write 6 ;f2 &n1'')')
      write(12,'('' write 6 ;f2 &n2'')')
      write(12,'('' write 6 ;10'')')
c#
      write(12,'('' do ;20 &n=1,&nnod'')')
      write(12,'('' if %ibc1(&ln,&n) ;20,;20,1'')')
      write(12,'('' let &nn=%ibc1(&ln,&n)'')')
      write(12,'('' write 6 ;f2 &nn'')')
      write(12,'('' conode &nn &n1 &n2 1 1.e+9'')')
      write(12,'('' ;20 continue'')')
      write(12,'('' let &d=%rfm(&ifn,1,0,&ln)'')')
c#
      write(12,'('' return'')')
      write(12,'('' end'')')
      write(12,'('' let &all=120.'')')
      dtmp=D*2.
      write(12,fmt1) dtmp
      write(12,fmt2) ALPHA
      write(12,'('' &beta=66.                    #angle beta'')')
      write(12,fmt3) R
```

```
      write(12,'('' &dum1=&alpha/2.'')')
      write(12,'('' &dum1=&dum1/360.'')')
      write(12,'('' &dum1=&dum1*2.'')')
      write(12,'('' &dum1=&dum1*%pi'')')
      write(12,'('' &s2=&al2/2.'')')
      write(12,'('' &s1=&s2*%tan(&dum1)'')')
      write(12,'('' &dum2=2.*&r'')')
      write(12,'('' &dum2=&dum2*%pi'')')
      write(12,'('' &dum3=&s1/&dum2'')')
      write(12,'('' &dum3=&dum3*360.'')')
      write(12,'('' &dum3=%abs(&dum3)'')')
      write(12,'('' &y(1)=&dum3/2. #mpt coord. in theta dir.'')')
      write(12,'('' &y(2)=&dum3'')')
      write(12,'('' &y(3)=&dum3+&y(1)'')')
      write(12,'('' &y(4)=&dum3*2.'')')
      write(12,'('' &z(1)=&s2/2.   #mpt coord. in z dir.'')')
      write(12,'('' &z(2)=&s2'')')
      write(12,'('' &z(3)=&s2+&z(1)'')')
      write(12,'('' &z(4)=&s2*2.'')')
c      write(12,'(''THIS IS A TEST OUTPUT'')')
c      write(12,*)T1
      write(12,fmt4)T1
      write(12,fmt5)T2
      write(12,'('' &t3=&t2/2.      #half thick. of stif.'')')
      write(12,'('' &of1=&t1/2.     #panel offset'')')
      write(12,'('' &of2=&t3/2.     #bottom stif. offset'')')
      write(12,'('' &of3=&t3/-2.    #top stif. offset'')')
      write(12,fmt6) B
      write(12,'('' &n=3'')')
      write(12,fmt7) NBY
      rnbz=NBZ*1./2.
      inbz=INT(rnbz)
      write(12,fmt8) inbz
      write(12,fmt9) NBC
      write(12,'('' &mw=2   #num. of wall layup'')')
      write(12,100)NMAT+1
      write(12,101)NMAT+2
      write(12,102)NMAT+3
      write(12,103)NMAT+4
      write(12,'('' DEFSYS  1 1  0.,0.,0., 1.,0.,0., 1.,1.,0.'')')
      write(12,'('' IJPOINT   1   1   1   &r  0.  0.  1'')')
      write(12,'('' 2   3   1    &r      0.      0.      1'')')
      write(12,'('' 3   5   1    &r      0.      0.      1'')')
      write(12,'('' 4   7   1    &r   &y(1)      0.      1'')')
      write(12,'('' 5   9   1    &r   &y(2)      0.      1'')')
      write(12,'('' 6  11   1    &r   &y(3)      0.      1'')')
      write(12,'('' 7  13   1    &r   &y(4)      0.      1'')')
      write(12,'('' 8  15   1    &r   &y(4)      0.      1'')')
      write(12,'('' 9  17   1    &r   &y(4)      0.      1'')')
      write(12,'('' 10  17   3    &r   &y(4)   &z(1)      1'')')
      write(12,'('' 11  17   5    &r   &y(4)   &z(2)      1'')')
      write(12,'('' 12  17   7    &r   &y(4)   &z(3)      1'')')
      write(12,'('' 13  17   9    &r   &y(4)   &z(4)      1'')')
      write(12,'('' 14  15   9    &r   &y(4)   &z(4)      1'')')
      write(12,'('' 15  13   9    &r   &y(4)   &z(4)      1'')')
      write(12,'('' 16  11   9    &r   &y(3)   &z(4)      1'')')
      write(12,'('' 17   9   9    &r   &y(2)   &z(4)      1'')')
      write(12,'('' 18   7   9    &r   &y(1)   &z(4)      1'')')
      write(12,'('' 19   5   9    &r      0.   &z(4)      1'')')
      write(12,'('' 20   3   9    &r      0.   &z(4)      1'')')
```

```
      write(12,'('' 21   1   9    &r      0.   &z(4)      1'')')
      write(12,'('' 22   1   7    &r      0.   &z(3)      1'')')
      write(12,'('' 23   1   5    &r      0.   &z(2)      1'')')
      write(12,'('' 24   1   3    &r      0.   &z(1)      1'')')
      write(12,'('' 25   5   3    &r   &y(1)   &z(1)      1'')')
      write(12,'('' 26   5   5    &r   &y(2)   &z(2)      1'')')
      write(12,'('' 27   5   7    &r   &y(1)   &z(3)      1'')')
      write(12,'('' 28   9   3    &r   &y(2)   &z(1)      1'')')
      write(12,'('' 29   9   5    &r   &y(2)   &z(2)      1'')')
      write(12,'('' 30   9   7    &r   &y(2)   &z(3)      1'')')
      write(12,'('' 31  13   3    &r   &y(3)   &z(1)      1'')')
      write(12,'('' 32  13   5    &r   &y(2)   &z(2)      1'')')
      write(12,'('' 33  13   7    &r   &y(3)   &z(3)      1'')')
      write(12,'('' 34   3   5    &r   &y(1)   &z(2)      1'')')
      write(12,'('' 35  15   5    &r   &y(3)   &z(2)      1'')')
      write(12,'('' 36   0   0    0.      0.      0.      1'')')
      write(12,'('' 37   0   0    0.      0.   &z(1)      1'')')
      write(12,'('' 38   0   0    0.      0.   &z(2)      1'')')
      write(12,'('' 39   0   0    0.      0.   &z(3)      1'')')
      write(12,'('' 40   0   0    0.      0.   &z(4)      1'')')
      write(12,'('' slines     1t3:7t15:19t24,1'')')
      write(12,'('' 5,28:28t30:30,17'')')
      write(12,'('' 26,29:29,32'')')
      write(12,'('' circles    3,4,36'')')
      write(12,'('' 4,5,36:5,6,36:6,7,36:24,25,37:25,28,37'')')
      write(12,'('' 28,31,37:31,10,37:23,34,38:34,26,38'')')
      write(12,'('' 32,35,38:35,11,38:22,27,39:27,30,39'')')
      write(12,'('' 30,33,39:33,12,39:19,18,40'')')
      write(12,'('' 18,17,40:17,16,40:16,15,40'')')
      write(12,'('' splines    3,26,1.,3,25,26'')')
      write(12,'('' 26,19,1.,26,27,19'')')
      write(12,'('' 7,32,1.,7,31,32'')')
      write(12,'('' 32,15,1.,32,33,15'')')
      write(12,'('' ksshell    1,20,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,LEFT'')')
      write(12,'('' ksshell    3,31,1,&tl,&tl,&tl,&tl,&ofl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,DOWN'')')
      write(12,'('' ksshell    8,13,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,RIGHT'')')
      write(12,'('' ksshell    27,15,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1,0,0,0.,UP'')')
      write(12,'('' ksshell    2,19,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1'')')
      write(12,'('' ksshell    25,33,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1'')')
      write(12,'('' ksshell    7,14,1,&tl,&tl,&tl,&tl,>'')')
      write(12,'('' &ofl,&ofl,&ofl,&ofl,&mwp,msh,1'')')
      write(12,'('' ijname 1 3 BTL'')')
      write(12,'('' 7 9 BTR'')')
      write(12,'('' 15 13 TOL'')')
      write(12,'('' 21 19 TOR'')')
      write(12,'('' mesh'')')
      write(12,'('' merge'')')
c...#######
c...#mesh2,3--stiffener
c...#######
      write(12,'('' ijpoint 1  1  1    &r   0.  &z(2)   1'')')
      write(12,'('' 2   3   1  &r    &y(1)    &z(2)   1'')')
      write(12,'('' 3   5   1  &r    &y(2)    &z(2)   1'')')
      write(12,'('' 4   7   1  &r    &y(1)    &z(1)   1'')')
```

```
      write(12,'('' 5   9   1  &r       0.       0.   1'')')
      write(12,'('' 6  11   1  &r    &y(1)       0.   1'')')
      write(12,'('' 7  13   1  &r    &y(2)       0.   1'')')
      write(12,'('' 8  15   1  &r    &y(3)       0.   1'')')
      write(12,'('' 9  17   1  &r    &y(4)       0.   1'')')
      write(12,'('' 10 19   1  &r    &y(3)    &z(1)   1'')')
      write(12,'('' 11 21   1  &r    &y(2)    &z(2)   1'')')
      write(12,'('' 12 23   1  &r    &y(3)    &z(2)   1'')')
      write(12,'('' 13 25   1  &r    &y(4)    &z(2)   1'')')
      write(12,'('' 14 25   3  &r-&b &y(4)    &z(2)   1'')')
      write(12,'('' 15 23   3  &r-&b &y(3)    &z(2)   1'')')
      write(12,'('' 16 21   3  &r-&b &y(2)    &z(2)   1'')')
      write(12,'('' 17 19   3  &r-&b &y(3)    &z(1)   1'')')
      write(12,'('' 18 17   3  &r-&b &y(4)       0.   1'')')
      write(12,'('' 19 15   3  &r-&b &y(3)       0.   1'')')
      write(12,'('' 20 13   3  &r-&b &y(2)       0.   1'')')
      write(12,'('' 21 11   3  &r-&b &y(1)       0.   1'')')
      write(12,'('' 22  9   3  &r-&b    0.       0.   1'')')
      write(12,'('' 23  7   3  &r-&b &y(1)    &z(1)   1'')')
      write(12,'('' 24  5   3  &r-&b &y(2)    &z(2)   1'')')
      write(12,'('' 25  3   3  &r-&b &y(1)    &z(2)   1'')')
      write(12,'('' 26  1   3  &r-&b    0.    &z(2)   1'')')
      write(12,'('' 27  0   0  0.       0.       0.   1'')')
      write(12,'('' 28  0   0  0.       0.    &z(1)   1'')')
      write(12,'('' 29  0   0  0.       0.    &z(2)   1'')')
      write(12,'('' slines    1,26:3,24:5,22:9,18:11,16:13,14'')')
      write(12,'('' circles   21,22,27:20,21,27:19,20,27:18,19,27'')')
      write(12,'('' 5,6,27:6,7,27'')')
      write(12,'('' 7,8,27:8,9,27'')')
      write(12,'('' 1,2,29:2,3,29:25,26,29:24,25,29'')')
      write(12,'('' 11,12,29:12,13,29:14,15,29:15,16,29'')')
      write(12,'('' splines   5,3,1.,5,4,3'')')
      write(12,'('' 22,24,1.,22,23,24'')')
      write(12,'('' 9,11,1.,9,10,11'')')
      write(12,'('' 16,18,1.,16,17,18'')')
      write(12,'('' ksshell    1,24,1,&t3,&t3,&t3,&t3,>'')')
      write(12,'('' &of3,&of3,&of3,&of3,&mwh,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    3,22,1,&t2,&t2,&t2,&t2,4m0.,>'')')
      write(12,'('' &mws,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    5,18,1,&t3,&t3,&t3,&t3,>'')')
      write(12,'('' &of2,&of2,&of2,&of2,&mwe,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    9,16,1,&t2,&t2,&t2,&t2,4m0.,>'')')
      write(12,'('' &mws,msh,1,-1,0,0.,STIF'')')
      write(12,'('' ksshell    11,14,1,&t3,&t3,&t3,&t3,>'')')
      write(12,'('' &of3,&of3,&of3,&of3,&mwh,msh,1,-1,0,0.,STIF'')')
      write(12,'('' mesh'')')
      write(12,'('' merge mesh=1'')')
      write(12,'('' ditto mesh=2'')')
      write(12,'('' mirror 3 0 1'')')
      write(12,'('' trans  2.*&s2 3'')')
      write(12,'('' merge mesh=1,2'')')
c...##########
c...#auto bay generation
c...##########
      write(12,'('' let &nm=3  #mesh number counter'')')
      write(12,'('' let &dz=&z(4)  #axial position'')')
      write(12,'('' let &dy=&y(4)  #theta position'')')
      write(12,'('' if &nbz-1,8,8,1'')')
      write(12,'('' do ;10 &i=1,&nbz-1,1'')')
      write(12,'('' ditto mesh=1t3'')')
```

```
      write(12,'('' trans &dz 3'')')
      write(12,'('' merge'')')
      write(12,'('' let &dz=&dz+&z(4)'')')
      write(12,'('' let &nm=&nm+1'')')
      write(12,'('' ;10 nop'')')
      write(12,'('' continue'')')
      write(12,'('' if &nby-1,10,10,1'')')
      write(12,'('' do ;11 &i=1,&nby-1,1'')')
      write(12,'('' let &nd=1 #ditto mesh number counter'')')
      write(12,'('' ditto mesh=&nd'')')
      write(12,'('' rotate &dy 3'')')
      write(12,'('' merge'')')
      write(12,'('' let &nd=&nd+1'')')
      write(12,'('' if &nd-&nm,-4,-4,1'')')
      write(12,'('' let &dy=&dy+&y(4)'')')
      write(12,'('' ;11 nop'')')
      write(12,'('' continue'')')
      write(12,'('' let &br1=0.'')')
      write(12,'('' &br2=&r+0.01'')')
      write(12,'('' &bly1=-0.1'')')
      write(12,'('' &bly2=0.1+&y(1)'')')
      write(12,'('' &bry1=&dy-&y(1)'')')
      write(12,'('' &bry1=&bry1-0.1'')')
      write(12,'('' &bry2=&dy+0.1'')')
      write(12,'('' &bdz1=-0.1'')')
      write(12,'('' &bdz2=0.1+&z(1)'')')
      write(12,'('' &buz1=&dz-&z(1)'')')
      write(12,'('' &buz1=&buz1-0.1'')')
      write(12,'('' &buz2=&dz+0.1'')')
      write(12,'('' mset 1 insert name=left  #edge 4'')')
c      write(12,'('' mset 1 insert name=stif'')')
      write(12,'('' mset 1 mask volu >'')')
      write(12,'('' &br1,&br2 &bly1,&bly2 &bdz1,&buz2 1'')')
      write(12,'('' mset 2 insert name=down  #edge 1'')')
      write(12,'('' mset 2 mask volu >'')')
      write(12,'('' &br1,&br2 &bly1,&bry2 &bdz1,&bdz2 1'')')
      write(12,'('' mset 3 insert name=right #edge 2'')')
      write(12,'('' mset 3 mask volu >'')')
      write(12,'('' &br1,&br2 &bry1,&bry2 &bdz1,&buz2 1'')')
      write(12,'('' mset 4 insert name=up    #edge 3'')')
      write(12,'('' mset 4 mask volu >'')')
      write(12,'('' &br1,&br2 &bly1,&bry2 &buz1,&buz2 1'')')
      write(12,'('' mset 5 insert name=stif  #stiffener'')')
      write(12,'('' mset 6 insert mat1=&mwp  #panel'')')
c...# b c conditions setup
c...# nbc=1 infinite panle.
c...# nbc=2 sides clamped.
c...# nbc=3 ends clamped.
c...########
       write(12,'('' nset 1 insert mset=6'')')
       write(12,'('' nset 1 insert mset=5'')')
       N=1
      write(12,'('' if &nbc-2  1,;ca2,;ca3'')')
c#case one, part of a larger panel
      write(12,'('' bcsys -4 1 2 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 3 0. 1.e+9 nset=1'')')
      write(12,'('' bcsys -4 1 2 &dy 1.e+9 nset=1'')')
      write(12,'('' bcsys -5 1 3 &dz 1.e+9 nset=1'')')
      write(12,'('' goto ;con'')')
c#case two, sides are clamped
```

```
       write(12,'('' ;ca2 bcsys -4 1 2 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -5 1 2 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -6 1 2 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -1 1 2 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -3 1 2 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -4 1 2 &dy 1.3+9 nset=1'')')
       write(12,'(''     bcsys -5 1 2 &dy 1.3+9 nset=1'')')
       write(12,'(''     bcsys -6 1 2 &dy 1.3+9 nset=1'')')
       write(12,'(''     bcsys -1 1 2 &dy 1.3+9 nset=1'')')
       write(12,'(''     bcsys -3 1 2 &dy 1.e+9 nset=1'')')
       write(12,'(''     goto ;con'')')
c#case three, ends are clamped
       write(12,'('' ;ca3 bcsys -4 1 3 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -5 1 3 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -6 1 3 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -2 1 3 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -3 1 3 0. 1.e+9 nset=1'')')
       write(12,'(''     bcsys -4 1 3 &dz 1.e+9 nset=1'')')
       write(12,'(''     bcsys -5 1 3 &dz 1.e+9 nset=1'')')
       write(12,'(''     bcsys -6 1 3 &dz 1.e+9 nset=1'')')
       write(12,'(''     bcsys -2 1 3 &dz 1.e+9 nset=1'')')
       write(12,'(''     bcsys -3 1 3 &dz 1.e+9 nset=1'')')
       write(12,'('' ;con continue'')')
c################
c#Conode scheme
c################
       write(12,'('' nset 13 insert mset 3'')')
       write(12,'('' nset 14 insert mset 4'')')
       write(12,'('' nset 99 insert name=BTL'')')
       write(12,'('' nset 98 insert name=BTR'')')
       write(12,'('' nset 97 insert name=TOL'')')
       write(12,'('' nset 96 insert name=TOR'')')
       write(12,'('' nset 99 list'')')
       write(12,'('' nset 98 list'')')
       write(12,'('' nset 97 list'')')
       write(12,'('' nset 96 list'')')
       write(12,'('' nset 98 mask nset 13'')')
       write(12,'('' nset 97 mask nset 13'')')
       write(12,'('' nset 97 mask volu -999.,999.,-999.,999.,&buz1,&buz2
      *'')')
       write(12,'('' nset 96 mask nset 14'')')
       write(12,'('' nset 98 list'')')
       write(12,'('' nset 97 list'')')
       write(12,'('' nset 96 list'')')
c#bottom left corner node
       write(12,'('' nlist 99 insert nset 99'')')
c#bottom right corner node
       write(12,'('' nlist 98 insert nset 98'')')
c#top right corner node
       write(12,'('' nlist 97 insert nset 97'')')
c#top left corner node
       write(12,'('' nlist 96 insert nset 96'')')
       write(12,'('' set echo on'')')
       write(12,'('' dofsup 1 nset=99'')')
       write(12,'('' dofsup 0 nset=98'')')
       write(12,'('' dofsup 1 nset=97'')')
       write(12,'('' dofsup 2 nset=97'')')
c#############
c#all nodes on bottom edge
       write(12,'('' nset 95 insert volu -999.,999.,-999.,999.,0.,0.'')')
```

```
      write(12,'('' nset 95 list'')')
c#all nodes on top edge
      write(12,'('' nset 94 insert volu -999.,999.,-999.,999.,&dz,&dz
     *'')')
      write(12,'('' nset 94 list'')')
c#all nodes on left edge
      write(12,'('' nset 93 insert volu -999.,999.,0.,0.,-999.,999.'')')
c#all nodes on right edge
      write(12,'('' nset 92 insert volu 0.,999.,&dy,&dy,-999.,999.,1,0.0
     *01'')')
      write(12,'('' nset 93 dele nset 95'')')
      write(12,'('' nset 93 dele nset 94'')')
      write(12,'('' nset 92 dele nset 95'')')
      write(12,'('' nset 92 dele nset 94'')')
      write(12,'('' nset 93 list'')')
      write(12,'('' nset 92 list'')')
      write(12,'('' nlist 95 insert nset 95'')')
      write(12,'('' nlist 94 insert nset 94'')')
      write(12,'('' nlist 93 insert nset 93'')')
      write(12,'('' nlist 92 insert nset 92'')')
      write(12,'('' show var dy'')')
      write(12,'('' call cono 95 99 98'')')
      write(12,'('' call cono 94 96 97'')')
      write(12,'('' call cono 93 99 96'')')
      write(12,'('' call cono 92 98 97'')')
      write(12,'('' FINISH'')')
      write(12,'('' STOP'')')
c      write(12,'(''!'')')
      close(unit=12)
      return
      end
c################
c FBSET
c################
      SUBROUTINE fbset
c ...
c ...
c .   fbset : DIAL band, setup generation.
c ...
c ...
      open(unit=12,status='unknown',file='fbset.com')
c      write(12,'(''#'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
c      write(12,'('' band << \!'')')
      write(12,'(''$BAND'')')
      write(12,'('' open 6 "band.out"'')')
      write(12,'('' START -1'')')
      write(12,'('' REGPS'')')
      write(12,'('' BAND'')')
      write(12,'('' STOP'')')
c      write(12,'(''!'')')
c      write(12,'('' setup << \!'')')
      write(12,'(''$SETUP'')')
      write(12,'('' open 6 "setup.out"'')')
      write(12,'('' START -1'')')
      write(12,'('' SETUP'')')
      write(12,'('' STOP'')')
c      write(12,'(''!'')')
      close(unit=12)
      return
```

