เอกสารอ้างอิง

1. B.M. Irons, "A Frontal Solution Program for Finite Element Analysis", International Journal For Numerical Methods in Engineering, Vol 2, page 5-32, 1970
2. C.P. Johnson, "A Frontal-Based Solver with Multi-Level Substructuring", Proceedings, North Carolina State University, May 23-25, page 381-384, 1977
3. A. Bykat, "A Note on An Element Ordering Scheme", International Journal For Numerical Methods in Engineering, Vol 11(1), page 194-198, 1977
4. Erik Thompson and Yoji Shimazaki, "A Frontal Procedure Using Skyline Storage", International Journal For Numerical Methods in Engineering, Vol 15, page 889-910, 1980
5. ภาณุวัฒน์ คุรุรัตน์, "โปรแกรมไมโครคอมพิวเตอร์ซึ่งใช้วิธีฟรอนทัลในการวิเคราะห์โครงสร้างชนิดโครงระนาบ", วิทยานิพนธ์วิศวกรรมศาสตร์มหาบัณฑิต, ภาควิชาวิศวกรรมโยธา จุฬาลงกรณ์มหาวิทยาลัย, 2527
6. Alan Jennings, "Matrix Computation for Engineers and Scientists", 1978
7. E.Hinton and D.R.J. Owen, "Finite Element Programming", 1977
8. R.D. Cook, "Concept and Applications of Finite Element Analysis", 2nd Edition, John Wiley & Sons, 1974
9. Roy E.Myers, "Microcomputer Graphics", Addison - Wesley, 1983
10. Chan S.Park, "Interactive Microcomputer Graphics", Addison - Wesley, 1985

11. G. Beer and W. HAAS, "A Partitioned Frontal Solver For Finite Element Analysis", International Journal For Numerical Methods in Engineering, Vol. 18, 1623-1654, 1982

12. ทักษิณ เทพชาตรี, "โปรแกรมไมโครคอมพิวเตอร์สำหรับวิเคราะห์โครงข้อหมุน 2 มิติ", เลขที่ คส 03/2526 ภาควิชาวิศวกรรมโยธา จุฬาลงกรณ์มหาวิทยาลัย

13. ทักษิณ เทพชาตรี, "โปรแกรมไมโครคอมพิวเตอร์สำหรับวิเคราะห์โครงข้อแข็ง 2 มิติ", เลขที่ คส 04/2526 ภาควิชาวิศวกรรมโยธา จุฬาลงกรณ์มหาวิทยาลัย

14. เริงเดชา รัชตโพธิ์, "โปรแกรมไมโครคอมพิวเตอร์สำหรับระบบไฟไนต์เอเลเมนต์โดยใช้วิธีโครงสร้างย่อย", รายงานวิจัย-คณะวิศวกรรมศาสตร์, 2526

15. นิวัฒน์ นิลสิทธานุเคราะห์, "แผนงานโครงสร้างย่อยแบบฟรอนทัลสำหรับวิเคราะห์ผนังต้านแรงเฉือนด้วยไมโครคอมพิวเตอร์", วิทยานิพนธ์วิศวกรรมศาสตร์มหาบัณฑิต ภาควิชาวิศวกรรมโยธา จุฬาลงกรณ์มหาวิทยาลัย, 2529

16. SAP4, "A STRUCTURAL ANALYSIS PROGRAM FOR STATIC AND DYNAMIC RESPONSE OF LINEAR SYSTEMS", K.J. BATHE, E.L. WILSON, F.E. PETERSON, UNIVERSITY OF CALIFORNIA, BERKELEY, IBM CONVERSION BY UNIVERSITY OF SOUTHERN CALIFORNIA, "AUGUST, 1973", "REVISED JULY, 1974"

ภาคผนวก ก

รายละเอียดโปรแกรมคอมพิวเตอร์

```
                    ***************************
                    *        REMARK           *
                    ***************************

====================
-:CONTROL PARAMETER:-
====================
NUMNP            : Total number of nodal points
NUMEL            : Total number of elements
NNPE             : Number of nodes per element
NPDOF            : Number of degrees of freedom per node
NDIME            : Number of coordinate dimensions
NMATS            : Total number of different materials
NTYPE            : Problem type parameter
                     1 = Plane stress
                     2 = Plane strain
                     3 = Plate bending
                     4 = Brick element
NCASE            : Total number of load cases to be analysed
NUMSEG           : Number of segments (computed by program)
NGAUSS           : Order of integration formula for numerical integration
IORDER           : Re-order elements numbering parameter
                     1 = by program
                     0 = as input

============================
-:SUBROUTINE INPUT,INPUT(2):-
============================
CO(NUMNP*NDIME)  : Coordinates of nodal point
NBCJ(NUMNP)      : Boundary condition parameter
                     1 = boundary conditions to be considered
                     0 = no boundary condition
KODE(NUMNP,NPDOF) : Condition of restraint on displacement
                     0 = no displacement  restraint
                     1 = nodal displacement restrained
NP(NUMEL,NNPE)   : Nodal connection numbers
MATNO(NUMEL)     : Material property number
PROPS(NMATS,I)   : Material properties
                     I = 1 Elastic modulus
                     I = 2 Poisson's ratio
                     I = 3 Material thickness
                     I = 4 Mass density
                     I = 5 Coefficient of thermal expansion

C$               : Title of the load case
IPLOD            : Applied point load control parameter
                     0 = no applied nodal loads to be input
                     1 = applied nodal loads to be input
IGRAV            : Gravity loading control parameter
                     0 = no gravity loads to be considered
                     1 = gravity loads to be considered
IEDGE            : Pressure load control parameter
                     0 = no pressure loads to be input
                     1 = pressure loads to be input
ITEMP            : Thermal loading control parameter
                     0 = no thermal loading to be considered
```

```
                    1 = thermal loading to be considered
IULOD             : Uniform load control parameter (for plate bending)
                    0 = no uniform loads to be considered
                    1 = uniform loads to be considered

NLOAD(NUMNP)      : Node number
                    0 = no nodal loads to be considered at that node
                    1 = nodal loads to be considered at that node
PLOAD(NUMNP,1)    : Load component in x direction
PLOAD(NUMNP,2)    : Load component in y direction
PLOAD(NUMNP,3)    : Load component in z direction

NEDGE             : Number of element edges on which distributed loads are
                    to be applied
NEASS(NEDGE)      : The element number with which the element edge is
                    associated
NSIDE(NEDGE)      : The edge number on which distributed loads are to be
                    applied
PRESS(NTOT)       :
PRESS((I-1)*NPRESS*2+J)
NPRESS = Number of node per side
                  Value of normal component of distributed load at node
                  Value of tangential component of distributed load at node

TEMP(NUMNP)       : Temperature at node
                    (Datum temperature is taken to be zero)
ULOAD(NUMEL)      : Value of uniform load (for plate bending)

=========================================================
              -: SUBPROGRAM ORDER :-
This subprogram computes number degree of freedom at nodes
& re-order elements numbering
=========================================================
NPDOF(NUMNP)      : Number of degree of freedom at that node
IORDER(NUMEL)     : Element number
SIDE(NUMEL,6)     : Element side number for side connectivity

=========================================
          -:SUBPROGRAM WAVE:-
This subprogram performs destination arrays
        for front processing
=========================================
IDA       : Total number of nodal points
IDB       : Total number of elements
IDC       : Number of nodal points per element
IDD       : Maximum number of equations in any disk segment
IDE       : Maximum number of new elements added in any disk disk segment
IDF       : Maximum number of nodal points represented in any disk segment

NUMNP     : Total number of nodal points.
NUMEL     : Total number of elements.
NNPE      : Number of nodal points per element.
NUMSEG    : Number of segments,or WAVE-records.

NP(I,J)   : The Jth nodal point number of element I.
IORDER(I) : The order that the elements will be assembled.
```

```
MAXVOL     : The storage that will be available for the stiffness matrix.

ISEG       : The current segment (or record) number
IELEX      : The number of new elements to be assembled for current segment
NEQ        : The number of equations in the stiffness matrix for current
             segment
ICOMP      : The number of fully assembled (or completed) equations in
             current segment
MOVEX      : The number of equations from the previous segment that need to
             be moved to new locations of disk for current segment
JUSTFY     : is a scalar that is used when,during the rearrangement of the
             equations remaining from the previous segment,it is not possible
             to leave the stiffness matrix right justified. The amount the
             matrix must then be shifted to right justify the matrix is
             indicated by this variable
IDIAG1(I)  : are the addresses of the diagonal terms for the previous segment
and          and the current segment, respectively
IDIAG2(I)
MOVE(I)    : The new column and row which the previous Ith column and Ith row
             will occupy in the current segment.
IEQS(I)    : The global equation number of the Ith equation in current
             segment. When there is only one degree-of-freedom per node,this
             corresponds to the nodal point number represented by the Ith
             equation in the current segment.
NPR(I)     : The location in the list of equations for the current segment of
             the first degree-of-freedom associated with the Ith nodal point.
               > 0 indicates the location
               = 0 indicates those nodes not yet reached by the front
               = -1 indicates those nodes for which the front has passed
IELE(I)    : The number of the Ith element to be assembled for the current
             segment.
NPT(I,J)   : The Jth nodal point number of the Ith element to be assembled
             for the current segment. This represents that part of the
             NP-array needed for the current segment. If the entire NP-array
             is retained in the main program, this array should be deleted.

LSTMAX     : Maximum nodes repersented in segment
NEQMAX     : Maximum equations repersented in segment
KMAX       : Maximum usage storage

================================================
          -: SUBPROGRAM STIFF :-
This subprogram performs element stiffness matrix
          for segment number ISEG
     save data into file name "GST.ISEG"
================================================
NGAUSS       : Order of integration formula for numerical integration
ELCOD(I,J)   : Nodal coordinates of current element
WGT(I,J)     : Array stores weights of gauss quadrature
PLACE(I,J)   : Array stores locations of gauss quadrature
NSTRE        : Number of independent stress components
DMATX(I,J)   : The elasticity matrix D
SHAPE(I)     : Array stores shape functions of isoparametric element
DERIV(I,J)   : Array stores derivative of shape function
XJACM(I,J)   : Array stores Jacobian matrix
DJACB        : Determinant of Jacobian matrix
```

```
XJACI(I,J)   : Array stores inverse of Jacobian matrix
CARTD(I,J)   : Cartesian shape function derivatives
BMATX(I,J)   : The element strain matrix B
DBMAT(I,J)   : The product of DB matrix
ESTIF(I,J)   : Element stiffness matrix


=========================================
        -: SUBPROGRAM LFORM,LB :-
         (LB for brick element)
   This subprogram performs load vector
   save data into file name "LOAD.ICASE"
   where ICASE = load case no.
=========================================
ST(I)        : Array stores load vector of elements
STRAN(I)     : Array stores initial strains
STRES(I)     : Array stores initial stresses


=================================================
-: SUBPROGRAM SOLVER :-
    This subprogram is a linear equations solver
    for 1-d symmetric skyline storage by
    "GAUSS ELIMINATION METHOD"
    solution for X of equation
              AX = B
    store X in B
=================================================
LNEQ(NEQMAX) : Max. equation no. shared with current equation
SK(MAXVOL)   : Active coefficient of stiffness matrix or left side A
B(NEQMAX)    : Load vector or right side B (or X)
ST(24, 24)   : Element stiffness matrix
SL(24)       : Element load vector
I1DOF(8)     : Location of first dof. at that node
```

```
'*************************************************************************
'*                    This is the FIRST program                          *
'*                       File name "HELLO"                               *
'*************************************************************************

ON ERROR GOTO 1000

DEFINT I-N
DEFDBL A-H, O-Z
DIM A!(40), A$(36)

DEF SEG = VARSEG(A!(0))
BLOAD "MEMO.DAT", VARPTR(A!(0))
FOR I = 1 TO 36 STEP 2: A$(I) = CHR$(A!(I) - 1): NEXT I
FOR I = 2 TO 36 STEP 2: A$(I) = CHR$(A!(I) + 1): NEXT I
N$ = "": FOR I = 1 TO 36: N$ = N$ + A$(I): NEXT I
D$ = CHR$(A!(40))

L$ = STRING$(60, "=")

CLS : SCREEN 0: WIDTH 80: KEY OFF

GOTO 120

'<---------------------- SUBROUTINE AREA ---------------------------->
100 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
110 LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

'-------------------------------------------------------------------------
120 CLS : GOSUB 100: GOSUB 110
    LOCATE 4, 27: COLOR 11: PRINT "S U P A T F E A P - I I I"
    PRINT
    COLOR 12: PRINT TAB(25); "AUTHORITY : "; N$
    COLOR 15: PRINT TAB(10); L$
    PRINT
    COLOR 14: PRINT TAB(27); "<<< U S E R   M E N U >>>"
    PRINT : PRINT
    PRINT TAB(25); "S  =  START SUPATFEAP III"
    PRINT TAB(25); "H  =  HARDWARE CONFIGURATION"
    COLOR 13: PRINT TAB(25); "E  =  EXIT TO SYSTEM"
    PRINT
    COLOR 15: PRINT TAB(10); L$
    PRINT
    COLOR 11: PRINT TAB(25); "====> SELECT ? ";

130 K$ = INKEY$: LOCATE 18, 40, 1, 0, 7: IF K$ = "" THEN 130
    IF ASC(K$) = 0 THEN PLAY "L40 8": GOTO 130
    COLOR 11: LOCATE 18, 40: PRINT K$;

    IF K$ = "S" OR K$ = "s" THEN RUN "HEAD"
    IF K$ = "H" OR K$ = "h" THEN 140
    IF K$ = "E" OR K$ = "e" THEN CLS : SYSTEM
    PLAY "L40 8": GOTO 130

    '<-------------------- HARDWAVE CONFIGURATION ---------------------->
140 CLS : GOSUB 100: GOSUB 110
```

```
    COLOR 15: LOCATE 3, 28: PRINT "HARDWARE CONFIGURATION"
    COLOR 12: LOCATE 4, 24: PRINT "SPECIFY DATA DISK DRIVE (A/B/C)"
    COLOR 15: PRINT TAB(10); L$
    LOCATE 10, 10: PRINT L$.
    COLOR 13: PRINT TAB(18); "Enter a character to specify data disk drive"
    LOCATE 7, 25: COLOR 11: PRINT TAB(25); "CURRENT DRIVE : "; : COLOR 12: PRINT D$
    LOCATE 8, 25: COLOR 11: PRINT TAB(25); "NEW DRIVE     : ";
    COLOR 12

150 K$ = INKEY$: IF K$ = "" THEN 150
    IF K$ = "A" OR K$ = "a" THEN K$ = "A": GOTO 160
    IF K$ = "B" OR K$ = "b" THEN K$ = "B": GOTO 160
    IF K$ = "C" OR K$ = "c" THEN K$ = "C": GOTO 160
    IF K$ = CHR$(13) THEN K$ = D$: GOTO 160
    PLAY "L40 B": GOTO 150
160 LOCATE 8, 41: PRINT K$

    COLOR 14: LOCATE 24, 15: PRINT "<Return> = Accept     R = Reenter"; : COLOR 7
170 P$ = INKEY$: IF P$ = "" THEN 170
    IF P$ = "R" OR P$ = "r" THEN LOCATE 24, 15: PRINT SPC(35); : GOTO 140
    IF P$ = CHR$(13) THEN LOCATE 24, 15: PRINT SPC(35); : GOTO 180
    BEEP: GOTO 170

180 IF D$ = K$ THEN 190
    A!(40) = ASC(K$)
    DEF SEG = VARSEG(A!(0))
    BSAVE "MEMO.DAT", VARPTR(A!(0)), VARPTR(A!(40)) - VARPTR(A!(0)) + 4
190 D$ = K$: GOTO 120

    '<------------------------ ERROR DATA ------------------------------>
1000 CLS
    COLOR 12: LOCATE 10, 1: PRINT "INSERT PROGRAM DISK IN DRIVE A:"
    PRINT "Press any key to continue"
1010 P$ = INKEY$: IF P$ = "" THEN 1010
    RESUME
```

```
'*************************************************************************
'*                                                                       *
'*              HEAD PROGRAM (called after HELLO)                        *
'*                      file name "HEAD"                                 *
'*                                                                       *
'*************************************************************************

DEFINT I-N
DEFDBL A-H, O-Z

COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
COMMON NCASE, NUMSEG, NGAUSS, IORDER, T$, E$, MAXVOL, F$()
COMMON IOPTION
COMMON LSTMAX, NEQMAX, KMAX
COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
'      Maximum nodes repersented in segment
'      Maximum equations repersented in segment
'      Maximum usage storage

VIEW PRINT 1 TO 25
ON ERROR GOTO 1000
IF LEN(F$) > 0 THEN 130

DIM A!(40), A$(36), F$(16)

DEF SEG = VARSEG(A!(0))
BLOAD "MEMO.DAT", VARPTR(A!(0))
FOR I = 1 TO 36 STEP 2: A$(I) = CHR$(A!(I) - 1): NEXT I
FOR I = 2 TO 36 STEP 2: A$(I) = CHR$(A!(I) + 1): NEXT I
N$ = "": FOR I = 1 TO 36: N$ = N$ + A$(I): NEXT I
D$ = CHR$(A!(40)) + ":"

100  OPEN "I", 1, D$ + "DIRECTOR": INPUT #1, F$: CLOSE 1

     OPEN "I", 1, D$ + F$ + ".CON"
     INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
     INPUT #1, NCASE
     INPUT #1, NUMSEG, NGAUSS, IORDER
     INPUT #1, T$
     INPUT #1, E$
     CLOSE 1

     GOTO 130

     '<---------------------- SUBROUTINE AREA ---------------------------->
110  LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
120  LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

     '-------------------------------------------------------------------
130  CLS : GOSUB 110: GOSUB 120
     LOCATE 2, 20: COLOR 11: PRINT "S U P A T F E A P - I I I": PRINT
     COLOR 12: PRINT TAB(12); "AUTHORITY : "; N$
     COLOR 15: PRINT TAB(12); "CURRENT PROJECT MASTER FILE NAME ==> "; F$
     PRINT
     COLOR 14: PRINT TAB(5); "ACTIVITY MENU :     D  = DATA MODE"
     PRINT TAB(5); "================     S  = SOLUTION MODE"
```

```
      PRINT TAB(25); "R  = RESULT MODE"
      PRINT TAB(25); "G  = GRAPHIC MODE"
      PRINT
      COLOR 13: PRINT TAB(25); "Q -> QUIT TO USER MENU"
      PRINT
      PRINT TAB(25); "E -> EXIT TO SYSTEM"
      LOCATE 20, 25: COLOR 11: PRINT "====> SELECT ?";

140   K$ = INKEY$: LOCATE 20, 40, 1, 0, 7: IF K$ = "" THEN 140
      IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 140
      COLOR 27: LOCATE 20, 40: PRINT K$;

      IF K$ = "D" OR K$ = "d" THEN RUN "MAININP"
      IF K$ = "Q" OR K$ = "q" THEN RUN "HELLO"
      IF K$ = "E" OR K$ = "e" THEN 250
      IF F$ = "UNDEFINED" THEN 150
      IF K$ = "S" OR K$ = "s" THEN 160
      IF K$ = "R" OR K$ = "r" THEN 230
      IF K$ = "G" OR K$ = "g" THEN CHAIN "PLOT"
150   PLAY "L40 B": GOTO 140

      '<---------------------- SOLUTION MODE ----------------------------->
160   MAXVOL = 7000

170   IF IORDER = 1 THEN ORDER$ = "Re-order of element by program"
      IF IORDER = 0 THEN ORDER$ = "Order of element as input"
      IF NGAUSS = 0 THEN NGAUSS = 2

      CLS : GOSUB 110: GOSUB 120
      LOCATE 5, 5: COLOR 11: PRINT "< < <  SOLUTION MODE  > > >"
      PRINT : COLOR 14
      PRINT TAB(5); "N = Number of gauss point in each direction ("; NGAUSS; ")"
      PRINT TAB(5); "O = "; ORDER$
      PRINT TAB(5); "E = EXECUTION"
      PRINT TAB(5); "M = MAX. VOLUME OF CORE STORAGE ("; MAXVOL; ")"
      PRINT : COLOR 12
      PRINT TAB(5); "Q = Quit to ACTIVITY MENU "
      LOCATE 20, 5: COLOR 11: PRINT "====> SELECT ?"

180   K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 180
      IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 180
      COLOR 27: LOCATE 20, 20: PRINT K$;
      IF K$ = "N" OR K$ = "n" THEN 190
      IF K$ = "O" OR K$ = "o" THEN 200
      IF K$ = "M" OR K$ = "m" THEN 210
      IF K$ = "E" OR K$ = "e" THEN 220
      IF K$ = "Q" OR K$ = "q" THEN 130
      PLAY "L40 B": GOTO 180

190   IF NGAUSS = 2 THEN NGAUSS = 3 ELSE NGAUSS = 2
      GOTO 170

200   IF IORDER = 1 THEN IORDER = 0 ELSE IORDER = 1
      GOTO 170

210   COLOR 15: LOCATE 20, 5
```

```
    INPUT "INPUT MAXIMUM VOLUME OF CORE STORAGE = ", MAXVOL
    GOTO 170

220 LOCATE 15, 1: COLOR 15: PRINT STRING$(80, "-")
    COLOR 13
    PRINT "    C = COMPLETE SOLUTION"
    PRINT "    W = UP TO WAVE DATA CALCULATION"
    COLOR 15: PRINT STRING$(80, "-")
225 K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 225
    COLOR 11: LOCATE 20, 20: PRINT K$;
    IF K$ = "C" OR K$ = "c" THEN IOPTION = 0: CHAIN "ORDER"
    IF K$ = "W" OR K$ = "w" THEN IOPTION = 1: CHAIN "ORDER"
    GOTO 170

    '<------------------------ RESULT MODE ------------------------------>
230 CLS : GOSUB 110: GOSUB 120
    COLOR 11: PRINT "<<< RESULT MODE >>>"
    PRINT : COLOR 14
    PRINT "D = DISPLACEMENTS"
    PRINT "S = STRESSES"
    PRINT : COLOR 13
    PRINT "Q -> QUIT TO ACTIVITY MENU"
    PRINT
    PRINT "====> SELECT ?"

240 K$ = INKEY$: LOCATE 10, 16, 1, 0, 7: IF K$ = "" THEN 240
    IF ASC(K$) = 0 THEN PLAY "L40 8": GOTO 240
    LOCATE 10, 16: PRINT K$;

    IF K$ = "D" OR K$ = "d" THEN CHAIN "PRTDIS"
    IF K$ = "S" OR K$ = "s" THEN 245
    IF K$ = "Q" OR K$ = "q" THEN 130
    PLAY "L40 8": GOTO 240

245 OPEN "I", 1, D$ + F$ + ".CON"
    INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    INPUT #1, NCASE
    INPUT #1, NUMSEG, NGAUSS, IORDER
    INPUT #1, T$
    INPUT #1, E$
    CLOSE 1
    IF NTYPE = 4 THEN RUN "STRESS-8"
    RUN "STRESS"

    '<--------------------------- EXIT ---------------------------------->
250 IF F$ = "UNDEFINED" THEN COLOR 7, 0: SYSTEM

    OPEN "I", 1, D$ + F$ + ".DIR"
    FOR I = 1 TO 16: IF NOT EOF(1) THEN INPUT #1, F$(I)
    NEXT I
    CLOSE 1

    CLS : GOSUB 110: GOSUB 120
    LOCATE 1, 1: COLOR 14
    PRINT " MASTER FILENAME : "; F$
    PRINT
```

```
PRINT TAB(11); "LATEST TIMING REPORT OF THE CURRENT PROJECT"
PRINT : COLOR 13
PRINT " DATE "; F$(11)
COLOR 11
PRINT " MODULE       ELAPSING"
PRINT STRING$(68, "="): COLOR 15
PRINT " INPUT"
PRINT " WAVE "
PRINT " STIFFNESS"
PRINT " FORCES"
PRINT " SOLVER"
LOCATE 10, 1

FOR I = 1 TO 5: II = I + 11
GOSUB 320
LOCATE I + 7, 15: PRINT TI$
NEXT I
PRINT
COLOR 12
PRINT " Number of core storage ..................... ="; MAXVOL
PRINT " Number of segments ......................... ="; NUMSEG
PRINT " Maximum nodes repersented in segment ....... ="; LSTMAX
PRINT " Maximum equations repersented in segment .... ="; NEQMAX
PRINT " Maximum usage storage ...................... ="; KMAX
PRINT " I/O on WAVE ................................ ="; WAVECOUNT
PRINT " I/O on STIFF ............................... ="; STIFFCOUNT
PRINT " I/O on SOLVER ......................... ..... ="; SKCOUNT

COLOR 14: PRINT
PRINT " SELECT      H  =  HARD COPY"
PRINT "             E  =  EXIT TO SYSTEM ";
PLAY "L40 BGBGBGBGBGBG"

280  K$ = INKEY$: IF K$ = "" THEN 280
     IF K$ = "E" OR K$ = "e" THEN COLOR 7, 0: SYSTEM
     IF K$ = "H" OR K$ = "h" THEN 290
     IF K$ = CHR$(32) THEN 130
     PLAY "L40 B": GOTO 280

290  WIDTH "LPT1:", 255
     LPRINT CHR$(15)
     LPRINT " MASTER FILENAME : "; F$
     LPRINT
     LPRINT TAB(11); "LATEST TIMING REPORT OF THE CURRENT PROJECT"
     LPRINT TAB(11); "DATE "; F$(11)
     LPRINT " MODULE          ELAPSING                I/O"
     LPRINT "-------------------------------------------"
     II = 12: GOSUB 320
     LPRINT "INPUT    "; TAB(18); TI$
     II = 13: GOSUB 320
     LPRINT "WAVE     "; TAB(18); TI$; TAB(40); USING "######"; WAVECOUNT
     II = 14: GOSUB 320
     LPRINT "STIFFNESS"; TAB(18); TI$; TAB(40); USING "######"; STIFFCOUNT
     II = 15: GOSUB 320
     LPRINT "FORCES   "; TAB(18); TI$; TAB(40); USING "######"; FORCECOUNT
     II = 16: GOSUB 320
```

```
    LPRINT "SOLVER   "; TAB(18); TI$; TAB(40); USING "######"; SKCOUNT
    LPRINT "-------------------------------------------"
    LPRINT " MAXVOL   NUMSEG   LSTMAX   NEQMAX     KMAX"
    LPRINT USING " ######      ###     ####    #####   ######"; MAXVOL; NUMSEG; LSTMAX; NEQMAX; KMAX
    COLOR 7, 0: SYSTEM
    END

320 IF LEN(F$(II)) = 0 THEN TI$ = "   --   ": RETURN
    TOTALtime! = VAL(F$(II))
    NM = INT(TOTALtime! / 60)
    NS = TOTALtime! - NM * 60
    NH = INT(NM / 60)
    NM = NM - NH * 60
    TI$ = ""
    H$ = RIGHT$(STR$(NH), LEN(STR$(NH)) - 1): IF LEN(H$) = 1 THEN H$ = "0" + H$
    M$ = RIGHT$(STR$(NM), LEN(STR$(NM)) - 1): IF LEN(M$) = 1 THEN M$ = "0" + M$
    S$ = RIGHT$(STR$(NS), LEN(STR$(NS)) - 1): IF LEN(S$) = 1 THEN S$ = "0" + S$
    TI$ = H$ + ":" + M$ + ":" + S$
    RETURN

    '<---------------------- RESUME ERROR AREA ---------------------------->
    '-----> ERR = 71                                          'Disk not ready
1000 IF ERR <> 71 THEN 1200
    CLS : COLOR 12: LOCATE 12, 12: PRINT "DISK NOT READY"
    LOCATE 13, 12: PRINT "Press any key to continue": COLOR 7
1100 K$ = INKEY$: IF K$ = "" THEN 1100
    CLS : RESUME

    '-----> ERR = 53                                          'File not found
1200 IF ERR <> 53 THEN 1400
    IF ERL = 100 THEN F$ = "UNDEFINED": RESUME 130

    CLS : COLOR 12: LOCATE 10, 12: PRINT "Insert program disk in drive A:"
    LOCATE 11, 12: PRINT "Press any key to continue"
1300 K$ = INKEY$: IF K$ = "" THEN 1300
    CLS
    RESUME

    '-----> Other error
1400 CLS : COLOR 12: LOCATE 15, 5: PRINT "PLEASE CHECK YOUR DEVICE"
    PRINT TAB(5); "Press any key to continue"
1410 K$ = INKEY$: IF K$ = "" THEN 1410
    RESUME
```

```
'*************************************************************************
'*                                                                       *
'*              MAIN INPUT PROGRAM (called after HEAD)                   *
'*                     file name "MAININP"                               *
'*                                                                       *
'*************************************************************************

ON ERROR GOTO 1000

CLS : SCREEN 0: WIDTH 80: KEY OFF
DEFINT I-N
DEFDBL A-H, O-Z

DIM A!(40), A$(36), F$(10)
DEF SEG = VARSEG(A!(0))

BLOAD "MEMO.DAT", VARPTR(A!(0))
FOR I = 1 TO 36 STEP 2: A$(I) = CHR$(A!(I) - 1): NEXT I
FOR I = 2 TO 36 STEP 2: A$(I) = CHR$(A!(I) + 1): NEXT I
N$ = "": FOR I = 1 TO 36: N$ = N$ + A$(I): NEXT I
D$ = CHR$(A!(40)) + ":"

100 OPEN "I", 1, D$ + "DIRECTOR"
    INPUT #1, F$
    CLOSE 1

110 L$ = STRING$(79, "=")
    GOTO 170

    '<---------------------- SUBROUTINE AREA ---------------------------->
120 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
130 LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

140 COLOR 11: LOCATE 23, 12: PRINT "<Return> = Accept    R = Re-enter";
150 K$ = INKEY$: IF K$ = "" THEN 150
    IF K$ = CHR$(13) OR K$ = "r" OR K$ = "R" THEN 160 ELSE BEEP: GOTO 150
160 LOCATE 23, 12: PRINT SPC(35); : RETURN

    '-------------------------------------------------------------------
170 CLS : GOSUB 120: GOSUB 130
    LOCATE 1, 10: COLOR 11: PRINT STRING$(54, "*")
    PRINT TAB(10); "*              D A T A     M O D E          *"
    PRINT TAB(10); "*                                           *"
    PRINT TAB(10); "*  PROGRAM MODULE-1  [ISOPARAMETRIC FINITE ELEMENT]  *"
    PRINT TAB(10); STRING$(54, "*")
    PRINT : PRINT
    COLOR 15: PRINT TAB(5); "CURRENT PROJECT MASTER FILENAME ==> "; F$
    PRINT : PRINT : COLOR 14
    PRINT TAB(5); "OPTIONS :      C  = CONTINUE"
    PRINT TAB(20); "N  = NEW PROJECT"
    PRINT TAB(20); "E  = OTHER EXISTING PROJECT"
    PRINT
    COLOR 13: PRINT TAB(20); "Q -> QUIT TO ACTIVITY MENU"
    PRINT
    LOCATE 20, 20: COLOR 11: PRINT "====> SELECT ?";
```

```
180  K$ = INKEY$: LOCATE 20, 35, 1, 0, 7: IF K$ = "" THEN 180
     IF ASC(K$) = 0 THEN PLAY "L40 8": GOTO 180
     COLOR 27: LOCATE 20, 35: PRINT K$

     IF K$ = "N" OR K$ = "n" THEN 200
     IF K$ = "E" OR K$ = "e" THEN 250
     IF K$ = "Q" OR K$ = "q" THEN RUN "A:HEAD"
     IF F$ = "UNDEFINED" THEN 190
     IF K$ = "C" OR K$ = "c" THEN 270
190  PLAY "L40 8": GOTO 180

     '<------------------------ NEW PROJECT ------------------------------>
200  F$ = "": T$ = "": E$ = ""
     CLS : GOSUB 120: GOSUB 130: COLOR 15
     LOCATE 4, 1: PRINT L$
     COLOR 14: PRINT "MASTER FILENAME : "
     PRINT
     PRINT "PROJECT TITLE   : "
     PRINT
     PRINT "ENGINEER        : "
     PRINT
     COLOR 12: PRINT "AUTHORITY       : "; N$
     COLOR 15: PRINT L$;
     LOCATE 20, 1: PRINT STRING$(80, "-");
     PRINT "MAX. NO.OF CHARACTERS FOR :"
     PRINT
     PRINT "   Filename = 8,   Title = 40,   Engineer = 20"

     LOCATE 5, 19, 1, 0, 7: INPUT "", F$: IF LEN(F$) = 0 THEN BEEP: GOTO 100
     LOCATE 7, 19, 1, 0, 7: INPUT "", T$
     LOCATE 9, 19, 1, 0, 7: INPUT "", E$

     LOCATE 17, 5: COLOR 11: PRINT "<Return> = Accept above data      R = Reenter"

210  K$ = INKEY$: IF K$ = "" THEN 210
     IF K$ = "R" OR K$ = "r" THEN 200
     NUMNP = 0: NUMEL = 0: NNPE = 0: NPDOF = 0: NDIME = 0: NMATS = 0: NTYPE = 0
     NCASE = 0: NUMSEG = 0: NGAUSS = 0: IORDER = 0

220  CLS : GOSUB 120: GOSUB 130
     COLOR 11: LOCATE 1, 1: PRINT "<<< SELECT STRUCTURE TYPE >>>"
     COLOR 14: PRINT
     PRINT "1...PLANE STRESS ELEMENT  (4 NODE)"
     PRINT "2...                      (8 NODE)"
     PRINT "3...PLANE STRAIN ELEMENT  (4 NODE)"
     PRINT "4...                      (8 NODE)"
     PRINT "5...PLATE BENDING ELEMENT (4 NODE)"
     PRINT "6...                      (8 NODE)"
     PRINT "7...BRICK ELEMENT ELEMENT (8 NODE)"
     PRINT
     COLOR 13: PRINT "====> SELECT ?"