```
      end
c
c######################
cFMATL
c######################
      SUBROUTINE fmatl
      common /matl/ NMAT,El(3,6,5),Vl(3,6,5),Gl(3,6,5),
     *E(6),V(6),MM(6)
      common /wall/ MWP,MWS,NWP(16),NWS(16),PT(16),PA(16),ST(16),SA(16)
c ...
c ...
c .   fmatl : DIAL matl generation.
c ...
c ...
c ...
c           NMAT -- Number of Material.
c           E    -- Elastic Modulus.
c           V    -- Poissons' Ratio.
c           G    -- Shear Modulus.
c           MWP  -- Number of Plies in Panel Wall.
c           MWS  -- Number of Plies in Stiffener Wall.
c           NWP  -- Material Composition of Panel Wall.
c           NWS  -- Material Composition of Stiffener Wall.
c           PT   -- Panel Ply Thickness.
c           PA   -- Panel Ply Angle.
c           ST   -- Stiffener Ply Thickness.
c           SA   -- Stiffener Ply Angle.
c           MM   -- Iso vs. Ortho material indicator.
c ...
      open(unit=12,status='unknown',file='fmat.com')
c      write(12,'(''#'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
c      write(12,'('' matl << \!'')')
      write(12,'(''$MATL'')')
      write(12,'('' open 6 "matl.out"'')')
      write(12,'('' START -1'')')
      write(12,'('' set syntax on'')')
100   format(' MATISO ',i2,2x,e12.5,2x,f7.5)
101   format(' MATORT ',i2,2x,e12.5,2x,e12.5,2x,e12.5,',>')
102   format(1x,e12.5,2x,e12.5,2x,e12.5,',>')
103   format(1x,e12.5,2x,e12.5,2x,e12.5)
      do 10 i=1,NMAT
         if (MM(i) .EQ. 1) then
            write(12,100)i,E(i),V(i)
         elseif (MM(i) .EQ. 2) then
            write(12,101)i,
     *El(1,i,1),El(2,i,1),El(3,i,1)
            write(12,102)Vl(1,i,1),Vl(2,i,1),Vl(3,i,1)
            write(12,103)Gl(1,i,1),Gl(2,i,1),Gl(3,i,1)
      endif
10    continue
      NMAT=NMAT+1
104   format(' WALLAM',2x,i2,2x,i2,',>')
105   format(1x,f16.5,',>')
106   format(1x,f16.3)
107   format(1x,i2,',>')
c
c..
c     panel layup
c..
```

```
c
      write(12,104)NMAT,MWP
      do 20 i=1,MWP
         write(12,107)NWP(i)
20    continue
      do 21 i=1,MWP
         write(12,105)PT(i)
21    continue
      do 22 i=1,MWP-1
         write(12,105)PA(i)
22    continue
      write(12,106)PA(MWP)
c
c..
c     whole stiffener layup
c..
c
      NMAT=NMAT+1
      write(12,104)NMAT,MWS
      do 30 i=1,MWS
         write(12,107)NWS(i)
30    continue
      do 31 i=1,MWS
         write(12,105)ST(i)
31    continue
      do 32 i=1,MWS-1
         write(12,105)SA(i)
32    continue
      write(12,106)SA(MWS)
c
c..
c     half stiffener layup
c..
c
      TEMP=REAL(MWS)
      TEMP1=TEMP/2.
      TEMP2=TEMP1-INT(TEMP1)
      NEMP=INT(TEMP1)
      IF (MOD(TEMP,2.) .GT. 0.)THEN
         NEMP=NEMP+1
         write(12,104)NMAT+1,NEMP
         do 40 i=1,NEMP
            write(12,107)NWS(i)
40       continue
         do 41 i=1,NEMP-1
            write(12,105)ST(i)
41       continue
         write(12,105)ST(NEMP)/2.
         do 42 i=1,NEMP-1
            write(12,105)SA(i)
42       continue
         write(12,106)SA(NEMP)
c
c
         write(12,104)NMAT+2,NEMP
         do 50 i=NEMP,MWS
            write(12,107)NWS(i)
50       continue
         write(12,105)ST(NEMP)/2.
         do 51 i=NEMP+1,MWS
```

```
            write(12,105)ST(i)
51        continue
          do 52 i=NEMP,MWS-1
             write(12,105)SA(i)
52        continue
          write(12,106)SA(MWS)
       else
          write(12,104)NMAT+1,NEMP
          do 60 i=1,NEMP
            write(12,107)NWS(i)
60        continue
          do 61 i=1,NEMP
             write(12,105)ST(i)
61        continue
          do 62 i=1,NEMP-1
             write(12,105)SA(i)
62        continue
          write(12,106)SA(NEMP)
c
c
C.....................................BEGIN SECT TEMP REMOVED (JEI 3/6/91)
C
CC          write(12,104)NMAT+2,NEMP
CC          do 70 i=NEMP,MWS
CC             write(12,107)NWS(i)
CC70        continue
CC          do 71 i=NEMP,MWS
CC             write(12,105)ST(i)
CC71        continue
CC          do 72 i=NEMP,MWS-1
CC             write(12,105)SA(i)
CC72        continue
CC          write(12,106)SA(MWS)
CC
C.....................................END SECT TEMP. REMOVED  (JEI 3/6/91
C
C.....................................BEGIN SECT TEMP ADDED (JEI 3/6/91)
C
          write(12,104)NMAT+2,NEMP
          do 70 i=NEMP,MWS
             write(12,107)NWS(i)
70        continue
          do 71 i=NEMP+1,MWS
             write(12,105)ST(i)
71        continue
          do 72 i=NEMP+1,MWS
             write(12,105)SA(i)
72        continue
C          write(12,106)SA(MWS)
C
C.....................................END SECT TEMP. ADDED (JEI 3/6/91)
       endif
       write(12,'('' MATL'')')
       write(12,'('' STOP'')')
c       write(12,'(''!'')')
       close(unit=12)
       return
       end
c################
c FLOAD
```

```
c#################
      SUBROUTINE fload
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /load/ XL(4),SL(4),PSURF,X,L,TEMP
      open(unit=12,status='unknown',file='fload.com')
100   format(' pline ',f10.3,1x,i2,1x,i2,1x,i2,' mset=',i2)
101   format(' psurf ',f10.3,1x,i2,1x,i2,' mset=',i2)
c      write(12,'(''#'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
c      write(12,'('' load << \!'')')
      write(12,'('' $load'')')
      write(12,'('' open 6 "load.out"'')')
      write(12,'('' start'')')
      write(12,'('' lcase 1'')')
      write(12,'(''# SHEAR LOADS'')')
      SX1=SL(1)
      SX2=SL(1)*-1.
      write(12,100)SX1,3,0,4,1
      write(12,100)SX2,3,0,1,2
      write(12,100)SX1,3,0,2,3
      write(12,100)SX2,3,0,3,4
      write(12,'(''# END LOADS'')')
      write(12,100)XL(2),2,0,1,2
      write(12,100)XL(4),2,0,3,4
      write(12,'(''# PRESSURE LOAD'')')
      write(12,101)PSURF,1,-1,6
      write(12,'(''# REACTIONARY LOAD TO COUNTER THE PRESSURE LOAD'')')
      psl=PSURF*R
      write(12,100)psl,2,0,4,1
      write(12,100)psl,2,0,2,3
      write(12,'('' load'')')
      write(12,'('' stop'')')
c      write(12,'(''!'')')
      return
      end
c#########################
c FSOLVE
c#########################
      SUBROUTINE fsolve
      common /geom/ NBY,NBZ,NWAL,D,ALPHA,R,T1,T2,B,
     *              NBC,NMESH
      common /load/ XL(4),SL(4),PSURF,X,S,TEMP
      open(unit=12,status='unknown',file='fsolve.com')
c      write(12,'(''#'')')
c      write(12,'('' source /usr/local/dial/init.com'')')
c      write(12,'('' solve << \!'')')
      write(12,'(''$SOLVE'')')
      write(12,'('' open 6 "solve.out"'')')
      write(12,'('' start'')')
100   format(' loads ',i2,f10.3)
      if (NMESH.eq.2) then
         write(12,100)1,1.
         write(12,'('' SAVE,D'')')
         write(12,'('' SAVE,S'')')
         write(12,'('' ASSIGN IMPR=1'')')
         write(12,'('' EIGEN 2'')')
         write(12,'('' solve'')')
         write(12,'('' stop'')')
c         write(12,'(''!'')')
```

```
c          write(12,'('' cluster << \!'')')
          write(12,'(''$cluster'')')
          write(12,'('' start'')')
          write(12,'('' shift 1'')')
          write(12,'('' save'')')
          write(12,'('' matrix:file,ki ki'')')
          write(12,'('' clust 2'')')
          write(12,'('' stop'')')
c          write(12,'(''!'')')
       else
          write(12,100)1,1.
          write(12,'('' SAVE,D'')')
          write(12,'('' SAVE,S'')')
          write(12,'('' solve'')')
          write(12,'('' stop'')')
c          write(12,'(''!'')')
       endif
       close(unit=12)
       return
       end

       subroutine nreal(ntk,ra)
       character arec*80, a*127, tkns(10)*10
       dimension lentk(10),ra(10)
c
  100   ierr=0
       call getlin( 5, 0, arec, a, irdata, *910, *998 )
       call getkns( a, irdata, tkns, lentk, ntk, *999 )
c        - - - - -
         do 200 i=1,ntk
         r = ra(i)
         call rintrp( tkns(i), lentk(i), r, *991 )
c         write(6,'(12x, ''real = '', 1pe10.3)') r
         ra(i) = r
         go to 200
  991      ierr=ierr+1
           write(6,'( 19x, ''Not a real '')')
  200 continue
       if (ierr .ge. 1) go to 100
       return
  998 continue
       write(6,'('' ERROR return from getlin'')')
       stop
  999 continue
       write(6,'('' ERROR return from getkns'')')
       stop
  910 continue
       write(6,'('' End-of-File return from "getlin"'')')
       return
       end
       subroutine nintg(ntk,ia)
       character a*127, arec*80, tkns(10)*10
       dimension lentk(10),ia(10)
c      ---------
  100 ierr=0
       call getlin( 5, 0, arec, a, irdata, *910, *998 )
       call getkns( a, irdata, tkns, lentk, ntk, *999 )
c
       do 200 i=1,ntk
         intgr = ia(i)
```

```
        call iintrp( tkns(i), lentk(i), intgr, *992 )
c        write(6,'(9x, ''integer = '', i10)') intgr
        ia(i) = intgr
c-debug        write(6,'('' In nintg: ia('', i5, '')= '', i5)') i, intgr
        go to 200
  992     ierr=ierr+1
          write(6,'( 19x, ''Not an integer '')')
  200 continue
      if ( ierr .ge. 1) go to 100
      return
c     ---------
  998 continue
      write(6,'('' ERROR return from getlin'')')
      stop
  999 continue
      write(6,'('' ERROR return from getkns'')')
      stop
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
      subroutine stitle(sline)
c
      character arec*80, a*127, tkns(1)*60, sline*(*)
      dimension lentk(1)
c     ---------
   1  ierr=0
      call getlin( 5, 0, arec, a, irdata, *910, *998 )
      ntk = 1
      call getkns( a, irdata, tkns, lentk, ntk, *999 )
      if (lentk(1) .eq. 0) return
      sline=tkns(1)(1:lentk(1))
      return
c
  998 continue
c ...     write(6,'('' ERROR return from getlin'')')
      write(6,'(10x,'' !!! Illegal Input, Try Again !!!'')')
      go to 1
  999 continue
c ...     write(6,'('' ERROR return from getkns'')')
      write(6,'(10x,'' !!! Illegal Input, Try Again !!!'')')
      go to 1
  910 continue
      write(6,'('' End-of-File return from "getlin"'')')
      return
      end
      subroutine getlin( nunit, igblnk, arec, aline, irdata, *, * )
c
c ---------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     nunit ... The Fortran logical unit number of the file from which
c#                     the input record(s) will be read.
c
c#    in     igblnk .. A integer flag that must be either zero or one.
c#                     It indicates the interpretation of input records that
c#                     are blank, or contain only a comment, and that do not
c#                     follow a "continued" record (in which case they always
c#                     indicate end of input for the continuation set).
c#                     1 ==> ignore (i.e., read past) them
```

```
c#                      0 ==> consider them to be a completed data line with
c#                            only "null" tokens.
c
c#      -     arec .... A work string supplied to this routine.  Each single
c#                      input record (i.e., up to the carriage return) is read
c#                      into this string.
c
c#    out     aline ... The returned string containing the "input line".  This
c#                      string will be the data string contained within a single
c#                      uncontinued input record, or it will be the concatenation
c#                      of the data strings from continued records.  The
c#                      accumulated lengths of the data strings (and any implied
c#                      delimiters) in a "continuation set" is limited by the
c#                      length of this string.
c
c#    out     irdata .. The returned data line, aline, extends to this character
c#                      position in string "aline".
c
c#    -       * ....... First alternare RETURN in the event that End-of-File is
c#                      encountered on attempt to read an uncontinued record
c#                      from unit NUNIT.
c
c#    -       * ....... Second alternare RETURN in the event of an error on
c#                      reading an input line.  This could be End-of-File on
c#                      trying to read a continued line, or lack of space in
c#                      aline for all the data strings.
c
c ------------------------------------------------------------
c#  PURPOSE:
c
c#  Reads record(s) from unit NUNIT and returns the "input line".
c#  Define the following terms:
c
c#    input record
c#        All input up to a carriage return.
c
c#    data string
c#        The string beginning at the first non-blank on an input record and
c#        extending to the final non-blank before any continuation character
c#        or comment character (which ever is first).
c
c#    continuation directive
c#        A continuation character that is the final non-blank before any
c#        comment (if present).
c
c#    continuation set
c#        A set of consecutive records, all but the last of which have
c#        continuation directives.  The "data strings" from these lines
c#        are what make up the final "data line".
c
c#    data line
c#        The concatenation of the data strings from all records in a
c#        continuation set.  (Note: a blank space is inserted between data
c#        strings unless this blank would be adjacent to a comma in the
c#        resulting string.
c
c ------------------------------------------------------------
c
      character aline*(*), arec*(*), cont*1, comnt*1, comma*1
      save cont, comnt, comma
```

```
      data cont:/'>'/, comnt /'#'/, comma /','/
c
c ---------------------------------------------------------------
c#                Read past blank or comment-only lines to
c#                a "data" record or End-of-File.
c ---------------------------------------------------------------
c
      irdata = 0
  100 continue
      read( nunit, '(a)', end=910 ) arec
c
c#      -- Locate the "data string" on this record and look for continuation.
      call getds( arec, locds, lends, icont )
c
      if (lends .eq. 0) then
        aline = ' '
c#        -- No data string on the record.  What we do depends on whether we
c#           are ignoring blank (or comment-only) lines:
        if (igblnk .eq. 1) then
c#          -- If no continuation directive is present, ignore the record
c#             and go on to read the next record.
          if (icont .eq. 0) go to 100
        else
c#          -- If no continuation directive is present, take the record
c#             as a line of "null" data.
          if (icont .eq. 0) return
        endif
      else
        if (icont .eq. 0) then
c#          -- Data string present and no continuation indicated.
          aline = arec( locds : locds+lends-1 )
          irdata = lends
          return
        endif
      endif
c
c#    -- The record just read is to be continued on the next record.
      more = 1
c
c#    -- We will not need to inset a space, as a delimiter, at the current
c#    -- end of "aline" before appending the data string to "aline".
      needlm = 0
c
c ---------------------------------------------------------------
c#           We have just read in a record that is one of a set of
c#           "continued" records used to input a single "data line".
c ---------------------------------------------------------------
c
      maxlen = len( aline )
  200 continue
      if (lends .ne. 0) then
c#      -- Append the data string to the data line we are accumulating.
        if ( needlm .eq. 1  .and.  arec(locds:locds) .ne. comma ) then
          if ( irdata + 1 .gt. maxlen ) return 2
          irdata = irdata + 1
          aline(irdata : irdata) = ' '
        endif
c
        if ( irdata + lends .gt. maxlen ) return 2
        aline(irdata+1 : irdata+lends) = arec(locds : locds+lends-1)
```

```
        irdata = irdata + lends
      endif
c
      if (more .eq. 0) return
c
      if (lends .ne. 0) then
        if ( aline(irdata:irdata) .eq. comma ) then
          needlm = 0
        else
          needlm = 1
        endif
      endif
c
c#      -- Read the next record from disk.
      read( nunit, '(a)', end=990 ) arec
c
      call getds( arec, locds, lends, icont )
c
      if (icont .eq. 0) more = 0
      go to 200
c
c -------- RETURN on End-of-File -------------------------
  910 continue
      return 1
c
c -------- RETURN on Error ------- -------------------------
  990 continue
      return 2
c
      end
      subroutine getds( arec, locds, lends, icont )
c
c ---------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     arec .... The given character string containing a single input
c#                     record, i.e., input up to a carriage return.  This may
c#                     contain a continuation dierctive following the "data
c#                     string" and also a trailing comment.
c
c#    out    locds ... The location in the string "arec" where the "data string"
c#                     begins.  This is undefined if no data string is present.
c
c#    out    lends ... The string length of the "data string" in string "arec".
c#                     This is set to zero if nno data string is present.
c
c#    out    icont ... =0 , if the string "arec" does not have a continuation
c#                          directive.
c#                     =1 , if the string "arec" does contain a continuation
c#                          directive indicating that the user's "input line"
c#                          will be continued on the next record.
c
c ---------------------------------------------------------------
c#  PURPOSE:
c
c#  Locates the data string in the given string, "arec", containing an input
c#  record.  If a continuation directive is present in the given string, this is
c#  is noted by setting the returned flag, "icont", to 1.
c
c#  The "data string" is the section of the record remaining after
```

```
c#  conceptually:
c
c#    1)  Removing trailing comments, i.e. finding the first occurance of
c#        a "comment character" in the given string and removing it and all
c#        characters that follow it.
c
c#    2)  If (after the previous step) the last character in the string is
c#        a "continuation character", it is removed.
c
c#    3)  Removing leading and trailing blanks from the string that remains
c#        after the previous two steps.
c
c#  While the above helps to envision the method for determining the "data
c#  string", in actuality we locate the beginning location of the data string
c#  in the given string and dtermine the character length of the data string.
c#  The location and length are returned to the calling routines.
c
c ------------------------------------------------------------------
      character arec*(*), cont*1, comnt*1
      save cont, comnt
      data cont /'>'/, comnt /'#'/
c
c ------------------------------------------------------------------
c
      lends = 0
      icont = 0
c
      locds = ileft( arec )
      if (locds .eq. 0) return
c
      lcom = index( arec, comnt )
      if (lcom .eq. locds) return
c
      if (lcom .eq. 0) then
        lr = iright( arec )
      else
        lr = iright( arec(1: lcom-1) )
      endif
c
      if (arec(lr:lr) .eq. cont) then
        icont = 1
        if (lr .eq. locds) return
c
        lr = iright( arec(1: lr-1) )
      endif
c
      lends = lr - locds + 1
      return
c
      end
      subroutine getkns( aline, irdata, tkns, lentk, ntk, * )
c
c ------------------------------------------------------------------
c#  ARGUMENTS:
c
c#    in     aline ... The given character string containing the input line.
c#                     If this string is blank (input IRDATA = 0) then all
c#                     NTK returned tokens will be blank with zero length.
c
c#    in     irdata .. The input line in string "aline" extends to this
```