230  K$ = INKEY$: LOCATE 11, 16, 1, 0, 7: IF K$ = "" THEN 230
     LOCATE 11, 16, 1, 0, 7: PRINT K$;
     IF K$ = "1" THEN NNPE = 4: NPDOF = 2: NDIME = 2: NTYPE = 1: GOTO 240
     IF K$ = "2" THEN NNPE = 8: NPDOF = 2: NDIME = 2: NTYPE = 1: GOTO 240
```

```
      IF K$ = "3" THEN NNPE = 4: NPDOF = 2: NDIME = 2: NTYPE = 2: GOTO 240
      IF K$ = "4" THEN NNPE = 8: NPDOF = 2: NDIME = 2: NTYPE = 2: GOTO 240
      IF K$ = "5" THEN NNPE = 4: NPDOF = 3: NDIME = 2: NTYPE = 3: GOTO 240
      IF K$ = "6" THEN NNPE = 8: NPDOF = 3: NDIME = 2: NTYPE = 3: GOTO 240
      IF K$ = "7" THEN NNPE = 8: NPDOF = 3: NDIME = 3: NTYPE = 4: GOTO 240
      BEEP: GOTO 230

240   PRINT : PRINT : COLOR 15
      PRINT "NO.OF NODAL POINTS PER ELEMENT ..... = "; NNPE
      PRINT "NO.OF DEGREE OF FREEDOM PER NODE .... = "; NPDOF
      PRINT "NO.OF COORDINATE OF NODAL POINT ..... = "; NDIME

      GOSUB 140
      IF K$ <> CHR$(13) THEN 220

      FOR I = 1 TO 10: F$(I) = "": NEXT I
      F$(1) = F$ + ".CON"
      OPEN "O", 1, D$ + F$ + ".DIR"
      FOR I = 1 TO 10: WRITE #1, F$(I): NEXT I
      CLOSE 1

      OPEN "O", 1, D$ + F$ + ".CON"
      WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
      WRITE #1, NCASE
      WRITE #1, NUMSEG, NGAUSS, IORDER
      WRITE #1, T$
      WRITE #1, E$
      CLOSE 1

      OPEN "O", 1, D$ + "DIRECTOR"
      WRITE #1, F$
      CLOSE 1

      RUN "INPUT"

      '<-------------------- OTHER EXISTING PROJECT ------------------------>
250   CLS : GOSUB 120: GOSUB 130: COLOR 15
      LOCATE 3, 1: COLOR 14: PRINT "OTHER EXISTING FILES"
      PRINT
      PRINT L$
260   FILES D$ + "*.DIR"
      LOCATE 4, 1: COLOR 12: PRINT "SELECT---> ";
      INPUT "", F2$
      IF LEN(F2$) = 0 THEN 170
      GOSUB 140
      IF K$ <> CHR$(13) THEN 250

      F$ = F2$
      OPEN "I", 1, D$ + F$ + ".CON"
      INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
      INPUT #1, NCASE
      INPUT #1, NUMSEG, NGAUSS, IORDER
      INPUT #1, T$
      INPUT #1, E$
      CLOSE 1
```

```
      GOTO 280

      '<---------------------- CONTINUE PROJECT ---------------------------->
270   OPEN "I", 1, D$ + F$ + ".CON"
      INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
      INPUT #1, NCASE
      INPUT #1, NUMSEG, NGAUSS, IORDER
      INPUT #1, T$
      INPUT #1, E$
      CLOSE 1

280   CLS : GOSUB 120: GOSUB 130: COLOR 15
      LOCATE 4, 1: PRINT L$;
      COLOR 14: PRINT "MASTER FILENAME : "; F$
      PRINT
      PRINT "PROJECT TITLE   : "; T$
      PRINT
      PRINT "ENGINEER        : "; E$
      PRINT
      COLOR 12: PRINT "AUTHORITY       : "; N$
      COLOR 15: PRINT L$;
      LOCATE 20, 1: PRINT STRING$(80, "-");
      PRINT "MAX. NO.OF CHARACTERS FOR :"
      PRINT
      PRINT "   Filename = 8,   Title = 40,   Engineer = 20"

290   LOCATE 17, 5: COLOR 11: PRINT "<Return> = Accept above data    R = Reenter";
300   K$ = INKEY$: IF K$ = "" THEN 300
      IF K$ = "R" OR K$ = "r" THEN LOCATE 17, 5: PRINT SPC(50); : GOTO 310
      IF K$ = CHR$(13) THEN 320 ELSE BEEP: GOTO 300

310   LOCATE 7, 19, 1, 0, 7: PRINT SPC(40);
      LOCATE 7, 19, 1, 0, 7: INPUT "", T$
      LOCATE 9, 19, 1, 0, 7: PRINT SPC(40);
      LOCATE 9, 19, 1, 0, 7: INPUT "", E$

      GOTO 290

320   OPEN "O", 1, D$ + "DIRECTOR"
      WRITE #1, F$
      CLOSE 1

      OPEN "O", 1, D$ + F$ + ".CON"
      WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
      WRITE #1, NCASE
      WRITE #1, NUMSEG, NGAUSS, IORDER
      WRITE #1, T$
      WRITE #1, E$
      CLOSE 1
      RUN "INPUT"

      '<------------------------ ERROR CHECK ------------------------------->
1000  IF ERR = 71 THEN 1400

      '                    FILE NOT FOUND
```

```
    IF ERL = 100 THEN F$ = "UNDEFINED": RESUME 110

    IF ERL <> 260 THEN 1200
    COLOR 12: LOCATE 7, 1: PRINT "NO EXISTING FILES"
    COLOR 11: PRINT "Press any key to continue"
1100 K$ = INKEY$: IF K$ = "" THEN 1100
    RESUME 170

1200 CLS : COLOR 12: LOCATE 10, 10: PRINT "INSERT PROGRAM DISK IN DRIVE A:"
    LOCATE 11, 10: PRINT "Press any key to continue"
1300 K$ = INKEY$: IF K$ = "" THEN 1300
    RESUME

    '                    DISK NOT READY

1400 CLS : COLOR 12: LOCATE 10, 10: PRINT "DISK NOT READY"
    LOCATE 11, 10: PRINT "Press any key to continue"
1500 K$ = INKEY$: IF K$ = "" THEN 1500
    IF ERL = 260 THEN RESUME 170
    RESUME
```

```
DECLARE SUB FIRST ()
DECLARE SUB HEADING ()
     '*************************************************************************
     '*                                                                       *
     '*                      file name "INPUT"                                *
     '*                         INPUT DATA                                    *
     '*                                                                       *
     '*************************************************************************

     DEFINT I-N
     DEFDBL A-H, O-Z

     DIM A!(40), A$(36), F$(25)

     DEF SEG = VARSEG(A!(0))
     BLOAD "MEMO.DAT", VARPTR(A!(0))
     FOR I = 1 TO 36 STEP 2: A$(I) = CHR$(A!(I) - 1): NEXT I
     FOR I = 2 TO 36 STEP 2: A$(I) = CHR$(A!(I) + 1): NEXT I
     N$ = "": FOR I = 1 TO 36: N$ = N$ + A$(I): NEXT I
     D$ = CHR$(A!(40)) + ":"

     OPEN "I", 1, D$ + "DIRECTOR": INPUT #1, F$: CLOSE 1

     OPEN "I", 1, D$ + F$ + ".DIR"
     FOR I = 1 TO 12: IF NOT EOF(1) THEN INPUT #1, F$(I)
     NEXT I
     CLOSE 1

     OPEN "I", 1, D$ + F$ + ".CON"
     INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
     INPUT #1, NCASE
     INPUT #1, NUMSEG, NGAUSS, IORDER
     INPUT #1, T$
     INPUT #1, E$
     CLOSE 1

     DIM CO(NUMNP * NDIME), HCO(5), D(3)
     DIM NBCJ(NUMNP), KODE(NUMNP, NPDOF)
     DIM MATNO(NUMEL), NP(NUMEL, NNPE), IP(8)
     DIM PROPS(NMATS, 5)

     L$ = STRING$(79, "=")
     F$(11) = DATE$
     TOTALtime = VAL(F$(12))
     STARTtime = TIMER

     GOTO 160

     '<---------------------- SUBROUTINE AREA ---------------------------->
100  COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue  OR <ESC> to quit";
110  K$ = INKEY$: IF K$ = "" THEN 110 ELSE RETURN

120  COLOR 14: LOCATE 24, 15: PRINT "<Return> = Accept     R = Reenter"; : COLOR 7
130  K$ = INKEY$: IF K$ = "" THEN 130
     IF K$ = "R" OR K$ = "r" OR K$ = CHR$(13) THEN LOCATE 24, 15: PRINT SPC(35); : RETURN
     PLAY "L40 B": GOTO 130
```

```
140 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
150 LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

    '<=============================================================/=========>
160 CLS : GOSUB 140: GOSUB 150
    LOCATE 5, 5: COLOR 11: PRINT "<<< D A T A      M E N U >>>"
    COLOR 14: PRINT : PRINT TAB(5); "Master file name --> "; F$
    PRINT : PRINT
    PRINT TAB(5); "N  = NODE DATA .............["; NUMNP; "NODES ]"
    PRINT TAB(5); "E  = ELEMENT DATA ..........["; NUMEL; "ELEMENTS ]"
    PRINT TAB(5); "L  = LOAD DATA .............["; NCASE; "CASES ]"
    PRINT
    COLOR 13: PRINT TAB(5); "Q -> QUIT TO ACTIVITY MENU"
    LOCATE 20, 5: COLOR 13: PRINT "====> SELECT ? ";

170 K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 170
    IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 170
    LOCATE 20, 20: PRINT K$;

    IF K$ = "Q" OR K$ = "q" THEN 180
    IF K$ = "N" OR K$ = "n" THEN 220
    IF K$ = "E" OR K$ = "e" THEN 480
    IF K$ = "L" OR K$ = "l" THEN 210
    PLAY "L40 B": GOTO 170

    '<---------------------------- QUIT ---------------------------------->
180 CLS : GOSUB 140: GOSUB 150
    COLOR 11
    PRINT "   <<< DIRECTORY OF DATA FILES >>>"
    PRINT
    PRINT "FILE TYPE                          STATUS"
    COLOR 15: PRINT STRING$(40, "=")
    COLOR 14
    PRINT "1)  CONTROL PARAMETER"
    PRINT "2)  COORDINATE"
    PRINT "3)  BOUNDATA RESTRAIN"
    PRINT "4)  ELEMENT CONNECTIVITY"
    PRINT "5)  MATERIAL PROPERTY"
    PRINT "6)  LOAD CASE #1"
    PRINT "              #2"
    PRINT "              #3"
    PRINT "              #4"
    PRINT "              #5"
    COLOR 15: PRINT STRING$(40, "=")
    COLOR 12: FOR I = 1 TO 10
    IF LEN(F$(I)) > 0 THEN S$ = "O.K." ELSE S$ = "NONE"
    LOCATE I + 6, 33: PRINT S$: NEXT I
    LOCATE 19, 1: COLOR 11: PRINT "GO BACK TO INPUT DATA AGAIN <Y/N>?"

190 K$ = INKEY$: LOCATE 19, 36, 1, 0, 7: IF K$ = "" THEN 190
    LOCATE 19, 36: PRINT K$;
    IF K$ = "Y" OR K$ = "y" THEN 160
    IF K$ = "N" OR K$ = "n" THEN 200
    PLAY "L40 B": GOTO 190
```

```
200  FINISHtime = TIMER
     TOTALtime = FINISHtime - STARTtime + TOTALtime
     F$(12) = STR$(TOTALtime)
     OPEN "O", 1, D$ + F$ + ".DIR"
     FOR I = 1 TO 12: WRITE #1, F$(I): NEXT I
     CLOSE 1

     OPEN "O", 1, D$ + F$ + ".CON"
     WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
     WRITE #1, NCASE
     WRITE #1, NUMSEG, NGAUSS, IORDER
     WRITE #1, T$
     WRITE #1, E$
     CLOSE 1

     RUN "HEAD"

     '<------------------------- INPUT LOAD ------------------------------>
210  FINISHtime = TIMER
     TOTALtime = FINISHtime - STARTtime + TOTALtime
     F$(12) = STR$(TOTALtime)

     OPEN "O", 1, D$ + F$ + ".DIR"
     FOR I = 1 TO 12: WRITE #1, F$(I): NEXT I
     CLOSE 1

     OPEN "O", 1, D$ + F$ + ".CON"
     WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
     WRITE #1, NCASE
     WRITE #1, NUMSEG, NGAUSS, IORDER
     WRITE #1, T$
     WRITE #1, E$
     CLOSE 1

     RUN "INPUT2"

     '<------------------------- NODE DATA ------------------------------>
220  IF LEN(F$(2)) = 0 THEN 230                'without coordinates data
     DEF SEG = VARSEG(CO(1))
     BLOAD D$ + F$ + ".COO", VARPTR(CO(1))

230  IF LEN(F$(3)) = 0 THEN 240                'without boundary data
     OPEN "I", 1, D$ + F$ + ".BOU"
     FOR I = 1 TO NUMNP
     INPUT #1, NBCJ(I)
     NEXT I
     FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
     INPUT #1, KODE(I, J)
     NEXT J, I
     CLOSE 1

240  CLS : GOSUB 140: GOSUB 150
     LOCATE 5, 5: COLOR 11: PRINT "<<<< NODE DATA >>>>"
     COLOR 14: PRINT : PRINT TAB(5); "Master file name --> "; F$
     PRINT : PRINT
     PRINT TAB(5); "N  = NO.OF NODES ..........["; NUMNP; "NODES ]"
```

```
    PRINT TAB(5); "C  = COORDINATE DATA"
    PRINT TAB(5); "B  = BOUNDARY DATA"
    PRINT TAB(5); "O  = OUTPUT"
    PRINT
    COLOR 13: PRINT TAB(5); "Q -> QUIT TO DATA MENU"
    LOCATE 20, 5: COLOR 13: PRINT "====> SELECT ?"

250 K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 250
    IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 250
    LOCATE 20, 20: PRINT K$

    IF K$ = "N" OR K$ = "n" THEN 260
    IF K$ = "C" OR K$ = "c" THEN 290 '          input coordinate
    IF K$ = "B" OR K$ = "b" THEN 330 '          input boudary data
    IF K$ = "O" OR K$ = "o" THEN 360 '          output
    IF K$ = "Q" OR K$ = "q" THEN 450
    PLAY "L40 B": GOTO 250

    '<-------------------- INPUT NUMBER OF NODES -------------------------->
260 LOCATE 19, 1: COLOR 11: PRINT L$: PRINT : PRINT L$
    LOCATE 20, 5: COLOR 15: INPUT "INPUT NO.OF NODES = ", NUMNP1
    GOSUB 120: IF K$ <> CHR$(13) THEN 260
    IF NUMNP1 <= NUMNP THEN NUMNP = NUMNP1: GOTO 240

    '              Node data need redimension

    ERASE CO, N8CJ, KODE
    DIM CO(NDIME * NUMNP1), N8CJ(NUMNP1), KODE(NUMNP1, NPDOF)
    IF LEN(F$(2)) = 0 THEN 270                  'without coordinates data
    DEF SEG = VARSEG(CO(1))
    BLOAD D$ + F$ + ".COO", VARPTR(CO(1))

270 IF LEN(F$(3)) = 0 THEN 280                  'without boundary data
    OPEN "I", 1, D$ + F$ + ".BOU"
    FOR I = 1 TO NUMNP
    INPUT #1, N8CJ(I)
    NEXT I
    FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
    INPUT #1, KODE(I, J)
    NEXT J, I
    CLOSE 1

280 NUMNP = NUMNP1
    GOTO 240

    '<---------------- INPUT COORDINATES OF NODAL POINTS ------------------>
290 CLS : GOSUB 140: GOSUB 150
    PRINT
    PRINT ">COORDINATES OF NODAL POINT"
    PRINT
    PRINT "   1.Press <RETURN> after entering each data to move cursor"
    PRINT "   2.Input <Ele. = 0> to terminate this mode"
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$
    COLOR 11
    LOCATE 10, 1: PRINT "  Node";
```

```
    FOR I = 1 TO NDIME: PRINT "    COOR-"; I; : NEXT I: PRINT "    NODAL-GEN."
    I1 = 0: I2 = 0: IGEN1 = 0
    FOR I = 1 TO NDIME: HCO(I) = 0: NEXT I
    .
300 COLOR 14: LOCATE 11, 1: PRINT SPC(78); : LOCATE 11, 2: PRINT I1
    FOR J = 1 TO NDIME: LOCATE 11, (J - 1) * 12 + 10: PRINT HCO(J): NEXT J
    LOCATE 11, (J - 1) * 12 + 10: PRINT IGEN1
310 LOCATE 12, 1: PRINT SPC(78);

    COLOR 12: LOCATE 12, 3: INPUT "", I2
    IF I2 > NUMNP OR I2 < 1 THEN F$(2) = F$ + ".COO": GOTO 240
    FOR J = 1 TO NDIME
    LOCATE CSRLIN - 1, (J - 1) * 12 + 11: INPUT "", HCO(J)
    IF HCO(J) = 0 THEN LOCATE CSRLIN - 1, (J - 1) * 12 + 10: PRINT HCO(J)
    NEXT J
    LOCATE CSRLIN - 1, (J - 1) * 12 + 11: INPUT "", IGEN2
    IGEN2 = ABS(IGEN2): LOCATE CSRLIN - 1, (J - 1) * 12 + 10: PRINT IGEN2
    GOSUB 120: IF K$ <> CHR$(13) THEN 310

    FOR J = 1 TO NDIME: CO((I2 - 1) * NDIME + J) = HCO(J): NEXT J

    '           Check if generation is required

    IF IGEN1 <= 0 THEN 320
    NG = ABS(I2 - I1)
    IF I2 > I1 THEN IFAC = 1 ELSE IFAC = -1
    FOR J = 1 TO NDIME
        D(J) = (HCO(J) - CO((I1 - 1) * NDIME + J)) * IGEN1 / NG: NEXT J
    K1 = I1
    ISTEP = ABS(INT((I2 - I1) / IGEN1)) - 1
    FOR K = 1 TO ISTEP
    K2 = K1 + IFAC * IGEN1
    FOR J = 1 TO NDIME
        CO((K2 - 1) * NDIME + J) = CO((K1 - 1) * NDIME + J) + D(J): NEXT J
    K1 = K2
    NEXT K

320 I1 = I2: IGEN1 = IGEN2: GOTO 300

    '<-------------------- INPUT BOUNDARY DATA -------------------------->
330 CLS : GOSUB 140: GOSUB 150
    PRINT ">BOUNDARY DATA"
    PRINT "             ( 0 = Free , 1 = Lock )"
    PRINT "   1.Press <RETURN> after entering each data to move cursor"
    PRINT "   2.Input <Node = 0> to terminate this mode"
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$
    COLOR 11
    LOCATE 10, 1: PRINT "  Node";
    FOR I = 1 TO NPDOF: PRINT "    DOF.-"; I; : NEXT I: PRINT
    I = 0: J = 0

340 COLOR 14: LOCATE 11, 1: PRINT SPC(78); : LOCATE 11, 2: PRINT I
    FOR J = 1 TO NPDOF: LOCATE 11, (J - 1) * 12 + 10: PRINT IP(J): NEXT J
350 LOCATE 12, 1: PRINT SPC(78);
```

```
    COLOR 12: LOCATE 12, 3: INPUT "", I
    IF I > NUMNP OR I < 1 THEN F$(3) = F$ + ".BOU": GOTO 240
    FOR J = 1 TO NPDOF
    LOCATE CSRLIN - 1, (J - 1) * 12 + 11: INPUT "", IP(J)
    IF IP(J) = 0 THEN LOCATE CSRLIN - 1, (J - 1) * 12 + 10: PRINT IP(J)
    NEXT J
    GOSUB 120
    IF K$ <> CHR$(13) THEN 350
    NBCJ(I) = 1
    FOR J = 1 TO NPDOF: KODE(I, J) = IP(J): NEXT J
    GOTO 340

    '<---------------------- OUTPUT NODE DATA ---------------------------->
360 CLS : GOSUB 140: GOSUB 150: COLOR 11
    PRINT "                                    "
    PRINT "   OUTPUT NODE DATA                 "
    PRINT "                                    "
    PRINT : COLOR 14
    PRINT "O  = ONSCREEN"
    PRINT "H  = HARD COPY"
    PRINT : COLOR 13
    PRINT "Q -> QUIT TO NODE DATA MENU"
    PRINT
    PRINT "===> SELECT ?"

370 K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 370
    LOCATE 12, 15: PRINT K$
    IF K$ = "O" OR K$ = "o" THEN 380
    IF K$ = "H" OR K$ = "h" THEN 410
    IF K$ = "Q" OR K$ = "q" THEN 240
    PLAY "L40 B": GOTO 370

    '<----------------- OUTPUT NODE DATA ON SCREEN ---------------------->
380 INUM = 0
390 CLS : GOSUB 140: GOSUB 150
    COLOR 13: LOCATE 1, 1: PRINT "** NODE  DATA **";
    PRINT TAB(4 + 13 * NDIME); "** BOUNDARY DATA **"
    PRINT "Node";
    FOR I = 1 TO NDIME: PRINT "       CORD-"; USING "#"; I; : NEXT I
    FOR I = 1 TO NPDOF: PRINT "   B-"; USING "#"; I; : NEXT I: PRINT
    FOR I = 1 TO (4 + 13 * NDIME + 6 * NPDOF): PRINT "-"; : NEXT I: PRINT

    COLOR 15
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NUMNP THEN 400
    PRINT USING "####"; INUM;
    FOR J = 1 TO NDIME: PRINT USING "##########.###"; CO((INUM - 1) * NDIME + J); : NEXT J
    FOR J = 1 TO NPDOF
    IF KODE(INUM, J) = 0 THEN PRINT "      "; ELSE PRINT "     L";
    NEXT J: PRINT : NEXT I
400 GOSUB 100: IF K$ = CHR$(27) THEN 360
    IF INUM >= NUMNP THEN 360 ELSE 390

    '<------------------- NODE DATA ON HARD COPY ----------------------->
410 CLS : GOSUB 140: GOSUB 150: COLOR 11
    PRINT "                                    "
```

```
    PRINT "|   HARD COPY for NODE DATA   |"
    PRINT "|_____________________________|"
    PRINT
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 420
    LPRINT CHR$(12);
    ILINE = 0
    CALL HEADING

420 GOSUB 440
    FOR I = 1 TO NUMNP
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 430
    LPRINT CHR$(12); : ILINE = 0                    'form feed
    CALL HEADING                                    'print heading
    GOSUB 440                                       'print title
430 LPRINT USING "####"; I;
    FOR J = 1 TO NDIME: LPRINT USING "##########.###"; CO((I - 1) * NDIME + J); : NEXT J
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 0 THEN LPRINT "      "; ELSE LPRINT "    L";
    NEXT J: LPRINT : NEXT I
    GOTO 360
    '       Print title
440 LPRINT
    LPRINT "** NODE  DATA **";
    LPRINT TAB(4 + 13 * NDIME); "** BOUNDARY DATA **"
    LPRINT "Node";
    FOR K = 1 TO NDIME: LPRINT "       CORD-"; USING "#"; K; : NEXT K
    FOR K = 1 TO NPDOF: LPRINT "   B-"; USING "#"; K; : NEXT K: LPRINT
    FOR K = 1 TO (4 + 13 * NDIME + 6 * NPDOF): LPRINT "-"; : NEXT K: LPRINT
    ILINE = ILINE + 4
    RETURN

    '<---------------------------- QUIT ---------------------------------->
450 IF LEN(F$(2)) = 0 THEN 460
    DEF SEG = VARSEG(CO(1))
    BSAVE D$ + F$ + ".COO", VARPTR(CO(1)), NUMNP * NDIME * 8

460 IF LEN(F$(3)) = 0 THEN 470
    OPEN "O", 1, D$ + F$ + ".BOU"
    FOR I = 1 TO NUMNP
    WRITE #1, NBCJ(I)
    NEXT I
    FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
    WRITE #1, KODE(I, J)
    NEXT J, I
    CLOSE 1

470 GOTO 160

    '<------------------------ ELEMENT DATA ------------------------------->
480 IF LEN(F$(4)) = 0 THEN 490                  'without Element connectivity
    OPEN "I", 1, D$ + F$ + ".ELE"
    FOR I = 1 TO NUMEL
    FOR J = 1 TO NNPE
```

```
    INPUT #1, NP(I, J)
    NEXT J, I
    FOR I = 1 TO NUMEL
    INPUT #1, MATNO(I)
    NEXT I
    CLOSE 1

490 IF LEN(F$(5)) = 0 THEN 500                  'without material properties
    OPEN "I", 1, D$ + F$ + ".MAT"
    FOR I = 1 TO NMATS
    FOR J = 1 TO 5
    INPUT #1, PROPS(I, J)
    NEXT J, I
    CLOSE 1

500 CLS : GOSUB 140: GOSUB 150
    LOCATE 5, 5: COLOR 11: PRINT "<<<< ELEMENT DATA >>>>"
    COLOR 14: PRINT : PRINT TAB(5); "Master file name --> "; D$ + F$
    PRINT : PRINT
    PRINT TAB(5); "N  = NO.OF ELEMENTS ........["; NUMEL; "ELEMENTS]"
    PRINT TAB(5); "S  = NO.OF MATERIAL SETS ...["; NMATS; "SETS]"
    PRINT TAB(5); "E  = ELEMENT CONNECTIVITY"
    PRINT TAB(5); "M  = MATERIAL PROPERTY"
    PRINT TAB(5); "O  = OUTPUT"
    PRINT
    COLOR 13: PRINT TAB(5); "Q -> QUIT TO DATA MENU"
    LOCATE 20, 5: COLOR 13: PRINT "====> SELECT ?"

510 K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 510
    IF ASC(K$) = 0 THEN PLAY "L40 8": GOTO 510
    LOCATE 20, 20: PRINT K$

    IF K$ = "N" OR K$ = "n" THEN 520
    IF K$ = "S" OR K$ = "s" THEN 540
    IF K$ = "E" OR K$ = "e" THEN 600
    IF K$ = "M" OR K$ = "m" THEN 670
    IF K$ = "O" OR K$ = "o" THEN 900
    IF K$ = "Q" OR K$ = "q" THEN 640
    PLAY "L40 8": GOTO 510

    '<------------------- INPUT NUMBER OF ELEMENTS ----------------------->
520 LOCATE 19, 1: COLOR 11: PRINT L$: PRINT : PRINT L$
    LOCATE 20, 5: COLOR 15: INPUT "INPUT NO.OF ELEMENTS = ", NUMEL1
    GOSUB 120: IF K$ <> CHR$(13) THEN 520
    IF NUMEL1 <= NUMEL THEN NUMEL = NUMEL1: GOTO 500

    '              Element data need redimension

    ERASE NP, MATNO
    DIM NP(NUMEL1, NNPE), MATNO(NUMEL1)

    IF LEN(F$(4)) = 0 THEN 530                  'without Element connectivity
    OPEN "I", 1, D$ + F$ + ".ELE"
    FOR I = 1 TO NUMEL
    FOR J = 1 TO NNPE
    INPUT #1, NP(I, J)
```

```
    NEXT J, I
    FOR I = 1 TO NUMEL
    INPUT #1, MATNO(I)
    NEXT I
    CLOSE 1

530 NUMEL = NUMEL1
    GOTO 500

    '<--------------- INPUT NUMBER OF DIFFERENT MATERIAL ----------------->
540 LOCATE 19, 1: COLOR 11: PRINT L$: PRINT : PRINT L$
    LOCATE 20, 5: COLOR 15: INPUT "INPUT NO.OF MATERIALS = ", NMATS1
    GOSUB 120: IF K$ <> CHR$(13) THEN 540
    IF NMATS1 <= NMATS THEN NMATS = NMATS1: GOTO 500

    '                   Material data need redimension

    ERASE PROPS
    DIM PROPS(NMATS1, 5)

    IF LEN(F$(5)) = 0 THEN 550                  'without material properties
    OPEN "I", 1, D$ + F$ + ".MAT"
    FOR I = 1 TO NMATS
    FOR J = 1 TO 5
    INPUT #1, PROPS(I, J)
    NEXT J, I
    CLOSE 1

550 NMATS = NMATS1
    GOTO 500

    '<----------------- INPUT ELEMENT CONNECTIVITY ----------------------->
600 CLS : GOSUB 140: GOSUB 150
    FOR I = 1 TO NNPE: IP(I) = 0: NEXT I: MATNO = 0
    PRINT
    PRINT ">ELEMENT CONNECTIVITY"
    PRINT
    PRINT "   1.Press <RETURN> after entering each data to move cursor"
    PRINT "   2.Input <Ele. = 0> to terminate this mode"
    PRINT "   3.Input (E-GEN. => 1) if genration is required"
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$

    COLOR 11
    LOCATE 10, 1: PRINT " Ele.";
    FOR I = 1 TO NNPE: PRINT "    N-"; USING "#"; I; : NEXT I
    PRINT "    E-GEN. N-GEN."