```
c#                      character position.
c
c#    out     tkns .... The character array of tokens.  The character lengths
c#                      of these array elements limit the lengths of input
c#                      tokens; tokens are correctly parsed but then truncated
c#                      to this length.
c
c#    out     lentk ... An integer array with the string lengths of the tokens
c#                      (after any necessary truncation).
c
c#    in      ntk ..... The number of tokens to be parsed from the line.
c
c#    -       * ....... Alternate RETURN in the event that a string token
c#                      initiated by an apostrophy had no trailing apostrophy, or
c#                      the next character after the trailing apostrophy was not
c#                      a delimiter (or end-of-line).
c#                      NOTE: The arguments that had been interpreted up to the
c#                            point that the error occured are returned; the
c#                            rest will be blank.
c
c -------------------------------------------------------------
c#  PURPOSE:
c
c#  Parses an "input line" contained in a given string to find the first NTK
c#  tokens on the line.
c
c#  Tokens are delimited by beginning-of-line, commas, spaces, tabs, or
c#  end-of-line.  Tabs are equivalent to (single) spaces.
c
c#  A single asterisk (*) input token is recognized as a "null" token, for
c#  which the retuned string is blank and returned length is zero.  Some users
c#  may find entry of the null token preferable to that of a comma with no
c#  preceeding token, or two adjacent apostrophies (see the next
c#  paragraph).
c
c#  If a token is enclosed within apostrophies then it may contain commas,
c#  spaces, or tabs.  The returned length of such a token is the number
c#  number of character positions between the pair of apostrophies.
c#  Note that, for this length calculation, tabs are a single character.
c#  Contigous alphanumeric strings (no embedded delimiters) do not need to be
c#  enclosed within apostrophies.
c
c#  There is no provision for including apostrophies (') or the comment
c#  character (the hash symbol, #; see routine GETDS) within a token,
c#  i.e., they may not be "escaped".  Strings enclosed within apostrophies
c#  may contain asterisks; these will not be misinterpreted as
c#  null tokens.
c
c#  A line with only "null" data will minimally contain (before an optional
c#  comment) either a null character, a comma or two adjacent apostrophies.
c
c#  Tokens are placed, left justified, into the character array TKNS.
c#  The tokens may be truncated in order to fit into the given TKNS array.
c
c#  The lengths of the returned tokens are placed in the LENTK array.
c
c -------------------------------------------------------------
c
c
c -------------------------------------------------------------
```

```
c#       Arrays for the returned tokens and their correcponding
c#       character lengths.
c ------------------------------------------------------------------
c
      character tkns(*)*(*)
      dimension lentk(*)
c
c ------------------------------------------------------------------
c#       Strings for the input line, the "curent" character on the
c#       line and delimiter characters (space, tab, and comma).
c ------------------------------------------------------------------
c
      character aline*(*), c*1, tab*1, space*1, comma*1, apos*1, null*1
c
      save  space      , comma      , apos       , null
      data  space /' '/, comma /','/, apos /''''/, null /'*'/
c
      tab = char(9)
c
      maxlen = len( tkns(1) )
c
c ------------------------------------------------------------------
c#                Initialize ntk tokens to blanks.
c ------------------------------------------------------------------
c
      do 50 i=1,ntk
        tkns(i) = ' '
        lentk(i) = 0
   50 continue
c
      if ( irdata .eq. 0 ) return
c
c ------------------------------------------------------------------
c#         Something is on the line (before any comment).
c#         Parse the tokens (up to NTKN of them).
c ------------------------------------------------------------------
c
      is = 0
      itkn = 1
c
  200 continue
c#      -- Move along the line looking for a token or a comma (null token).
      is = is + 1
      c = aline(is:is)
c
c     ---------
      if ( c.eq.tab .or. c.eq.space ) then
c#        -- Move to next character; to continue looking for token.
        go to 200
c     ---------
      else if ( c.eq.comma ) then
c#        -- No token found before this comma indicates null token.
        if ( itkn .eq. ntkn ) return
        itkn = itkn + 1
c
        if ( is .eq. irdata ) return
c#        -- Move to next character; look for next token.
        go to 200
c     ---------
      else
```

```
  300   continue
c#        -- We are at the left-most character of a non-null token.
        il = is
c
c       - - - - -
        if ( c .eq. apos ) then
c#        -- We look for the trailing apostrophy for this "quoted string".
          if ( is .eq. irdata ) return 1
  450     continue
          is = is + 1
          c = aline(is:is)
          if ( c .eq. apos ) then
            length = is - il - 1
            if ( length .gt. maxlen ) then
              length = maxlen
            endif
            lentk( itkn ) = length
            if ( length .gt. 0 ) tkns( itkn ) = aline( il+1: il+length )
            if ( is .eq. irdata ) return
            is = is + 1
            c = aline(is:is)
            if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) )
     *        return 1
          else
            if ( is .eq. irdata ) return 1
            go to 450
          endif
c       - - - - -
        else
  400     continue
          if ( is .eq. irdata ) then
c#          -- The token extends to the end of the input data.
            if ( aline(il:is) .eq. null ) return
            tkns( itkn ) = aline(il: is)
c#            -- The input token length may exceed the length of the variable.
            length = is - il + 1
            if ( length .le. maxlen ) then
              lentk( itkn ) = length
            else
              lentk( itkn ) = maxlen
            endif
            return
          endif
c
c#          -- Move along the line looking for the end of the token.
          is = is + 1
          c = aline(is:is)
c
          if ( .not. (c.eq.space .or. c.eq.tab .or. c.eq.comma) ) then
            if ( c .eq. apos ) return 1
c#            -- Move to next character; continue looking for end of token.
            go to 400
          endif
c
c#          -- The previous character was the right-most character of the token
          if ( aline(il: is-1) .ne. null ) then
            tkns( itkn ) = aline(il: is-1)
c#            -- The input token length may exceed the length of the variable.
            length = is - il
            if ( length .le. maxlen ) then
```

```
              lentk( itkn ) = length
            else
              lentk( itkn ) = maxlen
            endif
          endif
c       - - - - -
        endif
c
        if ( itkn .eq. ntkn ) return
c#        -- Find the next location at which we should begin looking for the
c#        -- next token, i.e., after a following comma or at the next token.
        itkn = itkn + 1
        if ( is .eq. irdata ) return
        if ( c .eq. comma ) then
c#          -- Go back and look for a token following this comma.
          go to 200
        else
c#          -- Curent character was a space or a tab.
c#          -- Look for a comma, the next token, or end of line.
  500     continue
          is = is + 1
          c = aline(is:is)
          if ( c .eq. comma ) then
            if ( is .eq. irdata ) return
            go to 200
          else if ( c.eq.space .or. c.eq.tab ) then
            if ( is .eq. irdata ) return
            go to 500
          else
            go to 300
          endif
        endif
c     ---------
      endif
c
      end
      subroutine iintrp( tkn, lentk, i, * )
c
c#  Read a integer number, I, from a character string, TKN, of length LENTK.
c#  (Note that blanks in the strings are ignored. Leading and/or trailing
c#   blanks are ignored, which is OK, but interior blanks are squeezed out,
c#   which is not OK; try to fix this sometime.)
c
      character tkn*(*)
c
      if ( lentk .eq. 0 ) return
c
      read( tkn(1:lentk), '(bn,i80)', err=990 ) i
c
      return
c
c -----------
  990 continue
      return 1
c
      end
      subroutine rintrp( tkn, lentk, r, * )
c
c#  Read a real number, R, from a character string, TKN, of length LENTK.
c#  (Note that blanks in the strings are ignored. Leading and/or trailing
```

```
c#   blanks are ignored, which is OK, but interior blanks are squeezed out,
c#   which is not OK; try to fix this sometime.)
c
      character tkn*(*)
c
      if ( lentk .eq. 0 ) return
c
      read( tkn(1:lentk), '(bn,f80.0)', err=990 ) r
c
      return
c
c -----------
  990 continue
      return 1
c
      end
      function ileft( string )
c
c  WRITTEN by:  D. T. Rickhoff
c
c ---------------------------------------------
c  Find the position of the leftmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c                                                                      ---
c ---------------------------------------------
c
      character*(*) string
      character*1 tab
c
      tab = char(9)
c
c --------------------------------
c  Find the location of the first nonblank character.
c
      ls = len( string )
c
      do 100 i=1,ls
        if(      string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          ileft = i
          return
        endif
  100 continue
c
c --------------------------------
c  String was blank.  Set ileft to zero (as a flag).
      ileft = 0
c
      return
      end
      function iright( string )
c
c  WRITTEN by:  D. T. Rickhoff
c
c ---------------------------------------------
c  Find the position of the rightmost nonblank character
c  in a string of length ls.
c  NOTE: Tab characters, char(9), are considered equivalent to only one blank.
c                                                                      ---
```

```
c ------------------------------------------------
c
      character*(*) string
      character*1 tab
c
      tab = char(9)
c
c --------------------------------
c  Find the location of the last nonblank character.
c
      ls = len( string )
c
      do 100 i=ls,1,-1
        if(      string(i:i) .ne. ' '
     *     .and. string(i:i) .ne. tab ) then
          iright = i
          return
        endif
  100 continue
c
c --------------------------------
c  String was blank.  Set iright to zero (as a flag).
      iright = 0
c
      return
      end
```

A3.4.1 FLAT STIFFENED PANEL MODULE # 1
POST PROCESSING DRIVER

```
$!
$!                  DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO PERFORM
$!  POST PROCESSING FOR MODELS GENERATED WITH
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.2   6-28-91
$!  AUTHOR: WAYNE M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#########################################################################
$START:
$  W := WRITE SYS$OUTPUT
$W ""
$W "         Advanced Composites Structural Concept"
$W "             and Materials Technologies"
$W "         Automated DIAL finite element analysis"
$W "               POST PROCESSING MODULE"
$W ""
$W "                  VAX VERSION 1.2"
$W "                    JUNE 1991"
$W ""
$W ""
$W ""
$INQUIRE DB "What is the name of the DIAL data base file?"
$ASSIGN 'DB' FILOO2
$W ""
$!#########################################################################
$! THE FOLLOWING COMMAND MAY NEED TO BE MODIFIED TO RUN
$! THE DIAL SCOPE PROCESSOR IF THE EQUIVALENT NAME IS NOT
$! ALREADY SETUP IN THE USER LOGIN.COM
$!  EXAMPLE: CHANGE TO $RUN DIAL$DIR:SCOPE
$!#########################################################################
$SCOPE
START -1
##################################################################
#
#  THIS SCOPE PROCEDURE IS FOR MODULE 1
#  STRINGER STIFFENED FLAT PANEL
#
##################################################################
SET SYNTAX ON
LET &IF1=0
DATA &NB B
DATA &IYES Y
;1000 FORMAT'(1X,'GIVE A SHORT NAME FOR THIS ANALYSIS')
WRITE 6 ;1000
QUERY !NAME "  PRECEDE MULTIPLE WORDS WITH APOSTROPHE >> "
QUERY !DEV "WHAT TYPE OF TERMINAL?(TEK,SEL,RETR,VT24) >> "
DEFINE SHCD VERS
QUERY &NBW  "COLOR OR BLACK & WHITE? (C/B) >> "
IF &NBW-&NB ;91,;90,;91
;90 DSPECT SET 0 WHITE
DSPECT SET 1 BLACK
;91 CONTINUE
QUERY &NHC "WILL YOU WANT TO PLOT HARD COPIES?(Y/N) >> "
IF &NHC-&IYES ;30,;10,;30
;10 QUERY !HCD "INPUT HARD COPY DEVICE(VERS,QMS) >> "
DEFINE SHCD !HCD
```

```
;1000 FORMAT'(/1X,'WHAT OUTPUT OPTION WOULD YOU LIKE?')
;1001 FORMAT'(1X,'  1) SCREEN PLOTS ONLY, NO HARD COPIES')
;1002 FORMAT'(1X,'  2) ONLY HARD COPY PLOTS')
;1003 FORMAT'(1X,'  3) SCREEN PLOTS WITH HARD COPY PLOTS OPTIONAL')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
QUERY &N1 "OPTION NUMBER? >> "
IF &N1-2 ;30,;40,;50
;30 DEVICE OPEN !DEV
LET &NPL=0
GOTO ;60
;40 DEVICE OPEN !HCD
LET &NPL=0
GOTO ;60
;50 DEVICE OPEN !DEV
LET &NPL=1
META OPEN SEL
;60 CONTINUE
TITLE ON
TITLE1 !NAME
#
# SUBROUTINE FOR CREATING METAFILE PLOTS ON DEMAND
#
SUB PLOT &NPL
DATA &IYES Y
IF &NPL ;100 ;100 ;10
;10 QUERY &NX "DO YOU WISH A HARD COPY PLOT OF THE LAST PLOT?(Y/N) >> "
IF &NX-&IYES ;100 ;20 ;100
;20 METASAVE
;100 CONTINUE
RETURN
END
#
# SUBROUTINE FOR SETTING VIEWPOINT
#
SUB VIEW &X,&Y,&Z
DATA &IYES Y
;1000 FORMAT'(/1X,'CURRENT VIEWPOINT IS: X ='F5.1,'  Y ='F5.1,'  Z ='F5.1,/)
;1001 FORMAT'(/1X,'VIEWING DIRECTION IS FROM THE VIEWPOINT COORDINATES TO'/)
;1002 FORMAT'(1X,'THE ORIGIN, ACTUAL DISTANCE BETWEEN THE POINTS IS NOT USED'/)
WRITE 6 ;1000 &X,&Y,&Z
;10 QUERY &NX "DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT?(Y/N) >>"
IF &NX-&IYES ;100 ;20 ;100
;20 WRITE 6 ;1001:WRITE 6 ;1002
QUERY &X "INPUT NEW VIEWPOINT X COORD.(INCLUDE DECIMAL POINT) >>  "
QUERY &Y "INPUT NEW VIEWPOINT Y COORD.(INCLUDE DECIMAL POINT) >>  "
QUERY &Z "INPUT NEW VIEWPOINT Z COORD.(INCLUDE DECIMAL POINT) >>  "
WRITE 6 ;1000 &X,&Y,&Z
;100 RETURN
END
#
#  SUBROUTINE FOR PRINTING LOAD SELECTION MENU
#
SUB LMNU
;2000 FORMAT'(//1X,'SELECT LOAD FACTOR LEVEL TO BE PLOTTED:')
;2001 FORMAT'(15X,'LINEAR SOLUTION          = 0')
;2002 FORMAT'(15X,'MAX LOAD STEP ATTAINED   = 1')
;2003 FORMAT'(15X,'INTERMEDIATE LOAD FACTOR = 2'/)
WRITE 6 ;2000:WRITE 6 ;2001:WRITE 6 ;2002:WRITE 6 ;2003
RETURN
```

```
STOP
#
#  SUBROUTINE FOR SELECTING NONLINEAR LOAD STEP
#
sub STAT &pain &iseq,&paot
#-----------------------------------------------------------------------
#     Return the Solution Sequence number and Load Factor thta are
#     closest to a requested Load Factor.
#
#      in   &pain  Requested Load Factor
#
#      out  &iseq  Solution sequence number for output Load Factor
#      out  &paot  Solution Load Factor closest to Requested Load Factor
#-----------------------------------------------------------------------
#                  Initialize Variables
      let &iseq=0
      let &paot=0.0
      let &is=0
      let &pa=0.0
      let &ismn=0
      let &pamn=0.0
#
      let &lstp = %icfl(PCT.HED.SOLV,0,0,1)
      let &istp = %icfl(BRCH.HED,0,1,2)
      let &nseq = %max(&lstp,&istp)
#
      do ;loop &is=1,&nseq
        let &ifn = %ifl(D.SV,0,&is)
        let &pa = %fafm(&ifn,1)
        if &pa-&pain 1,;mach,;find
          let &pamn = &pa
          let &ismn = &is
;loop continue
#                  Requested Load Factor is .gt. Final Solution Load Factor
        let &iseq = &is
        let &paot = &pa
        goto ;clos
;find continue
#                  Find Solution Load Factor closest to Requested Load Factor
      let &pdmn = &pain-&pamn
      let &pdmx = &pa-&pain
      if &pdmn-&pdmx 1,1,;max
        let &iseq = &ismn
        let &paot = &pamn
        goto ;clos
;max  continue
        let &iseq = &is
        let &paot = &pa
        goto ;clos
;mach continue
#                  Solution Load Factor matches Requested Load Factor
        let &iseq = &is
        let &paot = &pa
#
;clos continue
;110  format '(' Analysis State set to Solution Sequence Number ',I4)
;111  format '('                  with Solution Load Factor ',F10.3)
      write 6,;110 &iseq
      write 6,;111 &paot
#
      return
      end
```

```
#
#  BASIC GEOM PLOT SUBROUTINE
#
SUB GEO1 &NPL
DATA &IYES Y
;999 FORMAT'(/1X,'AFTER EACH PLOT IS GENERATED HIT A RETURN TO CONTINUE'/)
WRITE 6 ;999
TITLE2 'GEOMETRY PLOT
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
;2000 FORMAT'(/1X,'PICK THE PARTS TO BE DISPLAYED:')
;2001 FORMAT'(/15X,'ALL PARTS (SKIN + STIFFENERS) =  0')
;2002 FORMAT'(15X,'SKIN ONLY                      = -1')
;2003 FORMAT'(15X,'SINGLE STIFFENER  (1 THRU N)  =  ?'/)
WRITE 6 ;2000:WRITE 6 ;2001:WRITE 6 ;2002:WRITE 6 ;2003
QUERY &N1 " INPUT OPTION >> "
IF &N1 ;100 ;110 ;120
;100 MSET 20 INSERT NAME=BASE
GOTO ;200
;110 MSET 20 INSERT ALL
GOTO ;200
;120 MSET 20 INSERT MSET=&N1
;200 CONTINUE
SCREEN VIRT -.9,.9,-.9,.9
;1000 FORMAT'(/1X,'PICK ONE OF THE FOLLOWING LABEL OPTIONS:')
;1001 FORMAT'(/15X,'NO NODE OR ELEMENT NUMBERING = 0')
;1002 FORMAT'(15X,'NODE NUMBERING ONLY          = 1')
;1003 FORMAT'(15X,'ELEMENT NUMBERING ONLY       = 2')
;1004 FORMAT'(15X,'NODE AND ELEMENT NUMBERING   = 3'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003:WRITE 6 ;1004
QUERY &N1 " INPUT OPTION >> "
IF &N1-1 ;10 ;20 ;30
;10 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB:OVEND
GOTO ;60
;20 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
 OVNODE 0 0 MSET=20:OVEND
GOTO ;60
;30 IF &N1-2 ;60 ;40 ;50
;40 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
  OVELEM 0 0 MSET=20:OVEND
GOTO ;60
;50 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
  OVNODE 0 0 MSET=20:OVELEM 0 0 MSET=20:OVEND
;60 CONTINUE
MSET 20 DELETE ALL
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT SHOWING LOAD VECTORS
#
SUB GEO2 &NPL
TITLE2 'LOAD PLOT
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
```

```
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
DSET UL 0 1
VECT 0 1 0.
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GEO3 &XX &NPL
DATA &IYES Y
LEGEND OFF
TITLE2 'DEFLECTED GEOMETRY PLOT - LINEAR STRESS
DSET D 0 1
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED BUCKLED SHAPE
#
SUB GE04 &XX &NPL
DATA &IYES Y
LEGEND OFF
TITLE2 'DEFLECTED GEOMETRY PLOT - BIFURCATION BUCKLING
DSET BUCK 0 1
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;170 ;160 ;170
;160 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;180
;170 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
:OVMESH:OVEND
```

```
CALL PLOT &NPL
;180 CONTINUE
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GE05 &XX &NPL
DATA &IYES Y
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
IF &NS-1 ;10 ;20 ;30
;10 DSET D,0,1
TITLE2 'DEFLECTED GEOM PLOT - NONLINEAR ANALYSIS - LINEAR SOLUTION
GOTO ;100
;20 LET &MLS = %ICFL(BRCH.HED,0,1,2)
DSET D,0,&MLS
TITLE2 'DEFLECTED GEOM PLOT - NONLINEAR ANALYSIS - FINAL STEP
GOTO ;100
;30 CONTINUE
;1005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;1005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
DSET D,0,&ISEQ
;2000 FORMAT'('DEFLECTED GEOM - NONLINEAR ANALYSIS - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2000 &PAOT
TITLE2 !TIT2
;100 CONTINUE
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR SELECTING DEFLECTION PLOTS
#
SUB DEFL &NG &XX
DATA &IYES Y
;1000 FORMAT'(//1X,'SELECT DESIRED DEFLECTION PLOT TYPE:')
;1001 FORMAT'(/15X,'LINEAR ANALYSIS                = 0')
;1002 FORMAT'(15X,'BUCKLING ANALYSIS MODE SHAPE  = 1')
;1003 FORMAT'(15X,'NONLINEAR ANALYSIS             = 2'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
QUERY &NS " INPUT SELECTION >> "
```