    I1 = 0: I2 = 0: IE1 = 0: IN1 = 0
    FOR I = 1 TO NNPE: IP(I) = 0: NEXT I

610 COLOR 14: LOCATE 11, 1: PRINT SPC(78); : LOCATE 11, 1: PRINT I1
    FOR J = 1 TO NNPE: LOCATE 11, (J - 1) * 7 + 9: PRINT IP(J): NEXT J
    LOCATE 11, (J - 1) * 7 + 9: PRINT IE1: J = J + 1
    LOCATE 11, (J - 1) * 7 + 9: PRINT IN1
620 LOCATE 12, 1: PRINT SPC(78);
```

```
    COLOR 12: LOCATE 12, 2: INPUT "", I2
    IF I2 > NUMEL OR I2 < 1 THEN F$(4) = F$ + ".ELE": GOTO 500

    FOR J = 1 TO NNPE
    LOCATE 12, (J - 1) * 7 + 10: INPUT "", IP(J)
    IF IP(1) = 0 THEN
         IF IE1 > 0 THEN IE2 = (I2 - I1) / IE1
         IN2 = IN1 * IE2
         FOR K = 1 TO NNPE: IP(K) = NP(I1, K) + IN2: NEXT K
         FOR K = 1 TO NNPE: LOCATE 12, (K - 1) * 7 + 9: PRINT IP(K): NEXT K
         J = NNPE
         END IF
    NEXT J
    LOCATE 12, (J - 1) * 7 + 10: INPUT "", IE2
    IF IE2 = 0 THEN LOCATE 12, (J - 1) * 7 + 9: PRINT IE2
    J = J + 1: LOCATE 12, (J - 1) * 7 + 10: INPUT "", IN2
    IF IN2 = 0 THEN LOCATE 12, (J - 1) * 7 + 9: PRINT IN2
    GOSUB 120: IF K$ <> CHR$(13) THEN 620

    FOR J = 1 TO NNPE: NP(I2, J) = IP(J): NEXT J

    '              Check if generation is required

    IF IE1 = 0 OR I2 <= I1 THEN 630
    FOR K = (I1 + IE1) TO (I2 - 1) STEP IE1
    FOR J = 1 TO NNPE: NP(K, J) = NP(K - IE1, J) + IN1: NEXT J
    NEXT K

630 I1 = I2: IE1 = IE2: IN1 = IN2: GOTO 610

    '<----------------------------- QUIT ---------------------------------->
640 IF LEN(F$(4)) = 0 THEN 650

    OPEN "O", 1, D$ + F$ + ".ELE"
    FOR I = 1 TO NUMEL
    FOR J = 1 TO NNPE
    WRITE #1, NP(I, J)
    NEXT J, I
    FOR I = 1 TO NUMEL
    WRITE #1, MATNO(I)
    NEXT I
    CLOSE 1

650 IF LEN(F$(5)) = 0 THEN 660
    OPEN "O", 1, D$ + F$ + ".MAT"
    FOR I = 1 TO NMATS
    FOR J = 1 TO 5
    WRITE #1, PROPS(I, J)
    NEXT J, I
    CLOSE 1

660 GOTO 160

    '<-------------------- MATERIAL PROPERTIES -------------------------->
670 CLS : GOSUB 140: GOSUB 150
    COLOR 11: PRINT "> MATERIAL PROPERTIES"
```

```
    COLOR 15: PRINT "("; NMATS; "SETS DEFINED)"
    PRINT : COLOR 14
    PRINT "E = EDIT"
    PRINT "O = OUTPUT"
    PRINT
    COLOR 13: PRINT "Q = QUIT TO ELEMENT DATA"
    PRINT : COLOR 11: PRINT "===> SELECT ?"
680 K$ = INKEY$: LOCATE 11, 15, 1, 0, 7: IF K$ = "" THEN 680
    LOCATE 11, 15: PRINT K$
    IF K$ = "E" OR K$ = "e" THEN 690
    IF K$ = "O" OR K$ = "o" THEN 800
    IF K$ = "Q" OR K$ = "q" THEN 500
    PLAY "L40 8": GOTO 680

    '<-------------------- EDIT MATERIAL PROPERTY ------------------------>
690 CLS : GOSUB 140: GOSUB 150
    COLOR 11: PRINT "> MATERIAL PROPERTIES"
    COLOR 15: PRINT "("; NMATS; "SETS DEFINED)"
    PRINT
    COLOR 14: PRINT "Mat. set .. "
    PRINT
    PRINT "ELASTIC MODULUS ..... :"
    PRINT "Poisson's ratio ..... :"
    PRINT "Material thickness ..... :"
    PRINT "Mass density ..... :"
    PRINT "Coef. of thermal expansion .... :"
    PRINT
    COLOR 12: LOCATE 6, 13: INPUT "", INUM
    IF INUM < 1 OR INUM > NMATS THEN F$(5) = F$ + ".MAT": GOTO 670

    FOR I = 1 TO 5
    IF I <> 3 THEN LOCATE 7 + I, 36: INPUT "", HCO(I): GOTO 700
    IF NTYPE = 2 THEN HCO(3) = 1: GOTO 700
    IF NTYPE = 4 THEN HCO(4) = 0: GOTO 700
    LOCATE 7 + I, 36: INPUT "", HCO(I)
700 LOCATE 7 + I, 36: PRINT HCO(I)
    NEXT I

    COLOR 11: LOCATE 15, 1: PRINT ">ELEMENT LIST :"
    PRINT
    L$ = STRING$(78, "=")
    PRINT L$
    PRINT "    EXAMPLE:        1) ELEMENT LIST : 1 3 5 6 7 8 9    (INDIVIDUAL)"
    PRINT TAB(16); "OR   2) ELEMENT LIST : 1/5/2 6/9/1      (GENERATION)"
    PRINT TAB(16); "OR   3) ELEMENT LIST : 1 2 4 5/9/2      (COMBINATION)"
    PRINT TAB(16); "OR   3) ELEMENT LIST : A                (A = ALL)"

    LOCATE 15, 18: INPUT "", A$

    GOSUB 120: IF K$ <> CHR$(13) THEN 690

    FOR I = 1 TO 5: PROPS(INUM, I) = HCO(I): NEXT I

    IF A$ = "A" THEN A$ = "1/" + RIGHT$(STR$(NUMEL), LEN(STR$(NUMEL)) - 1) + "/1"
    L = LEN(A$)
    IL = 0
```

```
710  IBGN$ = "": IEND$ = "": ISTEP$ = ""
720  IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 750
     IF IL$ <> "/" THEN IBGN$ = IBGN$ + IL$: GOTO 720
730  IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 750
     IF IL$ <> "/" THEN IEND$ = IEND$ + IL$: GOTO 730
740  IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 750
     IF IL$ <> "/" THEN ISTEP$ = ISTEP$ + IL$: GOTO 740
750  GOSUB 760
     IF IL < L THEN 710 ELSE GOTO 690

760  IBGN = VAL(IBGN$): IEND = VAL(IEND$): ISTEP = VAL(ISTEP$)
     IF IBGN < 1 OR IBGN > NUMEL THEN RETURN
     IF IEND > NUMEL THEN IEND = NUMEL
     IF ISTEP < 1 THEN MATNO(IBGN) = 1: RETURN
     FOR K = IBGN TO IEND STEP ISTEP
     MATNO(K) = INUM: NEXT K
     RETURN

     GOTO 690
     '<------------------------- OUTPUT MAT ------------------------------>
800  CLS : GOSUB 140: GOSUB 150: COLOR 11
     PRINT "                                "
     PRINT "  OUTPUT MATERIAL DATA          "
     PRINT "                                "
     PRINT : COLOR 14
     PRINT "O  = ONSCREEN"
     PRINT "H  = HARD COPY"
     PRINT : COLOR 13
     PRINT "Q -> QUIT TO MATERIAL DATA MENU"
     PRINT
     PRINT "===> SELECT ?"

810  K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 810
     LOCATE 12, 15: PRINT K$
     IF K$ = "O" OR K$ = "o" THEN 820
     IF K$ = "H" OR K$ = "h" THEN 850
     IF K$ = "Q" OR K$ = "q" THEN 670
     PLAY "L40 8": GOTO 810

     '<------------------- MATERIAL DATA ON SCREEN ----------------------->
820  INUM = 0
830  CLS : GOSUB 140: GOSUB 150
     COLOR 13: LOCATE 1, 1: PRINT "** MATERIAL PROPERTIES **"
     PRINT "Mat. E          POISS   THICK   WEIGHT      ALPHA"
     PRINT STRING$(55, "-")
     COLOR 15
     FOR I = 1 TO 20
     INUM = INUM + 1: IF INUM > NMATS THEN 840
     LOCATE I + 3, 1: PRINT USING "###"; INUM;
     PRINT USING "  #.####^^^^"; PROPS(INUM, 1);
     PRINT USING "  #.####"; PROPS(INUM, 2);
     PRINT USING "  #.####"; PROPS(INUM, 3);
     PRINT USING "  #.####^^^^"; PROPS(INUM, 4);
     PRINT USING "  #.####^^^^"; PROPS(INUM, 5);
     NEXT I
```

```
840  GOSUB 100: IF K$ = CHR$(27) THEN 670
     IF INUM >= NMATS THEN 670
     GOTO 830

     '<------------------ MATERIAL DATA ON HARD COPY ----------------------->
850  CLS : GOSUB 140: GOSUB 150: COLOR 11
     PRINT "                                          "
     PRINT "  HARD COPY for MATERIAL PROPERTIES DATA  "
     PRINT "                                          "
     PRINT
     H$ = "DATA"
     IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
     IF ILINE < 56 THEN 860
     LPRINT CHR$(12);
     ILINE = 0
     CALL HEADING

860  GOSUB 880
     FOR I = 1 TO NMATS
     ILINE = ILINE + 1
     IF ILINE <= 60 THEN 870
     LPRINT CHR$(12); : ILINE = 0                    'form feed
     CALL HEADING                                    'print heading
     GOSUB 880                                       'print title
870  LPRINT USING "###"; I;
     LPRINT USING "  #.####^^^^"; PROPS(I, 1);
     LPRINT USING "  #.####"; PROPS(I, 2);
     LPRINT USING "  #.####"; PROPS(I, 3);
     LPRINT USING "  #.####^^^^"; PROPS(I, 4);
     LPRINT USING "  #.####^^^^"; PROPS(I, 5);
     LPRINT
     NEXT I
     GOTO 800

     '         Print title
880  LPRINT
     LPRINT "** MATERIAL PROPERTIES **"
     LPRINT "Mat. E           POISS   THICK   WEIGHT      ALPHA"
     LPRINT STRING$(55, "-")
     ILINE = ILINE + 4
     RETURN

     '<--------------------- OUTPUT ELEMENT DATA --------------------------->
900  CLS : GOSUB 140: GOSUB 150: COLOR 11
     PRINT "                                  "
     PRINT "  OUTPUT ELEMENT DATA             "
     PRINT "                                  "
     PRINT : COLOR 14
     PRINT "O  = ONSCREEN"
     PRINT "H  = HARD COPY"
     PRINT : COLOR 13
     PRINT "Q -> QUIT TO ELEMENT DATA MENU"
     PRINT
     PRINT "===> SELECT ?"

910  K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 910
```

```
    LOCATE 12, 15: PRINT K$
    IF K$ = "O" OR K$ = "o" THEN 920
    IF K$ = "H" OR K$ = "h" THEN 950
    IF K$ = "Q" OR K$ = "q" THEN 500
    PLAY "L40 8": GOTO 910

    '<------------------- ELEMENT DATA ON SCREEN ------------------------->
920 INUM = 0
930 CLS : GOSUB 140: GOSUB 150
    COLOR 13: LOCATE 1, 1: PRINT "** ELEMENT DATA **"
    PRINT " Ele.   Mat.";
    FOR I = 1 TO NNPE: PRINT "    N-"; USING "#"; I; : NEXT I: PRINT
    FOR I = 1 TO (12 + 7 * NNPE): PRINT "-"; : NEXT I: PRINT

    COLOR 15
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NUMEL THEN 940
    PRINT USING "######"; INUM; MATNO(INUM);
    FOR J = 1 TO NNPE: PRINT USING "#######"; NP(INUM, J); : NEXT J
    PRINT
    NEXT I
940 GOSUB 100: IF K$ = CHR$(27) THEN 900
    IF INUM >= NUMEL THEN 900 ELSE 930

    '<------------------ ELEMENT DATA ON HARD COPY ----------------------->
950 CLS : GOSUB 140: GOSUB 150: COLOR 11
    PRINT "                                        "
    PRINT "  HARD COPY for ELEMENT DATA            "
    PRINT "                                        "
    PRINT
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 960
    LPRINT CHR$(12);
    ILINE = 0
    CALL HEADING

960 GOSUB 980
    FOR I = 1 TO NUMEL
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 970
    LPRINT CHR$(12); : ILINE = 0                    'form feed
    CALL HEADING                                    'print heading
    GOSUB 980                                       'print title
970 LPRINT USING "######"; I; MATNO(I);
    FOR J = 1 TO NNPE: LPRINT USING "#######"; NP(I, J); : NEXT J
    LPRINT
    NEXT I
    GOTO 900

    '         Print title
980 LPRINT
    LPRINT "** ELEMENT  DATA **"
    LPRINT "  Ele.  Mat.";
    FOR K = 1 TO NNPE: LPRINT "    N-"; USING "#"; K; : NEXT K: LPRINT
    FOR K = 1 TO (12 + 7 * NNPE): LPRINT "-"; : NEXT K: LPRINT
```

```
     ILINE = ILINE + 4
     RETURN


DEFINT I-N
DEFDBL A-H, O-Z
     '---------------------- Initialize printer ----------------------------
SUB FIRST STATIC
     COLOR 12
     LOCATE 10, 25: PRINT "PLEASE ADJUST PARER"
     LOCATE 11, 25: PRINT "Press any key to continue"
6000 K$ = INKEY$: IF K$ = "" THEN 6000
     LPRINT CHR$(27) + "C" + CHR$(66);
     LOCATE 10, 25: COLOR 28: PRINT "┌──────────────────────────┐"
     LOCATE 11, 25: PRINT "│                          │"
     LOCATE 12, 25: PRINT "└──────────────────────────┘"
     LOCATE 11, 29: COLOR 12: PRINT "WAIT : data being print"
     LPRINT CHR$(15);                          'set to compressed mode
     WIDTH "LPT1:", 132
END SUB

DEFINT I-N
DEFDBL A-H, O-Z
     '------------------------ Print heading -----------------------------
SUB HEADING STATIC
SHARED ILINE, IPAGE, F$, T$, H$
     ILINE = ILINE + 4: IPAGE = IPAGE + 1
     FOR I = 1 TO 130: LPRINT "="; : NEXT I
     LPRINT CHR$(18);                                'release compressed mode
     LPRINT CHR$(27) + "W" + CHR$(1); "SUPATFEAP"; CHR$(27) + "W" + CHR$(0);
     LPRINT CHR$(15);
     LPRINT TAB(91); "<"; H$; ">Page"; IPAGE
     LPRINT "PROJECT  : "; T$; : LPRINT TAB(112); "FILE NAME: "; F$
     FOR I = 1 TO 130: LPRINT "="; : NEXT I: LPRINT
END SUB
```

```
DECLARE SUB FIRST ()
DECLARE SUB HEADING ()
    '*************************************************************************
    '*                                                                       *
    '*                    file name "INPUT(2)"                               *
    '*                   Input LOAD DATA for 5 load case                     *
    '*                                                                       *
    '*************************************************************************
    L$ = STRING$(79, "=")
    GOTO 150

    '<---------------------- SUBROUTINE AREA ---------------------------->
100 COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue  OR <ESC> to quit";
110 K$ = INKEY$: IF K$ = "" THEN 110 ELSE RETURN

120 COLOR 14: LOCATE 24, 15: PRINT "<Return> = Accept     R = Reenter"; : COLOR 7
130 K$ = INKEY$: IF K$ = "" THEN 130
    IF K$ = "R" OR K$ = "r" OR K$ = CHR$(13) THEN LOCATE 24, 15: PRINT SPC(35); : RETURN
    BEEP: GOTO 130

140 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$
    LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

    '-------------------------------------------------------------------------
150 DEFINT I-N
    DEFDBL A-H, O-Z

    DIM A!(40), F$(12), H(9), IFILE(5)              'H(9) = help vector

    DEF SEG = VARSEG(A!(0))
    BLOAD "MEMO.DAT", VARPTR(A!(0))
    D$ = CHR$(A!(40)) + ":"

    OPEN "I", 1, D$ + "DIRECTOR": INPUT #1, F$: CLOSE 1

    OPEN "I", 1, D$ + F$ + ".CON"
    INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    INPUT #1, NCASE
    INPUT #1, NUMSEG, NGAUSS, IORDER
    INPUT #1, T$
    INPUT #1, E$
    CLOSE 1

    DIM NLOAD(NUMNP), PLOAD(NUMNP, NPDOF), TEMP(NUMNP), ULOAD(NUMEL)
    DIM NEASS(NUMEL), NSIDE(NUMEL), PRESS(600)

    OPEN "I", 1, D$ + F$ + ".DIR"
    FOR I = 1 TO 12: INPUT #1, F$(I): NEXT I: CLOSE 1
    F$(11) = DATE$
    TOTALtime! = VAL(F$(12))
    STARTtime! = TIMER

    FOR I = 1 TO 5: IF LEN(F$(I + 5)) > 0 THEN IFILE(I) = 1
    NEXT I

160 CLS : GOSUB 140
```

```
      PRINT "<<< LOAD CASE MENU >>>"
      PRINT : COLOR 14
      PRINT "E = EDIT LOAD CASE ("; NCASE; "CASES)";
      COLOR 12
      FOR I = 1 TO 5: IF IFILE(I) = 1 THEN PRINT " #"; RIGHT$(STR$(I), 1);
      NEXT I
      PRINT : COLOR 14
      PRINT "D = DELETE LOAD CASE"
      PRINT : COLOR 13
      PRINT "Q = QUIT TO DATA MENU"
      PRINT
      PRINT "===> SELECT ?"

170   K$ = INKEY$: LOCATE 10, 15, 1, 0, 7: IF K$ = "" THEN 170
      LOCATE 10, 15: PRINT K$

      IF K$ = "Q" OR K$ = "q" THEN 180
      IF K$ = "D" OR K$ = "d" THEN 240
      IF K$ = "E" OR K$ = "e" THEN 300
      PLAY "L40 B": GOTO 170

      '<------------------------- DATA MENU ------------------------------->
180   CLS : GOSUB 140
      LOCATE 5, 5: COLOR 11: PRINT "<<< D A T A      M E N U >>>"
      COLOR 14: PRINT : PRINT TAB(5); "Master file name --> "; F$
      PRINT : PRINT
      PRINT TAB(5); "N  = NODE DATA .............["; NUMNP; "NODES ]"
      PRINT TAB(5); "E  = ELEMENT DATA ..........["; NUMEL; "ELEMENTS ]"
      PRINT TAB(5); "L  = LOAD DATA .............["; NCASE; "CASES ]"
      PRINT
      COLOR 13: PRINT TAB(5); "Q -> QUIT TO ACTIVITY MENU"
      LOCATE 20, 5: COLOR 13: PRINT "====> SELECT ? ";

190   K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 190
      IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 190
      LOCATE 20, 20: PRINT K$;

      IF K$ = "N" OR K$ = "n" THEN 200
      IF K$ = "E" OR K$ = "e" THEN 200
      IF K$ = "L" OR K$ = "l" THEN 160
      IF K$ = "Q" OR K$ = "q" THEN 210
      PLAY "L40 B": GOTO 190

200   OPEN "O", 1, D$ + F$ + ".CON"
      WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
      WRITE #1, NCASE
      WRITE #1, NUMSEG, NGAUSS, IORDER
      WRITE #1, T$
      WRITE #1, E$
      CLOSE 1

      FINISHtime! = TIMER
      TOTALtime! = FINISHtime! - STARTtime! + TOTALtime!
      F$(12) = STR$(TOTALtime!)
      OPEN "O", 1, D$ + F$ + ".DIR"
      FOR I = 1 TO 12: WRITE #1, F$(I): NEXT I
```

```
    CLOSE 1

    RUN "A:INPUT"

    '<-------------------- QUIT TO ACTIVITY MENU ------------------------->
210 CLS : GOSUB 140
    COLOR 11
    PRINT "   <<< DIRECTORY OF DATA FILES >>>"
    PRINT
    PRINT "FILE TYPE                    .   STATUS"
    COLOR 15: PRINT STRING$(40, "=")
    COLOR 14
    PRINT "1)  CONTROL PARAMETER"
    PRINT "2)  COORDINATE"
    PRINT "3)  BOUNDATA RESTRAIN"
    PRINT "4)  ELEMENT CONNECTIVITY"
    PRINT "5)  MATERIAL PROPERTY"
    PRINT "6)  LOAD CASE #1"
    PRINT "              #2"
    PRINT "              #3"
    PRINT "              #4"
    PRINT "              #5"
    COLOR 15: PRINT STRING$(40, "=")
    COLOR 12: FOR I = 1 TO 10
    IF LEN(F$(I)) > 0 THEN S$ = "O.K." ELSE S$ = "NONE"
    LOCATE I + 6, 33: PRINT S$: NEXT I
    LOCATE 19, 1: COLOR 11: PRINT "GO BACK TO INPUT DATA AGAIN <Y/N>?";

220 K$ = INKEY$: IF K$ = "" THEN 220
    IF K$ = "Y" OR K$ = "y" THEN 180
    IF K$ = "N" OR K$ = "n" THEN 230
    PLAY "L40 8": GOTO 220

230 OPEN "O", 1, D$ + F$ + ".CON"
    WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    WRITE #1, NCASE
    WRITE #1, NUMSEG, NGAUSS, IORDER
    WRITE #1, T$
    WRITE #1, E$
    CLOSE 1

    FINISHtime! = TIMER
    TOTALtime! = FINISHtime! - STARTtime! + TOTALtime!
    F$(12) = STR$(TOTALtime!)
    OPEN "O", 1, D$ + F$ + ".DIR"
    FOR I = 1 TO 12: WRITE #1, F$(I): NEXT I
    CLOSE 1

    RUN "HEAD"

    '<---------------------- DELETE LOAD CASE ---------------------------->
240 CLS : GOSUB 140
    COLOR 11: PRINT "<<< DELETE LOAD CASE >>>"
    COLOR 15: INPUT "    LOAD CASE #", ICASE
    LOCATE 10, 10: COLOR 12: PRINT "HIT <Y> TO CONFIRM or ANY KEY TO CANCEL"
```

```
250  K$ = INKEY$: IF K$ = "" THEN 250
     IF K$ <> "Y" THEN 160
     IF IFILE(ICASE) <> 1 THEN 160
     CASE$ = RIGHT$(STR$(ICASE), 1)

     OPEN "I", 1, D$ + F$ + ".LC" + CASE$
     INPUT #1, C$
     INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
     CLOSE 1

     KILL D$ + F$ + ".LC" + CASE$

     IF IPLOD <> 1 THEN 260
     KILL D$ + F$ + ".P" + CASE$

260  IF IGRAV <> 1 THEN 270
     KILL D$ + F$ + ".G" + CASE$

270  IF IEDGE <> 1 THEN 280
     KILL D$ + F$ + ".E" + CASE$

280  IF ITEMP <> 1 THEN 290
     KILL D$ + F$ + ".T" + CASE$

290  IF IULOD <> 1 THEN 295
     KILL D$ + F$ + ".U" + CASE$

295  IFILE(ICASE) = 0
     F$(ICASE + 5) = ""

     IPLOD = 0: IGRAV = 0: IEDGE = 0: ITEMP = 0: IULOD = 0
     NCASE = 0
     FOR I = 1 TO 5: NCASE = NCASE + IFILE(I): NEXT I

     GOTO 160
     '<---------------------- EDIT LOAD CASE ------------------------------>
300  CLS : GOSUB 140
     INPUT "LOAD CASE #", ICASE: CASE$ = RIGHT$(STR$(ICASE), 1)
     IF IFILE(ICASE) <> 1 THEN 310
     OPEN "I", 1, D$ + F$ + ".LC" + CASE$
     INPUT #1, C$
     INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
     CLOSE 1
     PRINT
     PRINT "TITLE : "; C$
     GOTO 330

310  PRINT
320  COLOR 15: LOCATE 5, 9: PRINT SPC(60); : LOCATE 5, 1: INPUT "TITLE : ", C$
330  GOSUB 120: IF K$ <> CHR$(13) THEN 320
     IFILE(ICASE) = 1

340  CLS : GOSUB 140
     LOCATE 1, 1: COLOR 11: PRINT "<<< L O A D      D A T A >>>"
     COLOR 15: PRINT : PRINT "Master file name --> "; F$
     COLOR 12: PRINT "LOAD CASE #"; CASE$; " : "; C$
```

```
    PRINT : COLOR 14
    PRINT "N = NODAL LOAD       "; : IF IPLOD = 1 THEN PRINT "(exist)" ELSE PRINT
    PRINT "G = GRAVITY LOADING  "; : IF IGRAV = 1 THEN PRINT "(exist)" ELSE PRINT
    PRINT "P = PRESSURE LOADING "; : IF IEDGE = 1 OR IULOD = 1 THEN PRINT "(exist)" ELSE PRINT
    PRINT "T = THERMAL LOADING  "; : IF ITEMP = 1 THEN PRINT "(exist)" ELSE PRINT
    PRINT : COLOR 13
    PRINT "Q -> QUIT TO LOAD CASE MENU"
    LOCATE 15, 1: COLOR 11: PRINT "====> SELECT ? ";

350 K$ = INKEY$: LOCATE 15, 16, 1, 0, 7: IF K$ = "" THEN 350
    IF ASC(K$) = 0 THEN PLAY "L40 8": GOTO 350
    LOCATE 15, 16: PRINT K$

    IF K$ = "Q" OR K$ = "q" THEN 360
    IF K$ = "N" OR K$ = "n" THEN 400
    IF K$ = "G" OR K$ = "g" THEN 1040
    IF K$ = "P" OR K$ = "p" THEN IF NTYPE = 3 THEN 890
    IF K$ = "P" OR K$ = "p" THEN IF NTYPE = 4 THEN 1090
    IF K$ = "P" OR K$ = "p" THEN 550
    IF K$ = "T" OR K$ = "t" THEN 740
    PLAY "L40 8": GOTO 350

    '<-------------------- QUIT TO LOAD CASE MENU ------------------------>
360 F$(ICASE + 5) = F$ + ".LC" + CASE$

    OPEN "O", 1, D$ + F$ + ".LC" + CASE$
    WRITE #1, C$
    WRITE #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1
    IPLOD = 0: IGRAV = 0: IEDGE = 0: ITEMP = 0

    NCASE = 0
    FOR I = 1 TO 5: NCASE = NCASE + IFILE(I): NEXT I

    GOTO 160

    '<========================== NODAL LOAD ================================>

400 ICHECK = 0: IF IPLOD <> 1 THEN 410
    OPEN "I", 1, D$ + F$ + ".p" + CASE$
    FOR I = 1 TO NUMNP: INPUT #1, NLOAD(I)
    FOR J = 1 TO NPDOF: INPUT #1, PLOAD(I, J)
    NEXT J, I
    CLOSE 1

410 CLS : GOSUB 140
    COLOR 11: PRINT "> NODAL FORCES"
    COLOR 12: PRINT "LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT "E = EDIT"
    PRINT "O = OUTPUT ONSCREEN"
    PRINT "H = HARD COPY"
    PRINT
    COLOR 13: PRINT "Q = QUIT TO LOAD DATA"
    PRINT : COLOR 11: PRINT "===> SELECT ?"
420 K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 420
```

```
    LOCATE 12, 15: PRINT K$
    IF K$ = "E" OR K$ = "e" THEN 430
    IF K$ = "O" OR K$ = "o" THEN 460
    IF K$ = "H" OR K$ = "h" THEN 490
    IF K$ = "Q" OR K$ = "q" THEN 530
    PLAY "L40 B": GOTO 420

    '<---------------------- EDIT NODAL LOAD ---------------------------->
430 FOR I = 1 TO 6: H(I) = 0: NEXT I
    ICHECK = 1: IPLOD = 1
    CLS : GOSUB 140
    COLOR 11: PRINT "> NODAL FORCES"
    PRINT : COLOR 12
    PRINT "   1.Press <RETURN> after entering each data to move cursor"
    PRINT "   2.Input <Node. = 0> to terminate this mode".
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$
    COLOR 11
    LOCATE 10, 1: PRINT "  Node";
    FOR I = 1 TO NPDOF: PRINT "     Forde-"; USING "#"; I; : NEXT I: PRINT
    I = 0: J = 0

440 COLOR 14: LOCATE 11, 1: PRINT SPC(78); : LOCATE 11, 2: PRINT I
    FOR J = 1 TO NPDOF: LOCATE 11, (J - 1) * 12 + 11: PRINT H(J): NEXT J
450 LOCATE 12, 1: PRINT SPC(78);

    COLOR 12: LOCATE 12, 3: INPUT "", I
    IF I > NUMNP OR I < 1 THEN 410
    FOR J = 1 TO NPDOF
    LOCATE CSRLIN - 1, (J - 1) * 12 + 12: INPUT "", H(J)
    IF H(J) = 0 THEN LOCATE CSRLIN - 1, (J - 1) * 12 + 11: PRINT H(J)
    NEXT J
    GOSUB 120
    IF K$ <> CHR$(13) THEN 450
    NLOAD(I) = 1
    FOR J = 1 TO NPDOF: PLOAD(I, J) = H(J): NEXT J
    GOTO 440

    '<-------------------- NODAL LOAD ON SCREEN --------------------------->
460 INUM = 0
470 CLS : GOSUB 140
    COLOR 13: LOCATE 2, 1: PRINT "** NODE  FORCES DATA **"
    PRINT "  Node";
    FOR I = 1 TO NPDOF: PRINT USING "        FORCE-#"; I; : NEXT I
    PRINT

    COLOR 15
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NUMNP THEN 480
    LOCATE I + 3, 1: PRINT USING "  ####"; INUM;
    FOR J = 1 TO NPDOF: PRINT USING "    #.#####^^^^"; PLOAD(INUM, J); : NEXT J
    PRINT
    NEXT I
480 GOSUB 100: IF K$ = CHR$(27) THEN 410
    IF INUM >= NUMNP THEN 410 ELSE CLS : GOTO 470
```

```
    '<------------------- NODAL LOAD ON HARD COPY ------------------------>
490 CLS : GOSUB 140: COLOR 11
    PRINT "                                          "
    PRINT "  HARD COPY for NODE FORCES DATA          "
    PRINT "                                          "
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 500
    LPRINT CHR$(12);
    ILINE = 0: CALL HEADING

500 GOSUB 520
    FOR I = 1 TO NUMNP
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 510
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    GOSUB 520
510 LPRINT USING "####"; I;
    FOR J = 1 TO NPDOF: LPRINT USING "    #.#####^^^^"; PLOAD(I, J); : NEXT J
    LPRINT
    NEXT I
    GOTO 410

    '          Print title
520 LPRINT
    LPRINT CHR$(18); "LOAD CASE #"; CASE$; " : "; C$
    LPRINT "** NODAL LOAD **"; CHR$(15)
    LPRINT "NODE";
    FOR K = 1 TO NPDOF: LPRINT USING "        FORCE-#"; K; : NEXT K: LPRINT
    LPRINT STRING$((4 + 15 * NPDOF), "-")
    ILINE = ILINE + 5
    RETURN

    '<--------------------------- QUIT ---------------------------------->
530 IF ICHECK = 0 THEN 540
    OPEN "O", 1, D$ + F$ + ".P" + CASE$
    FOR I = 1 TO NUMNP: WRITE #1, NLOAD(I)
    FOR J = 1 TO NPDOF: WRITE #1, PLOAD(I, J)
    NEXT J, I
    CLOSE 1

540 FOR I = 1 TO NUMNP: NLOAD(I) = 0
    FOR J = 1 TO NPDOF: PLOAD(I, J) = 0
    NEXT J, I

    GOTO 340

    '<=============== PRESSURE LOAD FOR PLANE STRESS/STRAIN ================>
    '                    (Distributed edge load)

550 IF NNPE = 8 THEN NPRESS = 3
    IF NNPE = 4 THEN NPRESS = 2
    ICHECK = 0: IF IEDGE <> 1 THEN 560
    OPEN "I", 1, D$ + F$ + ".E" + CASE$
    INPUT #1, NEDGE
```

```
    NTOT = NPRESS * 2 * NEDGE
    FOR I = 1 TO NEDGE: INPUT #1, NEASS(I), NSIDE(I): NEXT I
    FOR I = 1 TO NTOT: INPUT #1, PRESS(I): NEXT I
    CLOSE 1

560 CLS : GOSUB 140
    COLOR 11: PRINT "> DISTRIBUTED EDGE LOAD"
    COLOR 12: PRINT "LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT "E = EDIT"
    PRINT "O = OUTPUT ON SCREEN"
    PRINT "H = HARD COPY"
    PRINT
    COLOR 13: PRINT "Q = QUIT TO LOAD DATA"
    PRINT : COLOR 11: PRINT "===> SELECT ?"

570 K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 570
    LOCATE 12, 15: PRINT K$
    IF K$ = "E" OR K$ = "e" THEN 580
    IF K$ = "O" OR K$ = "o" THEN 650
    IF K$ = "H" OR K$ = "h" THEN 680
    IF K$ = "Q" OR K$ = "q" THEN 720
    PLAY "L40 B": GOTO 570

    '<---------- EDIT PRESSURE LOAD ---------->
580 ICHECK = 1: IEDGE = 1
    M$ = "Nor-I      Tan-I      Nor-J      Tan-J      Nor-K      Tan-K"
    CLS : GOSUB 140
    COLOR 11: PRINT "> DISTRIBUTED EDGE LOAD"
    COLOR 12: PRINT "LOAD CASE$ #"; CASE$; " : "; C$
    PRINT
    PRINT "   1.Press <RETURN> after entering each data to move cursor"
    PRINT "   2.Input <Elem. no. = 0> to terminate this mode"
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$
    COLOR 11
    LOCATE 10, 1: PRINT " Ele.  Side  ";
    PRINT LEFT$(M$, NPRESS * 2 * 11)
    IE = 0: ISIDE = 0
    FOR I = 1 TO NPRESS * 2: H(I) = 0: NEXT I

590 COLOR 14: LOCATE 11, 1: PRINT SPC(78);
    LOCATE 11, 1: PRINT IE; TAB(7); ISIDE
    LOCATE 11, 12
    FOR I = 1 TO NPRESS * 2: PRINT USING "  #.###^^^^"; H(I); : NEXT I
600 LOCATE 12, 1: PRINT SPC(78);

    COLOR 12: LOCATE 12, 2: INPUT "", IE
    IF IE < 1 OR IE > NUMEL THEN 620

610 LOCATE 12, 8: INPUT "", ISIDE
    IF ISIDE < 1 OR ISIDE > 4 THEN BEEP: GOTO 610

    FOR I = 1 TO NPRESS * 2
    LOCATE CSRLIN - 1, (I - 1) * 11 + 14: INPUT "", H(I)
    IF H(I) = 0 THEN LOCATE CSRLIN - 1, (I - 1) * 11 + 13: PRINT H(I)
```

```
    NEXT I

    GOSUB 120: IF K$ <> CHR$(13) THEN 600

    '           Semble load
    FOR I = 1 TO NEDGE
      IF NEASS(I) = IE AND NSIDE(I) = ISIDE THEN
      FOR J = 1 TO NPRESS * 2: PRESS((I - 1) * NPRESS * 2 + J) = H(J): NEXT J: GOTO 590
      END IF
    NEXT I
    NEDGE = NEDGE + 1
    NEASS(NEDGE) = IE: NSIDE(NEDGE) = ISIDE
    FOR I = 1 TO NPRESS * 2: PRESS((NEDGE - 1) * NPRESS * 2 + I) = H(I): NEXT I
    GOTO 590

    '           Check if zero pressure
620 FOR I = 1 TO NEDGE
    SUM = 0
    FOR L = 1 TO NPRESS * 2: SUM = SUM + PRESS((I - 1) * NPRESS * 2 + L): NEXT L
    IF SUM <> 0 THEN 630
    FOR J = I TO NEDGE - 1
    K = J + 1
    NEASS(J) = NEASS(K): NSIDE(J) = NSIDE(K)
    FOR L = 1 TO NPRESS * 2
    PRESS((J - 1) * NPRESS * 2 + L) = PRESS((K - 1) * NPRESS * 2 + L): NEXT L
    NEXT J
    NEDGE = NEDGE - 1
630 NEXT I

    '           Re-order element-side list
    FOR I = 1 TO NEDGE - 1: II = I: IO = I: IMIN = NEASS(I) * 5 + NSIDE(I)
    FOR J = I + 1 TO NEDGE
    NDIF = NEASS(J) * 5 + NSIDE(J)
    IF NDIF < IMIN THEN IMIN = NDIF: IO = J
    NEXT J
    GOSUB 640
    NEXT I
    GOTO 560

640 IE = NEASS(IO): ISIDE = NSIDE(IO)
    FOR K = 1 TO NPRESS * 2: H(K) = PRESS((IO - 1) * NPRESS * 2 + K): NEXT K
    NEASS(IO) = NEASS(II): NSIDE(IO) = NSIDE(II)
    FOR K = 1 TO NPRESS * 2
    PRESS((IO - 1) * NPRESS * 2 + K) = PRESS((II - 1) * NPRESS * 2 + K): NEXT K
    NEASS(II) = IE: NSIDE(II) = ISIDE
    FOR K = 1 TO NPRESS * 2: PRESS((II - 1) * NPRESS * 2 + K) = H(K): NEXT K
    RETURN

    '<------------------ PRESSURE LOAD ON SCREEN ------------------------>
650 INUM = 0
    M$ = "      Nor-I      Tan-I      Nor-J      Tan-J      Nor-K      Tan-K"
660 CLS : GOSUB 140
    LOCATE 1, 1: COLOR 11: PRINT "> OUTPUT DISTRIBUTED EDGE LOAD"
    COLOR 15: PRINT "No.of edge loads ="; NEDGE
    COLOR 13: PRINT "Ele.  Side";
    PRINT LEFT$(M$, NPRESS * 2 * 11)
```

```
    PRINT STRING$(NPRESS * 2 * 11 + 10, "-"): COLOR 15

    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NEDGE THEN 670
    PRINT USING "####  ####"; NEASS(INUM), NSIDE(INUM);
    FOR J = 1 TO NPRESS * 2
    PRINT USING "  #.###^^^^"; PRESS((INUM - 1) * NPRESS * 2 + J);
    NEXT J: PRINT
    NEXT I
670 GOSUB 100: IF K$ = CHR$(27) THEN 560
    IF INUM >= NEDGE THEN 560 ELSE GOTO 660

    '<----------------- PRESSURE LOAD ON HARD COPY ----------------------->
680 M$ = "      Nor-I      Tan-I      Nor-J      Tan-J      Nor-K      Tan-K"
    CLS : GOSUB 140: COLOR 11
    PRINT "                                          "
    PRINT "  HARD COPY for DISTRIBUTED EDGE LOAD     "
    PRINT "                                          "
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 690
    LPRINT CHR$(12);
    ILINE = 0: CALL HEADING

690 GOSUB 710
    FOR I = 1 TO NEDGE
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 700
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    GOSUB 710
700 LPRINT USING "####  ####"; NEASS(I), NSIDE(I);
    FOR J = 1 TO NPRESS * 2
    LPRINT USING "  #.###^^^^"; PRESS((I - 1) * NPRESS * 2 + J);
    NEXT J: LPRINT
    NEXT I
    GOTO 560

    '          Print title
710 LPRINT
    LPRINT CHR$(18); "LOAD CASE #"; CASE$; " : "; C$
    LPRINT "** NODAL LOAD **"; CHR$(15)
    LPRINT "Ele.  Side";
    LPRINT LEFT$(M$, NPRESS * 2 * 11)
    LPRINT STRING$(NPRESS * 2 * 11 + 10, "-")
    ILINE = ILINE + 5
    RETURN

    '<--------------------------- QUIT ---------------------------------->
720 IF ICHECK = 0 THEN 730
    OPEN "O", 1, D$ + F$ + ".E" + CASE$
    WRITE #1, NEDGE
    NTOT = NPRESS * 2 * NEDGE
    FOR I = 1 TO NEDGE: WRITE #1, NEASS(I), NSIDE(I): NEXT I
    FOR I = 1 TO NTOT: WRITE #1, PRESS(I): NEXT I
    CLOSE 1
```

```
730  NEDGE = 0
     FOR I = 1 TO NEDGE: NEASS(I) = 0: NSIDE(I) = 0: NEXT I
     FOR I = 1 TO NTOT: PRESS(I) = 0: NEXT I

     GOTO 340

     '<======================== THERMAL LOADING ============================>

740  ICHECK = 0: IF ITEMP <> 1 THEN 750
     OPEN "I", 1, D$ + F$ + ".T" + CASE$
     FOR I = 1 TO NUMNP: INPUT #1, TEMP(I): NEXT I
     CLOSE 1

750  CLS : GOSUB 140
     COLOR 11: PRINT "> THERMAL LOADING"
     COLOR 12: PRINT "LOAD CASE #"; CASE$; " : "; C$
     PRINT : COLOR 14
     PRINT "E = EDIT"
     PRINT "O = OUTPUT ONSCREEN"
     PRINT "H = HARD COPY"
     PRINT
     COLOR 13: PRINT "Q = QUIT TO LOAD DATA"
     PRINT : COLOR 11: PRINT "===> SELECT ?"

760  K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 760
     IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 760
     LOCATE 12, 15: PRINT K$
     IF K$ = "E" OR K$ = "e" THEN 770
     IF K$ = "O" OR K$ = "o" THEN 800
     IF K$ = "H" OR K$ = "h" THEN 830
     IF K$ = "Q" OR K$ = "q" THEN 870
     PLAY "L40 B": GOTO 760

     '<---------------------- EDIT THERMAL LOAD --------------------------->
770  ICHECK = 1: ITEMP = 1
     CLS : GOSUB 140
     COLOR 11: PRINT ">THERMAL LOADING"
     PRINT : COLOR 12
     PRINT "    1.Press <RETURN> after entering each data to move cursor"
     PRINT "    2.Input <Node. = 0> to terminate this mode"
     LOCATE 9, 1: COLOR 15: PRINT L$
     LOCATE 13, 1: PRINT L$
     COLOR 11: LOCATE 10, 1: PRINT "  Node        Temp"
     I = 0: H(1) = 0

780  COLOR 14: LOCATE 11, 1: PRINT SPC(78);
     LOCATE 11, 2: PRINT I: LOCATE 11, 14: PRINT H(1)
790  LOCATE 12, 1: PRINT SPC(78);

     COLOR 12: LOCATE 12, 3: INPUT "", I
     IF I > NUMNP OR I < 1 THEN 750
     LOCATE 12, 15: INPUT "", H(1): IF H(1) = 0 THEN LOCATE 12, 14: PRINT H(1)
     GOSUB 120
     IF K$ <> CHR$(13) THEN 790
     TEMP(I) = H(1): GOTO 780
```

```
    '<------------------ THERMAL LOAD ON SCREEN -------------------------->
800 INUM = 0
810 CLS : GOSUB 140
    COLOR 13: LOCATE 2, 1: PRINT "** THERMAL LOADING DATA **"
    PRINT "  Node        Temp"
    COLOR 15
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NUMNP THEN 820
    LOCATE I + 3, 1: PRINT USING "######     ###.###"; INUM; TEMP(INUM)
    NEXT I
820 GOSUB 100: IF K$ = CHR$(27) THEN 750
    IF INUM >= NUMNP THEN 750 ELSE CLS : GOTO 810

    '<----------------- THERMAL LOAD ON HARD COPY ----------------------->
830 CLS : GOSUB 140: COLOR 11
    PRINT "                                        "
    PRINT "  HARD COPY for THERMAL LOADING         "
    PRINT "                                        "
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 840
    LPRINT CHR$(12);
    ILINE = 0: CALL HEADING

840 GOSUB 860
    FOR I = 1 TO NUMNP
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 850
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    GOSUB 860
850 LPRINT USING "####     ###.###"; I; TEMP(I)
    NEXT I
    GOTO 750

    '          Print title
860 LPRINT
    LPRINT CHR$(18); "LOAD CASE #"; CASE$; " : "; C$
    LPRINT "** THERMAL LOAD **"; CHR$(15)
    LPRINT "Node        Temp"
    LPRINT "---------------"
    ILINE = ILINE + 5
    RETURN

    '<--------------------------- QUIT ---------------------------------->
870 IF ICHECK = 0 THEN 880
    OPEN "O", 1, D$ + F$ + ".T" + CASE$
    FOR I = 1 TO NUMNP: WRITE #1, TEMP(I): NEXT I
    CLOSE 1

880 FOR I = 1 TO NUMNP: TEMP(I) = 0: NEXT I
    GOTO 340

    '<=============== PRESSURE LOAD FOR PLATE BENDING ====================>
890 IF IULOD = 0 THEN 900
```

```
    OPEN "I", 1, D$ + F$ + ".U" + CASE$
    FOR I = 1 TO NUMEL: INPUT #1, ULOAD(I): NEXT I
    CLOSE 1

900 CLS : GOSUB 140
    COLOR 11: PRINT "> UNIFORM LOADS"
    PRINT : COLOR 14
    PRINT "E = EDIT"
    PRINT "O = OUTPUT ONSCREEN"
    PRINT "H = HARD COPY"
    PRINT
    COLOR 13: PRINT "Q = QUIT TO LOAD DATA"
    PRINT : COLOR 11: PRINT "===> SELECT ?"
910 K$ = INKEY$: LOCATE 11, 15, 1, 0, 7: IF K$ = "" THEN 910
    IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 910
    LOCATE 11, 15: PRINT K$
    IF K$ = "E" OR K$ = "e" THEN ICHECK = 1: GOTO 920
    IF K$ = "O" OR K$ = "o" THEN 950
    IF K$ = "H" OR K$ = "h" THEN 980
    IF K$ = "Q" OR K$ = "q" THEN 1020
    PLAY "L40 B": GOTO 910

    '<---------- EDIT PRESSURE LOAD ---------->
920 CLS : GOSUB 140
    COLOR 11: PRINT "> UNIFORM LOADS"
    PRINT : COLOR 12
    PRINT "    1.Press <RETURN> after entering each data to move cursor"
    PRINT "    2.Input <Elem. = 0> to terminate this mode"
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$
    COLOR 11: LOCATE 10, 1: PRINT "  Elem.          Load"
    I = 0: H(1) = 0

930 COLOR 14: LOCATE 11, 1: PRINT SPC(78);
    LOCATE 11, 3: PRINT I: LOCATE 11, 20: PRINT H(1)
940 LOCATE 12, 1: PRINT SPC(78);

    COLOR 12: LOCATE 12, 4: INPUT "", I
    IF I > NUMEL OR I < 1 THEN 900
    LOCATE 12, 21: INPUT "", H(1): IF H(1) = 0 THEN LOCATE 12, 20: PRINT H(1)
    GOSUB 120
    IF K$ <> CHR$(13) THEN 940
    ULOAD(I) = H(1): GOTO 930

    '<---------- PRESSURE LOAD ON SCREEN ---------->
950 INUM = 0
960 CLS : GOSUB 140
    COLOR 13: LOCATE 1, 1: PRINT "** UNIFORM LOADS **"
    PRINT "  Elem.        Press"
    PRINT "  ------------------"
    COLOR 15
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NUMEL THEN 970
    PRINT USING "  ####    ##.####^^^^"; INUM; ULOAD(INUM)
    NEXT I
970 GOSUB 100: IF K$ = CHR$(27) THEN 900
```

```
    IF INUM >= NUMEL THEN 900 ELSE CLS : GOTO 960

    '<------------------ PRESSURE LOAD ON HARD COPY ----------------------->
980 CLS : GOSUB 140: COLOR 11
    PRINT "                                    "
    PRINT "  HARD COPY for PRESSURE LOADING    "
    PRINT "                                    "
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 990
    LPRINT CHR$(12);
    ILINE = 0: CALL HEADING

990 GOSUB 1010
    FOR I = 1 TO NUMNP
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 1000
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    GOSUB 1010
1000 LPRINT USING "####     ###.###"; I; ULOAD(I)
    NEXT I
    GOTO 900

    '          Print title
1010 LPRINT
    LPRINT CHR$(18); "LOAD CASE #"; CASE$; " : "; C$
    LPRINT "** PRESSURE LOAD **"; CHR$(15)
    LPRINT "Node        PRESS"
    LPRINT "-----------------"
    ILINE = ILINE + 5
    RETURN

    '<--------------------------- QUIT ---------------------------------->
1020 IF ICHECK = 0 THEN 1030
    IULOD = 1
    OPEN "O", 1, D$ + F$ + ".U" + CASE$
    FOR I = 1 TO NUMEL: WRITE #1, ULOAD(I): NEXT I
    CLOSE 1
    ICHECK = 0
1030 FOR I = 1 TO NUMEL: ULOAD(I) = 0: NEXT I
    GOTO 340

    '<======================== GRAVITY LOAD ================================>
1040 IF IGRAV <> 1 THEN 1050
    OPEN "I", 1, D$ + F$ + ".G" + CASE$
    INPUT #1, THETA, GRAVY
    CLOSE 1

1050 CLS : GOSUB 140
    LOCATE 1, 1: COLOR 11: PRINT "> GRAVITY LOADING"
    COLOR 12: PRINT "LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT "ANGLE(degree) = "; THETA
    PRINT "FRACTION      = "; GRAVY
```

```
    GOSUB 120: IF K$ = CHR$(13) THEN 1060
    COLOR 12: LOCATE 5, 18: INPUT "", THETA
    LOCATE 6, 18: INPUT "", GRAVY
    GOTO 1050

1060 IF GRAVY <> 0 THEN 1070
    IF IGRAV = 1 THEN KILL D$ + F$ + ".G" + CASE$
    IGRAV = 0: GOTO 1080
1070 IGRAV = 1
    OPEN "O", 1, D$ + F$ + ".G" + CASE$
    WRITE #1, THETA, GRAVY
    CLOSE 1
1080 GOTO 340

    '<================== PRESSURE LOAD FOR BRICK ELEMENT ==================>

1090 ICHECK = 0: IF IEDGE <> 1 THEN 1100
    OPEN "I", 1, D$ + F$ + ".E" + CASE$
    INPUT #1, NEDGE
    FOR I = 1 TO NEDGE: INPUT #1, NEASS(I), NSIDE(I), PRESS(I): NEXT I
    CLOSE 1

1100 CLS : GOSUB 140
    COLOR 11: PRINT "> PRESSURE LOAD"
    COLOR 12: PRINT "LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT "E = EDIT"
    PRINT "O = OUTPUT ONSCREEN"
    PRINT "H = HARD COPY"
    PRINT
    COLOR 13: PRINT "Q = QUIT TO LOAD DATA"
    PRINT : COLOR 11: PRINT "===> SELECT ?"

1110 K$ = INKEY$: LOCATE 12, 15, 1, 0, 7: IF K$ = "" THEN 1110
    LOCATE 12, 15: PRINT K$
    IF K$ = "E" OR K$ = "e" THEN 1120
    IF K$ = "O" OR K$ = "o" THEN 1190
    IF K$ = "H" OR K$ = "h" THEN 1220
    IF K$ = "Q" OR K$ = "q" THEN 1260
    PLAY "L40 B": GOTO 1110

    '<---------------------- EDIT PRESSURE LOAD ---------------------------->
1120 ICHECK = 1: IEDGE = 1
    CLS : GOSUB 140
    COLOR 11: PRINT "> EDIT PRESSURE LOAD"
    PRINT : COLOR 12
    PRINT "   1.Press <RETURN> after entering each data to move cursor"
    PRINT "   2.Input <Elem. no. = 0> to terminate this mode"
    LOCATE 9, 1: COLOR 15: PRINT L$
    LOCATE 13, 1: PRINT L$
    COLOR 11
    LOCATE 10, 1: PRINT " Ele.  Side    PRESSURE"
    IE = 0: ISIDE = 0: XPRESS = 0

1130 COLOR 14: LOCATE 11, 1: PRINT SPC(78);
    LOCATE 11, 1: PRINT IE; TAB(7); ISIDE; TAB(15); XPRESS
```

```
1140 LOCATE 12, 1: PRINT SPC(78);

     COLOR 12: LOCATE 12, 2: INPUT "", IE
     IF IE < 1 OR IE > NUMEL THEN 1160
1150 LOCATE 12, 8: INPUT "", ISIDE
     IF ISIDE < 1 OR ISIDE > 6 THEN BEEP: GOTO 1150
     LOCATE 12, 16: INPUT "", XPRESS: IF XPRESS = 0 THEN LOCATE 12, 16: PRINT "0"

     GOSUB 120: IF K$ <> CHR$(13) THEN 1140

     '            SEMBLE LOAD

     FOR I = 1 TO NEDGE
     IF NEASS(I) = IE AND NSIDE(I) = ISIDE THEN PRESS(I) = XPRESS: GOTO 1130
     NEXT I
     NEDGE = NEDGE + 1
     NEASS(NEDGE) = IE: NSIDE(NEDGE) = ISIDE: PRESS(NEDGE) = XPRESS
     GOTO 1130

     '            CHECK IF ZERO PRESSURE

1160 FOR I = 1 TO NEDGE
     IF PRESS(I) <> 0 THEN 1170
     FOR J = I TO NEDGE - 1
     K = J + 1
     NEASS(J) = NEASS(K): NSIDE(J) = NSIDE(K): PRESS(J) = PRESS(K)
     NEXT J
     NEDGE = NEDGE - 1
1170 NEXT I

     '            RE-ORDER ELEMENT LIST

     FOR I = 1 TO NEDGE - 1: II = I: IO = I: IMIN = NEASS(I) * 7 + NSIDE(I)
     FOR J = I + 1 TO NEDGE
     NDIF = NEASS(J) * 7 + NSIDE(J)
     IF NDIF < IMIN THEN IMIN = NDIF: IO = J
     NEXT J
     GOSUB 1180
     NEXT I

     GOTO 1100

1180 IE = NEASS(IO): ISIDE = NSIDE(IO): XPRESS = PRESS(IO)
     NEASS(IO) = NEASS(II): NSIDE(IO) = NSIDE(II): PRESS(IO) = PRESS(II)
     NEASS(II) = IE: NSIDE(II) = ISIDE: PRESS(II) = XPRESS
     RETURN

     '<-------------------- PRESSURE LOAD ON SCREEN ----------------------->
1190 INUM = 0

1200 CLS : GOSUB 140
     LOCATE 1, 1: COLOR 11: PRINT "> OUTPUT PRESSURE LOAD"
     COLOR 15: PRINT "No.of pressure loads :"; NEDGE
     COLOR 13: PRINT " No.   Ele.  Side    Pressure"
     PRINT STRING$(78, "-"): COLOR 15
```

```
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NEDGE THEN 1210
    LOCATE I + 4, 1: PRINT USING "####  "; INUM, NEASS(INUM), NSIDE(INUM);
    PRINT USING " #######.###"; PRESS(INUM)
    NEXT I
1210 GOSUB 100: IF K$ = CHR$(27) THEN 1100
    IF INUM >= NEDGE THEN 1100 ELSE GOTO 1200

    '<------------------ PRESSURE LOAD ON HARD COPY ----------------------->
1220 CLS : GOSUB 140: COLOR 11
    PRINT "                                        "
    PRINT "  HARD COPY for PRESSURE LOADING        "
    PRINT "                                        "
    H$ = "DATA"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 1230
    LPRINT CHR$(12);
    ILINE = 0: CALL HEADING

1230 GOSUB 1250
    FOR I = 1 TO NEDGE
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 1240
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    GOSUB 1250
1240 LPRINT USING " ####   ###     #.#####^^^^"; NEASS(I); NSIDE(I); PRESS(I)
    NEXT I
    GOTO 1100

    '          Print title
1250 LPRINT
    LPRINT CHR$(18); "LOAD CASE #"; CASE$; " : "; C$
    LPRINT "** PRESSURE LOAD **"; CHR$(15)
    LPRINT "Elem.  Side        Pressure"
    LPRINT "---------------------------"
    ILINE = ILINE + 5
    RETURN

    '<--------------------------- Quit ---------------------------------->
1260 IF ICHECK = 0 THEN 1270
    OPEN "O", 1, D$ + F$ + ".E" + CASE$
    WRITE #1, NEDGE
    FOR I = 1 TO NEDGE: WRITE #1, NEASS(I), NSIDE(I), PRESS(I): NEXT I
    CLOSE 1

1270 NEDGE = 0
    FOR I = 1 TO NEDGE: NEASS(I) = 0: NSIDE(I) = 0: PRESS(I) = 0: NEXT I

    GOTO 340


DEFINT I-N
DEFDBL A-H, O-Z
SUB FIRST STATIC
    COLOR 12
```

```
     LOCATE 10, 25: PRINT "PLEASE ADJUST PAPER"
     LOCATE 11, 25: PRINT "Press any key to continue"
7000 K$ = INKEY$: IF K$ = "" THEN 7000
     LPRINT CHR$(27) + "C" + CHR$(66);
     COLOR 28
     LOCATE 10, 25: PRINT "┌──────────────────────────┐"
     LOCATE 11, 25: PRINT "│                          │"
     LOCATE 12, 25: PRINT "└──────────────────────────┘"
     LOCATE 11, 29: COLOR 12: PRINT "WAIT : data being print"
     LPRINT CHR$(15);                                  'set to compress mode
     WIDTH "LPT1:", 132                                'set to 132 column
END SUB

DEFINT I-N
DEFDBL A-H, O-Z
SUB HEADING STATIC
SHARED ILINE, IPAGE, F$, T$, H$
     ILINE = ILINE + 4: IPAGE = IPAGE + 1
     FOR I = 1 TO 130: LPRINT "="; : NEXT I
     LPRINT CHR$(18);
     LPRINT CHR$(27) + "W" + CHR$(1); "SUPATFEAP"; CHR$(27) + "W" + CHR$(0);
     LPRINT CHR$(15);
     LPRINT TAB(91); "<"; H$; "> Page"; IPAGE
     LPRINT "PROJECT  : "; T$; : LPRINT TAB(112); "FILE NAME: "; F$
     FOR I = 1 TO 130: LPRINT "="; : NEXT I: LPRINT
END SUB
```

```
'*************************************************************************
'*    file name "ORDER"                                                  *
'*                                                                       *
'*    subprogram to compute no. degree of freedom at each node           *
'*************************************************************************

DEFDBL A-H, O-Z
DEFINT I-N

COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
COMMON NCASE, NUMSEG, NGAUSS, KORDER, T$, E$, MAXVOL, F$()
COMMON IOPTION
COMMON LSTMAX, NEQMAX, KMAX
COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
COMMON NPDOF(), IORDER()

GOTO 120

'<---------------------- SUBROUTINE AREA ----------------------------->
100 LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$: RETURN
110 LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$
    LOCATE 25, 70: PRINT FRE(-1); : RETURN

'<-------------------------------------------------------------------->
120 DD$ = D$

DIM NP(NUMEL, NNPE), NBCJ(NUMNP), KODE(NUMNP, NPDOF), NPDOF(NUMNP)
DIM IORDER(NUMEL), OORDER(NUMEL), SIDE(NUMEL, 6)

OPEN "I", 1, D$ + F$ + ".DIR"
FOR I = 1 TO 12: IF NOT EOF(1) THEN INPUT #1, F$(I)
NEXT I: CLOSE 1
F$(11) = DATE$
STARTtime! = TIMER

OPEN "I", 1, D$ + F$ + ".BOU"
FOR I = 1 TO NUMNP
INPUT #1, NBCJ(I)
NEXT I
FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
INPUT #1, KODE(I, J)
NEXT J, I
CLOSE 1

OPEN "I", 1, D$ + F$ + ".ELE"
FOR I = 1 TO NUMEL
FOR J = 1 TO NNPE
INPUT #1, NP(I, J)
NEXT J, I
CLOSE 1

CLS : GOSUB 100: GOSUB 110
LOCATE 3, 1
COLOR 12: PRINT "MASTER FILE NAME ---> "; F$
COLOR 15
IF NTYPE = 1 THEN PRINT "(PLANE STRESS PROBLEM)"
```

```
    IF NTYPE = 2 THEN PRINT "(PLANE STRAIN PROBLEM)"
    IF NTYPE = 3 THEN PRINT "(PLATE BENDING PROBLEM)"
    IF NTYPE = 4 THEN PRINT "(BRICK ELEMENT PROBLEM)"
    COLOR 14
    PRINT NUMEL; "Elements"; " ,"; NUMNP; "Nodes"; " ,"; NNPE; "Nodes/element";
    PRINT " ,"; NPDOF; "d.o.f/node"; " ,"; NDIME; "Coordinates"

    PRINT : COLOR 14
    PRINT "<<< COMPUTE DEGREE OF FREEDOM AT EACH NODE >>>"
    COLOR 15: PRINT TAB(5); "Node. No ..."
    FOR I = 1 TO NUMNP
    NPDOF(I) = NPDOF
    NEXT I

    FOR I = 1 TO NUMNP
    COLOR 15: LOCATE 8, 17: PRINT I;
    IF NBCJ(I) = 0 THEN 130
    FOR J = 1 TO NPDOF
    NPDOF(I) = NPDOF(I) - KODE(I, J)
    NEXT J
    IF NPDOF(I) < 0 THEN PRINT "ERROR": STOP
130 NEXT I

    '*************************************************************************
    '*    SUBPROGRAM TO RE-ORDER ELEMENTS NUMBERING                          *
    '*                                                                       *
    '*            IORDER-ARRAY CONTAINING NEW ORDER                          *
    '*************************************************************************

    COLOR 14: PRINT
    PRINT "<<< ELEMENT ORDERING SCHEME >>>"
    COLOR 15: PRINT TAB(5); "Elements ..."

    IF KORDER = 0 THEN 140 ELSE 150

140 FOR I = 1 TO NUMEL
    IORDER(I) = I
    COLOR 15: LOCATE 10, 17: PRINT I;
    NEXT I
    GOTO 170

150 B = NUMNP + 1: Q = 0
    NSIDE = 4
    IF NSIDE = 4 AND NNPE = 8 THEN IOFF = 2 ELSE IOFF = 1

    IF NTYPE = 4 THEN NSIDE = 6

    '------------------------ STEP 1 ----------------------------------
    '        ELEMENTS DESCRIBED BY SIDE NUMBERS BASE ON B

    COLOR 13
    FOR I = 1 TO NUMEL
    LOCATE 10, 17: PRINT I;
    K = NP(I, 1)
    FOR J = 1 TO (NSIDE - 1)
    SIDE(I, J) = B * NP(I, J + (J - 1) * (IOFF - 1)) + NP(I, J + (J - 1) * (IOFF - 1) + IOFF)
```

```
    NEXT J
    SIDE(I, NSIDE) = B * NP(I, NSIDE + (NSIDE - 1) * (IOFF - 1)) + K
    NEXT I

    '------------------------ STEP 2 -------------------------------
    'ELEMENTS DESCRIBED BY THE (NEGATED) INTERFACING ELEMENTS
    'ICR IS THE STARTING POINT FOR THE REORDERING SCHEME ;

    COLOR 11
    FOR I = 1 TO NUMEL
    LOCATE 10, 17: PRINT I;
    IP = 0
    FOR J = 1 TO NSIDE
    K = NP(I, J)
    IF K <= 0 THEN 160
    X = SIDE(I, J)
    M = INT(X / B)
    X = M + ((X - (M * B)) * B)
    FOR L = I + 1 TO NUMEL
    FOR M = 1 TO NSIDE
    IF SIDE(L, M) = X THEN NP(L, M) = -I: NP(I, J) = -L: GOTO 160
    NEXT M, L
    NP(I, J) = 0: IP = IP + 1
160 NEXT J
    IF IP > IQ THEN IQ = IP: ICR = I
    NEXT I
    '------------------------ STEP 3 -------------------------------
    FOR I = 1 TO NUMEL: IORDER(I) = 0: OORDER(I) = 0: NEXT I
    IBAND = 1
    K = 1
    M = 1
    IORDER(1) = ICR
    OORDER(0) = 1

    FOR I = 2 TO NUMEL
    LOCATE 10, 17: PRINT I;
    FOR J = 1 TO NSIDE
    L = ABS(NP(ICR, J))
    IF OORDER(L) = 0 THEN M = M + 1: IORDER(M) = L: OORDER(L) = 1
    NEXT J
    OORDER(ICR) = 1
    IF M - K + 1 > IBAND THEN IBAND = M - K + 1
    K = K + 1
    ICR = IORDER(K)
    NEXT I

    '       IORDER[.] CONTAINS THE PERMUTATION OF INITIAL ORDER

    '                END OF ORDER ELEMENTS
    '
170 FINISHtime! = TIMER
    TOTALtime! = FINISHtime! - STARTtime!
    F$(13) = STR$(TOTALtime!)