```
;1004 FORMAT'(/1X,'SELECT DEFLECTION MAGNIFICATION FACTOR TO BE USED:')
;1005 FORMAT'(/15X,'LOW (.10)          = 0')
;1006 FORMAT'(15X,'MED (.20)          = 1')
;1007 FORMAT'(15X,'HIGH (.50)         = 2')
;1008 FORMAT'(15X,'USER SELECTABLE    = 3'/)
WRITE 6 ;1004:WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008
QUERY &NM " INPUT SELECTION >> "
IF &NM-1 ;10 ;20 ;30
;10 LET &XX=.1
GOTO ;100
;20 LET &XX=.2
GOTO ;100
;30 IF &NM-2 ;100 ;40 ;50
;40 LET &XX=.5
GOTO ;100
;50 QUERY &XX " INPUT MAGNIFICATION FACTOR >> "
;100 CONTINUE
IF &NS-1 ;60 ;70 ;80
;60 LET &NG=3
GOTO ;200
;70 LET &NG=4
GOTO ;200
;80 LET &NG=5
;200 CONTINUE
RETURN
END
#
# SUBROUTINE FOR PRINTING STRESS QUANTITY MENU
#
SUB MNU2
;1000 FORMAT'(/1X,'QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3')
;1001 FORMAT'(1X, '              (LINE MOMENT) M-X =  4, M-Y =  5')
;1002 FORMAT'(1X, '              (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13')
;1003 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
RETURN
END
#
#  SUBROUTINE TO PRINT LAYER STRESS MENU
#
SUB MNU3
;1000 FORMAT'(/1X,'QUANTITY MENU:(LAYER STRESS)  S-X =  1, S-Y =  2, S-XY = 4')
;1001 FORMAT'(1X, '              (LAYER STRAIN)  E-X = 11, E-Y = 12, E-XY = 14')
;1002 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN AVG LOADS AND STRAIN
#
SUB CON1 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT X R
LET &X=0
LET &Y=0
```

```
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
DLABEL 1 'N-X AVG       LINE LOAD
DLABEL 2 'N-Y AVG       LINE LOAD
DLABEL 3 'N-XY AVG      LINE LOAD
DLABEL 4 'M-X AVG      LINE MOMENT
DLABEL 5 'M-Y AVG      LINE MOMENT
DLABEL 11 'E-X AVG        STRAIN
DLABEL 12 'E-Y AVG        STRAIN
DLABEL 13 'E-XY AVG       STRAIN
;25 CONTINUE
CALL MNU2
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;30
;30 DFROMS &N3,&N3,MSH,0,NAME=BASE
CONT &N3,0,15,0,NAME=BASE
CALL PLOT &NPL
GOTO ;25
;40 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN PLY STRESS AND STRAIN
#
SUB CON2 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT X R
LET &X=0
LET &Y=0
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
```

```
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
CALL MNU3
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSHELL 1 &NLN &NLS NAME=BASE
DFROMS &N3,&N3,MSH,0,NAME=BASE
;3000 FORMAT'('SKIN STRESS QUANTITY = 'I3,', LAYER = 'I3,',  SURFACE = 'I3)
WRITE !TIT2 ;3000 &N3 &NLN &NLS
TITLE2 !TIT2
CONT &N3,0,15,0,NAME=BASE
CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN MARGIN OF SAFETY
#
SUB CON3 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT X R
LET &X=0
LET &Y=0
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN M.S. AT ELEM QUAD POINTS - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN M.S. AT ELEM QUAD POINTS - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
```

```
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN M.S. AT ELEM QUAD POINTS - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
SSHELL/MQMS=1 1 1 -1 NAME=BASE
DFROME 1 1 MSH NAME=BASE
CONT 1,0,15,0,NAME=BASE
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF AVG LOADS & STRAIN
# STIFFENERS
#
#
SUB CON4 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
DLABEL 3 'N-XY AVG     LINE LOAD
DLABEL 11 'E-X AVG       STRAIN
DLABEL 12 'E-Y AVG       STRAIN
DLABEL 13 'E-XY AVG      STRAIN
;25 CONTINUE
CALL MNU2
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;40
;30 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('STIFFENER AVG LINE LOAD OR STRAIN - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
DFROMS &N3,&N3,MSH,0,MSET=&NS
CONT &N3,0,15,0,MSET=&NS
CALL PLOT &NPL
```

```
GOTO ;25
;40 CONTINUE
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF LAYER STRESS & STRAIN
# STIFFENERS
#
SUB CON5 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
;70 CONTINUE
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
CALL MNU3
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSHELL 1 &NLN &NLS MSET=&NS
DFROMS &N3,&N3,MSH,0,MSET=&NS
;3000 FORMAT'('STIFF NO.= 'I3,', STRESS QUAN= 'I3,', LAYER= 'I3', SURF= 'I3)
WRITE !TIT2 ;3000 &NS &N3 &NLN &NLS
TITLE2 !TIT2
CONT &N3,0,15,0,MSET=&NS
CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
```

```
END
#
#  SUBROUTINE FOR MAKING CONTOUR PLOTS OF MARGIN OF SAFETY
#  STIFFENERS
#
SUB CON6 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('AML MARGIN OF SAFETY - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
SSHELL/MQMS=1 1 1 -1 MSET=&NS
DFROME 1 1 MSH MSET=&NS
CONT 1,0,15,0,MSET=&NS
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR GENERATING MAX-MIN SUMMARY TABLES
#  FOR THE SKIN PANEL AND STIFFENERS AVG LAMINATE LOADS & STRAINS
#
SUB SUM1 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
```

```
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
PRINT OFF,0,LIST
SLABEL 1 'N-X          LINE LOAD
SLABEL 2 'N-Y          LINE LOAD
SLABEL 3 'N-XY         LINE LOAD
SLABEL 4 'M-X          MOMENT LOAD
SLABEL 5 'M-Y          MOMENT LOAD
SLABEL 11 'E-X AVG       STRAIN
SLABEL 12 'E-Y AVG       STRAIN
SLABEL 13 'E-XY AVG      STRAIN
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
DLABEL 3 'N-XY AVG     LINE LOAD
DLABEL 11 'E-X AVG       STRAIN
DLABEL 12 'E-Y AVG       STRAIN
DLABEL 13 'E-XY AVG      STRAIN
SPRINT 8,1,2,3,4,5,11,12,13,0,NAME=BASE
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('AVG LINE LOAD OR STRAIN AT INTEG PTS. - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
SPRINT 8,1,2,3,4,5,11,12,13,0,MSET=&NS
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
RETURN
END
#
# SUBROUTINE FOR MAKING SUMMARY TABLES OF PLY STRESS AND STRAIN
#
SUB SUM2 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
```

```
TITLE2 'SKIN PANEL PLY STRESS SUMMARY - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN PANEL PLY STRESS SUMMARY - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN PANEL PLY STRESS SUMMARY - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
PRINT OFF,0,LIST
SLABEL 1 'S-X PLY         STRESS
SLABEL 2 'S-Y PLY         STRESS
SLABEL 4 'S-XY PLY        STRESS
SLABEL 11 'E-X PLY        STRAIN
SLABEL 12 'E-Y PLY        STRAIN
SLABEL 14 'E-XY PLY       STRAIN
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE GIVEN IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
PSHELL 6,1,2,4,11,12,14,1,0,0,1,0,1,NAME=BASE
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('PLY STRESS SUMMARY AT ELEM QUAD POINTS - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
;60 TITLE2 'STIFFENER
PSHELL 6,1,2,4,11,12,14,1,0,0,1,0,1,MSET=&NS
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING SUMMARY TABLE OF MARGINS OF SAFETY
#
SUB SUM3 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
CALL LMNU
```

```
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN MARGIN OF SAFETY SUMMARY - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN MARGIN OF SAFETY SUMMARY - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN MARGIN OF SAFETY SUMMARY - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
FAIL ON 11
PRINT OFF,0,LIST
PSHELL 2,11,12,1,0,0,1,1,1,NAME=BASE
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('AML MARGIN OF SAFETY AT ELEM QUAD POINTS - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
PSHELL 2,11,12,1,0,0,1,1,1,MSET=&NS
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR TRANSFORMING METAPLOT FILES TO DEVICE PLOT FILES
#
SUB HC &NPL
IF &NPL ;100 ;100 ;10
;10 DEVICE OPEN <SHCD>
;1000 FORMAT'(/1X,'SAVED PLOTS ARE NOW BEING TRANSFORMED TO PLOT FILES')
;1001 FORMAT'(1X,'PLEASE WAIT UNTIL COMPLETION IS INDICATED.')
WRITE 6 ;1000
WRITE 6 ;1001
METAPLOT
;1000 FORMAT'(/1X,'PLOT FILE TRANSFORMATION COMPLETED'//)
WRITE 6 ;1000
;100 CONTINUE
RETURN
END
#
#
#  THE FOLLOWING PROCEDURE DISPLAYS A MENU FOR SELECTION OF
#  POST PROCESSING FUNCTIONS
#
SUB MENU
;1000 FORMAT'(//1X,'MENU FOR PLOTS AND SUMMARY TABLES:'/)
;1001 FORMAT'(10X,'1) BASIC GEOMETRY PLOT')
;1002 FORMAT'(10X,'2) GEOMETRY PLOT WITH LOAD VECTORS SHOWN')
;1003 FORMAT'(10X,'3) DEFLECTED GEOMETRY PLOT')
;1005 FORMAT'(10X,'4) CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS')
```

```
;1006 FORMAT'(10X,'5) CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN')
;1007 FORMAT'(10X,'6) CONTOUR PLOT OF SKIN MARGIN OF SAFETY')
;1008 FORMAT'(10X,'7) CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR STRAINS')
;1009 FORMAT'(10X,'8) CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN')
;1010 FORMAT'(10X,'9) CONTOUR PLOT OF STIFF. MARGIN OF SAFETY')
;1011 FORMAT'(9X,'10) SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS')
;1012 FORMAT'(9X,'11) SUMMARY TABLE OF PLY STRESSES AND STRAINS')
;1013 FORMAT'(9X,'12) SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY'/)
;1014 FORMAT'(9X,'99) TERMINATE POST PROCESSING SESSION'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008:WRITE 6 ;1009:WRITE 6 ;1010:WRITE 6 ;1011
WRITE 6 ;1012:WRITE 6 ;1013:WRITE 6 ;1014
RETURN
END
#
#MENU SELECTION SUBROUTINE
#
SUB SEL2 &NPL &IMN
IF &IMN-2 ;10 ;20 ;30
;10 CALL GE01 &NPL
GOTO ;500
;20 CALL GE02 &NPL
GOTO ;500
;30 IF &IMN-4 ;40 ;50 ;60
;40 CALL DEFL &NG &XX
IF &NG-4 ;41 ;42 ;43
;41 CALL GE03 &XX &NPL
GOTO ;500
;42 CALL GE04 &XX &NPL
GOTO ;500
;43 CALL GE05 &XX &NPL
GOTO ;500
;50 CALL CON1 &NPL
GOTO ;500
;60 IF &IMN-6 ;70 ;80 ;90
;70 CALL CON2 &NPL
GOTO ;500
;80 CALL CON3 &NPL
GOTO ;500
;90 IF &IMN-8 ;100 ;500 ;500
;100 CALL CON4 &NPL
;500 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MENU SELECTION AND REDIRECTION
#
SUB SEL1 &IF1 &NPL
;5 CONTINUE
CALL MENU
QUERY &IMN "INPUT NUMBER OF MENU ITEM DESIRED >> "
CALL SEL2 &NPL &IMN
IF &IMN-99 ;10 ;200 ;10
;10 IF &IMN-8 ;5 ;120 ;120
;120 IF &IMN-9 ;130 ;140 ;150
;130 CALL CON5 &NPL
GOTO ;5
;140 CALL CON6 &NPL
GOTO ;5
;150 IF &IMN-11 ;160 ;170 ;180
;160 CALL SUM1 &IF1
```

```
GOTO ;5
;170 CALL SUM2 &IF1
GOTO ;5
;180 CALL SUM3 &IF1
GOTO ;5
;200 CONTINUE
RETURN
END
#
# MAIN PROGRAM SUBROUTINE DRIVER
#
CALL SEL1 &IF1 &NPL
CALL HC &NPL
CLOSE 8
;1000 FORMAT'(///1X,'THIS COMPLETES THE POST PROCESSING SESSION.'/)
;1001 FORMAT'(1X,'ANY HARD COPY PLOTS GENERATED HERE WILL RESIDE ON')
;1002 FORMAT'(1X,'FILES IN THE WORKING DIRECTORY, USE NORMAL SUBMITTAL')
;1003 FORMAT'(1X,'PROCEDURES TO ACTUALLY PRINT THEM.'//)
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
STOP
$END:
$!     End of PPFLAT.COM
```

A3.4.2 CURVED STIFFENED PANEL MODULE # 2 POST PROCESSING DRIVER

```
$!                                                                    ###########
$!                    DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO PERFORM
$!  POST PROCESSING FOR MODELS GENERATED WITH
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.2   6-28-91
$!  AUTHOR: WAYNE M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#########################################################################
$START:
$  W := WRITE SYS$OUTPUT
$W ""
$W "         Advanced Composites Structural Concept"
$W "             and Materials Technologies"
$W "         Automated DIAL finite element analysis"
$W "               POST PROCESSING MODULE"
$W ""
$W "                   VAX VERSION 1.2"
$W "                     JUNE 1991"
$W ""
$W ""
$W ""
$INQUIRE DB "What is the name of the DIAL data base file?"
$ASSIGN 'DB' FIL002
$W ""
$!#########################################################################
$! THE FOLLOWING COMMAND MAY NEED TO BE MODIFIED TO RUN
$! THE DIAL SCOPE PROCESSOR IF THE EQUIVALENT NAME IS NOT
$! ALREADY SETUP IN THE USER LOGIN.COM
$!  EXAMPLE: CHANGE TO $RUN DIAL$DIR:SCOPE
$!#########################################################################
$SCOPE
START -1
#################################################################
#
#  THIS SCOPE PROCEDURE IS FOR THE CURVED,
#  STRINGER STIFFENED PANEL
#
#################################################################
SET SYNTAX ON
LET &IF1=0
DATA &NB B
DATA &IYES Y
;1000 FORMAT'(1X,'GIVE A SHORT NAME FOR THIS ANALYSIS')
WRITE 6 ;1000
QUERY !NAME "  PRECEDE MULTIPLE WORDS WITH APOSTROPHE >> "
QUERY !DEV "WHAT TYPE OF TERMINAL?(TEK,SEL,RETR,VT24) >> "
DEFINE SHCD VERS
QUERY &NBW  "COLOR OR BLACK & WHITE? (C/B) >> "
IF &NBW-&NB ;91,;90,;91
;90 DSPECT SET 0 WHITE
DSPECT SET 1 BLACK
;91 CONTINUE
QUERY &NHC "WILL YOU WANT TO PLOT HARD COPIES?(Y/N) >> "
IF &NHC-&IYES ;30,;10,;30
;10 QUERY !HCD "INPUT HARD COPY DEVICE(VERS,QMS) >> "
DEFINE SHCD !HCD
```

```
;1000 FORMAT'(/1X,'WHAT OUTPUT OPTION WOULD YOU LIKE?')
;1001 FORMAT'(1X,'  1) SCREEN PLOTS ONLY, NO HARD COPIES')
;1002 FORMAT'(1X,'  2) ONLY HARD COPY PLOTS')
;1003 FORMAT'(1X,'  3) SCREEN PLOTS WITH HARD COPY PLOTS OPTIONAL')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
QUERY &N1 "OPTION NUMBER? >> "
IF &N1-2 ;30,;40,;50
;30 DEVICE OPEN !DEV
LET &NPL=0
GOTO ;60
;40 DEVICE OPEN !HCD
LET &NPL=0
GOTO ;60
;50 DEVICE OPEN !DEV
LET &NPL=1
META OPEN SEL
;60 CONTINUE
TITLE ON
TITLE1 !NAME
#
# SUBROUTINE FOR CREATING METAFILE PLOTS ON DEMAND
#
SUB PLOT &NPL
DATA &IYES Y
IF &NPL ;100 ;100 ;10
;10 QUERY &NX "DO YOU WISH A HARD COPY PLOT OF THE LAST PLOT?(Y/N) >> "
IF &NX-&IYES ;100 ;20 ;100
;20 METASAVE
;100 CONTINUE
RETURN
END
#
# SUBROUTINE FOR SETTING VIEWPOINT
#
SUB VIEW &X,&Y,&Z
DATA &IYES Y
;1000 FORMAT'(/1X,'CURRENT VIEWPOINT IS: X ='F5.1,'  Y ='F5.1,'  Z ='F5.1,/)
;1001 FORMAT'(/1X,'VIEWING DIRECTION IS FROM THE VIEWPOINT COORDINATES TO'/)
;1002 FORMAT'(1X,'THE ORIGIN, ACTUAL DISTANCE BETWEEN THE POINTS IS NOT USED'/)
WRITE 6 ;1000 &X,&Y,&Z
;10 QUERY &NX "DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT?(Y/N) >>"
IF &NX-&IYES ;100 ;20 ;100
;20 WRITE 6 ;1001:WRITE 6 ;1002
QUERY &X "INPUT NEW VIEWPOINT X COORD.(INCLUDE DECIMAL POINT) >>  "
QUERY &Y "INPUT NEW VIEWPOINT Y COORD.(INCLUDE DECIMAL POINT) >>  "
QUERY &Z "INPUT NEW VIEWPOINT Z COORD.(INCLUDE DECIMAL POINT) >>  "
WRITE 6 ;1000 &X,&Y,&Z
;100 RETURN
END
#
#  SUBROUTINE FOR PRINTING LOAD SELECTION MENU
#
SUB LMNU
;2000 FORMAT'(//1X,'SELECT LOAD FACTOR LEVEL TO BE PLOTTED:')
;2001 FORMAT'(15X,'LINEAR SOLUTION          = 0')
;2002 FORMAT'(15X,'MAX LOAD STEP ATTAINED   = 1')
;2003 FORMAT'(15X,'INTERMEDIATE LOAD FACTOR = 2'/)
WRITE 6 ;2000:WRITE 6 ;2001:WRITE 6 ;2002:WRITE 6 ;2003
RETURN
```

```
STOP
#
#  SUBROUTINE FOR SELECTING NONLINEAR LOAD STEP
#
sub STAT &pain &iseq,&paot
#-------------------------------------------------------------------------
#     Return the Solution Sequence number and Load Factor thta are
#     closest to a requested Load Factor.
#
#      in   &pain  Requested Load Factor
#
#      out  &iseq  Solution sequence number for output Load Factor
#      out  &paot  Solution Load Factor closest to Requested Load Factor
#-------------------------------------------------------------------------
#                  Initialize Variables
      let &iseq=0
      let &paot=0.0
      let &is=0
      let &pa=0.0
      let &ismn=0
      let &pamn=0.0
#
      let &lstp = %icfl(PCT.HED.SOLV,0,0,1)
      let &istp = %icfl(BRCH.HED,0,1,2)
      let &nseq = %max(&lstp,&istp)
#
      do ;loop &is=1,&nseq
        let &ifn = %ifl(D.SV,0,&is)
        let &pa = %fafm(&ifn,1)
        if &pa-&pain 1,;mach,;find
          let &pamn = &pa
          let &ismn = &is
;loop continue
#                  Requested Load Factor is .gt. Final Solution Load Factor
        let &iseq = &is
        let &paot = &pa
        goto ;clos
;find continue
#                  Find Solution Load Factor closest to Requested Load Factor
      let &pdmn = &pain-&pamn
      let &pdmx = &pa-&pain
      if &pdmn-&pdmx 1,1,;max
        let &iseq = &ismn
        let &paot = &pamn
        goto ;clos
;max  continue
        let &iseq = &is
        let &paot = &pa
        goto ;clos
;mach continue
#                  Solution Load Factor matches Requested Load Factor
        let &iseq = &is
        let &paot = &pa
#
;clos continue
;110  format '(' Analysis State set to Solution Sequence Number ',I4)
;111  format '('                    with Solution Load Factor ',F10.3)
      write 6,;110 &iseq
      write 6,;111 &paot
#
      return
      end
```

```
#
#  BASIC GEOM PLOT SUBROUTINE
#
SUB GE01 &NPL
DATA &IYES Y
;999 FORMAT'(/1X,'AFTER EACH PLOT IS GENERATED HIT A RETURN TO CONTINUE'/)
WRITE 6 ;999
TITLE2 'GEOMETRY PLOT
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
;2000 FORMAT'(/1X,'PICK THE PARTS TO BE DISPLAYED:')
;2001 FORMAT'(/15X,'ALL PARTS (SKIN + STIFFENERS) =  0')
;2002 FORMAT'(15X,'SKIN ONLY                      = -1')
;2003 FORMAT'(15X,'SINGLE STIFFENER  (1 THRU N)  =  ?'/)
WRITE 6 ;2000:WRITE 6 ;2001:WRITE 6 ;2002:WRITE 6 ;2003
QUERY &N1 " INPUT OPTION >> "
IF &N1 ;100 ;110 ;120
;100 MSET 20 INSERT NAME=BASE
GOTO ;200
;110 MSET 20 INSERT ALL
GOTO ;200
;120 MSET 20 INSERT MSET=&N1
;200 CONTINUE
SCREEN VIRT -.9,.9,-.9,.9
;1000 FORMAT'(/1X,'PICK ONE OF THE FOLLOWING LABEL OPTIONS:')
;1001 FORMAT'(/15X,'NO NODE OR ELEMENT NUMBERING = 0')
;1002 FORMAT'(15X,'NODE NUMBERING ONLY          = 1')
;1003 FORMAT'(15X,'ELEMENT NUMBERING ONLY       = 2')
;1004 FORMAT'(15X,'NODE AND ELEMENT NUMBERING   = 3'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003:WRITE 6 ;1004
QUERY &N1 " INPUT OPTION >> "
IF &N1-1 ;10 ;20 ;30
;10 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB:OVEND
GOTO ;60
;20 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
 OVNODE 0 0 MSET=20:OVEND
GOTO ;60
;30 IF &N1-2 ;60 ;40 ;50
;40 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
  OVELEM 0 0 MSET=20:OVEND
GOTO ;60
;50 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
  OVNODE 0 0 MSET=20:OVELEM 0 0 MSET=20:OVEND
;60 CONTINUE
MSET 20 DELETE ALL
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT SHOWING LOAD VECTORS
#
SUB GE02 &NPL
TITLE2 'LOAD PLOT
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
```

```
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
DSET UL 0 1
VECT 0 1 0.
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GE03 &XX &NPL
DATA &IYES Y
LEGEND OFF
TITLE2 'DEFLECTED GEOMETRY PLOT - LINEAR STRESS
DSET D 0 1
DGLOBAL
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED BUCKLED SHAPE
#
SUB GE04 &XX &NPL
DATA &IYES Y
LEGEND OFF
TITLE2 'DEFLECTED GEOMETRY PLOT - BIFURCATION BUCKLING
DSET BUCK 0 1
DGLOBAL
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;170 ;160 ;170
;160 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;180
;170 CONTINUE
```

```
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
:OVMESH:OVEND
CALL PLOT &NPL
;180 CONTINUE
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GEO5 &XX &NPL
DATA &IYES Y
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
IF &NS-1 ;10 ;20 ;30
;10 DSET D,0,1
DGLOBAL
TITLE2 'DEFLECTED GEOM PLOT - NONLINEAR ANALYSIS - LINEAR SOLUTION
GOTO ;100
;20 LET &MLS = %ICFL(BRCH.HED,0,1,2)
DSET D,0,&MLS
DGLOBAL
TITLE2 'DEFLECTED GEOM PLOT - NONLINEAR ANALYSIS - FINAL STEP
GOTO ;100
;30 CONTINUE
;1005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;1005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
DSET D,0,&ISEQ
DGLOBAL
;2000 FORMAT'('DEFLECTED GEOM - NONLINEAR ANALYSIS - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2000 &PAOT
TITLE2 !TIT2
;100 CONTINUE
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR SELECTING DEFLECTION PLOTS
#
SUB DEFL &NG &XX
DATA &IYES Y
;1000 FORMAT'(//1X,'SELECT DESIRED DEFLECTION PLOT TYPE:')
```

```
;1001 FORMAT'(/15X,'LINEAR ANALYSIS                = 0')
;1002 FORMAT'(15X,'BUCKLING ANALYSIS MODE SHAPE  = 1')
;1003 FORMAT'(15X,'NONLINEAR ANALYSIS             = 2'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
QUERY &NS " INPUT SELECTION >> "
;1004 FORMAT'(/1X,'SELECT DEFLECTION MAGNIFICATION FACTOR TO BE USED:')
;1005 FORMAT'(/15X,'LOW (.10)          = 0')
;1006 FORMAT'(15X,'MED (.20)          = 1')
;1007 FORMAT'(15X,'HIGH (.50)         = 2')
;1008 FORMAT'(15X,'USER SELECTABLE    = 3'/)
WRITE 6 ;1004:WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008
QUERY &NM " INPUT SELECTION >> "
IF &NM-1 ;10 ;20 ;30
;10 LET &XX=.1
GOTO ;100
;20 LET &XX=.2
GOTO ;100
;30 IF &NM-2 ;100 ;40 ;50
;40 LET &XX=.5
GOTO ;100
;50 QUERY &XX " INPUT MAGNIFICATION FACTOR >> "
;100 CONTINUE
IF &NS-1 ;60 ;70 ;80
;60 LET &NG=3
GOTO ;200
;70 LET &NG=4
GOTO ;200
;80 LET &NG=5
;200 CONTINUE
RETURN
END
#
# SUBROUTINE FOR PRINTING STRESS QUANTITY MENU
#
SUB MNU2
;1000 FORMAT'(/1X,'QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3')
;1001 FORMAT'(1X, '              (LINE MOMENT) M-X =  4, M-Y =  5')
;1002 FORMAT'(1X, '              (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13')
;1003 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
RETURN
END
#
#  SUBROUTINE TO PRINT LAYER STRESS MENU
#
SUB MNU3
;1000 FORMAT'(/1X,'QUANTITY MENU:(LAYER STRESS)  S-X =  1, S-Y =  2, S-XY = 4')
;1001 FORMAT'(1X, '              (LAYER STRAIN)  E-X = 11, E-Y = 12, E-XY = 14')
;1002 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN AVG LOADS AND STRAIN
#
SUB CON1 &NPL
```

```
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT X R
LET &X=0
LET &Y=0
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
DLABEL 1 'N-X AVG       LINE LOAD
DLABEL 2 'N-Y AVG       LINE LOAD
DLABEL 3 'N-XY AVG      LINE LOAD
DLABEL 4 'M-X AVG      LINE MOMENT
DLABEL 5 'M-Y AVG      LINE MOMENT
DLABEL 11 'E-X AVG        STRAIN
DLABEL 12 'E-Y AVG        STRAIN
DLABEL 13 'E-XY AVG       STRAIN
;25 CONTINUE
CALL MNU2
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;30
;30 DFROMS &N3,&N3,MSH,0,NAME=BASE
CONT &N3,0,15,0,NAME=BASE
CALL PLOT &NPL
GOTO ;25
;40 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN PLY STRESS AND STRAIN
#
SUB CON2 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT X R
LET &X=0
LET &Y=0
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
```

```
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
CALL MNU3
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSHELL 1 &NLN &NLS NAME=BASE
DFROMS &N3,&N3,MSH,0,NAME=BASE
;3000 FORMAT'('SKIN STRESS QUANTITY = 'I3,', LAYER = 'I3,',  SURFACE = 'I3)
WRITE !TIT2 ;3000 &N3 &NLN &NLS
TITLE2 !TIT2
CONT &N3,0,15,0,NAME=BASE
CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN MARGIN OF SAFETY
#
SUB CON3 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT X R
LET &X=0
LET &Y=0
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN M.S. AT ELEM QUAD POINTS - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN M.S. AT ELEM QUAD POINTS - NONLINEAR - FINAL STEP
```

```
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN M.S. AT ELEM QUAD POINTS - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
SSHELL/MQMS=1 1 1 -1 NAME=BASE
DFROME 1 1 MSH NAME=BASE
CONT 1,0,15,0,NAME=BASE
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF AVG LOADS & STRAIN
# STIFFENERS
#
#
SUB CON4 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
DLABEL 1 'N-X AVG       LINE LOAD
DLABEL 2 'N-Y AVG       LINE LOAD
DLABEL 3 'N-XY AVG      LINE LOAD
DLABEL 11 'E-X AVG        STRAIN
DLABEL 12 'E-Y AVG        STRAIN
DLABEL 13 'E-XY AVG       STRAIN
;25 CONTINUE
CALL MNU2
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;40
;30 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('STIFFENER AVG LINE LOAD OR STRAIN - STIFF. NO.= 'I3)
```

```
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
DFROMS &N3,&N3,MSH,0,MSET=&NS
CONT &N3,0,15,0,MSET=&NS
CALL PLOT &NPL
GOTO ;25
;40 CONTINUE
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF LAYER STRESS & STRAIN
# STIFFENERS
#
SUB CON5 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
;70 CONTINUE
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
CALL MNU3
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSHELL 1 &NLN &NLS MSET=&NS
DFROMS &N3,&N3,MSH,0,MSET=&NS
;3000 FORMAT'('STIFF NO.= 'I3,', STRESS QUAN= 'I3,', LAYER= 'I3', SURF= 'I3)
WRITE !TIT2 ;3000 &NS &N3 &NLN &NLS
TITLE2 !TIT2
CONT &N3,0,15,0,MSET=&NS
CALL PLOT &NPL
```

```
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING CONTOUR PLOTS OF MARGIN OF SAFETY
#  STIFFENERS
#
SUB CON6 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;999 CONTINUE
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('AML MARGIN OF SAFETY - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
SSHELL/MQMS=1 1 1 -1 MSET=&NS
DFROME 1 1 MSH MSET=&NS
CONT 1,0,15,0,MSET=&NS
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR GENERATING MAX-MIN SUMMARY TABLES
#  FOR THE SKIN PANEL AND STIFFENERS AVG LAMINATE LOADS & STRAINS
#
SUB SUM1 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
```

```
;40 CONTINUE
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
PRINT OFF,0,LIST
SLABEL 1 'N-X          LINE LOAD
SLABEL 2 'N-Y          LINE LOAD
SLABEL 3 'N-XY         LINE LOAD
SLABEL 4 'M-X          MOMENT LOAD
SLABEL 5 'M-Y          MOMENT LOAD
SLABEL 11 'E-X AVG       STRAIN
SLABEL 12 'E-Y AVG       STRAIN
SLABEL 13 'E-XY AVG      STRAIN
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
DLABEL 3 'N-XY AVG     LINE LOAD
DLABEL 11 'E-X AVG       STRAIN
DLABEL 12 'E-Y AVG       STRAIN
DLABEL 13 'E-XY AVG      STRAIN
SPRINT 8,1,2,3,4,5,11,12,13,0,NAME=BASE
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('AVG LINE LOAD OR STRAIN AT INTEG PTS. - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
SPRINT 8,1,2,3,4,5,11,12,13,0,MSET=&NS
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
RETURN
END
#
# SUBROUTINE FOR MAKING SUMMARY TABLES OF PLY STRESS AND STRAIN
#
SUB SUM2 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
```

```
;40 CONTINUE
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN PANEL PLY STRESS SUMMARY - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN PANEL PLY STRESS SUMMARY - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN PANEL PLY STRESS SUMMARY - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
PRINT OFF,0,LIST
SLABEL 1 'S-X PLY         STRESS
SLABEL 2 'S-Y PLY         STRESS
SLABEL 4 'S-XY PLY        STRESS
SLABEL 11 'E-X PLY        STRAIN
SLABEL 12 'E-Y PLY        STRAIN
SLABEL 14 'E-XY PLY       STRAIN
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE GIVEN IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
PSHELL 6,1,2,4,11,12,14,1,0,0,1,0,1,NAME=BASE
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('PLY STRESS SUMMARY AT ELEM QUAD POINTS - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
;60 TITLE2 'STIFFENER
PSHELL 6,1,2,4,11,12,14,1,0,0,1,0,1,MSET=&NS
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING SUMMARY TABLE OF MARGINS OF SAFETY
#
SUB SUM3 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
```

```
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN MARGIN OF SAFETY SUMMARY - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN MARGIN OF SAFETY SUMMARY - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN MARGIN OF SAFETY SUMMARY - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
FAIL ON 11
PRINT OFF,0,LIST
PSHELL 2,11,12,1,0,0,1,1,1,NAME=BASE
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 QUERY &NS "INPUT STIFFENER NUMBER(ALL=10) >> "
;3000 FORMAT'('AML MARGIN OF SAFETY AT ELEM QUAD POINTS - STIFF. NO.= 'I3)
WRITE !TIT2 ;3000 &NS
TITLE2 !TIT2
PSHELL 2,11,12,1,0,0,1,1,1,MSET=&NS
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR TRANSFORMING METAPLOT FILES TO DEVICE PLOT FILES
#
SUB HC &NPL
IF &NPL ;100 ;100 ;10
;10 DEVICE OPEN <SHCD>
;1000 FORMAT'(/1X,'SAVED PLOTS ARE NOW BEING TRANSFORMED TO PLOT FILES')
;1001 FORMAT'(1X,'PLEASE WAIT UNTIL COMPLETION IS INDICATED.')
WRITE 6 ;1000
WRITE 6 ;1001
METAPLOT
;1000 FORMAT'(/1X,'PLOT FILE TRANSFORMATION COMPLETED'//)
WRITE 6 ;1000
;100 CONTINUE
RETURN
END
#
#
#  THE FOLLOWING PROCEDURE DISPLAYS A MENU FOR SELECTION OF
#  POST PROCESSING FUNCTIONS
#
SUB MENU
```

```
;1000 FORMAT'(//1X,'MENU FOR PLOTS AND SUMMARY TABLES:'/)
;1001 FORMAT'(10X,'1) BASIC GEOMETRY PLOT')
;1002 FORMAT'(10X,'2) GEOMETRY PLOT WITH LOAD VECTORS SHOWN')
;1003 FORMAT'(10X,'3) DEFLECTED GEOMETRY PLOT')
;1005 FORMAT'(10X,'4) CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS')
;1006 FORMAT'(10X,'5) CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN')
;1007 FORMAT'(10X,'6) CONTOUR PLOT OF SKIN MARGIN OF SAFETY')
;1008 FORMAT'(10X,'7) CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR STRAINS')
;1009 FORMAT'(10X,'8) CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN')
;1010 FORMAT'(10X,'9) CONTOUR PLOT OF STIFF. MARGIN OF SAFETY')
;1011 FORMAT'(9X,'10) SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS')
;1012 FORMAT'(9X,'11) SUMMARY TABLE OF PLY STRESSES AND STRAINS')
;1013 FORMAT'(9X,'12) SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY'/)
;1014 FORMAT'(9X,'99) TERMINATE POST PROCESSING SESSION'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008:WRITE 6 ;1009:WRITE 6 ;1010:WRITE 6 ;1011
WRITE 6 ;1012:WRITE 6 ;1013:WRITE 6 ;1014
RETURN
END
#
#MENU SELECTION SUBROUTINE
#
SUB SEL2 &NPL &IMN
IF &IMN-2 ;10 ;20 ;30
;10 CALL GE01 &NPL
GOTO ;500
;20 CALL GE02 &NPL
GOTO ;500
;30 IF &IMN-4 ;40 ;50 ;60
;40 CALL DEFL &NG &XX
IF &NG-4 ;41 ;42 ;43
;41 CALL GE03 &XX &NPL
GOTO ;500
;42 CALL GE04 &XX &NPL
GOTO ;500
;43 CALL GE05 &XX &NPL
GOTO ;500
;50 CALL CON1 &NPL
GOTO ;500
;60 IF &IMN-6 ;70 ;80 ;90
;70 CALL CON2 &NPL
GOTO ;500
;80 CALL CON3 &NPL
GOTO ;500
;90 IF &IMN-8 ;100 ;500 ;500
;100 CALL CON4 &NPL
;500 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MENU SELECTION AND REDIRECTION
#
SUB SEL1 &IF1 &NPL
;5 CONTINUE
CALL MENU
QUERY &IMN "INPUT NUMBER OF MENU ITEM DESIRED >> "
CALL SEL2 &NPL &IMN
IF &IMN-99 ;10 ;200 ;10
;10 IF &IMN-8 ;5 ;120 ;120
;120 IF &IMN-9 ;130 ;140 ;150
;130 CALL CON5 &NPL
```

```
GOTO ;5
;140 CALL CON6 &NPL
GOTO ;5
;150 IF &IMN-11 ;160 ;170 ;180
;160 CALL SUM1 &IF1
GOTO ;5
;170 CALL SUM2 &IF1
GOTO ;5
;180 CALL SUM3 &IF1
GOTO ;5
;200 CONTINUE
RETURN
END
#
# MAIN PROGRAM SUBROUTINE DRIVER
#
CALL SEL1 &IF1 &NPL
CALL HC &NPL
CLOSE 8
;1000 FORMAT'(///1X,'THIS COMPLETES THE POST PROCESSING SESSION.'/)
;1001 FORMAT'(1X,'ANY HARD COPY PLOTS GENERATED HERE WILL RESIDE ON')
;1002 FORMAT'(1X,'FILES IN THE WORKING DIRECTORY, USE NORMAL SUBMITTAL')
;1003 FORMAT'(1X,'PROCEDURES TO ACTUALLY PRINT THEM.'//)
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
STOP
$END:
$!      End of PP.COM
```

A3.4.3 TUBULAR TRUSS CORE PANEL MODULE # 3 POST PROCESSING DRIVER

```
$!
$! ######################################################################
$!
$!                    DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO PERFORM
$!  POST PROCESSING FOR MODELS GENERATED WITH
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.3   2-28-92
$!  AUTHOR: WAYNE M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!######################################################################
$START:
$  W := WRITE SYS$OUTPUT
$W ""
$W "         Advanced Composites Structural Concept"
$W "             and Materials Technologies"
$W "         Automated DIAL finite element analysis"
$W "               POST PROCESSING MODULE"
$W ""
$W "                  VAX VERSION 1.3 "
$W "                    February 1992 "
$W ""
$W ""
$W ""
$INQUIRE DB "What is the name of the DIAL data base file?"
$ASSIGN 'DB' FIL002
$W ""
$W "Which module was used to create this data base?"
$W "  1) FLAT RECTANGULAR STRINGER STIFFENED PANEL  "
$W "  2) CURVED RECTANGULAR STRINGER STIFFENED PANEL  "
$W "  3) FLAT RECTANGULAR TUBULAR PANEL"
$W "  4) GEODESICALLY STIFFENED PANEL  "
$W ""
$INQUIRE MODULE " NUMBER?>> "
$W ""
$IF MODULE .EQS. "1" THEN GOTO MOD1
$IF MODULE .EQS. "2" THEN GOTO MOD2
$IF MODULE .EQS. "3" THEN GOTO MOD3
$IF MODULE .EQS. "4" THEN GOTO MOD2
$MOD1:
$W "This post-processor will only handle the tubular core panel!"
$GOTO END
$MOD2:
$W "This post-processor will only handle the tubular core panel!"
$GOTO END
$!
$MOD3:
$! THE FOLLOWING COMMAND MAY NEED TO BE MODIFIED TO RUN
$! THE DIAL SCOPE PROCESSOR IF THE EQUIVALENT NAME IS NOT
$! ALREADY SETUP IN THE USER LOGIN.COM
$!  EXAMPLE: CHANGE TO $RUN DIAL$DIR:SCOPE
$!
$SCOPE
START -1
```