    CHAIN "WAVE"
```

```
'*************************************************************************
'*                                                                       *
'*                SUBPROGRAM WAVE                                        *
'*                file name "WAVE"                                       *
'*                                                                       *
'*     Modified for maximum storage and RANDOM FILE as wave  record      *
'*                                                                       *
'*************************************************************************

DEFINT I-N
DEFDBL A-H, O-Z

COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
COMMON NCASE, NUMSEG, NGAUSS, KORDER, T$, E$, MAXVOL, F$()
COMMON IOPTION
COMMON LSTMAX, NEQMAX, KMAX
'      Maximum nodes repersented in segment
'      Maximum equations repersented in segment
'      Maximum usage storage
COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
COMMON NPDOF(), IORDER()

IDA = NUMNP  'Total number of nodal points
IDB = NUMEL  'Total number of elements
IDC = NNPE   'Number of nodal points per element
IDD = 800  'Max number of equations in any tape segment
IDE = 300  'Max number of new elements added in any tape segment
IDF = 400  'Max number of nodal points represented in any tape segment

DD$ = D$

'                 Dimension array

DIM NP(NUMEL, NNPE)
DIM IDIAG1(IDD), IDIAG2(IDD), MOVE(IDD), IEQS(IDD)
DIM NPR(IDA)
DIM IELE(IDE), NPT(IDE, IDC)
DIM NPIX(IDA), LISTC(IDA), LISTCI(IDA), NPRC(IDA), LISTI(IDA)
DIM LISTL(IDF), LSTO(IDF)

IF LEN(F$(4)) = 0 THEN STOP                          'F$.ELE

TOTALtime! = VAL(F$(13))
STARTtime! = TIMER

OPEN "I", 1, D$ + F$ + ".ELE"
FOR I = 1 TO NUMEL
FOR J = 1 TO NNPE
INPUT #1, NP(I, J)
NEXT J, I
CLOSE 1

L$ = STRING$(79, "=")
GOTO 120

'<---------------------- SUBROUTINE AREA ----------------------------->
```

```
100  LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$: RETURN
110  LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$: RETURN

     '<-------------------------------------------------------------------->

120  GOSUB 100: GOSUB 110
     COLOR 14
     LOCATE 11, 1: PRINT "<<< COMPUTE FOR WAVE SEGMENT >>>"
     COLOR 15: PRINT TAB(5); "Elements ..."


     '<----------------- Compute no. degree of freedom --------------------->
     NUMEQ = 0
     FOR I = 1 TO NUMNP
     NUMEQ = NUMEQ + NPDOF(I)
     NEXT I

     'CALCULATE LISTC(COMPLETE LIST OF NODAL POINTS IN ORDER OF THEIR REDUCTION

     FOR I = 1 TO NUMNP
     NPIX(I) = 0
     NEXT I

     FOR I = 1 TO NUMEL
     FOR J = 1 TO NNPE
     J1 = NP(I, J)
     NPIX(J1) = NPIX(J1) + 1
     NEXT J, I

     '--------- re-order node to be used w.r.t order of element to be assembled
     '         IORDER(NUMEL)=order that elements will be assembled
     '         LISTC(NUMNP) =order that nodes will be assembled
     J2 = 0
     FOR I = 1 TO NUMEL
     I1 = IORDER(I)
     FOR J = 1 TO NNPE
     J1 = NP(I1, J)
     NPIX(J1) = NPIX(J1) - 1
     IF NPIX(J1) > 0 THEN 130
     J2 = J2 + 1
     LISTC(J2) = J1
130  NEXT J
     NEXT I

     '               START POINT FOR WAVE RECORD

     ISEG = 0
     KMAX = 0
     IRDRC = 0
     ICOMP = 0
     MOVEX = 0
     LISTX = 0
     LSTMAX = 0
     NEQMAX = 0
     IREC = 0
     WAVECOUNT = 0
```

```
    FOR I = 1 TO NUMNP
    NPR(I) = 0
    NPIX(I) = 0
    LISTI(I) = 0
    I1 = LISTC(I)
    LISTCI(I1) = I
    NEXT I

    FOR I = 1 TO NUMEL
    FOR J = 1 TO NNPE
    J1 = NP(I, J)
    NPIX(J1) = NPIX(J1) + 1
    NEXT J
    NP(I, 1) = -NP(I, 1)
    NEXT I

    ' BEGIN NEW SEGMENT

140 ICODE = 0
    IELEX = 0
    ISEG = ISEG + 1
    COLOR 14
    LOCATE 11, 30: PRINT "#"; RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1); " >>>"
    COLOR 15

    ' ADD NEW ELEMENT TO CURRENT SEGMENT

150 IRDRC = IRDRC + 1
    LOCATE 12, 17: PRINT IRDRC;
    LMENT = IORDER(IRDRC)
    NEWX = 0
    IELEX = IELEX + 1
    IF IELEX > IDE THEN 7009
    IELE(IELEX) = LMENT
    NP(LMENT, 1) = -NP(LMENT, 1)

    ' DETERMINE NEW NODAL POINTS ADDED TO LIST FROM CURRENT ELEMENT

    FOR I = 1 TO NNPE
    NPLI = NP(LMENT, I)
    NPIX(NPLI) = NPIX(NPLI) - 1
    IF NPR(NPLI) < 0 THEN 7002
    IF NPR(NPLI) > 0 THEN 160

    NPR(NPLI) = 1
    LISTX = LISTX + 1
    IF LISTX > IDF THEN 7007
    LSTO(LISTX) = NPLI
    NEWX = NEWX + 1

160 NEXT I

    ' CALCULATE NPR AND LISTI ARRAYS FOR CURRENT LSTO

170 FOR I = 1 TO NUMNP
```

```
    LISTC(I) = ABS(LISTC(I))
    NEXT I

    FOR I = 1 TO LISTX
    I1 = LST0(I)
    I2 = LISTCI(I1)
    LISTC(I2) = -LISTC(I2)
    NEXT I

    LST1 = 0
    I1 = 1
    I3 = 1
    FOR I = 1 TO NUMNP
    IF LISTC(I) > 0 THEN 180
    I2 = ABS(LISTC(I))
    IF LST1 = 0 THEN LST1 = I2
    IF NPR(I2) < 0 THEN 7003
    NPR(I2) = I1
    LISTI(I2) = I3
    I1 = I1 + NPDOF(I2)
    I3 = I3 + 1
180 NEXT I
    NEQ = I1 - 1
    IF NEQ > IDD THEN 7008

    ' CALCULATE HEIGHT OF EACH COLUMN FOR CURRENT ARRAY

    FOR I = 1 TO NEQ
    IDIAG2(I) = 1
    NEXT I

    FOR I = 1 TO NUMEL
    IF NP(I, 1) < 0 THEN 210
    FOR J = 1 TO NNPE
    NPJ = ABS(NPR(NP(I, J)))
    IF NPJ = 0 THEN 200

    FOR K = 1 TO NNPE
    NPK = NPR(NP(I, K))
    IF NPJ > NPK THEN 190
    NPK = NPK + NPDOF(NP(I, K)) - 1
    IHGHT = NPK - NPJ + 1
    IF IDIAG2(NPK) > IHGHT THEN 190
    ICOL = NPK
    LEND = NPDOF(NP(I, K))
    FOR L = 1 TO LEND
    IDIAG2(ICOL) = IHGHT
    ICOL = ICOL - 1
    IHGHT = IHGHT - 1
    NEXT L
190 NEXT K
200 NEXT J
210 NEXT I

    ' CALCULATE IDIAG FOR CURRENT ARRAY
```

```
      FOR I = 2 TO NEQ
      IDIAG2(I) = IDIAG2(I - 1) + IDIAG2(I)
      NEXT I
      KVOL = IDIAG2(NEQ)

      ' CHECK ON STORAGE NEEDED FOR CURRENT ARRAY

      IF KVOL > MAXVOL THEN 220
      IF IRDRC = NUMEL THEN 240
      IF ICODE <> 0 THEN 220

      ' TOTAL STORAGE NOT UTILIZED. RETURN FOR ADDITIONAL ELEMENT.

      GOTO 150

220   IF NPIX(LST1) = 0 THEN 230
      ICODE = 1

      ' NEED AT LEAST ONE NEW ELEMENT TO COMPLETE THE FIRST ROW OF MATRIX

      GOTO 150

230   IF ICODE = 1 THEN 240
      IF KVOL <= MAXVOL THEN 240
      IF IELEX = 1 THEN 240

      ' STORAGE REQUIREMENT IS EXCEEDED AND LAST ELEMENT ADDED IS NOT
      ' NEEDED TO COMPLETE FIRST ROW OF MATRIX

      ICODE = 2
      LMENT = IELE(IELEX)
      IELE(IELEX) = 0
      IELEX = IELEX - 1

      FOR I = 1 TO NNPE
      I1 = NP(LMENT, I)
      NPIX(I1) = NPIX(I1) + 1
      NEXT I
      NP(LMENT, 1) = -NP(LMENT, 1)

      IBGN = LISTX - NEWX + 1
      IF NEWX = 0 THEN 7004
      FOR I = IBGN TO LISTX
      I1 = LSTO(I)
      NPR(I1) = 0
      LISTI(I1) = 0
      LSTO(I) = 0
      NEXT I
      LISTX = LISTX - NEWX
      IRDRC = IRDRC - 1
      NEWX = 0

      ' RETURN TO REMOVE LAST ELEMENT

      GOTO 170
      '=================================================================
```

```
      '
      ' CURRENT TAPE SEGMENT COMPLETED
      '
      '=================================================================

  240 IF KMAX < KVOL THEN KMAX = KVOL


      FOR I = 1 TO IELEX
      I1 = IELE(I)
      FOR J = 1 TO NNPE
      NPT(I, J) = NP(I1, J)
      NEXT J, I


      ' FORMULATE NEW LIST AND MOVE ARRAYS


      IF MOVEX = 0 THEN 260

      FOR I = 1 TO MOVEX
      MOVE(I) = 0
      NEXT I

      IEMPT = ICOMP
      IBGN = LCOMP + 1
      JOLD = ICOMP

      FOR I = 1 TO LSTMAX
      IF I <= LCOMP THEN 250
      I1 = LISTL(I)
      IF I1 = 0 THEN 260
      IF NPR(I1) <= 0 THEN 7005
      JNEW = NPR(I1) - 1
      JEND = NPDOF(I1)
      FOR J = 1 TO JEND
      JNEW = JNEW + 1
      JOLD = JOLD + 1
      MOVE(JOLD) = JNEW
      NEXT J
  250 LISTL(I) = 0
      NEXT I


      ' SLIDE IDIAG TO LOWEST POSITION


  260 IDELT = MAXVOL - IDIAG2(NEQ)
      IF IDELT < 0 THEN 7006
      JUSTFY = 0
      IF MOVEX = 0 THEN 280
      IDLT = IDELT
      FOR I = 1 TO MOVEX
      IM = MOVE(I)
      IF IM = 0 THEN 270
      IDT = IDIAG1(I) - IDIAG2(IM)
```

```
    IF IDT < IDLT THEN IDLT = IDT
270 NEXT I

    IF IDLT = IDELT THEN 280
    JUSTFY = IDELT - IDLT
    IDELT = IDLT

280 FOR I = 1 TO NEQ
    IDIAG2(I) = IDIAG2(I) + IDELT
    NEXT I

    ' DETERMINE NUMBER OF FULLY ASSEMBLED EQUATIONS AND IEQS ARRAY


    IF IRDRC = NUMEL THEN JUSTFY = 0
    LCOMP = 0
    ICOMP = 0
    FOR I = 1 TO NUMNP
    IF LISTI(I) <= 0 THEN 290
    I1 = LISTI(I)
    LISTL(I1) = I
    J1 = 0
    JEND = I - 1
    FOR J = 1 TO JEND
    J1 = J1 + NPDOF(J)
    NEXT J
    JBGN = NPR(I)
    JEND = JBGN + NPDOF(I) - 1
    FOR J = JBGN TO JEND
    J1 = J1 + 1
    IEQS(J) = J1
    NEXT J

    IF NPIX(I) <> 0 THEN 290
    LCOMP = LCOMP + 1
    ICOMP = ICOMP + NPDOF(I)
290 NEXT I

    IF LISTX > LSTMAX THEN LSTMAX = LISTX

    IF NEQ > NEQMAX THEN NEQMAX = NEQ

    '-------------------------------------------------------------------
    '       DISK SEGMENT OUTPUT
    '-------------------------------------------------------------------
    '       ISEG,IELEX,NEQ,ICOMP,MOVEX,JUSTFY
    '       IDIAG1,IDIAG2,MOVE,IEQS,NPR,IELE,NPT
    EXT$ = STR$(ISEG)
    EXT$ = RIGHT$(EXT$, LEN(EXT$) - 1)
    OPEN "B", 2, DD$ + "WVE." + EXT$

    PUT #2, , ISEG
    PUT #2, , IELEX
    PUT #2, , NEQ
    PUT #2, , ICOMP
    PUT #2, , MOVEX
```

```
    PUT #2, , JUSTFY
    FOR I = 1 TO MOVEX: PUT #2, , IDIAG1(I): NEXT I
    FOR I = 1 TO NEQ: PUT #2, , IDIAG2(I): NEXT I
    FOR I = 1 TO MOVEX: PUT #2, , MOVE(I): NEXT I
    FOR I = 1 TO NEQ: PUT #2, , IEQS(I): NEXT I
    FOR I = 1 TO NUMNP: PUT #2, , NPR(I): NEXT I
    FOR I = 1 TO IELEX: PUT #2, , IELE(I): NEXT I
    FOR I = 1 TO IELEX
    FOR J = 1 TO NNPE
    PUT #2, , NPT(I, J)
    NEXT J, I

    CLOSE 2

WAVECOUNT = WAVECOUNT + 6 + 2 * MOVEX + 2 * NEQ + NUMNP + IELEX + IELEX * NNPE

    IF IRDRC = NUMEL THEN 300

    ' PREPARE FOR NEXT TAPE SEGMENT

    I1 = 0
    IBGN = LCOMP + 1
    FOR I = IBGN TO LISTX
    I1 = I1 + 1
    LSTO(I1) = LISTL(I)
    LSTO(I) = 0
    NEXT I

    FOR I = 1 TO NEQ
    IDIAG1(I) = IDIAG2(I) + JUSTFY
    IDIAG2(I) = 0
    NEXT I

    FOR I = 1 TO LCOMP
    I1 = LISTL(I)
    NPR(I1) = -1
    LISTI(I1) = -1
    NEXT I

    MOVEX = NEQ
    LISTX = LISTX - LCOMP
    GOTO 140

300 'CLS : GOSUB 100: GOSUB 110: COLOR 11
    'PRINT "Number of segment ..............................."; ISEG
    'PRINT "Maximum nodes repersented in segment ..........."; LSTMAX
    'PRINT "Maximum equations repersented in segment ......."; NEQMAX
    'PRINT "Maximum usage storage..........................."; KMAX

    NUMSEG = ISEG

    '  THIS PART FOR PRINT WAVE

'   OPEN "O", 1, D$ + F$ + ".CON"
'   WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
'   WRITE #1, NCASE
```

```
'     WRITE #1, NUMSEG, NGAUSS, KORDER
'     WRITE #1, T$
'     WRITE #1, E$
'     WRITE #1, MAXVOL
'     CLOSE 1

      FINISHtime! = TIMER
      TOTALtime! = FINISHtime! - STARTtime! + TOTALtime!
      F$(13) = STR$(TOTALtime!)

      IF IOPTION = 1 THEN CHAIN "PRTWAVE"
      CHAIN "STIFF"

      END
      '=====================================================================

      '                    ERROR MESSAGE

7002 PRINT "ERROR IN NPR-ARRAY      CODE 7002"
     STOP

7003 PRINT "ERROR IN NPR-ARRAY      CODE 7003"
     STOP

7004 PRINT "NEWX = 0                CODE 7004"
     STOP

7005 PRINT "ERROR IN NPR-ARRAY      CODE 7005"
     STOP

7006 PRINT "MAXVOL TOO SMALL FOR SPECIFIED MESH        CODE 7006"
     PRINT "MUST BE INCREASED AT LEAST TO "; IDIAG2(NEQ)
     STOP

7007 PRINT "NUMBER OF NODAL POINTS IN CURRENT TAPE SEGMENT"
     PRINT "EXCEEDS DIMENSION IDF WHICH EQUALS "; IDF; ""
     PRINT "MUST CHANGE DIMENSIONS OF ARRAYS:LISTL AND LSTO      CODE 7007"
     STOP

7008 PRINT "NUMBER OF EQUATIONS IN CURRENT TAPE SEGMENT"
     PRINT "EXCEEDS THE DIMENSION IDD WHICH EQUALS "; IDD
     PRINT "MUST CHANGE DIMENSIONS OF ARRAYS IDIAG1,IDIAG2,IEQS,MOVE CODE 7008"
     STOP

7009 PRINT "NUMBER OF ELEMENTS IN CURRENT TAPE SEGMENT"
     PRINT "EXCEEDS THE DIMENSION IDE WHICH EQUALS "; IDE
     PRINT "MUST CHANGE DIMENSION OF ARRAYS:IELE,NPT             CODE 7009"
     STOP
```

```
'*************************************************************************
'*                                                                       *
'*               file name "STIFF"                                       *
'*   Subprogram to generate stiffness matrix only                        *
'*   NTYPE = 1 ----> PLANE STRESS FINITE ELEMENT                         *
'*           2 ----> PLANE STRAIN FINITE ELEMENT                         *
'*           3 ----> PLATE BENDING FINITE ELEMENT                        *
'*           4 ----> BRICK FINITE ELEMENT                                *
'*                                                                       *
'*************************************************************************
DEFINT I-N
DEFDBL A-H, O-Z

COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
COMMON NCASE, NUMSEG, NGAUSS, IORDER, T$, E$, MAXVOL, F$()
COMMON IOPTION
COMMON LSTMAX, NEQMAX, KMAX
'      Maximum nodes repersented in segment
'      Maximum equations repersented in segment
'      Maximum usage storage
COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
COMMON NPDOF(), IORDER()

COMMON COORD(), NP(), MATNO(), PROPS(), NBCJ(), KODE()

DD$ = D$

DIM COORD(NUMNP * NDIME), NP(NUMEL, NNPE), MATNO(NUMEL), PROPS(10, 7)
DIM NBCJ(NUMNP), KODE(NUMNP, NPDOF)

DEF SEG = VARSEG(COORD(1))
BLOAD D$ + F$ + ".COO", VARPTR(COORD(1))

OPEN "I", 1, D$ + F$ + ".BOU"
FOR I = 1 TO NUMNP
INPUT #1, NBCJ(I)
NEXT I
FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
INPUT #1, KODE(I, J)
NEXT J, I
CLOSE 1

OPEN "I", 1, D$ + F$ + ".ELE"
FOR I = 1 TO NUMEL
FOR J = 1 TO NNPE
INPUT #1, NP(I, J)
NEXT J, I
FOR I = 1 TO NUMEL
INPUT #1, MATNO(I)
NEXT I
CLOSE 1

OPEN "I", 1, D$ + F$ + ".MAT"
FOR I = 1 TO NMATS
FOR J = 1 TO 5
INPUT #1, PROPS(I, J)
```

```
    NEXT J, I
    CLOSE 1

    STARTtime! = TIMER
    GOTO 150

    '------------------ Subroutine area ----------------------------------
100 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
110 LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN
    '---------------------------------------------------------------------

150 GOSUB 100: GOSUB 110
    COLOR 14
    LOCATE 13, 1: PRINT "<<< FORM ELEMENT STIFFNESS MATRIX >>>"
    COLOR 15: PRINT TAB(5); "Elements ..."

    '<------ Integration constants ---------------------------------------->

    DIM ESTIF(24, 24), WGT(3, 3), PLACE(3, 3)
    FOR I = 1 TO 3: FOR J = 1 TO 3: READ WGT(I, J): NEXT J, I
    FOR I = 1 TO 3: FOR J = 1 TO 3: READ PLACE(I, J): NEXT J, I
    DATA 2      ,1                  ,0.555555555555556
    DATA 0      ,1                  ,0.888888888888889
    DATA 0      ,0                  ,0.555555555555556
    DATA 0      ,-0.577350269189626 ,-0.774596669241483
    DATA 0      , 0.577350269189626 ,0
    DATA 0      ,0                  , 0.774596669241483

    DIM DMATX(6, 6), SHAPE(8), DERIV(3, 8), GPCOD(2), ELCOD(3, 8), XJACM(3, 3)
    DIM XJACI(3, 3), CARTD(3, 8), BMATX(6, 24), DBMAT(6, 24), STRAN(5), STRES(5)
    DIM LNODE(3), PGASH(2), DGASH(2), IEQ(24)

    IF NTYPE < 3 THEN NSTRE = 3
    IF NTYPE = 3 THEN NSTRE = 5
    IF NTYPE = 4 THEN NSTRE = 6

    NEDOF = NNPE * NPDOF
    ISEG = 0: IELEM = 0: STIFFCOUNT = 0

    '<--------------------- LOOP FOR EACH SEGMENT ------------------------->

1000 ISEG = ISEG + 1: ISEG$ = RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1)

    OPEN "B", 1, DD$ + "" + "WVE." + ISEG$
    GET #1, , ISEG
    GET #1, , IELEX
    CLOSE 1

    OPEN "B", 1, DD$ + "GST." + ISEG$
    '<------------- COMPUTE FOR EACH ELEMENT IN SEGMENT ------------------->
    FOR KELEM = 1 TO IELEX: IELEM = IELEM + 1
    LOCATE 14, 17: PRINT IELEM;
    LZ = IORDER(IELEM): LPROP = MATNO(LZ)

    FOR I = 1 TO NDIME: FOR J = 1 TO NNPE
    ELCOD(I, J) = COORD((NP(LZ, J) - 1) * NDIME + I)
```

```
    NEXT J, I
    YOUNG = PROPS(LPROP, 1)
    POISS = PROPS(LPROP, 2)
    THICK = PROPS(LPROP, 3)

    FOR I = 1 TO NSTRE: FOR J = 1 TO NSTRE: DMATX(I, J) = 0: NEXT J, I

    ON NTYPE GOTO 1230, 1300, 1370, 1440

    '<---------- D MATRIX FOR PLANE STRESS CASE ---------->
1230 C = YOUNG / (1 - POISS ^ 2)
    DMATX(1, 1) = C: DMATX(2, 2) = C
    DMATX(1, 2) = C * POISS: DMATX(2, 1) = C * POISS
    DMATX(3, 3) = (1 - POISS) * C / 2
    GOTO 1510

    '<---------- D MATRIX FOR PLANE STRAIN CASE ---------->
1300 C = YOUNG * (1 - POISS) / ((1 + POISS) * (1 - 2 * POISS))
    DMATX(1, 1) = C: DMATX(2, 2) = C
    DMATX(1, 2) = C * POISS / (1 - POISS): DMATX(2, 1) = C * POISS / (1 - POISS)
    DMATX(3, 3) = C * (1 - 2 * POISS) / (2 * (1 - POISS))
    GOTO 1510

    '<---------- D MATRIX FOR PLATE BENDING CASE ---------->
1370 DMATX(1, 1) = YOUNG * THICK ^ 3 / (12 * (1 - POISS ^ 2)): DMATX(1, 2) = POISS * DMATX(1, 1)
    DMATX(2, 2) = DMATX(1, 1): DMATX(2, 1) = DMATX(1, 2)
    DMATX(3, 3) = (1 - POISS) * DMATX(1, 1) / 2
    DMATX(4, 4) = YOUNG * THICK / (2.4 * (1 + POISS)): DMATX(5, 5) = DMATX(4, 4)
    GOTO 1510

    '<----------D MATRIX FOR BRICK ELEMENT---------->
1440 C = YOUNG * (1 - POISS) / ((1 + POISS) * (1 - 2 * POISS))
    DMATX(1, 1) = C: DMATX(1, 2) = C * POISS / (1 - POISS)
    DMATX(1, 3) = DMATX(1, 2): DMATX(2, 2) = C
    DMATX(2, 3) = DMATX(1, 2): DMATX(2, 1) = DMATX(1, 2)
    DMATX(3, 1) = DMATX(1, 3): DMATX(3, 2) = DMATX(2, 3): DMATX(3, 3) = C
    DMATX(4, 4) = C * (1 - 2 * POISS) / (2 * (1 - POISS))
    DMATX(5, 5) = DMATX(4, 4): DMATX(6, 6) = DMATX(4, 4)

1510 FOR I = 1 TO NEDOF: FOR J = 1 TO NEDOF
    ESTIF(I, J) = 0: NEXT J, I

    IF NTYPE < 4 THEN 1540
    FOR NO = 1 TO NGAUSS: R = PLACE(NO, NGAUSS)
1540 FOR NA = 1 TO NGAUSS: S = PLACE(NA, NGAUSS)
    FOR NB = 1 TO NGAUSS: T = PLACE(NB, NGAUSS)

    IF NTYPE = 4 THEN 1900
    IF NNPE = 8 THEN 1730

    '<---------- SHAPE FUNCTION & DERIVATIVES (Q4) ---------->
    ST = S * T

    SHAPE(1) = (1 - T - S + ST) / 4: SHAPE(2) = (1 - T + S - ST) / 4
    SHAPE(3) = (1 + T + S + ST) / 4: SHAPE(4) = (1 + T - S - ST) / 4
```

```
    DERIV(1, 1) = (-1 + T) / 4: DERIV(2, 1) = (-1 + S) / 4
    DERIV(1, 2) = (1 - T) / 4: DERIV(2, 2) = (-1 - S) / 4
    DERIV(1, 3) = (1 + T) / 4: DERIV(2, 3) = (1 + S) / 4
    DERIV(1, 4) = (-1 - T) / 4: DERIV(2, 4) = (1 - S) / 4

    GOTO 1950

    '<---------------- SHAPE FUNCTION & DERIVATIVES (Q8) ----------------->
1730 S2 = 2 * S:  T2 = 2 * T:     SS = S * S:     TT = T * T
    ST = S * T:  SST = S * S * T: STT = S * T * T: ST2 = 2 * S * T

    SHAPE(1) = (-1 + ST + SS + TT - SST - STT) / 4
    SHAPE(2) = (1 - T - SS + SST) / 2
    SHAPE(3) = (-1 - ST + SS + TT - SST + STT) / 4
    SHAPE(4) = (1 + S - TT - STT) / 2
    SHAPE(5) = (-1 + ST + SS + TT + SST + STT) / 4
    SHAPE(6) = (1 + T - SS - SST) / 2
    SHAPE(7) = (-1 - ST + SS + TT + SST - STT) / 4
    SHAPE(8) = (1 - S - TT + STT) / 2

    DERIV(1, 1) = (T + S2 - ST2 - TT) / 4: DERIV(2, 1) = (S + T2 - SS - ST2) / 4
    DERIV(1, 2) = -S + ST: DERIV(2, 2) = (-1 + SS) / 2
    DERIV(1, 3) = (-T + S2 - ST2 + TT) / 4: DERIV(2, 3) = (-S + T2 - SS + ST2) / 4
    DERIV(1, 4) = (1 - TT) / 2: DERIV(2, 4) = (-T - ST)
    DERIV(1, 5) = (T + S2 + ST2 + TT) / 4: DERIV(2, 5) = (S + T2 + SS + ST2) / 4
    DERIV(1, 6) = -S - ST: DERIV(2, 6) = (1 - SS) / 2
    DERIV(1, 7) = (-T + S2 + ST2 - TT) / 4: DERIV(2, 7) = (-S + T2 + SS - ST2) / 4
    DERIV(1, 8) = (-1 + TT) / 2: DERIV(2, 8) = -T + ST

    GOTO 1950
    '<--------SHAPE FUNCTION & DERIVATIVES FOR BRICK ELEMENT --------->
1900 ST = S * T: RT = R * T: RS = R * S: RST = R * S * T

    SHAPE(1) = (1 - T - S + ST + R - RT - RS + RST) / 8
    SHAPE(2) = (1 - T + S - ST + R - RT + RS - RST) / 8
    SHAPE(3) = (1 - T + S - ST - R + RT - RS + RST) / 8
    SHAPE(4) = (1 - T - S + ST - R + RT + RS - RST) / 8
    SHAPE(5) = (1 + T - S - ST + R + RT - RS - RST) / 8
    SHAPE(6) = (1 + T + S + ST + R + RT + RS + RST) / 8
    SHAPE(7) = (1 + T + S + ST - R - RT - RS - RST) / 8
    SHAPE(8) = (1 + T - S - ST - R - RT + RS + RST) / 8

    DERIV(1, 1) = (1 - T - S + ST) / 8: DERIV(2, 1) = (-1 + T - R + RT) / 8
    DERIV(1, 2) = (1 - T + S - ST) / 8: DERIV(2, 2) = (1 - T + R - RT) / 8
    DERIV(1, 3) = (-1 + T - S + ST) / 8: DERIV(2, 3) = (1 - T - R + RT) / 8
    DERIV(1, 4) = (-1 + T + S - ST) / 8: DERIV(2, 4) = (-1 + T + R - RT) / 8
    DERIV(1, 5) = (1 + T - S - ST) / 8: DERIV(2, 5) = (-1 - T - R - RT) / 8
    DERIV(1, 6) = (1 + T + S + ST) / 8: DERIV(2, 6) = (1 + T + R + RT) / 8
    DERIV(1, 7) = (-1 - T + S - ST) / 8: DERIV(2, 7) = (1 + T - R - RT) / 8
    DERIV(1, 8) = (-1 - T + S + ST) / 8: DERIV(2, 8) = (-1 - T + R + RT) / 8

    DERIV(3, 1) = (-1 + S - R + RS) / 8
    DERIV(3, 2) = (-1 - S - R - RS) / 8
    DERIV(3, 3) = (-1 - S + R + RS) / 8
    DERIV(3, 4) = (-1 + S + R - RS) / 8
    DERIV(3, 5) = (1 - S + R - RS) / 8
```

```
    DERIV(3, 6) = (1 + S + R + RS) / 8
    DERIV(3, 7) = (1 + S - R - RS) / 8
    DERIV(3, 8) = (1 - S - R + RS) / 8

    '<---------------- CREATE JACOBIAN MATRIX XJACM ------------------>
1950 FOR I = 1 TO NDIME: FOR J = 1 TO NDIME: XJACM(I, J) = 0
    FOR K = 1 TO NNPE: XJACM(I, J) = XJACM(I, J) + DERIV(I, K) * ELCOD(J, K)
    NEXT K, J, I

    IF NTYPE = 4 THEN 2000

    '     Calculate determinant & inverse of jacobian matrix (2X2)

    DJACB = XJACM(1, 1) * XJACM(2, 2) - XJACM(1, 2) * XJACM(2, 1)
    IF DJACB <= 0 THEN PRINT "ZERO OR NEGATIVE ": STOP

    XJACI(1, 1) = XJACM(2, 2) / DJACB: XJACI(2, 2) = XJACM(1, 1) / DJACB
    XJACI(1, 2) = -XJACM(1, 2) / DJACB: XJACI(2, 1) = -XJACM(2, 1) / DJACB
    GOTO 2070

    '     Calculate determinant & inverse of jacobian matrix (3X3)

2000 DJACB = XJACM(1, 1) * (XJACM(2, 2) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 2))
    DJACB = DJACB - XJACM(1, 2) * (XJACM(2, 1) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 1))
    DJACB = DJACB + XJACM(1, 3) * (XJACM(2, 1) * XJACM(3, 2) - XJACM(2, 2) * XJACM(3, 1))
    IF DJACB <= 0 THEN PRINT "ZERO OR NEGATIVE ": STOP

    XJACI(1, 1) = (XJACM(2, 2) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 2)) / DJACB
    XJACI(1, 2) = -(XJACM(1, 2) * XJACM(3, 3) -.XJACM(1, 3) * XJACM(3, 2)) / DJACB
    XJACI(1, 3) = (XJACM(1, 2) * XJACM(2, 3) - XJACM(1, 3) * XJACM(2, 2)) / DJACB
    XJACI(2, 1) = -(XJACM(2, 1) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 1)) / DJACB
    XJACI(2, 2) = (XJACM(1, 1) * XJACM(3, 3) - XJACM(1, 3) * XJACM(3, 1)) / DJACB
    XJACI(2, 3) = -(XJACM(1, 1) * XJACM(2, 3) - XJACM(1, 3) * XJACM(2, 1)) / DJACB
    XJACI(3, 1) = (XJACM(2, 1) * XJACM(3, 2) - XJACM(2, 2) * XJACM(3, 1)) / DJACB
    XJACI(3, 2) = -(XJACM(1, 1) * XJACM(3, 2) - XJACM(1, 2) * XJACM(3, 1)) / DJACB
    XJACI(3, 3) = (XJACM(1, 1) * XJACM(2, 2) - XJACM(1, 2) * XJACM(2, 1)) / DJACB

    '<-------- CALCULATE CARTESIANS DERIVATIVES --------------------------->

2070 FOR I = 1 TO NDIME: FOR J = 1 TO NNPE: CARTD(I, J) = 0
    FOR K = 1 TO NDIME
    CARTD(I, J) = CARTD(I, J) + XJACI(I, K) * DERIV(K, J)
    NEXT K, J, I
    IF NTYPE < 4 THEN DVOLU = DJACB * WGT(NA, NGAUSS) * WGT(NB, NGAUSS) * THICK: DAREA = DVOLU / THICK
    IF NTYPE = 4 THEN DVOLU = DJACB * WGT(NO, NGAUSS) * WGT(NA, NGAUSS) * WGT(NB, NGAUSS)

    '<------ EVALUATE THE B AND DB MATRICES ------------------------------>

    FOR I = 1 TO NSTRE: FOR J = 1 TO NEDOF: BMATX(I, J) = 0: NEXT J, I
    ON NTYPE GOTO 2180, 2180, 2270, 2390

    '<------ B MATRIX FOR PLANE STRESS , PLANE STRAIN -------------------->
2180 NGASH = 0
    FOR I = 1 TO NNPE: MGASH = NGASH + 1: NGASH = MGASH + 1
    BMATX(1, MGASH) = CARTD(1, I): BMATX(1, NGASH) = 0
    BMATX(2, MGASH) = 0: BMATX(2, NGASH) = CARTD(2, I)
```

```
    BMATX(3, MGASH) = CARTD(2, I): BMATX(3, NGASH) = CARTD(1, I)
    NEXT I
    GOTO 2480

    '<------ B MATRIX FOR PLATE BENDING ---------------------------------->
2270 JGASH = 0
    FOR I = 1 TO NNPE
    IGASH = JGASH + 1
    BMATX(4, IGASH) = CARTD(1, I): BMATX(5, IGASH) = CARTD(2, I)
    IGASH = IGASH + 1: JGASH = IGASH + 1
    BMATX(1, IGASH) = -CARTD(1, I): BMATX(3, IGASH) = -CARTD(2, I)
    BMATX(4, IGASH) = -SHAPE(I): BMATX(2, JGASH) = -CARTD(2, I)
    BMATX(3, JGASH) = -CARTD(1, I): BMATX(5, JGASH) = -SHAPE(I)
    NEXT I
    GOTO 2480

    '<------ B MATRIX FOR BRICK ELEMENT ---------------------------------->
2390 NGASH = 0
    FOR I = 1 TO NNPE
    LGASH = NGASH + 1: MGASH = LGASH + 1: NGASH = MGASH + 1
    BMATX(1, LGASH) = CARTD(1, I): BMATX(1, MGASH) = 0
    BMATX(1, NGASH) = 0
    BMATX(2, LGASH) = 0: BMATX(2, MGASH) = CARTD(2, I)
    BMATX(2, NGASH) = 0
    BMATX(3, LGASH) = 0: BMATX(3, MGASH) = 0
    BMATX(3, NGASH) = CARTD(3, I)
    BMATX(4, LGASH) = CARTD(2, I): BMATX(4, MGASH) = CARTD(1, I)
    BMATX(4, NGASH) = 0
    BMATX(5, LGASH) = 0: BMATX(5, MGASH) = CARTD(3, I)
    BMATX(5, NGASH) = CARTD(2, I)
    BMATX(6, LGASH) = CARTD(3, I): BMATX(3, MGASH) = 0
    BMATX(6, NGASH) = CARTD(1, I)
    NEXT I

    '<------ CALCULATE D*B ----------------------------------------------->
2480 FOR I = 1 TO NSTRE: FOR J = 1 TO NEDOF: DBMAT(I, J) = 0
    FOR K = 1 TO NSTRE: DBMAT(I, J) = DBMAT(I, J) + DMATX(I, K) * BMATX(K, J)
    NEXT K, J, I

    '<------ CALCULATE THE ELEMENT STIFFNESS ---------------------------->
    IF NTYPE < 3 THEN DV = DVOLU ELSE DV = DAREA
    IF NTYPE = 4 THEN DV = DVOLU
    FOR I = 1 TO NEDOF: FOR J = I TO NEDOF: FOR K = 1 TO NSTRE
    ESTIF(I, J) = ESTIF(I, J) + BMATX(K, I) * DBMAT(K, J) * DV: NEXT K, J, I

    FOR I = 1 TO NEDOF: FOR J = 1 TO (I - 1): ESTIF(I, J) = ESTIF(J, I): NEXT J, I

    NEXT NB, NA
    IF NTYPE < 4 THEN 2610
    NEXT NO

2610 FOR I = 1 TO NEDOF: IEQ(I) = 0: NEXT I

    FOR I = 1 TO NNPE: FOR J = 1 TO NPDOF
    IF KODE(NP(LZ, I), J) = 1 THEN IEQ((I - 1) * NPDOF + J) = 1
    NEXT J, I
```

```
        FOR I = 1 TO NEDOF
        IF IEQ(I) = 1 THEN 2730
        FOR J = I TO NEDOF
        IF IEQ(J) = 1 THEN 2720
        PUT #1, , ESTIF(I, J)
        STIFFCOUNT = STIFFCOUNT + 1
2720    NEXT J
2730    NEXT I

        NEXT KELEM
        CLOSE 1

        IF ISEG < NUMSEG THEN 1000

        FINISHtime! = TIMER
        TOTALtime! = FINISHtime! - STARTtime!
        F$(14) = STR$(TOTALtime!)
        IF NTYPE = 4 THEN CHAIN "L8"
        CHAIN "LFORM"
```

```
'*************************************************************************
'*                                                                       *
'*               file name "LFORM"                                       *
'*   Subprogram to generate LOAD for 5 load cases                        *
'*   NTYPE = 1 ----> PLANE STRESS FINITE ELEMENT                         *
'*           2 ----> PLANE STRAIN FINITE ELEMENT                         *
'*           3 ----> PLATE BENDING FINITE ELEMENT                        *
'*                                                                       *
'*************************************************************************
DEFINT I-N
DEFDBL A-H, O-Z

COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
COMMON NCASE, NUMSEG, NGAUSS, IORDER, T$, E$, MAXVOL, F$()
COMMON IOPTION
COMMON LSTMAX, NEQMAX, KMAX
'      Maximum nodes repersented in segment
'      Maximum equations repersented in segment
'      Maximum usage storage
COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
COMMON NPDOF(), IORDER()

COMMON COORD(), NP(), MATNO(), PROPS(), NBCJ(), KODE()

GOTO 200

'<---------------------- SUBROUTINE AREA ----------------------------->
100 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
110 LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

'-------------------------------------------------------------------------
200 DD$ = D$

DIM NLOAD(NUMNP), PLOAD(NUMNP, NPDOF), TEMP(NUMNP), ULOAD(300)
DIM NEASS(100), NSIDE(100), PRESS(100)
DIM IFILE(5)

STARTtime! = TIMER

FOR I = 1 TO 5
IF LEN(F$(I + 5)) > 0 THEN IFILE(I) = 1
NEXT I

'<------ Integration constants ------------------------------------->
DIM ESTIF(24, 24), ST(24), WGT(3, 3), PLACE(3, 3)
FOR I = 1 TO 3: FOR J = 1 TO 3: READ WGT(I, J): NEXT J, I
FOR I = 1 TO 3: FOR J = 1 TO 3: READ PLACE(I, J): NEXT J, I
DATA 2      ,1                  ,0.555555555555556
DATA 0      ,1                  ,0.888888888888889
DATA 0      ,0                  ,0.555555555555556
DATA 0      ,-0.577350269189626 ,-0.774596669241483
DATA 0      , 0.577350269189626 ,0
DATA 0      ,0                  , 0.774596669241483

NEDOF = NNPE * NPDOF
FORCECOUNT = 0
```

```
      DIM DMATX(6, 6), SHAPE(8), DERIV(3, 8), GPCOD(2), ELCOD(3, 8), XJACM(3, 3)
      DIM XJACI(3, 3), CARTD(3, 8), BMATX(6, 24), DBMAT(6, 24), STRAN(5), STRES(5)
      DIM LNODE(3), PGASH(2), DGASH(2), IEQ(24)

      GOSUB 100: GOSUB 110
      COLOR 14
      LOCATE 15, 1: PRINT "<<< FORM LOAD VECTOR ("; NCASE; "CASES) >>>"

      '<---------------------- LOOP OVER LOAD CASE ------------------------->
      FOR ICASE = 1 TO 5
      IF IFILE(ICASE) = 0 THEN 3660
      CASE$ = RIGHT$(STR$(ICASE), 1)

      OPEN "I", 1, D$ + F$ + ".LC" + CASE$
      INPUT #1, C$
      INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
      CLOSE 1

      LOCATE 16, 1: PRINT SPC(70);
      LOCATE 16, 1: COLOR 14
      PRINT TAB(5); "LOAD #"; CASE$;
      PRINT " : "; C$
      COLOR 15: PRINT TAB(5); "Elements ..."

      IF IPLOD = 0 THEN 210                          'without node loaded

      OPEN "I", 1, D$ + F$ + ".P" + CASE$
      FOR I = 1 TO NUMNP
      INPUT #1, NLOAD(I)
      FOR J = 1 TO NPDOF
      INPUT #1, PLOAD(I, J)
      NEXT J, I
      CLOSE 1

210   IF IGRAV = 0 THEN 220                          'without gravity load

      OPEN "I", 1, D$ + F$ + ".G" + CASE$
      INPUT #1, THETA, GRAVY
      CLOSE 1

220   IF IEDGE = 0 THEN 230                          'without press load

      IF NNPE = 8 THEN NPRESS = 3
      IF NNPE = 4 THEN NPRESS = 2
      OPEN "I", 1, D$ + F$ + ".E" + CASE$
      INPUT #1, NEDGE
      NTOT = NPRESS * 2 * NEDGE
      FOR I = 1 TO NEDGE: INPUT #1, NEASS(I), NSIDE(I): NEXT I
      FOR I = 1 TO NTOT: INPUT #1, PRESS(I): NEXT I
      CLOSE 1

230   IF ITEMP = 0 THEN 240                          'without thermal load
      OPEN "I", 1, D$ + F$ + ".T" + CASE$
      FOR I = 1 TO NUMNP: INPUT #1, TEMP(I): NEXT I
      CLOSE 1
```

```
240  IF IULOD = 0 THEN 1470                          'without thermal load
     OPEN "I", 1, D$ + F$ + ".U" + CASE$
     FOR I = 1 TO NUMEL: INPUT #1, ULOAD(I): NEXT I
     CLOSE 1

1470 OPEN "O", 1, DD$ + "LOAD." + CASE$
     IF ITEMP = 1 THEN OPEN "O", 2, D$ + "STRESS." + CASE$

     IF NTYPE < 3 THEN NSTRE = 3
     IF NTYPE = 3 THEN NSTRE = 5

     '<-------------------- LOOP OVER EACH ELEMENT ------------------------>

     FOR IELEM = 1 TO NUMEL: LOCATE 17, 17: PRINT IELEM;
     LZ = IORDER(IELEM): LPROP = MATNO(LZ)

     FOR I = 1 TO NDIME: FOR J = 1 TO NNPE
     ELCOD(I, J) = COORD((NP(LZ, J) - 1) * NDIME + I)
     NEXT J, I

     FOR I = 1 TO NEDOF: ST(I) = 0: NEXT I

     YOUNG = PROPS(LPROP, 1)
     POISS = PROPS(LPROP, 2)
     THICK = PROPS(LPROP, 3)
     ALPHA = PROPS(LPROP, 5)

     IF IGRAV = 0 AND ITEMP = 0 AND IULOD = 0 THEN 3130

     IF ITEMP = 0 THEN 2090

     FOR I = 1 TO NSTRE: FOR J = 1 TO NSTRE: DMATX(I, J) = 0: NEXT J, I

     ON NTYPE GOTO 1790, 1860, 1930

     '<--------------- D MATRIX FOR PLANE STRESS CASE ---------------->
1790 C = YOUNG / (1 - POISS ^ 2)
     DMATX(1, 1) = C: DMATX(2, 2) = C
     DMATX(1, 2) = C * POISS: DMATX(2, 1) = C * POISS
     DMATX(3, 3) = (1 - POISS) * C / 2
     GOTO 2090

     '<--------------- D MATRIX FOR PLANE STRAIN CASE ---------------->
1860 C = YOUNG * (1 - POISS) / ((1 + POISS) * (1 - 2 * POISS))
     DMATX(1, 1) = C: DMATX(2, 2) = C
     DMATX(1, 2) = C * POISS / (1 - POISS): DMATX(2, 1) = C * POISS / (1 - POISS)
     DMATX(3, 3) = C * (1 - 2 * POISS) / (2 * (1 - POISS))
     GOTO 2090

     '<--------------- D MATRIX FOR PLATE BENDING CASE --------------->
1930 DMATX(1, 1) = YOUNG * THICK ^ 3 / (12 * (1 - POISS ^ 2)): DMATX(1, 2) = POISS * DMATX(1, 1)
     DMATX(2, 2) = DMATX(1, 1): DMATX(2, 1) = DMATX(1, 2)
     DMATX(3, 3) = (1 - POISS) * DMATX(1, 1) / 2
     DMATX(4, 4) = YOUNG * THICK / (2.4 * (1 + POISS)): DMATX(5, 5) = DMATX(4, 4)
     GOTO 2090
```

```
2090 FOR NA = 1 TO NGAUSS: S = PLACE(NA, NGAUSS)
     FOR NB = 1 TO NGAUSS: T = PLACE(NB, NGAUSS)

     IF NNPE = 8 THEN 2260

     '<---------------- SHAPE FUNCTION & DERIVATIVES (Q4) ----------------->
     ST = S * T

     SHAPE(1) = (1 - T - S + ST) / 4: SHAPE(2) = (1 - T + S - ST) / 4
     SHAPE(3) = (1 + T + S + ST) / 4: SHAPE(4) = (1 + T - S - ST) / 4

     DERIV(1, 1) = (-1 + T) / 4: DERIV(2, 1) = (-1 + S) / 4
     DERIV(1, 2) = (1 - T) / 4: DERIV(2, 2) = (-1 - S) / 4
     DERIV(1, 3) = (1 + T) / 4: DERIV(2, 3) = (1 + S) / 4
     DERIV(1, 4) = (-1 - T) / 4: DERIV(2, 4) = (1 - S) / 4

     GOTO 2480

     '<---------------- SHAPE FUNCTION & DERIVATIVES (Q8) ----------------->
2260 S2 = 2 * S:  T2 = 2 * T:     SS = S * S:     TT = T * T
     ST = S * T:  SST = S * S * T: STT = S * T * T: ST2 = 2 * S * T

     SHAPE(1) = (-1 + ST + SS + TT - SST - STT) / 4
     SHAPE(2) = (1 - T - SS + SST) / 2
     SHAPE(3) = (-1 - ST + SS + TT - SST + STT) / 4
     SHAPE(4) = (1 + S - TT - STT) / 2
     SHAPE(5) = (-1 + ST + SS + TT + SST + STT) / 4
     SHAPE(6) = (1 + T - SS - SST) / 2
     SHAPE(7) = (-1 - ST + SS + TT + SST - STT) / 4
     SHAPE(8) = (1 - S - TT + STT) / 2

     DERIV(1, 1) = (T + S2 - ST2 - TT) / 4: DERIV(2, 1) = (S + T2 - SS - ST2) / 4
     DERIV(1, 2) = -S + ST: DERIV(2, 2) = (-1 + SS) / 2
     DERIV(1, 3) = (-T + S2 - ST2 + TT) / 4: DERIV(2, 3) = (-S + T2 - SS + ST2) / 4
     DERIV(1, 4) = (1 - TT) / 2: DERIV(2, 4) = (-T - ST)
     DERIV(1, 5) = (T + S2 + ST2 + TT) / 4: DERIV(2, 5) = (S + T2 + SS + ST2) / 4
     DERIV(1, 6) = -S - ST: DERIV(2, 6) = (1 - SS) / 2
     DERIV(1, 7) = (-T + S2 + ST2 - TT) / 4: DERIV(2, 7) = (-S + T2 + SS - ST2) / 4
     DERIV(1, 8) = (-1 + TT) / 2: DERIV(2, 8) = -T + ST

     GOTO 2480

     '<---------------- CREATE JACOBIAN MATRIX XJACM ----------------->
2480 FOR I = 1 TO NDIME: FOR J = 1 TO NDIME: XJACM(I, J) = 0
     FOR K = 1 TO NNPE: XJACM(I, J) = XJACM(I, J) + DERIV(I, K) * ELCOD(J, K)
     NEXT K, J, I

     '------- Calculate determinant & inverse of jacobian matrix (2x2)-------

     DJACB = XJACM(1, 1) * XJACM(2, 2) - XJACM(1, 2) * XJACM(2, 1)
     IF DJACB <= 0 THEN PRINT "ZERO OR NEGATIVE ": STOP

     XJACI(1, 1) = XJACM(2, 2) / DJACB: XJACI(2, 2) = XJACM(1, 1) / DJACB
     XJACI(1, 2) = -XJACM(1, 2) / DJACB: XJACI(2, 1) = -XJACM(2, 1) / DJACB
```

```
    '<-------- CALCULATE CARTESIANS DERIVATIVES -------------------------->
2600 FOR I = 1 TO NDIME: FOR J = 1 TO NNPE: CARTD(I, J) = 0
    FOR K = 1 TO NDIME
    CARTD(I, J) = CARTD(I, J) + XJACI(I, K) * DERIV(K, J)
    NEXT K, J, I

    IF NTYPE < 4 THEN DVOLU = DJACB * WGT(NA, NGAUSS) * WGT(NB, NGAUSS) * THICK: DAREA = DVOLU / THICK
    IF NTYPE < 3 THEN DV = DVOLU ELSE DV = DAREA

    IF ITEMP = 0 THEN 2920

    '<-------- EVALUATE THE INITIAL THERMAL STRAINS --------------------->
    TEMP = 0
    FOR I = 1 TO NNPE: TEMP = TEMP + TEMP(NP(LZ, I)) * SHAPE(I): NEXT I
    EIGEN = TEMP * ALPHA: IF NTYPE = 3 THEN EIGEN = EIGEN * 2 / THICK
    ON NTYPE GOTO 2760, 2770, 2780

2760 STRAN(1) = -EIGEN: STRAN(2) = -EIGEN: STRAN(3) = 0: GOTO 2800
2770 STRAN(1) = -(1 + POISS) * EIGEN: STRAN(2) = -(1 + POISS) * EIGEN: STRAN(3) = 0
    GOTO 2800
2780 STRAN(1) = -EIGEN: STRAN(2) = -EIGEN: STRAN(3) = 0: STRAN(4) = 0: STRAN(5) = 0
    GOTO 2800

2800 FOR I = 1 TO NSTRE: STRES(I) = 0
    FOR J = 1 TO NSTRE: STRES(I) = STRES(I) + DMATX(I, J) * STRAN(J): NEXT J, I
    IF NTYPE = 2 THEN STRES(4) = -YOUNG * EIGEN
    IF NTYPE = 1 THEN STRES(4) = 0
    FOR I = 1 TO (NSTRE + 1): WRITE #2, STRES(I): NEXT I
    FOR I = 1 TO NNPE: NGASH = (I - 1) * NPDOF + 1: MGASH = (I - 1) * NPDOF + 2
    IF NTYPE = 3 THEN 2890
    ST(NGASH) = ST(NGASH) - (CARTD(1, I) * STRES(1) + CARTD(2, I) * STRES(3)) * DV
    ST(MGASH) = ST(MGASH) - (CARTD(1, I) * STRES(3) + CARTD(2, I) * STRES(2)) * DV: GOTO 2900
2890 ST(MGASH) = ST(MGASH) - (-CARTD(1, I) * STRES(1)) * DV: ST(MGASH + 1) = ST(MGASH + 1) - (-CARTD(2, I) * ST
ES(2)) * DV
2900 NEXT I

2920 IF IGRAV = 0 THEN 3040
    '---------------------- Gravity loading -------------------------
    DENSE = PROPS(LPROP, 4)
    IF DENSE = 0 THEN 3040
    GXCOM = DENSE * GRAVY * SIN(THETA)
    GYCOM = -DENSE * GRAVY * COS(THETA)
    FOR I = 1 TO NNPE
    NGASH = (I - 1) * NPDOF + 1: MGASH = (I - 1) * NPDOF + 2: LGASH = (I - 1) * NPDOF + 3
    IF NTYPE = 4 THEN ST(LGASH) = ST(LGASH) + GYCOM * SHAPE(I) * DV: GOTO 3020
    IF NTYPE = 3 THEN ST(NGASH) = ST(NGASH) + GYCOM * SHAPE(I) * DV: GOTO 3020
    ST(NGASH) = ST(NGASH) + GXCOM * SHAPE(I) * DV
    ST(MGASH) = ST(MGASH) + GYCOM * SHAPE(I) * DV
3020 NEXT I

3040 IF IULOD = 0 THEN 3110
    '----------ADD CONTRIBUTION OF BODY FORCES OF UNIFORM LOAD -------------
    BODYFX = ULOAD(LZ)
    FOR I = 1 TO NNPE: NGASH = (I - 1) * NPDOF + 1: MGASH = (I - 1) * NPDOF + 2
    ST(NGASH) = ST(NGASH) + BODYFX * SHAPE(I) * DV: IF NTYPE = 3 THEN 3100
    ST(MGASH) = ST(MGASH) + BODYFY * SHAPE(I) * DV
```

```
3100 NEXT I

3110 NEXT NB, NA

3130 IF IPLOD = 0 THEN 3230
     '<-------- SEMBLE NODAL LOADS INTO LOAD VECTOR ----------------------->
     FOR I = 1 TO NNPE
     IF NLOAD(NP(LZ, I)) <> 1 THEN 3210
     FOR J = 1 TO NPDOF: NGASH = (I - 1) * NPDOF + J
     ST(NGASH) = ST(NGASH) + PLOAD(NP(LZ, I), J)
     NEXT J
     NLOAD(NP(LZ, I)) = -NLOAD(NP(LZ, I))
3210 NEXT I

3230 IF IEDGE = 0 THEN 3510
     '<-------- DISTRIBUTED EDGE LOADING ---------------------------------->
     FOR I = 1 TO NEDGE: IF NEASS(I) <> LZ THEN 3490
     J1 = (NPRESS - 1) * (NSIDE(I) - 1) + 1
     LNODE(1) = J1
     LNODE(2) = J1 + 1: IF LNODE(2) > NNPE THEN LNODE(2) = 1
     LNODE(3) = LNODE(2) + 1: IF LNODE(3) > NNPE THEN LNODE(3) = 1
     FOR J = 1 TO NGAUSS: S = PLACE(J, NGAUSS)

     IF NNPE = 8 THEN 3240
     SHAPE(1) = (1 - S) / 2: SHAPE(2) = (1 + S) / 2
     DERIV(1, 1) = -.5: DERIV(1, 2) = .5
     GOTO 3250

3240 SHAPE(1) = -.5 * (S - S ^ 2)
     SHAPE(2) = 1 - S ^ 2
     SHAPE(3) = .5 * (S + S ^ 2)
     DERIV(1, 1) = S - .5
     DERIV(1, 2) = -2 * S
     DERIV(1, 3) = .5 + S

3250 FOR K = 1 TO 2: PGASH(K) = 0: DGASH(K) = 0
     FOR L = 1 TO NPRESS
     PGASH(K) = PGASH(K) + PRESS((I - 1) * NPRESS * 2 + (L - 1) * 2 + K) * SHAPE(L)
     DGASH(K) = DGASH(K) + ELCOD(K, LNODE(L)) * DERIV(1, L): NEXT L, K
     DVOLU = WGT(J, NGAUSS)
     PXCOM = DGASH(1) * PGASH(2) - DGASH(2) * PGASH(1)
     PYCOM = DGASH(1) * PGASH(1) + DGASH(2) * PGASH(2)
     FOR L = 1 TO NPRESS: NGASH = 2 * (LNODE(L) - 1) + 1
     ST(NGASH) = ST(NGASH) + SHAPE(L) * PXCOM * DVOLU * THICK
     ST(NGASH + 1) = ST(NGASH + 1) + SHAPE(L) * PYCOM * DVOLU * THICK: NEXT L
     NEXT J
3490 NEXT I

3510 FOR I = 1 TO NEDOF: IEQ(I) = 0: NEXT I

     FOR I = 1 TO NNPE: FOR J = 1 TO NPDOF
     IF KODE(NP(LZ, I), J) = 1 THEN IEQ((I - 1) * NPDOF + J) = 1
     NEXT J, I

     FOR I = 1 TO NEDOF
     IF IEQ(I) = 1 THEN 3600
```

```
      WRITE #1, ST(I)
      FORCECOUNT = FORCECOUNT + 1
3600 NEXT I

      NEXT IELEM

      CLOSE 1, 2

3660 NEXT ICASE

      FINISHtime! = TIMER
      TOTALtime! = FINISHtime! - STARTtime!
      F$(15) = STR$(TOTALtime!)
      CHAIN "SOLVER"
```

```
DECLARE SUB DERIVE (R#, S#, T#)
    '*************************************************************************
    '*                                                                       *
    '*              file name "L8"                                           *
    '*   Subprogram to generate LOAD for 5 load cases                        *
    '*   NTYPE = 4 ----> BRICK FINITE ELEMENT                                *
    '*                                                                       *
    '*************************************************************************
    DEFINT I-N
    DEFDBL A-H, O-Z

    COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    COMMON NCASE, NUMSEG, NGAUSS, IORDER, T$, E$, MAXVOL, F$()
    COMMON IOPTION
    COMMON LSTMAX, NEQMAX, KMAX
    COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
    COMMON NPDOF(), IORDER()
    COMMON COORD(), NP(), MATNO(), PROPS(), NBCJ(), KODE()

    GOTO 180

    '<---------------------- SUBROUTINE AREA ----------------------------->
150 LOCATE 1, 66: COLOR 14: PRINT "DATE "; : COLOR 15: PRINT DATE$: RETURN
160 LOCATE 2, 66: COLOR 14: PRINT "TIME "; : COLOR 15: PRINT TIME$: RETURN

170 LOCATE 24, 20: PRINT "PRESS ANY KEY TO CONTINUE";
171 K$ = INKEY$: IF K$ = "" THEN 171
    LOCATE 24, 20: PRINT SPC(50);
    RETURN

    '-------------------------------------------------------------------------
180 DD$ = D$

    DIM IORDER(NUMEL)
    DIM NLOAD(NUMNP), PLOAD(NUMNP, NPDOF), TEMP(NUMNP)
    DIM NEASS(100), NSIDE(100), PRESS(100)
    DIM IFILE(5)

    STARTtime! = TIMER

    FOR I = 1 TO 5
    IF LEN(F$(I + 5)) > 0 THEN IFILE(I) = 1
    NEXT I

    '<------ Integration constants ------------------------------------->
    DIM ST(24), WGT(3, 3), PLACE(3, 3)
    FOR I = 1 TO 3: FOR J = 1 TO 3: READ WGT(I, J): NEXT J, I
    FOR I = 1 TO 3: FOR J = 1 TO 3: READ PLACE(I, J): NEXT J, I
    DATA 2      ,1                  ,0.555555555555556
    DATA 0      ,1                  ,0.888888888888889
    DATA 0      ,0                  ,0.555555555555556
    DATA 0      ,-0.577350269189626 ,-0.774596669241483
    DATA 0      , 0.577350269189626 ,0
    DATA 0      ,0                  , 0.774596669241483

    DIM KFACE(6, 4), KCRD(6), FVAL(6), IPERM(3), ETA(3)
```

```
    FOR I = 1 TO 6: FOR J = 1 TO 4: READ KFACE(I, J): NEXT J, I
    FOR I = 1 TO 6: READ KCRD(I): NEXT I
    FOR I = 1 TO 6: READ FVAL(I): NEXT I
    FOR I = 1 TO 3: READ IPERM(I): NEXT I
    DATA 1,2,6,5
    DATA 4,3,7,8
    DATA 2,3,7,6
    DATA 1,4,8,5
    DATA 6,7,8,5
    DATA 2,3,4,1
    DATA 1,1,2,2,3,3
    DATA 1,-1,1,-1,1,-1
    DATA 2,3,1

    NEDOF = NNPE * NPDOF

    DIM DMATX(6, 6), SHAPE(8), DERIV(3, 8), GPCOD(2), ELCOD(3, 8), XJACM(3, 3)
    DIM XJACI(3, 3), CARTD(8, 3), BMATX(6, 24), DBMAT(6, 24), STRAN(5), STRES(5)
    DIM LNODE(3), PGASH(2), QGASH(2), IEQ(24)

    CLS : GOSUB 150: GOSUB 160
    COLOR 11: PRINT "<<< FORM LOAD VECTOR ("; NCASE; "CASES) >>>"

    '<-------------------- LOOP OVER LOAD CASE -------------------------->
    FOR ICASE = 1 TO 5
    IF IFILE(ICASE) = 0 THEN 3000
    CASE$ = RIGHT$(STR$(ICASE), 1)

    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, ULOD
    CLOSE 1

    LOCATE 5, 1: COLOR 14: PRINT "LOAD #"; CASE$
    PRINT SPC(70);
    LOCATE 6, 1: COLOR 15: PRINT "TITLE :"; C$

    IF IPLOD = 0 THEN 1210
    OPEN "I", 1, D$ + F$ + ".P" + CASE$
    FOR I = 1 TO NUMNP
    INPUT #1, NLOAD(I)
    FOR J = 1 TO NPDOF
    INPUT #1, PLOAD(I, J)
    NEXT J, I
    CLOSE 1

1210 IF IGRAV = 0 THEN 1220
    OPEN "I", 1, D$ + F$ + ".G" + CASE$
    INPUT #1, THETA, GRAVY
    CLOSE 1

1220 IF IEDGE = 0 THEN 1230
    OPEN "I", 1, D$ + F$ + ".E" + CASE$
    INPUT #1, NEDGE
    FOR I = 1 TO NEDGE: INPUT #1, NEASS(I), NSIDE(I), PRESS(I): NEXT I
    CLOSE 1
```

```
1230 IF ITEMP = 0 THEN 1240
     OPEN "I", 1, D$ + F$ + ".T" + CASE$
     FOR I = 1 TO NUMNP: INPUT #1, TEMP(I): NEXT I
     CLOSE 1

1240 OPEN "O", 1, DD$ + "LOAD." + CASE$

     '<-------------------- LOOP OVER EACH ELEMENT ---------------------->
     FOR IELEM = 1 TO NUMEL: LOCATE 7, 1: PRINT IELEM
     KELEM = IORDER(IELEM): LPROP = MATNO(KELEM)

     FOR I = 1 TO NDIME: FOR J = 1 TO NNPE
     ELCOD(I, J) = COORD((NP(KELEM, J) - 1) * NDIME + I)
     NEXT J, I

     FOR I = 1 TO NEDOF: ST(I) = 0: NEXT I

     IF IPLOD = 0 THEN 1260

     '<--------------- SEMBLE NODAL LOADS IN LOAD VECTOR ------------------>
     FOR I = 1 TO NNPE
     IF NLOAD(NP(KELEM, I)) <> 1 THEN 1250
     FOR J = 1 TO NPDOF: NGASH = (I - 1) * NPDOF + J
     ST(NGASH) = ST(NGASH) + PLOAD(NP(KELEM, I), J)
     NEXT J
     NLOAD(NP(KELEM, I)) = -NLOAD(NP(KELEM, I))
1250 NEXT I

1260 IF IGRAV = 0 AND ITEMP = 0 THEN 2940

     '<---------------- GRAVITY LOADING & THERMAL LOADING ----------------->
     IF ITEMP = 0 THEN 1270
     YOUNG = PROPS(LPROP, 1)
     POISS = PROPS(LPROP, 2)
     ALPHA = PROPS(LPROP, 5)
     TEMP = 0
     FOR I = 1 TO 8: TEMP = TEMP + TEMP(NP(KELEM, I)): NEXT I
     TEMP = TEMP * .125
     FACT = YOUNG * ALPHA * TEMP / (1 - 2 * POISS)
1270 DENSE = PROPS(LPROP, 4)
     FOR LX = 1 TO NGAUSS: ETA(1) = PLACE(LX, NGAUSS)
     FOR LY = 1 TO NGAUSS: ETA(2) = PLACE(LY, NGAUSS)
     FOR LZ = 1 TO NGAUSS: ETA(3) = PLACE(LZ, NGAUSS)
     CALL DERIVE(ETA(1), ETA(2), ETA(3))
     DVOLU = WGT(LX, NGAUSS) * WGT(LY, NGAUSS) * WGT(LZ, NGAUSS) * DET
     IF IGRAV = 0 THEN 1280
     FOR I = 1 TO 8
     ST(I * 3) = ST(I * 3) + SHAPE(I) * DENSE * DVOLU
     NEXT I
1280 IF ITEMP = 0 THEN 1300
     L = 0
     FOR I = 1 TO 8
     FOR K = 1 TO 3
     L = L + 1
     ST(L) = ST(L) + FACT * DVOLU * CARTD(I, K)
```

```
      NEXT K, I
1300  NEXT LZ, LY, LX

2940  IF IEDGE = 0 THEN 2980

      '<---------- PRESSURE LOAD ----------------------------------->
      FOR IN = 1 TO NEDGE: IF NEASS(IN) <> KELEM THEN 2970

      KF = NSIDE(IN)          'FACE
      PR = PRESS(IN)          'PRESSURE

'     INTEGRATE OVER THE SURFACE

      ML = KCRD(KF)
      MM = IPERM(ML)
      MN = IPERM(MM)
      ETA(ML) = FVAL(KF)
      FOR LX = 1 TO NGAUSS
      ETA(MM) = PLACE(LX, NGAUSS)
      FOR LY = 1 TO NGAUSS
      ETA(MN) = PLACE(LY, NGAUSS)

      CALL DERIVE(ETA(1), ETA(2), ETA(3))

'     COMPUTE DIRECTION COSINES OF NORMAL TO SURFACE

      A1 = (XJACM(MM, 2) * XJACM(MN, 3) - XJACM(MM, 3) * XJACM(MN, 2))
      A2 = (XJACM(MM, 3) * XJACM(MN, 1) - XJACM(MM, 1) * XJACM(MN, 3))
      A3 = (XJACM(MM, 1) * XJACM(MN, 2) - XJACM(MM, 2) * XJACM(MN, 1))
      AA = SQR(A1 ^ 2 + A2 ^ 2 + A3 ^ 2)
      A1 = A1 / AA
      A2 = A2 / AA
      A3 = A3 / AA

'     COMPUTE FIRST FUND. FORM   (SIN /)

      AA = 0
      BB = 0
      CC = 0
      FOR I = 1 TO 3
      AA = AA + XJACM(MM, I) ^ 2
      CC = CC + XJACM(MN, I) ^ 2
      BB = BB + XJACM(MM, I) * XJACM(MN, I)
      NEXT I
      C = SQR(AA * CC - BB * BB)

'     COMPUTE PRESSURE,LOAD COMPONENTS, STORE IN R

      IF KTYPE = 2 GOTO 2950
      FORCE = PR
      GOTO 2960
2950  YY = 0!
      FOR I = 1 TO 8
      YY = YY + SHAPE(I) * ELCOD(2, I)
      NEXT I
      YY = YY - YREF
```

```
      FORCE = -PR * YY
      IF YY > 0 THEN FORCE = 0!