```
#################################################################
#
#  THIS SCOPE PROCEDURE IS FOR MODULE 3
#  TUBULAR FLAT PANEL
#
#################################################################
SET SYNTAX ON
#
LET &IF1=0
DATA &IYES Y
DATA &NB B
;1000 FORMAT'(1X,'GIVE A SHORT NAME FOR THIS ANALYSIS')
WRITE 6 ;1000
QUERY !NAME "  PRECEDE MULTIPLE WORDS WITH APOSTROPHE >> "
QUERY !DEV "WHAT TYPE OF TERMINAL?(TEK,SEL,RETR,VT24) >> "
DEFINE SHCD VERS
QUERY  &NBW  "COLOR OR BLACK AND WHITE ? (C/B) >> "
IF &NBW-&NB ;91, ;90, ;91
;90 DSPECT SET 0 WHITE
DSPECT SET 1 BLACK
;91 CONTINUE
QUERY &NHC "WILL YOU WANT TO PLOT HARD COPIES?(Y/N) >> "
IF &NHC-&IYES ;30,;10,;30
;10 QUERY !HCD "INPUT HARD COPY DEVICE(VERS,QMS) >> "
DEFINE SHCD !HCD
;1000 FORMAT'(/1X,'WHAT OUTPUT OPTION WOULD YOU LIKE?')
;1001 FORMAT'(1X,'  1) SCREEN PLOTS ONLY, NO HARD COPIES')
;1002 FORMAT'(1X,'  2) ONLY HARD COPY PLOTS')
;1003 FORMAT'(1X,'  3) SCREEN PLOTS WITH HARD COPY PLOTS OPTIONAL')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
QUERY &N1 "OPTION NUMBER? >> "
IF &N1-2 ;30,;40,;50
;30 DEVICE OPEN !DEV
LET &NPL=0
GOTO ;60
;40 DEVICE OPEN !HCD
LET &NPL=0
GOTO ;60
;50 DEVICE OPEN !DEV
LET &NPL=1
META OPEN SEL
;60 CONTINUE
TITLE ON
TITLE1 !NAME
#
# SUBROUTINE FOR CREATING METAFILE PLOTS ON DEMAND
#
SUB PLOT &NPL
DATA &IYES Y
IF &NPL ;100 ;100 ;10
;10 QUERY &NX "DO YOU WISH A HARD COPY PLOT OF THE LAST PLOT?(Y/N) >> "
IF &NX-&IYES ;100 ;20 ;100
;20 METASAVE
;100 CONTINUE
RETURN
END
#
# SUBROUTINE FOR SETTING VIEWPOINT
#
```

```
SUB VIEW &X,&Y,&Z
DATA &IYES Y
;1000 FORMAT'(/1X,'CURRENT VIEWPOINT IS: X ='F5.1,'  Y ='F5.1,'  Z ='F5.1,/)
;1001 FORMAT'(/1X,'VIEWING DIRECTION IS FROM THE VIEWPOINT COORDINATES TO'/)
;1002 FORMAT'(1X,'THE ORIGIN, ACTUAL DISTANCE BETWEEN THE POINTS IS NOT USED'/)
WRITE 6 ;1000 &X,&Y,&Z
;10 QUERY &NX "DO YOU WISH TO CHANGE THE VIEWPOINT FOR THIS PLOT?(Y/N) >>"
IF &NX-&IYES ;100 ;20 ;100
;20 WRITE 6 ;1001:WRITE 6 ;1002
QUERY &X "INPUT NEW VIEWPOINT X COORD.(INCLUDE DECIMAL POINT) >>  "
QUERY &Y "INPUT NEW VIEWPOINT Y COORD.(INCLUDE DECIMAL POINT) >>  "
QUERY &Z "INPUT NEW VIEWPOINT Z COORD.(INCLUDE DECIMAL POINT) >>  "
WRITE 6 ;1000 &X,&Y,&Z
;100 RETURN
END
#
# SUBROUTINE TO PRINT MESH QUERY FOR PLOTTING
#
SUB PRNT
;1000 FORMAT'(/1X,'PICK ONE OF THE FOLLOWING LABEL OPTIONS:')
;1001 FORMAT'(/15X,'NO NODE OR ELEMENT NUMBERING = 0')
;1002 FORMAT'(15X,'NODE NUMBERING ONLY          = 1')
;1003 FORMAT'(15X,'ELEMENT NUMBERING ONLY       = 2')
;1004 FORMAT'(15X,'NODE AND ELEMENT NUMBERING   = 3'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003:WRITE 6 ;1004
RETURN
END
#
#  SUBROUTINE FOR PRINTING LOAD SELECTION MENU
#
SUB LMNU
;2000 FORMAT'(//1X,'SELECT LOAD FACTOR LEVEL TO BE PLOTTED:')
;2001 FORMAT'(15X,'LINEAR SOLUTION          = 0')
;2002 FORMAT'(15X,'MAX LOAD STEP ATTAINED   = 1')
;2003 FORMAT'(15X,'INTERMEDIATE LOAD FACTOR = 2'/)
WRITE 6 ;2000:WRITE 6 ;2001:WRITE 6 ;2002:WRITE 6 ;2003
RETURN
END
#
#  SUBROUTINE FOR SELECTING NONLINEAR LOAD STEP
#
sub STAT &pain &iseq,&paot
#-----------------------------------------------------------------------
#     Return the Solution Sequence number and Load Factor thta are
#     closest to a requested Load Factor.
#
#      in   &pain  Requested Load Factor
#
#      out  &iseq  Solution sequence number for output Load Factor
#      out  &paot  Solution Load Factor closest to Requested Load Factor
#-----------------------------------------------------------------------
#                  Initialize Variables
      let &iseq=0
      let &paot=0.0
      let &is=0
      let &pa=0.0
      let &ismn=0
      let &pamn=0.0
#
      let &lstp = %icfl(PCT.HED.SOLV,0,0,1)
      let &istp = %icfl(BRCH.HED,0,1,2)
      let &nseq = %max(&lstp,&istp)
```

```
#
      do ;loop &is=1,&nseq
        let &ifn = %ifl(D.SV,0,&is)
        let &pa = %fafm(&ifn,1)
        if &pa-&pain 1,;mach,;find
          let &pamn = &pa
          let &ismn = &is
;loop continue
#                   Requested Load Factor is .gt. Final Solution Load Factor
        let &iseq = &is
        let &paot = &pa
        goto ;clos
;find continue
#                   Find Solution Load Factor closest to Requested Load Factor
      let &pdmn = &pain-&pamn
      let &pdmx = &pa-&pain
      if &pdmn-&pdmx 1,1,;max
        let &iseq = &ismn
        let &paot = &pamn
        goto ;clos
;max  continue
        let &iseq = &is
        let &paot = &pa
        goto ;clos
;mach continue
#                   Solution Load Factor matches Requested Load Factor
        let &iseq = &is
        let &paot = &pa
#
;clos continue
;110  format '(' Analysis State set to Solution Sequence Number ',I4)
;111  format '('                  with Solution Load Factor ',F10.3)
      write 6,;110 &iseq
      write 6,;111 &paot
#
      return
      end
#
#  BASIC GEOM PLOT SUBROUTINE
#
SUB GE01 &NPL
DATA &IYES Y
;999 FORMAT'(/1X,'AFTER EACH PLOT IS GENERATED HIT A RETURN TO CONTINUE'/)
WRITE 6 ;999
TITLE2 'GEOMETRY PLOT
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
;2000 FORMAT'(/1X,'PICK THE PARTS TO BE DISPLAYED:'/)
#
;2001 FORMAT'(15X,'ALL PARTS (SKINS + TUBES)          =  0')
;2002 FORMAT'(15X,'TOP SKIN ONLY                      =  1')
;2003 FORMAT'(15X,'BOTTOM SKIN ONLY                   =  2')
;2004 FORMAT'(15X,'ALL TUBES (webs + Top & Bot. walls =  3')
;2005 FORMAT'(15X,'Fillers only                       =  4')
;2006 FORMAT'(15X,'All tube webs only                 =  5'/)
#
WRITE 6 ;2000 :WRITE 6 ;2001 :WRITE 6 ;2002 :WRITE 6 ;2003
```

```
WRITE 6 ;2004 :WRITE 6 ;2005 :WRITE 6 ;2006
QUERY &N1 " INPUT OPTION >> "
#
IF &N1-3 1 ;120 ;120
IF &N1-1 1 ;110 ;130
     MSET 20 INSERT ALL
GOTO ;200
;110 MSET 20 INSERT MSET=1
GOTO ;200
;130 MSET 20 INSERT MSET=3
GOTO ;200
#
;120 IF &N1-4 1 ;140 ;150
     MSET 20 INSERT MSET=11
GOTO ;200
;140 MSET 20 INSERT MSET=4
GOTO ;200
;150 MSET 20 INSERT MSET=2
#
;200 CONTINUE
SCREEN VIRT -.9,.9,-.9,.9
CALL PRNT
QUERY &N1 " INPUT OPTION >> "
IF &N1-1 ;10 ;20 ;30
;10 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB:OVEND
GOTO ;60
;20 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
 OVNODE 0 0 MSET=20:OVEND
GOTO ;60
;30 IF &N1-2 ;60 ;40 ;50
;40 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
  OVELEM 0 0 MSET=20:OVEND
GOTO ;60
;50 OVBEGI:OVGEOM 0,0,0,MSET=20:OVMESH MSET=20:OVGLOB
  OVNODE 0 0 MSET=20:OVELEM 0 0 MSET=20:OVEND
;60 CONTINUE
MSET 20 DELETE ALL
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT SHOWING LOAD VECTORS
#
SUB GEO2 &NPL
TITLE2 'LOAD PLOT
LEGEND OFF
ORIENT Z UP
LET &X=1.
LET &Y=-1.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
DSET UL 0 1
VECT 0 1 0.
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GEO3 &XX &NPL
DATA &IYES Y
```

```
LEGEND OFF
ORIENT Z UP
LET &X=1.
LET &Y=-1.
LET &Z=1.5
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
TITLE2 'DEFLECTED GEOMETRY PLOT - LINEAR STRESS
DSET D 0 1
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED BUCKLED SHAPE
#
SUB GE04 &XX &NPL
DATA &IYES Y
LEGEND OFF
#
QUERY &NMOD " HOW MANY MODES WERE DETERMINED? >> "
      let &IMOD=1
;888   CONTINUE
;200   FORMAT'(' BIFURCATION BUCKLING - MODE SHAPE No. ',I2)
      WRITE  !TIT2  ;200  &IMOD
      TITLE2 !TIT2
DSET BUCK 0  &IMOD
#
ORIENT Z UP
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;170 ;160 ;170
;160 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;180
;170 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
#
;180  CONTINUE
      let &IMOD=&IMOD+1
      IF &IMOD-&NMOD  ;111 ;111 ;222
;111  QUERY &NX " DO YOU WANT TO PLOT THE NEXT MODE SHAPE? (Y/N) >> "
```

```
      IF &NX-&IYES  ;222  ;888  ;222
;222  CONTINUE
   RETURN
   END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GE05 &XX &NPL
DATA &IYES Y
LEGEND OFF
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
IF &NS-1 ;10 ;20 ;30
;10 DSET D,0,1
TITLE2 'DEFLECTED GEOM PLOT - NONLINEAR ANALYSIS - LINEAR SOLUTION
GOTO ;100
;20 LET &MLS = %ICFL(BRCH.HED,0,1,2)
DSET D,0,&MLS
TITLE2 'DEFLECTED GEOM PLOT - NONLINEAR ANALYSIS - FINAL STEP
GOTO ;100
;30 CONTINUE
;1005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;1005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
DSET D,0,&ISEQ
;2000 FORMAT'('DEFLECTED GEOM - NONLINEAR ANALYSIS - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2000 &PAOT
TITLE2 !TIT2
;100 CONTINUE
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVGLOB:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 &XX:OVGLOB
OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR PRINTING STRESS QUANTITY MENU
#
SUB MNU2
;1000 FORMAT'(/1X,'QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3')
;1001 FORMAT'(1X, '              (LINE MOMENT) M-X =  4, M-Y =  5')
;1002 FORMAT'(1X, '              (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13')
;1003 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
```

```
WRITE 6 ;1003
RETURN
END
#
#  SUBROUTINE TO PRINT LAYER STRESS MENU
#
SUB MNU3
;1000 FORMAT'(/1X,'QUANTITY MENU:(LAYER STRESS)  S-X =  1, S-Y =  2, S-XY = 4')
;1001 FORMAT'(1X, '               (LAYER STRAIN)  E-X = 11, E-Y = 12, E-XY = 14')
;1002 FORMAT'(1X, '         -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN AVG LOADS AND STRAIN
#
SUB CON1 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Y UP
LET &X=0.
LET &Y=0.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN AVG LINE LOAD OR STRAIN - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
DLABEL 1 'N-X AVG       LINE LOAD
DLABEL 2 'N-Y AVG       LINE LOAD
DLABEL 3 'N-XY AVG      LINE LOAD
DLABEL 4 'M-X AVG      LINE MOMENT
DLABEL 5 'M-Y AVG      LINE MOMENT
DLABEL 11 'E-X AVG        STRAIN
DLABEL 12 'E-Y AVG        STRAIN
DLABEL 13 'E-XY AVG       STRAIN
;25 CONTINUE
CALL MNU2
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;70 ;30
;30 CONTINUE
QUERY &N1 "TOP OR BOTTOM FACE SHEET?(TOP=0,BOTTOM=1)>> "
IF &N1 ;70 ;40 ;50
```

```
;40 DFROMS &N3,&N3,MSH,0,MSET=1
OVBEGI:OVGEOM 0,0,0,MSET=1:OVCONT &N3,0,10,0,MSET=1
OVANNO 0 "TOP SHEET":OVMESH MSET=1:OVGLOB:OVEND
GOTO ;60
;50 DFROMS &N3,&N3,MSH,0,MSET=3
OVBEGI:OVGEOM 0,0,0,MSET=3:OVCONT &N3,0,10,0,MSET=3
OVANNO 0 "BOTTOM SHEET":OVMESH MSET=3:OVGLOB:OVEND
;60 CALL PLOT &NPL
GOTO ;25
;70 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN PLY STRESS AND STRAIN
#
SUB CON2 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Y U
LET &X=0.
LET &Y=0.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
#LET !ANO='LINEAR ANALYSIS
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
#LET &PAIN=100.
#CALL STAT &PAIN &ISEQ,&PAOT
#;2100 FORMAT'('LOAD STEP=',F5.2)
#WRITE !ANO ;2100 &PAOT
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
#;2101 FORMAT'('LOAD STEP=',F5.2)
#WRITE !ANO ;2101 &PAOT
;999 CONTINUE
DLABEL 1 'S-X PLY        STRESS
DLABEL 2 'S-Y PLY        STRESS
DLABEL 4 'S-XY PLY       STRESS
DLABEL 11 'E-X PLY       STRAIN
DLABEL 12 'E-Y PLY       STRAIN
DLABEL 14 'E-XY PLY      STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &N1 "INPUT TOP OR BOTTOM FACE SHEET(TOP=0,BOTTOM=1) >> "
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
CALL MNU3
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
```

```
IF &N3-99 ;90 ;200 ;90
;90 CONTINUE
IF &N1 ;200 ;100 ;110
;100 CONTINUE
;2000 FORMAT'('TOP F.S. STRESS QUANTITY = 'I3,', LAYER = 'I3,',  SURFACE = 'I3)
WRITE !TIT2 ;2000 &N3 &NLN &NLS
TITLE2 !TIT2
SSHELL 1 &NLN &NLS MSET=1
DFROMS &N3,&N3,MSH,0,MSET=1
OVBEGI:OVGEOM 0,0,0,MSET=1:OVCONT &N3,0,10,0,MSET=1
OVMESH MSET=1:OVGLOB:OVEND
GOTO ;120
;110 CONTINUE
;3000 FORMAT'('BOTTOM F.S. STRESS QUANT. = 'I3,', LAYER = 'I3,',  SURFACE = 'I3)
WRITE !TIT2 ;3000 &N3 &NLN &NLS
TITLE2 !TIT2
SSHELL 1 &NLN &NLS MSET=3
DFROMS &N3,&N3,MSH,0,MSET=3
OVBEGI:OVGEOM 0,0,0,MSET=3:OVCONT &N3,0,10,0,MSET=3
OVMESH MSET=3:OVGLOB:OVEND
;120 CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;200 ;130 ;200
;130 GOTO ;80
;200 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN MARGIN OF SAFETY
#
SUB CON3 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Y UP
LET &X=0.
LET &Y=0.
LET &Z=1.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
TITLE2 'SKIN M.S. AT ELEM QUAD POINTS - LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
TITLE2 'SKIN M.S. AT ELEM QUAD POINTS - NONLINEAR - FINAL STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('SKIN M.S. AT ELEM QUAD POINTS - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE !TIT2 ;2006 &PAOT
TITLE2 !TIT2
;999 CONTINUE
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
QUERY &N1 "TOP OR BOTTOM FACE SHEET?(TOP=0,BOTTOM=1)>> "
```

```
IF &N1 ;70 ;40 ;50
;40 SSHELL/MQMS=1 1 1 -1 MSET=1
DFROME 1 1 MSH MSET=1
OVBEGI:OVGEOM 0,0,0,MSET=1:OVCONT 1,0,10,0,MSET=1
OVANNO 0 "TOP SHEET":OVMESH MSET=1:OVGLOB:OVEND
GOTO ;60
;50 SSHELL/MQMS=1 1 1 -1 MSET=3
DFROME 1 1 MSH MSET=3
OVBEGI:OVGEOM 0,0,0,MSET=3:OVCONT 1,0,10,0,MSET=3
OVANNO 0 "BOTTOM SHEET":OVMESH MSET=3:OVGLOB:OVEND
;60 CALL PLOT &NPL
;70 CONTINUE
FAIL OFF  11
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF AVG LOADS & STRAIN
#             TUBES
#
SUB CON4 &NPL  &NTBW &NPRT
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
#
;101 FORMAT'('Tube ',I2,'-',I1,' AVG L.LOAD OR STRAIN - lin. ana.')
WRITE  !ST1  ;101 &NTBW &NPRT
TITLE2 !ST1
#
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
#
;102 FORMAT'('Tube ',I2,'-',I1,' AVG L.LOAD/STRAIN - final step ')
WRITE  !ST2  ;102 &NTBW &NPRT
TITLE2 !ST2
#
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
#
;2006 FORMAT'('Tube ',I2,'-',I1,' AVG L.LOAD/STRAIN, L. fac.= ',F5.2)
WRITE !TIT2 ;2006 &NTBW &NPRT &PAOT
TITLE2 !TIT2
#
;999 CONTINUE
#TITLE2 'TUBE AVG LINE LOAD OR STRAIN
#
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
```

```
DLABEL 3 'N-XY AVG      LINE LOAD
DLABEL 11 'E-X AVG        STRAIN
DLABEL 12 'E-Y AVG        STRAIN
DLABEL 13 'E-XY AVG       STRAIN
;25 CONTINUE
CALL MNU2
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;30
;30 DFROMS &N3,&N3,MSH,0, MSET=12
CONT &N3,0,10,0, MSET=12
CALL PLOT &NPL
GOTO ;25
;40 CONTINUE
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF LAYER STRESS & STRAIN
#    TUBES
#
SUB CON5 &NPL  &NTBW  &NPRT
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z UP
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
LET !ANO='LINEAR SOLUTION
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
LET !ANO= 'FINAL LOAD STEP
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2100 FORMAT'('LOAD STEP=',F5.2)
WRITE !ANO ;2000 &PAOT
;999 CONTINUE
;70 CONTINUE
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
CALL MNU3
```

```
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSHELL 1 &NLN &NLS    MSET=12
DFROMS &N3,&N3,MSH,0,     MSET=12
#
;3000 FORMAT'('Tube ',I2,'-',I1,' Stress quan.='I3,', lay.=',I3,', surf.='I3)
WRITE !TIT2 ;3000 &NTBW &NPRT &N3 &NLN &NLS
TITLE2 !TIT2
#
OVBEGI:OVGEOM 0,0,0,    MSET=12:OVCONT &N3,0,10,0,    MSET=12
OVANNO 0 !ANO:OVMESH    MSET=12:OVGLOB:OVEND
CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL TUBE PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING CONTOUR PLOTS OF MARGIN OF SAFETY
#     TUBE
#
SUB CON6 &NPL   &NTBW &NPRT
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z U
LET &X=1.
LET &Y=-1.
LET &Z=2.
CALL VIEW &X &Y &Z
VIEWPT &X &Y &Z
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
#
;500 SSET S,0,1
;101 FORMAT'('Tube ',I2,'-',I1,' Stress M.S. - linear ana.')
WRITE  !ST1  ;101 &NTBW &NPRT
TITLE2 !ST1
#
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
#
;102 FORMAT'('Tube ',I2,'-',I1,' Stress M.S., nonl. ana., - final step')
WRITE  !ST2  ;102 &NTBW &NPRT
TITLE2 !ST2
#
GOTO ;999
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
#
;2006 FORMAT'('Tube ',I2,'-',I1,' M.S.-Nonl. ana. - load fac.=',F5.2)
WRITE !TIT2 ;2006 &NTBW &NPRT &PAOT
TITLE2 !TIT2
#
;999 CONTINUE
DLABEL 1 'MARGIN OF SAFETY
```