2960  TS = FORCE * WGT(LX, NGAUSS) * WGT(LY, NGAUSS) * C

      FOR I = 1 TO 4
      N = KFACE(KF, I)
      QQ = TS * SHAPE(N)
      K = 3 * N
      ST(K - 2) = ST(K - 2) + QQ * A1
      ST(K - 1) = ST(K - 1) + QQ * A2
      ST(K) = ST(K) + QQ * A3
      NEXT I

      NEXT LY, LX

2970  NEXT IN

2980  FOR I = 1 TO NEDOF: IEQ(I) = 0: NEXT I
      FOR I = 1 TO NNPE: FOR J = 1 TO NPDOF
      IF KODE(NP(KELEM, I), J) = 1 THEN IEQ((I - 1) * NPDOF + J) = 1
      NEXT J, I
      FOR I = 1 TO NEDOF
      IF IEQ(I) = 1 THEN 2990
      WRITE #1, ST(I)
2990  NEXT I

      NEXT IELEM

      CLOSE 1
3000  NEXT ICASE

      FINISHtime! = TIMER
      TOTALtime! = FINISHtime! - STARTtime!
      F$(15) = STR$(TOTALtime!)
      CHAIN "SOLVER"

DEFINT I-N
DEFDBL A-H, O-Z
'-----------------------------------------------------------------------------
SUB DERIVE (R, S, T) STATIC
      SHARED SHAPE(), DERIV(), ELCOD(), XJACM(), IPERM(), CARTD(), DET
      DEFINT I-N
      DEFDBL A-H, O-Z
      DIM B(3, 3)

      RP = (1! + R) * .125
      RM = (1! - R) * .125
      SP = 1! + S
      SM = 1! - S
      TP = 1! + T
      TM = 1! - T

'     SHAPE FUNCTIONS

      SHAPE(1) = RP * SM * TM
```

```
      SHAPE(2) = RP * SP * TM
      SHAPE(3) = RM * SP * TM
      SHAPE(4) = RM * SM * TM
      SHAPE(5) = RP * SM * TP
      SHAPE(6) = RP * SP * TP
      SHAPE(7) = RM * SP * TP
      SHAPE(8) = RM * SM * TP

'     DERIVATIVES OF SHAPE FUNCTIONS

      DERIV(1, 1) = SM * TM * .125
      DERIV(1, 2) = SP * TM * .125
      DERIV(1, 3) = -DERIV(1, 2)
      DERIV(1, 4) = -DERIV(1, 1)
      DERIV(1, 5) = SM * TP * .125
      DERIV(1, 6) = SP * TP * .125
      DERIV(1, 7) = -DERIV(1, 6)
      DERIV(1, 8) = -DERIV(1, 5)
'      DERIV(1,9) = -R
'      DERIV(1,10)= 0.
'      DERIV(1,11)= 0.

      DERIV(2, 1) = -RP * TM
      DERIV(2, 2) = -DERIV(2, 1)
      DERIV(2, 3) = RM * TM
      DERIV(2, 4) = -DERIV(2, 3)
      DERIV(2, 5) = -RP * TP
      DERIV(2, 6) = -DERIV(2, 5)
      DERIV(2, 7) = RM * TP
      DERIV(2, 8) = -DERIV(2, 7)
'      DERIV(2,9) = 0.
'      DERIV(2,10)=-S
'      DERIV(2,11)= 0.

      DERIV(3, 1) = -RP * SM
      DERIV(3, 2) = -RP * SP
      DERIV(3, 3) = -RM * SP
      DERIV(3, 4) = -RM * SM
      DERIV(3, 5) = -DERIV(3, 1)
      DERIV(3, 6) = -DERIV(3, 2)
      DERIV(3, 7) = -DERIV(3, 3)
      DERIV(3, 8) = -DERIV(3, 4)
'      DERIV(3,9) = 0.
'      DERIV(3,10)= 0.
'      DERIV(3,11)= -T

'     JACOBIAN MATRIX A

      FOR I = 1 TO 3
      FOR J = 1 TO 3
      C = 0!
      FOR L = 1 TO 8: C = C + DERIV(I, L) * ELCOD(J, L): NEXT L
      XJACM(I, J) = C
      NEXT J, I

'     INVERT JACOBIAN
```

```
    FOR I = 1 TO 3
    J = IPERM(I)
    K = IPERM(J)
    B(I, I) = XJACM(J, J) * XJACM(K, K) - XJACM(K, J) * XJACM(J, K)
    B(I, J) = XJACM(K, J) * XJACM(I, K) - XJACM(I, J) * XJACM(K, K)
    B(J, I) = XJACM(J, K) * XJACM(K, I) - XJACM(J, I) * XJACM(K, K)
    NEXT I
    DET = XJACM(1, 1) * B(1, 1) + XJACM(1, 2) * B(2, 1) + XJACM(1, 3) * B(3, 1)

'   MATRIX OF X-Y-Z DERIVATIVES

    FOR I = 1 TO 3
    FOR J = 1 TO 8
    C = 0
    FOR K = 1 TO 3: C = C + B(I, K) * DERIV(K, J): NEXT K
    CARTD(J, I) = C / DET
    NEXT J, I

END SUB
```

```
DECLARE SUB FIRST ()
DECLARE SUB HEADING ()
    '*************************************************************************
    '*                file name "PRTWAVE"                                    *
    '*                                                                       *
    '*    OBJECTS:-shows profile storage of structure,segment by segment     *
    '*                                                                       *
    '*************************************************************************

    DEFINT I-N
    DEFDBL A-H, O-Z

    COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    COMMON NCASE, NUMSEG, NGAUSS, KORDER, T$, E$, MAXVOL, F$()
    COMMON IOPTION

    COMMON LSTMAX, NEQMAX, KMAX
    '      Maximum nodes repersented in segment
    '      Maximum equations repersented in segment
    '      Maximum usage storage

    DIM IDIAG1(NEQMAX), IDIAG2(NEQMAX), MOVE(NEQMAX), IEQS(NEQMAX), NPR(NUMNP), IELE(NUMEL)
    DIM NPT(NUMEL, NNPE)
    DIM LPLOT(640)
    DIM LNEQ(NEQMAX)

    DOT$ = "C3 BR3 R0 BR3 R0 BR3 R0 BR3 R0 BR3 R0 BR3 R0"
    NUM$(0) = "BR3 U4 R3 D4 L3 BR3"
    NUM$(1) = "BR3 U4 BD4"
    NUM$(2) = "BR3 BU4 R3 D2 L3 D2 R3"
    NUM$(3) = "BR3 BU4 R3 D2 L2 BR2 D2 L3 BR3"
    NUM$(4) = "BR3 BU4 D3 R3 BL1 BU1 D2 BR1"
    NUM$(5) = "BR3 BU4 BR3 L3 D2 R3 D2 L3 BR3"
    NUM$(6) = "BR3 BU4 BR3 L3 D4 R3 U2 L3 BR3 BD2"
    NUM$(7) = "BR3 BU4 R3 D4"
    NUM$(8) = "BR3 U4 R3 D4 L3 U2 R3 BD2"
    NUM$(9) = "BR3 R3 U4 L3 D2 R3 BD2"

    DO$ = D$
    GOTO 200

    '<------------------ Subroutine area ---------------------------------
100 COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue  OR <ESC> to quit";
110 K$ = INKEY$: IF K$ = "" THEN 110
    LOCATE 25, 2: PRINT SPC(43); : PRINT : RETURN

120 COLOR 28: LOCATE 19, 1
    PRINT "
    PRINT "
    PRINT "
    COLOR 12: LOCATE 20, 8: PRINT "WAIT : data being read";
    RETURN

130 LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$: RETURN
140 LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$: RETURN
```

```
     '                    DISK SEGMENT INPUT
150  OPEN "B", 2, DD$ + "WVE." + ISEG$
     GET #2, , ISEG
     GET #2, , IELEX
     GET #2, , NEQ
     GET #2, , ICOMP
     GET #2, , MOVEX
     GET #2, , JUSTFY
     FOR I = 1 TO MOVEX: GET #2, , IDIAG1(I): NEXT I
     FOR I = 1 TO NEQ: GET #2, , IDIAG2(I): NEXT I
     FOR I = 1 TO MOVEX: GET #2, , MOVE(I): NEXT I
     FOR I = 1 TO NEQ: GET #2, , IEQS(I): NEXT I
     FOR I = 1 TO NUMNP: GET #2, , NPR(I): NEXT I
     FOR I = 1 TO IELEX: GET #2, , IELE(I): NEXT I
     FOR I = 1 TO IELEX
     FOR J = 1 TO NNPE
     GET #2, , NPT(I, J)
     NEXT J, I
     CLOSE 2
     RETURN

     '=========================================================================
200  ISEG = 1: W = 1
     ISEG$ = RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1)
     CLS : GOSUB 120
     GOSUB 150
210  CLS : GOSUB 130: GOSUB 140
     COLOR 11: PRINT "<<< PROFILE OF STORAGE MENU >>>"
     COLOR 15: PRINT TAB(5); "Master file name --> "; F$
     PRINT TAB(5); "TOTAL NUMBER OF SEGMENT = "; NUMSEG
     COLOR 14: PRINT
     PRINT TAB(5); "S  = SELECT SEGMENT #"; ISEG; "( Total "; NUMSEG; "segment )"
     PRINT TAB(5); "W  = ";
     IF W = 1 THEN PRINT "SHOW PROFILE ";  ELSE PRINT "SHOW WAVE DATA ";
     PRINT "(PROFILE or WAVE DATA)"
     PRINT TAB(5); "O  = ON SCREEN"
     PRINT TAB(5); "H  = HARD COPY"
     PRINT : COLOR 13
     PRINT TAB(5); "Q -> QUIT TO ACTIVITY MENU"
     LOCATE 20, 5: COLOR 13: PRINT "====> SELECT ? ";

220  K$ = INKEY$: LOCATE 20, 20, 1, 0, 7: IF K$ = "" THEN 220
     IF ASC(K$) = 0 THEN PLAY "L40 B": GOTO 220
     LOCATE 20, 20: PRINT K$;

     IF K$ = "S" OR K$ = "s" THEN 230
     IF K$ = "W" OR K$ = "w" THEN
          IF W = 1 THEN W = 0 ELSE W = 1
          GOTO 210
          END IF
     IF K$ = "O" OR K$ = "o" THEN
          IF W = 0 THEN 500 ELSE 300
          END IF
     IF K$ = "H" OR K$ = "h" THEN
          IF W = 0 THEN 700 ELSE 400
          END IF
```

```
    IF K$ = "Q" OR K$ = "q" THEN RUN "HEAD"
    PLAY "L40 B": GOTO 220

230 LOCATE 20, 1: PRINT SPC(70);
    LOCATE 20, 5: COLOR 15
    INPUT "INPUT SEGMENT NUMBER #", I
    IF I = ISEG THEN 210
    IF I > NUMSEG OR I < 1 THEN PLAY "L40 B": BEEP: GOTO 230
    ISEG$ = RIGHT$(STR$(I), LEN(STR$(I)) - 1)
    GOSUB 120
    GOSUB 150
    GOTO 210

    '---------------------- Profile on screen ----------------------------
300 SCREEN 1: CLS : NPAGE = 0: IEQ = 0
    WINDOW SCREEN (1, 1)-(320, 200)

    IEND = NEQ + 1
    FOR J = NEQ TO 2 STEP -1
    IHIGHT = IDIAG2(J) - IDIAG2(J - 1)
    IBGN = J - IHIGHT + 1
    IF IBGN >= IEND THEN 305
    FOR I = IBGN TO IEND - 1
    LNEQ(I) = J
    NEXT I
    IEND = IBGN
305 NEXT J

310 CLS
    NPAGE = NPAGE + 1
    FOR I = 1 TO 200
    IEQ = IEQ + 1: IF IEQ > NEQ THEN 330
    IF IEQ <= ICOMP THEN MCOLOR = 1 ELSE MCOLOR = 2
    FOR J = IEQ TO LNEQ(IEQ)
    KHIGHT = IDIAG2(J) - IDIAG2(J - 1)
    LHIGHT = KHIGHT - J + IEQ
    IF LHIGHT >= 1 THEN PSET (J - (NPAGE - 1) * 200, I), MCOLOR
    NEXT J
    IF IEQ MOD 20 > 0 THEN 320
    DRAW DOT$
    NUMBER$ = STR$(IEQ)
    FOR J = 2 TO LEN(NUMBER$)              'show equation number
    K = VAL(MID$(NUMBER$, J, 1))
    DRAW NUM$(K)
    NEXT J
320 NEXT I
330 K$ = INKEY$: IF K$ = "" THEN 330
    IF K$ = CHR$(27) THEN SCREEN 0: WIDTH 80: GOTO 210
    IF IEQ < NEQ THEN 310
    SCREEN 0: WIDTH 80: GOTO 210

    '-------- Profile HARD COPY by bit image on Epson Fx80 ---------------
400 CLS : GOSUB 130: GOSUB 140
    COLOR 15
    PRINT "┌──────────────────────────────────┐"
    PRINT "│      HARD COPY  for  PROFILE     │"
```

```
      PRINT "└──────────────────────────────────┘"
      PRINT
      COLOR 11: PRINT "Equation ..."
      H$ = "PROFILE"
      IPAGE = 0
      CALL FIRST: CALL HEADING
      LPRINT
      COLOR 15

      IEND = NEQ + 1
      FOR J = NEQ TO 2 STEP -1
      IHIGHT = IDIAG2(J) - IDIAG2(J - 1)
      IBGN = J - IHIGHT + 1
      IF IBGN >= IEND THEN 410
      FOR I = IBGN TO IEND - 1
      LNEQ(I) = J
      NEXT I
      IEND = IBGN
410   NEXT J

      DOT1 = (LEN(STR$(NEQ))) * 6 - 1 + 30          'no. of dot for eq. marking
      ONEQ = 0
      LPRINT CHR$(14);
      LPRINT "SEGMENT #"; ISEG
      CALL P(27): CALL P(51): CALL P(24)            'set line spacing

      LBIT = 0                                      ' counter of 8 bit print
      FOR I = 1 TO NEQ
      LOCATE 7, 13: PRINT I
      LBIT = LBIT + 1
      IF LBIT > 8 THEN GOSUB 430                    ' have to print data
      OLDDOT = MAXDOT
      IF LNEQ(I) - ONEQ > MAXDOT THEN MAXDOT = LNEQ(I) - ONEQ 'max. dot per line
      IF MAXDOT > 500 - DOT1 THEN
          MAXDOT = OLDDOT
      '    IF (I - 1) MOD 20 > 0 THEN 415
      '    KK = LNEQ(I - 1)
      '    FOR K = 1 TO 10
      '    LPLOT(KK + 3 * K - ONEQ) = LPLOT(KK + 3 * K - ONEQ) + 2 ^ (8 - LBIT)
      '    NEXT K
      '    MAXDOT = MAXDOT + 30
415   LFACT = I - 1
          GOSUB 430
          LPRINT CHR$(12);
          LPRINT CHR$(27) + "2";
          CALL HEADING
          LPRINT
          LPRINT CHR$(14);
          LPRINT "SEGMENT #"; ISEG; "(continue)"
          CALL P(27): CALL P(51): CALL P(24)        'set line spacing
          ONEQ = I - 1
          END IF
      FOR J = I TO LNEQ(I)
      KHIGHT = IDIAG2(J) - IDIAG2(J - 1)
      LHIGHT = KHIGHT - J + I
      IF LHIGHT < 1 THEN IFACT = 0 ELSE IFACT = 1
```

```
    LPLOT(J - ONEQ) = LPLOT(J - ONEQ) + (2 ^ (8 - LBIT)) * IFACT
    NEXT J
    IF I MOD 20 > 0 THEN 420
    LFACT = I                                   'mark of every 20 eq.
    KK = LNEQ(I)
    FOR K = 1 TO 10
    LPLOT(KK + 3 * K - ONEQ) = LPLOT(KK + 3 * K - ONEQ) + 2 ^ (8 - LBIT)
    NEXT K
    MAXDOT = MAXDOT + 30
420 NEXT I

    GOSUB 430                                   'print last data
    LPRINT CHR$(27) + "2";
    LPRINT CHR$(12);
    IPAGE = 0
    GOTO 210

430 CALL P(27): CALL P(42): CALL P(5)       'set dot density = 576  1:1
    N2 = INT(MAXDOT / 256): N1 = MAXDOT - (256 * N2)
    CALL P(N1): CALL P(N2)                  'set MAXDOT dot printed
    FOR L = 1 TO MAXDOT
    CALL P(LPLOT(L)): LPLOT(L) = 0
    NEXT L
    IF LFACT = 0 THEN 440
'    LPRINT CHR$(15);
'    LPRINT LFACT;
    LFACT = 0
440 LPRINT CHR$(10);
    LBIT = 1: MAXDOT = 0
    RETURN

    '------------------ Print WAVE DATA on screen -------------------------
500 CLS : GOSUB 130: GOSUB 140
    COLOR 11: LOCATE 1, 1: PRINT "SEGMENT #"; ISEG;
    COLOR 13: PRINT "NUMNP ="; NUMNP; ",NUMEL ="; NUMEL; ",NNPE ="; NNPE; ",IELEX ="; IELEX
    PRINT "NEQ ="; NEQ; ",ICOMP ="; ICOMP; ",MOVEX ="; MOVEX; ",JUSTFY ="; JUSTFY
    COLOR 15: PRINT STRING$(80, "-")

    VIEW PRINT 4 TO 25
    COLOR 11: PRINT "IDIAG1(MOVEX)"
    VIEW PRINT 5 TO 25
    K = 0
510 CLS : COLOR 15
    FOR I = 1 TO 20: FOR J = 1 TO 10
    K = K + 1: IF K > MOVEX THEN 520
    PRINT USING "#######"; IDIAG1(K);
    NEXT J
    PRINT USING "  ... ####"; K
    NEXT I
520 GOSUB 100: IF K$ = CHR$(27) THEN 650
    IF K < MOVEX THEN 510

    VIEW PRINT 4 TO 25: CLS
    COLOR 11: PRINT "IDIAG2(NEQ)"
    VIEW PRINT 5 TO 25
    K = 0
```

```
530  CLS : COLOR 15
     FOR I = 1 TO 20: FOR J = 1 TO 10
     K = K + 1: IF K > NEQ THEN 540
     PRINT USING "#######"; IDIAG2(K);
     NEXT J
     PRINT USING "  ... ####"; K
     NEXT I
540  GOSUB 100: IF K$ = CHR$(27) THEN 650
     IF K < NEQ THEN 530

     VIEW PRINT 4 TO 25: CLS
     COLOR 11: PRINT "MOVE(MOVEX)"
     VIEW PRINT 5 TO 25
     K = 0
550  CLS : COLOR 15
     FOR I = 1 TO 20: FOR J = 1 TO 10
     K = K + 1: IF K > MOVEX THEN 560
     PRINT USING "######"; MOVE(K);
     NEXT J
     PRINT USING "  ... ####"; K
     NEXT I
560  GOSUB 100: IF K$ = CHR$(27) THEN 650
     IF K < MOVEX THEN 550

     VIEW PRINT 4 TO 25: CLS
     COLOR 11: PRINT "IEQS(NEQ)"
     VIEW PRINT 5 TO 25
     K = 0
570  CLS : COLOR 15
     FOR I = 1 TO 20: FOR J = 1 TO 10
     K = K + 1: IF K > NEQ THEN 580
     PRINT USING "######"; IEQS(K);
     NEXT J
     PRINT USING "  ... ####"; K
     NEXT I
580  GOSUB 100: IF K$ = CHR$(27) THEN 650
     IF K < NEQ THEN 570

     VIEW PRINT 4 TO 25: CLS
     COLOR 11: PRINT "NPR(NUMNP)"
     VIEW PRINT 5 TO 25
     K = 0
590  CLS : COLOR 15
     FOR I = 1 TO 20: FOR J = 1 TO 10
     K = K + 1: IF K > NUMNP THEN 600
     PRINT USING "######"; NPR(K);
     NEXT J
     PRINT USING "  ... ####"; K
     NEXT I
600  GOSUB 100: IF K$ = CHR$(27) THEN 650
     IF K < NUMNP THEN 590

     VIEW PRINT 4 TO 25: CLS
     COLOR 11: PRINT "IELE(IELEX)"
     VIEW PRINT 5 TO 25
     K = 0
```

```
610  CLS : COLOR 15
     FOR I = 1 TO 20: FOR J = 1 TO 10
     K = K + 1: IF K > IELEX THEN 620
     PRINT USING "######"; IELE(K);
     NEXT J
     PRINT USING "  ... ####"; K
     NEXT I
620  GOSUB 100: IF K$ = CHR$(27) THEN 650
     IF K < IELEX THEN 610

     VIEW PRINT 4 TO 25: CLS
     COLOR 11: PRINT "NPT(IELEX,NNPE)"
     VIEW PRINT 5 TO 25
     K = 0
630  CLS : COLOR 15
     FOR I = 1 TO 20
     K = K + 1: IF K > IELEX THEN 640
     FOR J = 1 TO NNPE: PRINT USING "######"; NPT(K, J); : NEXT J
     PRINT USING "  ... ####"; K
     NEXT I
640  GOSUB 100: IF K$ = CHR$(27) THEN 650
     IF K < IELEX THEN 630

650  VIEW PRINT 1 TO 25
     GOTO 210

     '-------------------- Wave data HARDCOPY ----------------------------
700  CLS : GOSUB 130: GOSUB 140
     COLOR 15
     PRINT "                                          "
     PRINT "   HARD COPY   for   WAVE DATA            "
     PRINT "                                          "
     PRINT
     H$ = "WAVE DATA"
     IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
     IF ILINE < 56 THEN 710
     LPRINT CHR$(12);
     ILINE = 0
     CALL HEADING

710  LPRINT
     LPRINT CHR$(14); "** WAVE DATA **"; CHR$(15)
     LPRINT "SEGMENT #"; ISEG
     LPRINT "IELEX ="; IELEX; "   NEQ ="; NEQ; "   ICOMP ="; ICOMP;
     LPRINT "    MOVEX ="; MOVEX; "   JUSTFY ="; JUSTFY
     ILINE = ILINE + 4

     N$ = "IDIAG1(MOVEX)"
     ILINE = ILINE + 1
     IF ILINE <= 60 THEN 720
     LPRINT CHR$(12); : ILINE = 0
     CALL HEADING
     ILINE = ILINE + 1
720  LPRINT N$
     I = 0
730  GOSUB 1000
```

```
    FOR J = 1 TO 15
    I = I + 1: IF I > MOVEX THEN 740
    LPRINT USING "#######"; IDIAG1(I);
    NEXT J
    LPRINT USING " .... ####"; I
    GOTO 730
740 IF J > 1 THEN LPRINT

    N$ = "IDIAG2(NEQ)"
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 750
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    ILINE = ILINE + 1
750 LPRINT N$
    I = 0
760 GOSUB 1000
    FOR J = 1 TO 15
    I = I + 1: IF I > NEQ THEN 770
    LPRINT USING "#######"; IDIAG2(I);
    NEXT J
    LPRINT USING " .... ####"; I
    GOTO 760
770 IF J > 1 THEN LPRINT

    N$ = "MOVE(MOVEX)"
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 780
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    ILINE = ILINE + 1
780 LPRINT N$
    I = 0
790 GOSUB 1000
    FOR J = 1 TO 15
    I = I + 1: IF I > MOVEX THEN 800
    LPRINT USING "#######"; MOVE(I);
    NEXT J
    LPRINT USING " .... ####"; I
    GOTO 790
800 IF J > 1 THEN LPRINT

    N$ = "IEQS(NEQ)"
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 810
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    ILINE = ILINE + 1
810 LPRINT N$
    I = 0
820 GOSUB 1000
    FOR J = 1 TO 15
    I = I + 1: IF I > NEQ THEN 830
    LPRINT USING "#######"; IEQS(I);
    NEXT J
    LPRINT USING " .... ####"; I
```

```
      GOTO 820
830   IF J > 1 THEN LPRINT

      N$ = "NPR(NUMNP)"
      ILINE = ILINE + 1
      IF ILINE <= 60 THEN 840
      LPRINT CHR$(12); : ILINE = 0
      CALL HEADING
      ILINE = ILINE + 1
840   LPRINT N$
      I = 0
850   GOSUB 1000
      FOR J = 1 TO 15
      I = I + 1: IF I > NUMNP THEN 860
      LPRINT USING "#######"; NPR(I);
      NEXT J
      LPRINT USING " .... ####"; I
      GOTO 850
860   IF J > 1 THEN LPRINT

      N$ = "IELE(IELEX)"
      ILINE = ILINE + 1
      IF ILINE <= 60 THEN 870
      LPRINT CHR$(12); : ILINE = 0
      CALL HEADING
      ILINE = ILINE + 1
870   LPRINT N$
      I = 0
880   GOSUB 1000
      FOR J = 1 TO 15
      I = I + 1: IF I > IELEX THEN 890
      LPRINT USING "#######"; IELE(I);
      NEXT J
      LPRINT USING " .... ####"; I
      GOTO 880
890   IF J > 1 THEN LPRINT

      N$ = "NPT(IELEX,NNPE)"
      ILINE = ILINE + 1
      IF ILINE <= 60 THEN 900
      LPRINT CHR$(12); : ILINE = 0
      CALL HEADING
      ILINE = ILINE + 1
900   LPRINT N$
      FOR I = 1 TO IELEX
      GOSUB 1000
      FOR J = 1 TO NNPE: LPRINT USING "######"; NPT(I, J); : NEXT J
      LPRINT USING "  .... ####"; I
      NEXT I

      GOTO 210

1000  ILINE = ILINE + 1
      IF ILINE <= 60 THEN RETURN
      LPRINT CHR$(12); : ILINE = 0
      CALL HEADING
```

```
     LPRINT N$
     RETURN

SUB FIRST STATIC
     COLOR 12
     LOCATE 10, 25: PRINT "       PLEASE ADJUST PAPER"
     LOCATE 11, 25: PRINT "   Press any key to continue"
6000 K$ = INKEY$: IF K$ = "" THEN 6000
     LPRINT CHR$(27) + "C" + CHR$(66);
     COLOR 28
     LOCATE 10, 25: PRINT "┌──────────────────────────────┐"
     LOCATE 11, 25: PRINT "│                              │"
     LOCATE 12, 25: PRINT "└──────────────────────────────┘"
     LOCATE 11, 29: COLOR 12: PRINT "WAIT : data being print"
     LPRINT CHR$(15);
     WIDTH "LPT1:", 132
END SUB

DEFINT I-N
DEFDBL A-H, O-Z
SUB HEADING STATIC
SHARED ILINE, IPAGE, F$, T$, H$
     ILINE = ILINE + 4: IPAGE = IPAGE + 1
     FOR I = 1 TO 130: LPRINT "="; : NEXT I
     LPRINT CHR$(18);
     LPRINT CHR$(27) + "W" + CHR$(1); "SUPATFEAP"; CHR$(27) + "W" + CHR$(0);
     LPRINT CHR$(15);
     T = 110 - LEN(STR$(IPAGE)) - LEN(H$) - 7 + 1
     LPRINT TAB(T); "<"; H$; "> Page"; IPAGE
     LPRINT "PROJECT  : "; T$;
     LPRINT TAB(130 - LEN(F$) - 12 + 1); "FILE NAME : "; F$
     FOR I = 1 TO 130: LPRINT "="; : NEXT I: LPRINT
END SUB
```

```
DECLARE SUB FIRST ()
DECLARE SUB HEADING ()
    '*************************************************************************
    '*                                                                       *
    '*                    file name "PRTDIS"                                 *
    '*              Subprogram to print displacements                        *
    '*                                                                       *
    '*************************************************************************
    DEFINT I-N
    DEFDBL A-H, O-Z

    COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    COMMON NCASE, NUMSEG, NGAUSS, IORDER, T$, E$, MAXVOL, F$()
    COMMON IOPTION
    COMMON LSTMAX, NEQMAX, KMAX
    '      Maximum nodes repersented in segment
    '      Maximum equations repersented in segment
    '      Maximum usage storage

    DIM IFILE(5)
    DIM DIS(NUMNP, NPDOF), NBCJ(NUMNP), KODE(NUMNP, NPDOF)
    DIM NP(NUMEL, NNPE)

    GOTO 170

    '<------------------ Subroutine area ---------------------------------->
100 COLOR 11: LOCATE 24, 2: PRINT "Press any key to continue  OR <ESC> to quit";
110 K$ = INKEY$: IF K$ = "" THEN 110 ELSE RETURN

120 COLOR 14: LOCATE 23, 10: PRINT "<Return> = Accept     R = Reenter": COLOR 7
130 K$ = INKEY$: IF K$ = "" THEN 130
    IF K$ = "R" OR K$ = "r" OR K$ = CHR$(13) THEN 140 ELSE BEEP: GOTO 130
140 LOCATE 23, 10: PRINT SPC(40); : RETURN

150 LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$
    LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$: RETURN

    '*************************************************************************
    '*             MAIN PROGRAM to show DISPLACEMENT                         *
    '*                                                                       *
    '*************************************************************************

170 OPEN "I", 1, D$ + F$ + ".DIR"
    FOR I = 1 TO 25: IF NOT EOF(1) THEN INPUT #1, F$(I)
    NEXT I
    CLOSE 1

    FOR I = 1 TO 5: IF LEN(F$(I + 5)) > 1 THEN IFILE(I) = 1
    NEXT I

    OPEN "I", 1, D$ + F$ + ".ELE"
    FOR I = 1 TO NUMEL: FOR J = 1 TO NNPE: INPUT #1, NP(I, J): NEXT J, I
    CLOSE 1

    OPEN "I", 1, D$ + F$ + ".BOU"
    FOR I = 1 TO NUMNP: INPUT #1, NBCJ(I): NEXT I
```

```
    FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF: INPUT #1, KODE(I, J): NEXT J, I
    CLOSE 1

    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN 180
    NEXT I: STOP
180 ICASE = I: CASE$ = RIGHT$(STR$(ICASE), 1)

    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1

    OPEN "R", 1, D$ + F$ + ".D" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMNP
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 1 THEN 190
    GET #1
    DIS(I, J) = CVD(A$)
190 NEXT J
    NEXT I
    CLOSE 1

200 CLS : GOSUB 150
    COLOR 11: PRINT "<<< OUTPUT DISPLACEMENTS >>>"
    COLOR 13: PRINT "    LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT TAB(5); "L = CHANGE LOAD CASE ("; NCASE; "CASES:";
    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN PRINT " #"; RIGHT$(STR$(I), 1);
    NEXT I: PRINT " )"
    PRINT TAB(5); "O = ON SCREEN"
    PRINT TAB(5); "H = HARD COPY"
    PRINT : COLOR 12
    PRINT TAB(5); "Q -> QUIT TO ACTIVITY MENU"
    LOCATE 15, 5: PRINT "====> SELECT ?"