```
FAIL ON 11
SSHELL/MQMS=1 1 1 -1   MSET=12
DFROME 1 1 MSH    MSET=12
CONT 1,0,10,0,  MSET=12
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR GENERATING MAX-MIN SUMMARY TABLES
#  FOR THE FACE SHEETS AND TUBE STIFFENERS AVG LAMINATE LOADS & STRAINS
#
SUB SUM1 &IF1
    DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
#
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
;3000 FORMAT'(/1X,'AVG LINE LOAD OR STRAIN - LINEAR SOLUTION')
WRITE 6 ;3000
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
;3001 FORMAT'(/1X,'AVG LINE LOAD OR STRAIN - NONLINEAR - FINAL STEP')
WRITE 6 ;3000
GOTO ;999
#
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'(/1X'AVG LINE LOAD OR STRAIN - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE 6 ;2006 &PAOT
#
;999 CONTINUE
PRINT OFF,0,LIST
SLABEL 1 'N-X          LINE LOAD
SLABEL 2 'N-Y          LINE LOAD
SLABEL 3 'N-XY         LINE LOAD
SLABEL 4 'M-X          MOMENT LOAD
SLABEL 5 'M-Y          MOMENT LOAD
SLABEL 11 'E-X AVG       STRAIN
SLABEL 12 'E-Y AVG       STRAIN
SLABEL 13 'E-XY AVG      STRAIN
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
DLABEL 3 'N-XY AVG     LINE LOAD
DLABEL 11 'E-X AVG       STRAIN
DLABEL 12 'E-Y AVG       STRAIN
```

```
DLABEL 13 'E-XY AVG      STRAIN
TITLE2 'TOP FACE SHEET AVG LINE LOAD OR STRAIN AT INTEGRATION POINTS
SPRINT 8, 1,2,3,4,5,11,12,13, 0, MSET=1
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 TITLE2 'BOTTOM FACE SHEET AVG LINE LOAD OR STRAIN AT INTEGRATION POINTS
SPRINT 8, 1,2,3,4,5,11,12,13, 0, MSET=3
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
#
;80 TITLE2 'TUBE STIFFENER AVG LINE LOAD OR STRAIN AT INTEGRATION POINTS
SPRINT 8, 1,2,3,4,5,11,12,13, 0, MSET=11
#
IF &IF1 ;90 ;90 ;100
;90 QUERY &ID "HIT RETURN TO CONTINUE!"
;100 CONTINUE
   RETURN
   END
#
# SUBROUTINE FOR MAKING SUMMARY TABLES OF PLY STRESS AND STRAIN
#
SUB SUM2 &IF1
    DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
#
;40 CONTINUE
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
#
;520 CONTINUE
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
#
;999 CONTINUE
PRINT OFF,0,LIST
SLABEL 1 'S-X PLY        STRESS
SLABEL 2 'S-Y PLY        STRESS
SLABEL 4 'S-XY PLY       STRESS
SLABEL 11 'E-X PLY       STRAIN
SLABEL 12 'E-Y PLY       STRAIN
SLABEL 14 'E-XY PLY      STRAIN
DLABEL 1 'S-X PLY        STRESS
DLABEL 2 'S-Y PLY        STRESS
DLABEL 4 'S-XY PLY       STRESS
DLABEL 11 'E-X PLY       STRAIN
DLABEL 12 'E-Y PLY       STRAIN
```

```
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE GIVEN IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
#
TITLE2 'TOP FACE SHEET PLY STRESS SUMMARY AT ELEM QUAD POINTS
PSHELL 6, 1,2,4,11,12,14, 1,0,0, 1,0,1, MSET=1
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 TITLE2 'BOTTOM FACE SHEET PLY STRESS SUMMARY AT ELEM QUAD POINTS
PSHELL 6, 1,2,4,11,12,14, 1,0,0, 1,0,1, MSET=3
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
#
;80 TITLE2 'TUBE STIFFENER PLY STRESS SUMMARY AT ELEM QUAD POINTS
PSHELL 6, 1,2,4,11,12,14, 1,0,0, 1,0,1, MSET=11
#
IF &IF1 ;90 ;90 ;100
;90 QUERY &ID "HIT RETURN TO CONTINUE!"
;100 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING SUMMARY TABLE OF MARGINS OF SAFETY
#
SUB SUM3 &IF1
    DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
#
CALL LMNU
QUERY &NS " INPUT SELECTION >> "
IF &NS-1 ;500 ;510 ;520
;500 SSET S,0,1
GOTO ;999
;510 LET &MLS = %ICFL(BRCH.HED,0,1,2)
SSET S,0,&MLS
GOTO ;999
;520 CONTINUE
#
;2005 FORMAT'(/1X,'PROGRAM WILL FIND LOAD STEP CLOSEST TO SELECTED FACTOR.'/)
WRITE 6 ;2005
QUERY &PAIN " INPUT INTERMEDIATE LOAD FACTOR FRACTION (0 TO 1.0) >> "
CALL STAT &PAIN &ISEQ,&PAOT
SSET S,0,&ISEQ
;2006 FORMAT'('MARGIN OF SAFETY SUMMARY - NONLINEAR - LOAD FACTOR ='F5.2)
WRITE 6 ;2006 &PAOT
#
;999 CONTINUE
FAIL ON 11
PRINT OFF,0,LIST
TITLE2 'TOP FACE SHEET MARGIN OF SAFETY SUMMARY AT ELEMENT QUADRATURE POINTS
PSHELL 3, 11,12,13, 1,0,0, 1,1,1, MSET=1
IF &IF1 ;50 ;50 ;60
```

```
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 TITLE2 'BOTTOM FACE SHEET MARGIN OF SAFETY SUM. AT ELEMENT QUADRATURE POINTS
PSHELL 3, 11,12,13, 1,0,0, 1,1,1, MSET=3
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
#
;80 TITLE2 'TUBE STIFFENER MARGIN OF SAFETY SUMMARY AT ELEMENT QUADRATURE POINTS
PSHELL 3, 11,12,13, 1,0,0, 1,1,1, MSET=11
#
IF &IF1 ;90 ;90 ;100
;90 QUERY &ID "HIT RETURN TO CONTINUE!"
;100 CONTINUE
FAIL OFF 11
   RETURN
   END
#
#  SUBROUTINE FOR TRANSFORMING METAPLOT FILES TO DEVICE PLOT FILES
#
SUB HC &NPL
IF &NPL ;100 ;100 ;10
;10   DEVICE OPEN <SHCD>
;1000 FORMAT'(/1X,'SAVED PLOTS ARE NOW BEING TRANSFORMED TO PLOT FILES')
;1001 FORMAT'(1X,'PLEASE WAIT UNTIL COMPLETION IS INDICATED.')
WRITE 6 ;1000
WRITE 6 ;1001
METAPLOT
;1000 FORMAT'(/1X,'PLOT FILE TRANSFORMATION COMPLETED'//)
WRITE 6 ;1000
;100 CONTINUE
RETURN
END
#    ------------------------------------------------------------------
#   Sub DEFL should be deleted when this package is merged with
#            the post-processor for the MOD 1
#
SUB DEFL &NG &XX
DATA &IYES Y
;1000 FORMAT'(//1X,'SELECT DESIRED DEFLECTION PLOT TYPE:')
;1001 FORMAT'(/15X,'LINEAR ANALYSIS                = 0')
;1002 FORMAT'(15X,'BUCKLING ANALYSIS MODE SHAPE  = 1')
;1003 FORMAT'(15X,'NONLINEAR ANALYSIS             = 2'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
QUERY &NS " INPUT SELECTION >> "
;1004 FORMAT'(/1X,'SELECT DEFLECTION MAGNIFICATION FACTOR TO BE USED:')
;1005 FORMAT'(/15X,'LOW (.10)          = 0')
;1006 FORMAT'(15X,'MED (.20)          = 1')
;1007 FORMAT'(15X,'HIGH (.50)         = 2')
;1008 FORMAT'(15X,'USER SELECTABLE    = 3'/)
WRITE 6 ;1004:WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008
QUERY &NM " INPUT SELECTION >> "
IF &NM-1 ;10 ;20 ;30
;10 LET &XX=.1
GOTO ;100
;20 LET &XX=.2
GOTO ;100
;30 IF &NM-2 ;100 ;40 ;50
;40 LET &XX=.5
GOTO ;100
;50 QUERY &XX " INPUT MAGNIFICATION FACTOR >> "
;100 CONTINUE
IF &NS-1 ;60 ;70 ;80
```

```
;60 LET &NG=3
GOTO ;200
;70 LET &NG=4
GOTO ;200
;80 LET &NG=5
;200 CONTINUE
RETURN
END
#   ------------------------------------------------------------------------
SUB TUBT  &NTBW &NPRT
let &NTBW=1
    &NPRT=1
mset 12 delete all
mset 12 insert mset 9
return
end
#
SUB TUBB  &NTBW &NPRT
let &NTBW=1
    &NPRT=2
define TBPN "bot tube wall "
mset 12 delete all
mset 12 insert mset 10
return
end
#
SUB TUBW  &NTBW &NPRT
mset 12 delete all
;22 query &MTBW "Input the number of tubes in the panel: >> "
;10 FORMAT'(/3X,'The tube webs will be plotted ONE AT A TIME as you wish.'/)
WRITE 6  ;10
;11 FORMAT'(/3X,'If the the number of tubes in the panel is N,')
WRITE 6  ;11
;12 FORMAT'(/3X,'the number of tube webs in the panel is NW=N+1. Now ...'/)
WRITE 6  ;12
query &NTBW "Input the tube web ID number you want to plot (1 to NW): >> "
let &NW=&MTBW+1
IF &NTBW-&NW ;13 ;13 ;22
;13 continue
mset 12 insert mset 2
mset 12 mask   mesh &NTBW
let &NPRT=0
return
end
#   ------------------------------------------------------------------------
SUB MENX
;1011 FORMAT'(9X,'10) SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS')
;1012 FORMAT'(9X,'11) SUMMARY TABLE OF PLY STRESSES AND STRAINS')
;1013 FORMAT'(9X,'12) SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY')
#
;1014 FORMAT'(9X,'13) CON. PLOT OF top TUBE AVG LAM. LOADS OR STRAINS')
;1015 FORMAT'(9X,'14) CON. PLOT OF top TUBE PLY STRESS OR STRAIN')
;1016 FORMAT'(9X,'15) CON. PLOT OF top TUBE MARGIN OF SAFETY')
#
;1017 FORMAT'(9X,'16) CON. PLOT OF bot. TUBE AVG LAM. LOADS OR STRAINS')
;1018 FORMAT'(9X,'17) CON. PLOT OF bot. TUBE PLY STRESS OR STRAIN')
;1019 FORMAT'(9X,'18) CON. PLOT OF bot. TUBE MARGIN OF SAFETY')
WRITE 6 ;1011
WRITE 6 ;1012:WRITE 6 ;1013:WRITE 6 ;1014
WRITE 6 ;1015:WRITE 6 ;1016:WRITE 6 ;1017
WRITE 6 ;1018:WRITE 6 ;1019
RETURN
```

```
END
#-----------
#       SELECTION OF POST PROCESSING FUNCTIONS
SUB MENU
;1000 FORMAT'(//1X,'MENU FOR PLOTS AND SUMMARY TABLES:'/)
;1001 FORMAT'(10X,'1) BASIC GEOMETRY PLOT')
;1002 FORMAT'(10X,'2) GEOMETRY PLOT WITH LOAD VECTORS SHOWN')
;1003 FORMAT'(10X,'3) DEFLECTED GEOMETRY PLOT')
#
;1005 FORMAT'(10X,'4) CONTOUR PLOT OF FACE SHEET AVG LAMINATE LOADS OR STRAINS')
;1006 FORMAT'(10X,'5) CONTOUR PLOT OF FACE SHEET PLY STRESS OR STRAIN')
;1007 FORMAT'(10X,'6) CONTOUR PLOT OF FACE SHEET MARGIN OF SAFETY')
#
;1008 FORMAT'(10X,'7) CONTOUR PLOT OF TUBE web AVG LAM. LOADS OR STRAINS')
;1009 FORMAT'(10X,'8) CONTOUR PLOT OF TUBE web PLY STRESS OR STRAIN')
;1010 FORMAT'(10X,'9) CONTOUR PLOT OF TUBE web MARGIN OF SAFETY')
;1020 FORMAT'(9X,'99) TERMINATE POST PROCESSING SESSION'/)
#
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008:WRITE 6 ;1009:WRITE 6 ;1010
CALL MENX
WRITE 6 ;1020
RETURN
END
#       ------------------------------------------------------------------
#                       MENU SELECT. AND REDIRECT'N &IMN=(1,2,3) & (4,5,6)
SUB SEL2 &NPL &IMN
IF &IMN-3    1    1   ;50
IF &IMN-2    1 ;20   ;30
CALL GEO1 &NPL
GOTO ;500
;20 CALL GEO2 &NPL
GOTO ;500
#
;30 CALL DEFL &NG &XX
IF &NG-4 ;41 ;42 ;43
;41 CALL GEO3 &XX &NPL
GOTO ;500
;42 CALL GEO4 &XX &NPL
GOTO ;500
;43 CALL GEO5 &XX &NPL
GOTO ;500
#    ----------------------
;50 IF &IMN-5   1   ;60   ;80
CALL CON1 &NPL
GOTO ;500
;60 CALL CON2 &NPL
GOTO ;500
;80 CALL CON3 &NPL
;500 CONTINUE
RETURN
END
#
SUB SEL4 &IMN &NPL  &NPRT
#    ---------------------MENU SELECT.  &IMN=(7,8,9) TO (10,11,12)  -----
IF &IMN-9    1  1     ;150
IF &IMN-8    1  ;120 ;140
CALL TUBW        &NTBW  &NPRT
CALL CON4 &NPL   &NTBW  &NPRT
GOTO ;5
;120 CALL TUBW  &NTBW  &NPRT
```

```
CALL CON5 &NPL  &NTBW  &NPRT
GOTO ;5
;140 CALL TUBW  &NTBW  &NPRT
CALL CON6 &NPL  &NTBW  &NPRT
GOTO ;5
#     ----------------------
;150 IF &IMN-11 1 ;170 ;180
CALL SUM1 &IF1
GOTO ;5
;170 CALL SUM2 &IF1
GOTO ;5
;180 CALL SUM3 &IF1
;5 CONTINUE
RETURN
END
#  ------------- ------------------------------------------------------------
#                       MENU SELECT. AND REDIR. &IMN= (13,14,15) (16,17,18)
SUB SEL3 &IMN &NPL
# mset 9 : all top tube walls
#
IF &IMN-15  1 1    ;250
IF &IMN-14  1 ;230 ;240
CALL TUBT       &NTBW  &NPRT
CALL CON4 &NPL  &NTBW  &NPRT
GOTO ;5
;230 CALL TUBT  &NTBW  &NPRT
CALL CON5 &NPL  &NTBW  &NPRT
GOTO ;5
;240 CALL TUBT  &NTBW  &NPRT
CALL CON6 &NPL  &NTBW  &NPRT
GOTO ;5
#    ----------------------
# mset 10 : all bottom tube walls
#
;250 IF &IMN-17 1 ;430 ;440
CALL TUBB       &NTBW  &NPRT
CALL CON4 &NPL  &NTBW  &NPRT
GOTO ;5
;430 CALL TUBB  &NTBW  &NPRT
CALL CON5 &NPL  &NTBW  &NPRT
GOTO ;5
;440 CALL TUBB  &NTBW  &NPRT
CALL CON6 &NPL  &NTBW  &NPRT
GOTO ;5
;5 CONTINUE
RETURN
END
#-------------------------------------------------------------------------
#                    MENU SELECTION AND REDIRECTION
SUB SEL1 &IF1 &NPL
;5 CONTINUE
CALL MENU
QUERY &IMN "INPUT NUMBER OF MENU ITEM DESIRED >> "
#
IF &IMN-99  1  ;200 ;200
IF &IMN    ;5  ;5    1
IF &IMN-19  1  ;5   ;5
IF &IMN-12  1   1   ;13
IF &IMN-6   1   1   ;12
#                        SEL 2 (1 TO 6)
CALL SEL2 &NPL &IMN
GOTO ;5
```

```
#                          SEL 3 (13 to 18)
;13  CALL SEL3 &IMN &NPL
GOTO ;5
#                          SEL 4 (7 TO 12)
;12  CALL SEL4 &IMN &NPL
GOTO ;5
;200 CONTINUE
RETURN
END
#    ---------------------------------------------------------------------
#                     MAIN PROGRAM SUBROUTINE DRIVER
CALL SEL1 &IF1 &NPL
CALL HC &NPL
CLOSE 8
;1000 FORMAT'(///1X,'THIS COMPLETES THE POST PROCESSING SESSION.'/)
;1001 FORMAT'(1X,'ANY HARD COPY PLOTS GENERATED HERE WILL RESIDE ON')
;1002 FORMAT'(1X,'FILES IN THE WORKING DIRECTORY, USE NORMAL SUBMITTAL')
;1003 FORMAT'(1X,'PROCEDURES TO ACTUALLY PRINT THEM.'//)
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
STOP
$END:
$!     End of PP.COM
```

A.3.4.4 GEODESIC CURVED STIFFENED PANEL MODULE # 4 POST PROCESSING DRIVER

```
                    DIALAMATIC
$!
$!  THIS PROCEDURE IS DESIGNED TO PERFORM
$!  POST PROCESSING FOR MODELS GENERATED WITH
$!  DIALAMATIC, THE NASA ACT GENERIC MODEL PROGRAM
$!
$!  VAX VERSION V1.1   12-17-90
$!  AUTHOR: WAYNE M. BROWN  LMSC  81-12
$!  PHONE: (408) 756-1137
$!
$!#####################################################################
$START:
$  W := WRITE SYS$OUTPUT
$W ""
$W "        Advanced Composites Structural Concept"
$W "            and Materials Technologies"
$W "        Automated DIAL finite element analysis"
$W "              POST PROCESSING MODULE"
$W ""
$W "                 VAX VERSION 1.1"
$W ""
$W ""
$W ""
$INQUIRE DB "What is the name of the DIAL data base file?"
$ASSIGN 'DB' FILOO2
$W ""
$W "Which module was used to create this data base?"
$W "  1) FLAT RECTANGULAR STRINGER STIFFENED PANEL"
$W "  2) CURVED RECTANGULAR STRINGER STIFFENED PANEL"
$W "  3) FLAT RECTANGULAR TUBULAR PANEL"
$W "  4) CURVED GEODESICALLY STIFFENED RECTANGULAR PANEL"
$W ""
$INQUIRE MODULE " NUMBER?>> "
$W ""
$IF MODULE .EQS. "1" THEN GOTO END
$IF MODULE .EQS. "2" THEN GOTO END
$IF MODULE .EQS. "3" THEN GOTO MOD4
$GOTO END
$MOD4:
$! THE FOLLOWING COMMAND MAY NEED TO BE MODIFIED TO RUN
$! THE DIAL SCOPE PROCESSOR IF THE EQUIVALENT NAME IS NOT
$! ALREADY SETUP IN THE USER LOGIN.COM
$!  EXAMPLE: CHANGE TO $RUN DIAL$DIR:SCOPE
$SCOPE
START -1
##################################################################
#
#  THIS SCOPE PROCEDURE IS FOR MODULE 4
#  GEODESICALLY STIFFENED CURVED PANEL
#
##################################################################
SET SYNTAX ON
LET &IF1=0
DATA &IYES Y
;1000 FORMAT'(1X,'GIVE A SHORT NAME FOR THIS ANALYSIS')
WRITE 6 ;1000
QUERY !NAME "  PRECEDE MULTIPLE WORDS WITH APOSTROPHE >> "
QUERY !DEV "WHAT TYPE OF TERMINAL?(TEK,SEL,RETR,VT24)>> "
DEFINE SHCD VERS
QUERY &NHC "WILL YOU WANT TO PLOT HARD COPIES?(Y/N)>> "
```

```
IF &NHC-&IYES ;30,;10,;30
;10 QUERY !HCD "INPUT HARD COPY DEVICE(VERS,QMS)>> "
DEFINE SHCD !HCD
;1000 FORMAT'(/1X,'WHAT OUTPUT OPTION WOULD YOU LIKE?')
;1001 FORMAT'(1X,'  1) SCREEN PLOTS ONLY, NO HARD COPIES')
;1002 FORMAT'(1X,'  2) ONLY HARD COPY PLOTS')
;1003 FORMAT'(1X,'  3) SCREEN PLOTS WITH HARD COPY PLOTS OPTIONAL')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
QUERY &N1 "OPTION NUMBER?>> "
IF &N1-2 ;30,;40,;50
;30 DEVICE OPEN !DEV
LET &NPL=0
GOTO ;60
;40 DEVICE OPEN !HCD
LET &NPL=0
GOTO ;60
;50 DEVICE OPEN !DEV
LET &NPL=1
META OPEN SEL
;60 CONTINUE
TITLE ON
TITLE1 !NAME
#
# SUBROUTINE FOR CREATING METAFILE PLOTS ON DEMAND
#
SUB PLOT &NPL
DATA &IYES Y
IF &NPL ;100 ;100 ;10
;10 QUERY &NX "DO YOU WISH A HARD COPY PLOT OF THE LAST PLOT?(Y/N)>> "
IF &NX-&IYES ;100 ;20 ;100
;20 METASAVE
;100 CONTINUE
RETURN
END
#
#  BASIC GEOM PLOT SUBROUTINE
#
SUB GEO1 &NPL
DATA &IYES Y
;1000 FORMAT'(/1X,'AFTER EACH PLOT IS GENERATED HIT A RETURN TO CONTINUE'/)
WRITE 6 ;1000
TITLE2 'GEOMETRY PLOT
LEGEND OFF
ORIENT Z R
VIEWPT -1.,-.25,.3
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVMESH:OVEND
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT SHOWING LOAD VECTORS
#
SUB GEO2 &NPL
TITLE2 'LOAD PLOT
LEGEND OFF
ORIENT Z R
VIEWPT -1.,-.25,.3
DSET UL 0 1
VECT 0 1 0.
```

```
CALL PLOT &NPL
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED SHAPE
#
SUB GEO3 &NPL
DATA &IYES Y
LEGEND OFF
TITLE2 'DEFLECTED GEOMETRY PLOT - LINEAR STRESS
DSET D 0 1
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;130 ;120 ;130
;120 CONTINUE
ORIENT Z R
VIEWPT -1.,-.25,.3
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 .1
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;140
;130 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 .1:OVMESH:OVEND
CALL PLOT &NPL
;140 CONTINUE
RETURN
END
#
# SUBROUTINE FOR GEOMETRY PLOT OF DEFLECTED BUCKLED SHAPE
#
SUB GEO4 &NPL
DATA &IYES Y
LEGEND OFF
TITLE2 'DEFLECTED GEOMETRY PLOT - BIFURCATION BUCKLING
DSET BUCK 0 1
ORIENT Z R
VIEWPT -1.,-.25,.3
QUERY &NS " WOULD YOU LIKE TO SHOW THE UNDEFLECTED SHAPE?(Y/N)>> "
IF &NS-&IYES ;170 ;160 ;170
;160 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 .1
OVMESH:OVGEOM 2:LATT TYPE 2:OVMESH:OVEND
CALL PLOT &NPL
LATT TYPE 0
GOTO ;180
;170 CONTINUE
OVBEGI:SCREEN VIRT -.9,.9,-.9,.9:OVGEOM:OVGEOM 1 1 .1:OVMESH:OVEND
CALL PLOT &NPL
;180 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN AVG LOADS AND STRAIN
#
SUB CON1 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z R
VIEWPT -1.,-.5,0
SSET S,0,1
TITLE2 'SKIN PANEL AVG LINE LOAD OR STRAIN
DLABEL 1 'N-X AVG      LINE LOAD
```

```
DLABEL 2 'N-Y AVG       LINE LOAD
DLABEL 3 'N-XY AVG      LINE LOAD
DLABEL 4 'M-X AVG      LINE MOMENT
DLABEL 5 'M-Y AVG      LINE MOMENT
DLABEL 11 'E-X AVG        STRAIN
DLABEL 12 'E-Y AVG        STRAIN
DLABEL 13 'E-XY AVG       STRAIN
;25 CONTINUE
;1000 FORMAT'(/1X,'QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3')
;1001 FORMAT'(1X, '              (LINE MOMENT) M-X =  4, M-Y =  5')
;1002 FORMAT'(1X, '              (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13')
;1003 FORMAT'(1X, '          >   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;30
;30 DFROMS &N3,&N3,MSH,0,MSET=6
CONT &N3,0,15,0,MSET=6
CALL PLOT &NPL
GOTO ;25
;40 CONTINUE
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN MARGIN OF SAFETY
#
SUB CON3 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z R
VIEWPT -1.,-.5,0
SSET S,0,1
TITLE2 'SKIN PANEL MARGIN OF SAFETY AT ELEM QUAD POINTS
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
SSHELL/MQMS=1 1 1 -1 MSET=6
DFROME 1 1 MSH MSET=6
CONT 1,0,15,0,MSET=6
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
# SUBROUTINE FOR CONTOUR PLOTS OF SKIN PLY STRESS AND STRAIN
#
SUB CON2 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z R
VIEWPT -1.,-.5,0
SSET S,0,1
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
```

```
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
;1000 FORMAT'(/1X,'QUANTITY MENU:(LAYER STRESS)  S-X =  1, S-Y =  2, S-XY = 4')
;1001 FORMAT'(1X, '              (LAYER STRAIN)  E-X = 11, E-Y = 12, E-XY = 14')
;1002 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSET S,0,1
SSHELL 1 &NLN &NLS MSET=6
DFROMS &N3,&N3,MSH,0,MSET=6
;2000 FORMAT'('SKIN STRESS QUANTITY = 'I3,', LAYER = 'I3,',  SURFACE = 'I3)
WRITE !TIT2 ;2000 &N3 &NLN &NLS
TITLE2 !TIT2
CONT &N3,0,15,0,MSET=6
CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
END
#
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF AVG LOADS & STRAIN
# STIFFENERS
#
#
SUB CON4 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z R
VIEWPT -1.,-1.,-.5,
SSET S,0,1
TITLE2 'STIFFENER AVG LINE LOAD OR STRAIN
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
DLABEL 3 'N-XY AVG     LINE LOAD
DLABEL 11 'E-X AVG       STRAIN
DLABEL 12 'E-Y AVG       STRAIN
DLABEL 13 'E-XY AVG      STRAIN
;25 CONTINUE
;1000 FORMAT'(/1X,'QUANTITY MENU:(LINE LOAD)   N-X =  1, N-Y =  2, N-XY = 3')
;1001 FORMAT'(1X, '              (LINE MOMENT) M-X =  4, M-Y =  5')
;1002 FORMAT'(1X, '              (AVG STRAIN)  E-X = 11, E-Y = 12, E-XY = 13')
;1003 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;30 ;40 ;30
;30 DFROMS &N3,&N3,MSH,0,MSET=5
CONT &N3,0,15,0,MSET=5
CALL PLOT &NPL
GOTO ;25
;40 CONTINUE
```

```
RETURN
END
#
# SUBROUTINE TO CREATE CONTOUR PLOTS OF LAYER STRESS & STRAIN
# STIFFENERS
#
SUB CON5 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z R
VIEWPT -1.,-1.,-.5,
SSET S,0,1
;70 CONTINUE
DLABEL 1 'S-X PLY        STRESS
DLABEL 2 'S-Y PLY        STRESS
DLABEL 4 'S-XY PLY       STRESS
DLABEL 11 'E-X PLY       STRAIN
DLABEL 12 'E-Y PLY       STRAIN
DLABEL 14 'E-XY PLY      STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE PLOTTED OUT IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
;80 CONTINUE
QUERY &NLN "INPUT LAYER NUMBER TO BE PLOTTED >> "
QUERY &NLS "INPUT LAYER SURFACE: BOTTOM=-1, MIDDLE=0, TOP=1 >> "
;1000 FORMAT'(/1X,'QUANTITY MENU:(LAYER STRESS)  S-X =  1, S-Y =  2, S-XY = 4')
;1001 FORMAT'(1X, '              (LAYER STRAIN)  E-X = 11, E-Y = 12, E-XY = 14')
;1002 FORMAT'(1X, '        -->   INPUT 99 TO DISCONTINUE PLOTTING')
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
QUERY &N3 "INPUT DESIRED QUANTITY NUMBER >> "
IF &N3-99 ;90 ;100 ;90
;90 SSET S,0,1
SSHELL 1 &NLN &NLS MSET=5
DFROMS &N3,&N3,MSH,0,MSET=5
;2000 FORMAT'('STRINGER STRESS QUANTITY = 'I3,', LAYER = 'I3,',  SURFACE = 'I3)
WRITE !TIT2 ;2000 &N3 &NLN &NLS
TITLE2 !TIT2
CONT &N3,0,15,0,MSET=5
CALL PLOT &NPL
QUERY &N1 "DO YOU WISH ADDITIONAL PLY CONTOUR PLOTS?(Y/N)>> "
IF &N1-&IYES ;100 ;95 ;100
;95 GOTO ;80
;100 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING CONTOUR PLOTS OF MARGIN OF SAFETY
#  STIFFENERS
#
SUB CON6 &NPL
DATA &IYES Y
;10 LEGEND ON RIGHT
ORIENT Z R
VIEWPT -1.,-1.,-.5,
SSET S,0,1
TITLE2 'STIFFENER AML MARGIN OF SAFETY
DLABEL 1 'MARGIN OF     SAFETY
FAIL ON 11
SSHELL/MQMS=1 1 1 -1 MSET=5
```

```
DFROME 1 1 MSH MSET=5
CONT 1,0,15,0,MSET=5
CALL PLOT &NPL
;60 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR GENERATING MAX-MIN SUMMARY TABLES
#  FOR THE SKIN PANEL AND STIFFENERS AVG LAMINATE LOADS & STRAINS
#
SUB SUM1 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
SSET S,0,1
PRINT OFF,0,LIST
TITLE2 'SKIN PANEL AVG LINE LOAD OR STRAIN AT INTEGRATION POINTS
SLABEL 1 'N-X          LINE LOAD
SLABEL 2 'N-Y          LINE LOAD
SLABEL 3 'N-XY         LINE LOAD
SLABEL 4 'M-X          MOMENT LOAD
SLABEL 5 'M-Y          MOMENT LOAD
SLABEL 11 'E-X AVG       STRAIN
SLABEL 12 'E-Y AVG       STRAIN
SLABEL 13 'E-XY AVG      STRAIN
DLABEL 1 'N-X AVG      LINE LOAD
DLABEL 2 'N-Y AVG      LINE LOAD
DLABEL 3 'N-XY AVG     LINE LOAD
DLABEL 11 'E-X AVG       STRAIN
DLABEL 12 'E-Y AVG       STRAIN
DLABEL 13 'E-XY AVG      STRAIN
SPRINT 8,1,2,3,4,5,11,12,13,0,MSET=6
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 TITLE2 'STIFFENER AVG LINE LOAD OR STRAIN AT INTEGRATION POINTS
SPRINT 8,1,2,3,4,5,11,12,13,0,MSET=5
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
RETURN
END
#
# SUBROUTINE FOR MAKING SUMMARY TABLES OF PLY STRESS AND STRAIN
#
SUB SUM2 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
```

```
;40 CONTINUE
SSET S,0,1
PRINT OFF,0,LIST
SLABEL 1 'S-X PLY         STRESS
SLABEL 2 'S-Y PLY         STRESS
SLABEL 4 'S-XY PLY        STRESS
SLABEL 11 'E-X PLY        STRAIN
SLABEL 12 'E-Y PLY        STRAIN
SLABEL 14 'E-XY PLY       STRAIN
DLABEL 1 'S-X PLY         STRESS
DLABEL 2 'S-Y PLY         STRESS
DLABEL 4 'S-XY PLY        STRESS
DLABEL 11 'E-X PLY        STRAIN
DLABEL 12 'E-Y PLY        STRAIN
DLABEL 14 'E-XY PLY       STRAIN
;1000 FORMAT'(/1X,'LAYER OR PLY STRESSES ARE GIVEN IN THE LOCAL PLY')
;1001 FORMAT'(1X, ' MATERIAL SYSTEM DIRECTIONS (FIBER DIRECTIONS)'/)
WRITE 6 ;1000
WRITE 6 ;1001
SSET S,0,1
TITLE2 'SKIN PANEL PLY STRESS SUMMARY AT ELEM QUAD POINTS
PSHELL 6,1,2,4,11,12,14,1,0,0,1,0,1,MSET=6
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 TITLE2 'STIFFENER PLY STRESS SUMMARY AT ELEM QUAD POINTS
PSHELL 6,1,2,4,11,12,14,1,0,0,1,0,1,MSET=5
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MAKING SUMMARY TABLE OF MARGINS OF SAFETY
#
SUB SUM3 &IF1
DATA &IYES Y
;10 CONTINUE
IF &IF1 ;40 ;20 ;40
;20 QUERY &N2 "DO YOU WISH TO REDIRECTED THE PRINTOUT TO A FILE?(Y/N)>> "
IF &N2-&IYES ;40 ;30 ;40
;30 QUERY !OUT "INPUT FILE NAME FOR REDIRECTION OF PRINTED OUTPUT >> "
LET &IF1=1
OPEN 8 !OUT
UNIT 8
;40 CONTINUE
SSET S,0,1
FAIL ON 11
PRINT OFF,0,LIST
TITLE2 'SKIN PANEL MARGIN OF SAFETY SUMMARY AT ELEMENT QUADRATURE POINTS
PSHELL 1,1,0,0,0,1,1,1,MSET=5
IF &IF1 ;50 ;50 ;60
;50 QUERY &ID "HIT RETURN TO CONTINUE!"
;60 TITLE2 'STRINGER MARGIN OF SAFETY SUMMARY AT ELEMENT QUADRATURE POINTS
PSHELL 1,1,0,0,0,1,1,1,MSET=6
IF &IF1 ;70 ;70 ;80
;70 QUERY &ID "HIT RETURN TO CONTINUE!"
;80 CONTINUE
FAIL OFF 11
RETURN
END
#
#  SUBROUTINE FOR TRANSFORMING METAPLOT FILES TO DEVICE PLOT FILES
```

```
#
SUB HC &NPL
IF &NPL ;100 ;100 ;10
;10 DEVICE OPEN <SHCD>
;1000 FORMAT'(/1X,'SAVED PLOTS ARE NOW BEING TRANSFORMED TO PLOT FILES')
;1001 FORMAT'(1X,'PLEASE WAIT UNTIL COMPLETION IS INDICATED.')
WRITE 6 ;1000
WRITE 6 ;1001
METAPLOT
;1000 FORMAT'(/1X,'PLOT FILE TRANSFORMATION COMPLETED'//)
WRITE 6 ;1000
;100 CONTINUE
RETURN
END
#
#
#  THE FOLLOWING PROCEDURE DISPLAYS A MENU FOR SELECTION OF
#  POST PROCESSING FUNCTIONS
#
SUB MENU
;1000 FORMAT'(//1X,'MENU FOR PLOTS AND SUMMARY TABLES:'/)
;1001 FORMAT'(10X,'1) BASIC GEOMETRY PLOT')
;1002 FORMAT'(10X,'2) GEOMETRY PLOT WITH LOAD VECTORS SHOWN')
;1003 FORMAT'(10X,'3) DEFLECTED GEOMETRY PLOT - LINEAR ANALYSIS')
;1004 FORMAT'(10X,'4) DEFLECTED GEOMETRY PLOT - BUCKLING MODE')
;1005 FORMAT'(10X,'5) CONTOUR PLOT OF SKIN AVG LAMINATE LOADS OR STRAINS')
;1006 FORMAT'(10X,'6) CONTOUR PLOT OF SKIN PLY STRESS OR STRAIN')
;1007 FORMAT'(10X,'7) CONTOUR PLOT OF SKIN MARGIN OF SAFETY')
;1008 FORMAT'(10X,'8) CONTOUR PLOT OF STIFF. AVG LAMINATE LOADS OR STRAINS')
;1009 FORMAT'(10X,'9) CONTOUR PLOT OF STIFF. PLY STRESS OR STRAIN')
;1010 FORMAT'(9X,'10) CONTOUR PLOT OF STIFF. MARGIN OF SAFETY')
;1011 FORMAT'(9X,'11) SUMMARY TABLE OF AVG LAMINATE LOADS AND STRAINS')
;1012 FORMAT'(9X,'12) SUMMARY TABLE OF PLY STRESSES AND STRAINS')
;1013 FORMAT'(9X,'13) SUMMARY TABLE OF LAMINATE MARGINS OF SAFETY'/)
;1014 FORMAT'(9X,'99) TERMINATE POST PROCESSING SESSION'/)
WRITE 6 ;1000:WRITE 6 ;1001:WRITE 6 ;1002:WRITE 6 ;1003
WRITE 6 ;1004:WRITE 6 ;1005:WRITE 6 ;1006:WRITE 6 ;1007
WRITE 6 ;1008:WRITE 6 ;1009:WRITE 6 ;1010:WRITE 6 ;1011
WRITE 6 ;1012:WRITE 6 ;1013:WRITE 6 ;1014
RETURN
END
#
#MENU SELECTION SUBROUTINE
#
SUB SEL2 &NPL &IMN
IF &IMN-2 ;10 ;20 ;30
;10 CALL GE01 &NPL
GOTO ;500
;20 CALL GE02 &NPL
GOTO ;500
;30 IF &IMN-4 ;40 ;50 ;60
;40 CALL GE03 &NPL
GOTO ;500
;50 CALL GE04 &NPL
GOTO ;500
;60 IF &IMN-6 ;70 ;80 ;90
;70 CALL CON1 &NPL
GOTO ;500
;80 CALL CON2 &NPL
GOTO ;500
;90 IF &IMN-8 ;100 ;110 ;500
;100 CALL CON3 &NPL
```

```
GOTO ;500
;110 CALL CON4 &NPL
GOTO ;500
;500 CONTINUE
RETURN
END
#
#  SUBROUTINE FOR MENU SELECTION AND REDIRECTION
#
SUB SEL1 &IF1 &NPL
;5 CONTINUE
CALL MENU
QUERY &IMN "INPUT NUMBER OF MENU ITEM DESIRED >> "
CALL SEL2 &NPL &IMN
IF &IMN-8 ;5 ;5 ;120
;120 IF &IMN-10 ;130 ;140 ;150
;130 CALL CON5 &NPL
GOTO ;5
;140 CALL CON6 &NPL
GOTO ;5
;150 IF &IMN-12 ;160 ;170 ;180
;160 CALL SUM1 &IF1
GOTO ;5
;170 CALL SUM2 &IF1
GOTO ;5
;180 IF &IMN-14 ;190 ;200 ;200
;190 CALL SUM3 &IF1
;GOTO ;5
;200 CONTINUE
RETURN
END
#
# MAIN PROGRAM SUBROUTINE DRIVER
#
CALL SEL1 &IF1 &NPL
CALL HC &NPL
CLOSE 8
;1000 FORMAT'(///1X,'THIS COMPLETES THE POST PROCESSING SESSION.'/)
;1001 FORMAT'(1X,'ANY HARD COPY PLOTS GENERATED HERE WILL RESIDE ON')
;1002 FORMAT'(1X,'FILES IN THE WORKING DIRECTORY, USE NORMAL SUBMITTAL')
;1003 FORMAT'(1X,'PROCEDURES TO ACTUALLY PRINT THEM.'//)
WRITE 6 ;1000
WRITE 6 ;1001
WRITE 6 ;1002
WRITE 6 ;1003
STOP
$END:
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1992	Contractor (Report May 1989-May 1992)

4. TITLE AND SUBTITLE
Advanced Composites Structural Concepts and Materials Technologies for Primary Aircraft Structures - Structural Response and Failure Analysis - ISPAN Modules Users Manual

5. FUNDING NUMBERS
C NAS1-18888
WU 510-02-13-01

6. AUTHOR(S)
J.W. Hairr, J.T. Huang, J.E. Ingram, and B.M. Shah

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Lockheed Aeronautical Systems Company
86 South Cobb Drive
Marietta, GA 30063

8. PERFORMING ORGANIZATION REPORT NUMBER
LG92ER0036

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA CR-4449

11. SUPPLEMENTARY NOTES
Langley Technical Monitor: Randall C. Davis
Task 2 Users Manual

12a. DISTRIBUTION/AVAILABILITY STATEMENT
[redacted]
Subject Category 05

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The ISPAN Program is an interactive design tool that is intended to provide a means of performing simple and self contained preliminary analysis of aircraft primary structures made of composite materials. The program combines a series of modules with the finite element code DIAL as its backbone.
Four ISPAN Modules were developed and are documented. These include: 1) Flat stiffened panel; 2) Curved stiffened panel; 3) Flat tubular panel; and 4) Curved Geodesic Panel.
Users are instructed to input geometric and material properties, load information and types of analysis (Linear, Bifurcation buckling; or post-buckling) interactively. The program utilizing this information will generate finite element mesh and perform analysis. The output in the form of summary tables of stress or margins of safety, contour plots of loads or stress and deflected shape plots may be generated and used to evaluate specific design.

14. SUBJECT TERMS
Interactive design, composites, curved panels, stiffened panels, post buckling, finite element analysis, nonlinear analysis

15. NUMBER OF PAGES
390

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102