210 K$ = INKEY$: LOCATE 15, 20, 1, 0, 7: IF K$ = "" THEN 210
    LOCATE 15, 20: PRINT K$

    IF K$ = "L" OR K$ = "l" THEN 220
    IF K$ = "O" OR K$ = "o" THEN 240
    IF K$ = "H" OR K$ = "h" THEN 270
    IF K$ = "Q" OR K$ = "q" THEN RUN "HEAD"
    PLAY "L40 B": GOTO 210

    '<---------------------- SELECT LOAD CASE ---------------------------->
220 LOCATE 15, 1: PRINT SPC(78);
    LOCATE 15, 5: COLOR 15: INPUT "INPUT LOAD CASE NO.= ", I
    IF I < 1 OR I > 5 THEN PLAY "L40 B": GOTO 220
    IF IFILE(I) = 0 THEN PLAY "L40 B": GOTO 220
    GOSUB 120: IF K$ <> CHR$(13) THEN 220
    IF I = ICASE THEN 200

    ICASE = I: CASE$ = RIGHT$(STR$(ICASE), 1)
    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
```

```
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1

    OPEN "R", 1, D$ + F$ + ".D" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMNP
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 1 THEN 230
    GET #1
    DIS(I, J) = CVD(A$)
230 NEXT J
    NEXT I
    CLOSE 1
    GOTO 200

    '<---------------------- DISPL. ON SCREEN ---------------------------->
240 INUM = 0
250 CLS : GOSUB 150
    COLOR 13: LOCATE 1, 1: PRINT "** NODAL DISPLACEMENT **"
    PRINT "   LOAD CASE #"; CASE$; " : "; C$
    PRINT "  Node";
    FOR I = 1 TO NPDOF: PRINT "       DOF.-"; I; : NEXT I: PRINT

    COLOR 15
    FOR I = 1 TO 20
    INUM = INUM + 1: IF INUM > NUMNP THEN 260
    LOCATE I + 3, 1: PRINT USING "######"; INUM
    FOR J = 1 TO NPDOF
    LOCATE I + 3, (J - 1) * 15 + 10: PRINT USING "#.#####^^^^"; DIS(INUM, J): NEXT J
    NEXT I
260 GOSUB 100: IF K$ = CHR$(27) THEN 200
    IF INUM >= NUMNP THEN 200 ELSE 250

    '<---------------------- DISPL. ON HARD COPY ---------------------------->
270 CLS : GOSUB 150: COLOR 11
    PRINT "                                          "
    PRINT "  HARD COPY for NODAL DISPLACEMENT        "
    PRINT "                                          "
    PRINT
    H$ = "DISPL."
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 280
    LPRINT CHR$(12);
    ILINE = 0
    CALL HEADING

280 GOSUB 300
    FOR I = 1 TO NUMNP
    ILINE = ILINE + 1
    IF ILINE <= 60 THEN 290
    LPRINT CHR$(12); : ILINE = 0
    CALL HEADING
    GOSUB 300
290 LPRINT USING "####"; I;
    FOR J = 1 TO NPDOF
    LPRINT USING "         #.#####^^^^"; DIS(I, J); : NEXT J
```

```
    LPRINT
    NEXT I
    GOTO 200

    '           Print title
300 LPRINT
    LPRINT CHR$(14); "** NODAL DISPLACEMENT **"; CHR$(15)
    LPRINT "LOAD CASE #"; CASE$; " : "; C$
    LPRINT "Node";
    FOR K = 1 TO NPDOF: LPRINT USING "              DOF.- #"; K; : NEXT K: LPRINT
    FOR K = 1 TO (4 + (NPDOF * 20)): LPRINT "-"; : NEXT K: LPRINT
    ILINE = ILINE + 5
    RETURN


DEFINT I-N
DEFDBL A-H, O-Z
SUB FIRST STATIC
    COLOR 12
    LOCATE 10, 25: PRINT "PLEASE ADJUST PAPER"
    LOCATE 11, 25: PRINT "Press any key to continue"
6000 K$ = INKEY$: IF K$ = "" THEN 6000
    LPRINT CHR$(27) + "C" + CHR$(66);
    COLOR 28
    LOCATE 10, 25: PRINT "┌──────────────────────────┐"
    LOCATE 11, 25: PRINT "│                          │"
    LOCATE 12, 25: PRINT "└──────────────────────────┘"
    LOCATE 11, 29: COLOR 12: PRINT "WAIT : data being print"
    LPRINT CHR$(15);
    WIDTH "LPT1:", 132
END SUB

DEFINT I-N
DEFDBL A-H, O-Z
SUB HEADING STATIC
SHARED ILINE, IPAGE, F$, T$, H$
    ILINE = ILINE + 4: IPAGE = IPAGE + 1
    FOR I = 1 TO 130: LPRINT "="; : NEXT I
    LPRINT CHR$(18);
    LPRINT CHR$(27) + "W" + CHR$(1); "SUPATFEAP"; CHR$(27) + "W" + CHR$(0);
    LPRINT CHR$(15);
    LPRINT TAB(91); "<"; H$; ">Page"; IPAGE
    LPRINT "PROJECT  : "; T$; : LPRINT TAB(112); "FILE NAME: "; F$
    FOR I = 1 TO 130: LPRINT "="; : NEXT I: LPRINT
END SUB
```

```
DECLARE SUB FIRST ()
DECLARE SUB HEADING ()
    '*************************************************************************
    '*                                                                       *
    '*               file name "STRESS"                                      *
    '*   Subprogram to compute element stresses                              *
    '*   NTYPE = 1 ----> PLANE STRESS FINITE ELEMENT                         *
    '*           2 ----> PLANE STRAIN FINITE ELEMENT                         *
    '*           3 ----> PLATE BENDING FINITE ELEMENT                        *
    '*                                                                       *
    '*************************************************************************
    DEFINT I-N
    DEFDBL A-H, O-Z
    GOTO 170

    '<---------------------- SUBROUTINE AREA ---------------------------->
100 COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue OR <ESC> to quit";
110 K$ = INKEY$: IF K$ = "" THEN 110
    LOCATE 25, 2: PRINT SPC(45); : RETURN

120 COLOR 14: LOCATE 23, 10: PRINT "<Return> = Accept     R = Reenter"; : COLOR 7
130 K$ = INKEY$: IF K$ = "" THEN 130
    IF K$ = "R" OR K$ = "r" OR K$ = CHR$(13) THEN 140 ELSE BEEP: GOTO 130
140 LOCATE 23, 10: PRINT SPC(40); : RETURN

150 LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$: RETURN
160 LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$: RETURN

    '-------------------------------------------------------------------------

170 DIM A!(40), A$(36), F$(10), IFILE(5)

    DEF SEG = VARSEG(A!(0))
    BLOAD "MEMO.DAT", VARPTR(A!(0))
    D$ = CHR$(A!(40)) + ":"

    OPEN "I", 1, D$ + "DIRECTOR": INPUT #1, F$: CLOSE 1

    OPEN "I", 1, D$ + F$ + ".DIR"
    FOR I = 1 TO 10: INPUT #1, F$(I): NEXT I: CLOSE 1

    FOR I = 1 TO 5: IF LEN(F$(I + 5)) > 1 THEN IFILE(I) = 1
    NEXT I

    OPEN "I", 1, D$ + F$ + ".CON"
    INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    INPUT #1, NCASE
    INPUT #1, NUMSEG, NGAUSS, IORDER
    INPUT #1, T$
    INPUT #1, E$
    CLOSE 1

    DIM NP(NUMEL, NNPE), MATNO(NUMEL)
    DIM COORD(NUMNP * NDIME), PROPS(NMATS, 7)
    DIM NBCJ(NUMNP), KODE(NUMNP, NPDOF), TEMP(NUMNP), AL(6), OELEM(NUMEL)
```

```
    DIM ELDIS(NPDOF, NUMNP), STRSG(5), STRSP(3), STRIN(5)

    OPEN "I", 1, D$ + F$ + ".BOU"
    FOR I = 1 TO NUMNP
    INPUT #1, NBCJ(I)
    NEXT I
    FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
    INPUT #1, KODE(I, J)
    NEXT J, I
    CLOSE 1

    OPEN "I", 1, D$ + F$ + ".ELE"
    FOR I = 1 TO NUMEL: FOR J = 1 TO NNPE
    INPUT #1, NP(I, J)
    NEXT J, I
    FOR I = 1 TO NUMEL
    INPUT #1, MATNO(I)
    NEXT I
    CLOSE 1

    DEF SEG = VARSEG(COORD(1))
    BLOAD D$ + F$ + ".COO", VARPTR(COORD(1))

    OPEN "I", 1, D$ + F$ + ".MAT"
    FOR I = 1 TO NMATS: FOR J = 1 TO 5
    INPUT #1, PROPS(I, J)
    NEXT J, I
    CLOSE 1

    '<------ Integration constants ---------------------------------------->
    DIM ESTIF(24, 24), WGT(3, 3), PLACE(3, 3)
    FOR I = 1 TO 3: FOR J = 1 TO 3: READ WGT(I, J): NEXT J, I
    FOR I = 1 TO 3: FOR J = 1 TO 3: READ PLACE(I, J): NEXT J, I
    DATA 2      , 1                 , 0.555555555555556
    DATA 0      , 1                 , 0.888888888888889
    DATA 0      , 0                 , 0.555555555555556
    DATA 0      ,-0.577350269189626 ,-0.774596669241483
    DATA 0      , 0.577350269189626 , 0
    DATA 0      , 0                 , 0.774596669241483

    '<------------------ STRESS OUTPUT LOCATIONS ------------------------>
    DIM STPTS(9, 2), LOCOP(9), ILOCOP(9)
    FOR J = 1 TO 2: FOR I = 1 TO 9: READ STPTS(I, J): NEXT I, J
    DATA -1,  0,  1,  1,  1,  0, -1, -1,  0
    DATA -1, -1, -1,  0,  1,  1,  1,  0,  0

    DIM DMATX(5, 5), SHAPE(8), DERIV(2, 8), GPCOD(2), ELCOD(2, 8), XJACM(2, 2)
    DIM XJACI(2, 2), CARTD(2, 8), BMATX(5, 24), DBMAT(5, 24), STRAN(5), STRES(5)
    DIM LNODE(3), PGASH(2), DGASH(2), IEQ(24), XY(5, 3)

    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN 180
    NEXT I
180 ICASE = I: CASE$ = RIGHT$(STR$(ICASE), 1)

    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
```

```
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1

    OPEN "R", 1, D$ + F$ + ".D" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMNP
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 1 THEN 190
    GET #1
    ELDIS(J, I) = CVD(A$)
190 NEXT J
    NEXT I
    CLOSE 1

    '=====================================================================
200 CLS : GOSUB 150: GOSUB 160
    COLOR 11: PRINT "<<< OUTPUT ELEMENT STRESSES >>>"
    COLOR 15: PRINT "    LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT TAB(5); "L = CHANGE LOAD CASE ("; NCASE; "CASES:";
    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN PRINT " #"; RIGHT$(STR$(I), 1);
    NEXT I: PRINT ")"
    PRINT TAB(5); "O = ON SCREEN"
    PRINT TAB(5); "H = HARD COPY"
    PRINT TAB(5); "N = NO. OF OUTPUT LOCATIONS ("; NOP; ")"
    PRINT : COLOR 13
    PRINT TAB(5); "Q -> QUIT TO ACTIVITY MENU"
    LOCATE 15, 5: PRINT "====> SELECT ?"

210 K$ = INKEY$: LOCATE 15, 20, 1, 0, 7: IF K$ = "" THEN 210
    LOCATE 15, 20: PRINT K$;

    IF K$ = "L" OR K$ = "l" THEN 220
    IF K$ = "O" OR K$ = "o" THEN 410
    IF K$ = "H" OR K$ = "h" THEN 410
    IF K$ = "N" OR K$ = "n" THEN 800
    IF K$ = "Q" OR K$ = "q" THEN RUN "A:HEAD"
    PLAY "L40 B": GOTO 210

    '<------------------------ Select load case ----------------------->
220 LOCATE 15, 1: PRINT SPC(78);
    LOCATE 15, 5: COLOR 15: INPUT "INPUT LOAD CASE NO.= ", I
    IF I < 1 OR I > 5 THEN PLAY "L40 B": GOTO 220
    IF IFILE(I) = 0 THEN PLAY "L40 B": GOTO 220
    GOSUB 120: IF K$ <> CHR$(13) THEN 220
    IF I = ICASE THEN 200

    ICASE = I: CASE$ = RIGHT$(STR$(ICASE), 1)
    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1

    IF ITEMP = 0 THEN 230
    OPEN "I", 1, D$ + F$ + ".T" + CASE$
    FOR I = 1 TO NUMNP: INPUT #1, TEMP(I): NEXT I
```

```
    CLOSE 1

230 OPEN "R", 1, D$ + F$ + ".D" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMNP
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 1 THEN 240
    GET #1
    ELDIS(J, I) = CVD(A$)
240 NEXT J
    NEXT I
    CLOSE 1

    GOTO 200
    '------------------ Select element ----------------------------------
410 O$ = K$
    FOR I = 1 TO NUMEL: OELEM(I) = 0: NEXT I
    LOCATE 15, 1: PRINT SPC(30);
    COLOR 11: LOCATE 15, 1: PRINT ">ELEMENT LIST :"
    PRINT
    L$ = STRING$(78, "=")
    PRINT L$
    PRINT "    EXAMPLE:        1) ELEMENT LIST : 1 3 5 6 7 8 9    (INDIVIDUAL)"
    PRINT TAB(16); "OR   2) ELEMENT LIST : 1/5/2 6/9/1      (GENERATION)"
    PRINT TAB(16); "OR   3) ELEMENT LIST : 1 2 4 5/9/2      (COMBINATION)"
    PRINT TAB(16); "OR   3) ELEMENT LIST : A                (A = ALL)"

    LOCATE 15, 18: INPUT "", A$

    GOSUB 120: IF K$ <> CHR$(13) THEN 410

    IF A$ = "A" OR A$ = "a" THEN A$ = "1/" + RIGHT$(STR$(NUMEL), LEN(STR$(NUMEL)) - 1) + "/1"
    L = LEN(A$)
    IL = 0

420 IBGN$ = "": IEND$ = "": ISTEP$ = ""
430 IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 460
    IF IL$ <> "/" THEN IBGN$ = IBGN$ + IL$: GOTO 430
440 IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 460
    IF IL$ <> "/" THEN IEND$ = IEND$ + IL$: GOTO 440
450 IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 460
    IF IL$ <> "/" THEN ISTEP$ = ISTEP$ + IL$: GOTO 450
460 GOSUB 470
    IF IL < L THEN 420 ELSE GOTO 480

470 IBGN = VAL(IBGN$): IEND = VAL(IEND$): ISTEP = VAL(ISTEP$)
    IF IBGN < 1 OR IBGN > NUMEL THEN RETURN
    IF IEND > NUMEL THEN IEND = NUMEL
    IF ISTEP < 1 THEN OELEM(IBGN) = 1: RETURN
    FOR K = IBGN TO IEND STEP ISTEP
    OELEM(K) = 1: NEXT K
    RETURN

480 IF O$ = "O" OR K$ = "o" THEN 490
    IF O$ = "H" OR K$ = "h" THEN 590
    '<---------------------- Output on screen ---------------------------->
```

```
490  CLS : GOSUB 150: GOSUB 160
     COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue OR <ESC> to quit";
     LOCATE 1, 1: COLOR 11: PRINT "<<< OUTPUT ELEMENT STRESSES >>>"
     COLOR 15: PRINT "    LOAD CASE #"; CASE$; " : "; C$
     COLOR 13

     ON NTYPE GOTO 500, 500, 510

500  PRINT "  X-COORD.  Y-COORD.       X-STRESS       Y-STRESS      XY-STRESS       Z-STRESS"
     PRINT "                          MAX-STRESS     MIN-STRESS          ANGLE"
     PRINT STRING$(80, "-"): GOTO 520

510  PRINT "G.P X-COORD. Y-COORD.  X-MOMENT    Y-MOMENT   XY-MOMENT  XZ-S.FORCE  YZ-S.FORCE"
     PRINT STRING$(80, "-")

520  IF NTYPE < 3 THEN NSTRE = 3 ELSE NSTRE = 5
     NEDOF = NNPE * NPDOF
     NSTR1 = NSTRE + 1

     VIEW PRINT 6 TO 24

     '      Loop over each element

     FOR IELEM = 1 TO NUMEL
     IF OELEM(IELEM) = 0 THEN 570
     COLOR 14
     PRINT "Element no. "; IELEM
     PRINT "-----------"
     COLOR 15

     GOSUB 850                          'element coordinates
     GOSUB 900                          'comput D matrix

     '      Enter loop over each sampling point

     FOR L = 1 TO NOP
     GOSUB 1000                                    'compute stress
     PRINT USING "######.###"; GPCOD(1); GPCOD(2);
     ON NTYPE GOTO 530, 530, 540

530  FOR I = 1 TO 4: PRINT USING "    #.#####^^^^"; STRSG(I); : NEXT I: PRINT TAB(21);
     PRINT USING "    #.#####^^^^"; STRSP(1), STRSP(2); : PRINT USING "         ###.##"; STRSP(3): GOTO 550
540  FOR I = 1 TO 5: PRINT USING " #.#####^^^^"; STRSG(I); : NEXT I
550  PRINT
     NEXT L

560  K$ = INKEY$: IF K$ = "" THEN 560
     IF K$ = CHR$(27) THEN VIEW PRINT 1 TO 24: GOTO 580
570  NEXT IELEM
     VIEW PRINT 1 TO 24
580  GOTO 200

     '<-------------------- OUTPUT ON HARD COPY -------------------------->
590  CLS : GOSUB 150: GOSUB 160: COLOR 11
     IF NTYPE < 3 THEN NSTRE = 3 ELSE NSTRE = 5
     NEDOF = NNPE * NPDOF
```

```
    NSTR1 = NSTRE + 1
    PRINT "
    PRINT "  HARD COPY for ELEMENT STRESSES  "
    PRINT "
    PRINT
    H$ = "STRESS"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 600
    LPRINT CHR$(12);
    ILINE = 0
    CALL HEADING

600 GOSUB 670

    '      Loop over each element

    FOR IELEM = 1 TO NUMEL
    IF OELEM(IELEM) = 0 THEN 660
    IF (60 - ILINE) >= (NOP + 1) THEN 610
    LPRINT CHR$(12); : ILINE = 0: CALL HEADING: GOSUB 670          'page up
610 LPRINT "Element "; IELEM: ILINE = ILINE + 1
    GOSUB 850                          'element coordinates
    GOSUB 900                          'comput D matrix

    '      Enter loop over each sampling point

    FOR L = 1 TO NOP
    GOSUB 1000                                    'compute stress
    IF ILINE < 60 THEN 620
    LPRINT CHR$(12); : ILINE = 0: CALL HEADING: GOSUB 670          'page up
620 ILINE = ILINE + 1
    LPRINT USING "######.###"; GPCOD(1); GPCOD(2);
    ON NTYPE GOTO 630, 630, 640

630 FOR I = 1 TO 4: LPRINT USING "    #.#####^^^^"; STRSG(I); : NEXT I
    LPRINT USING "    #.#####^^^^"; STRSP(1), STRSP(2);
    LPRINT USING "    ###.##"; STRSP(3); : GOTO 650
640 FOR I = 1 TO 5: LPRINT USING "    #.#####^^^^"; STRSG(I); : NEXT I
650 LPRINT
    NEXT L
    LPRINT : ILINE = ILINE + 1
660 NEXT IELEM
    GOTO 200

    '           Print title
670 LPRINT
    LPRINT "** ELEMENT STRESS **"
    LPRINT "LOAD CASE #"; CASE$; " : "; C$
    ON NTYPE GOTO 680, 680, 690
680 LPRINT "  X-COORD.  Y-COORD.       X-STRESS       Y-STRESS       XY-STRESS       Z-STRESS       MAX P.S.
  MIN P.S.     ANGLE"
    LPRINT STRING$(120, "-"): GOTO 700
690 LPRINT "  X-COORD.  Y-COORD.       X-MOMENT       Y-MOMENT       XY-MOMENT     XZ-S.FORCE     YZ-S.FORCE"
    LPRINT STRING$(95, "-")

700 ILINE = ILINE + 5: RETURN
```

```
    '<---------------------- OUTPUT LOCATIONS ---------------------------->
800 CLS : GOSUB 150: GOSUB 160
    COLOR 11: PRINT "<<< OUTPUT LOCATIONS >>>"
    PRINT : COLOR 15
    PRINT "Maximum number of location = 9"
    PRINT "Location number is 1 to 9"
    PRINT : PRINT : COLOR 14
    PRINT "NUMBER OF OUTPUT LOCATIONS ="; NOP
    COLOR 15: PRINT STRING$(80, "="): COLOR 12
    PRINT "            ";
    FOR I = 1 TO 9: PRINT "   Loc"; USING "#"; I; : NEXT I: PRINT
    PRINT "LOC. LIST :";
    FOR I = 1 TO NOP: PRINT USING "#######"; LOCOP(I);
    NEXT I
    COLOR 15: PRINT STRING$(80, "=")
    GOSUB 120: IF K$ = CHR$(13) THEN 200

810 LOCATE 9, 30: INPUT "", NOP
    IF NOP > 9 THEN PLAY "L40 8": GOTO 810

    LOCATE 12, 10: PRINT SPC(65);
    FOR I = 1 TO NOP
820 LOCATE 12, 8 + I * 7: INPUT "", LOCOP(I)
    IF LOCOP(I) < 0 OR LOCOP(I) > 9 THEN 820
    NEXT I
    GOTO 800

    '<------------------- SET UP ELEMENT COORD. ---------------------------->
850 FOR I = 1 TO NDIME: FOR J = 1 TO NNPE
    ELCOD(I, J) = COORD((NP(IELEM, J) - 1) * NDIME + I)
    NEXT J, I
    RETURN

    '<---------------------- COMPUTE D MATRIX ---------------------------->
900 LPROP = MATNO(IELEM)
    YOUNG = PROPS(LPROP, 1)
    POISS = PROPS(LPROP, 2)
    THICK = PROPS(LPROP, 3)
    ALPHA = PROPS(LPROP, 5)
    FOR I = 1 TO NSTRE: FOR J = 1 TO NSTRE: DMATX(I, J) = 0: NEXT J, I
    ON NTYPE GOTO 910, 920, 930

   '------------------- D MATRIX FOR PLANE STRESS ------------------------
910 C = YOUNG / (1 - POISS ^ 2)
    DMATX(1, 1) = C: DMATX(2, 2) = C
    DMATX(1, 2) = C * POISS: DMATX(2, 1) = C * POISS
    DMATX(3, 3) = (1 - POISS) * C / 2
    GOTO 950

    '------------------- D MATRIX FOR PLANE STRAIN ------------------------
920 C = YOUNG * (1 - POISS) / ((1 + POISS) * (1 - 2 * POISS))
    DMATX(1, 1) = C: DMATX(2, 2) = C
    DMATX(1, 2) = C * POISS / (1 - POISS): DMATX(2, 1) = C * POISS / (1 - POISS)
    DMATX(3, 3) = C * (1 - 2 * POISS) / (2 * (1 - POISS))
    GOTO 950
```

```
    '------------------ D MATRIX FOR PLATE BENDING ------------------------
930 DMATX(1, 1) = YOUNG * THICK ^ 3 / (12 * (1 - POISS ^ 2)): DMATX(1, 2) = POISS * DMATX(1, 1)
    DMATX(2, 2) = DMATX(1, 1): DMATX(2, 1) = DMATX(1, 2)
    DMATX(3, 3) = (1 - POISS) * DMATX(1, 1) / 2
    DMATX(4, 4) = YOUNG * THICK / (2.4 * (1 + POISS)): DMATX(5, 5) = DMATX(4, 4)
    GOTO 950

950 RETURN

    '======================= COMPUTE STRESS ================================
1000 M = LOCOP(L)
    S = STPTS(M, 1): T = STPTS(M, 2)

    IF NNPE = 8 THEN 1010

    '<---------------- SHAPE FUNCTION & DERIVATIVES (Q4) ----------------->
    ST = S * T

    SHAPE(1) = (1 - T - S + ST) / 4: SHAPE(2) = (1 - T + S - ST) / 4
    SHAPE(3) = (1 + T + S + ST) / 4: SHAPE(4) = (1 + T - S - ST) / 4

    DERIV(1, 1) = (-1 + T) / 4: DERIV(2, 1) = (-1 + S) / 4
    DERIV(1, 2) = (1 - T) / 4: DERIV(2, 2) = (-1 - S) / 4
    DERIV(1, 3) = (1 + T) / 4: DERIV(2, 3) = (1 + S) / 4
    DERIV(1, 4) = (-1 - T) / 4: DERIV(2, 4) = (1 - S) / 4

    GOTO 1030

    '<---------------- SHAPE FUNCTION & DERIVATIVES (Q8) ----------------->
1010 S2 = 2 * S:  T2 = 2 * T:     SS = S * S:     TT = T * T
    ST = S * T:  SST = S * S * T: STT = S * T * T: ST2 = 2 * S * T

    SHAPE(1) = (-1 + ST + SS + TT - SST - STT) / 4
    SHAPE(2) = (1 - T - SS + SST) / 2
    SHAPE(3) = (-1 - ST + SS + TT - SST + STT) / 4
    SHAPE(4) = (1 + S - TT - STT) / 2
    SHAPE(5) = (-1 + ST + SS + TT + SST + STT) / 4
    SHAPE(6) = (1 + T - SS - SST) / 2
    SHAPE(7) = (-1 - ST + SS + TT + SST - STT) / 4
    SHAPE(8) = (1 - S - TT + STT) / 2

    DERIV(1, 1) = (T + S2 - ST2 - TT) / 4: DERIV(2, 1) = (S + T2 - SS - ST2) / 4
    DERIV(1, 2) = -S + ST: DERIV(2, 2) = (-1 + SS) / 2
    DERIV(1, 3) = (-T + S2 - ST2 + TT) / 4: DERIV(2, 3) = (-S + T2 - SS + ST2) / 4
    DERIV(1, 4) = (1 - TT) / 2: DERIV(2, 4) = (-T - ST)
    DERIV(1, 5) = (T + S2 + ST2 + TT) / 4: DERIV(2, 5) = (S + T2 + SS + ST2) / 4
    DERIV(1, 6) = -S - ST: DERIV(2, 6) = (1 - SS) / 2
    DERIV(1, 7) = (-T + S2 + ST2 - TT) / 4: DERIV(2, 7) = (-S + T2 + SS - ST2) / 4
    DERIV(1, 8) = (-1 + TT) / 2: DERIV(2, 8) = -T + ST

    '<---------------- CALCULATE COORDINATE OF SAMPLING POINT-------->
1030 FOR I = 1 TO NDIME: GPCOD(I) = 0
    FOR J = 1 TO NNPE: GPCOD(I) = GPCOD(I) + ELCOD(I, J) * SHAPE(J): NEXT J, I

    '<---------------- CREATE JACOBIAN MATRIX XJACM ----------------->
```

```
    FOR I = 1 TO NDIME: FOR J = 1 TO NDIME: XJACM(I, J) = 0
    FOR K = 1 TO NNPE: XJACM(I, J) = XJACM(I, J) + DERIV(I, K) * ELCOD(J, K)
    NEXT K, J, I

    '<-------- CALCULATE DETERMINANT AND INVERSE OF JACOBIAN MATRIX -------->
    DJACB = XJACM(1, 1) * XJACM(2, 2) - XJACM(1, 2) * XJACM(2, 1)
    IF DJACB <= 0 THEN PRINT "ZERO OR NEGATIVE ": STOP

    XJACI(1, 1) = XJACM(2, 2) / DJACB: XJACI(2, 2) = XJACM(1, 1) / DJACB
    XJACI(1, 2) = -XJACM(1, 2) / DJACB: XJACI(2, 1) = -XJACM(2, 1) / DJACB

    '<-------- CALCULATE CARTESIANS DERIVATIVES ---------------------------->
    FOR I = 1 TO NDIME: FOR J = 1 TO NNPE: CARTD(I, J) = 0
    FOR K = 1 TO NDIME
    CARTD(I, J) = CARTD(I, J) + XJACI(I, K) * DERIV(K, J)
    NEXT K, J, I

    '<------ EVALUATE THE B AND DB MATRICES ------------------------------->
    FOR I = 1 TO NSTRE: FOR J = 1 TO NEDOF: BMATX(I, J) = 0: NEXT J, I
    ON NTYPE GOTO 1060, 1060, 1070

    '<------ B MATRIX FOR PLANE STRESS , PLANE STRAIN -------------------->
1060 NGASH = 0
    FOR I = 1 TO NNPE: MGASH = NGASH + 1: NGASH = MGASH + 1
    BMATX(1, MGASH) = CARTD(1, I): BMATX(1, NGASH) = 0
    BMATX(2, MGASH) = 0: BMATX(2, NGASH) = CARTD(2, I)
    BMATX(3, MGASH) = CARTD(2, I): BMATX(3, NGASH) = CARTD(1, I)
    NEXT I
    GOTO 1090

    '<------ B MATRIX FOR PLATE BENDING ---------------------------------->
1070 JGASH = 0
    FOR I = 1 TO NNPE
    IGASH = JGASH + 1
    BMATX(4, IGASH) = CARTD(1, I): BMATX(5, IGASH) = CARTD(2, I)
    IGASH = IGASH + 1: JGASH = IGASH + 1
    BMATX(1, IGASH) = -CARTD(1, I): BMATX(3, IGASH) = -CARTD(2, I)
    BMATX(4, IGASH) = -SHAPE(I): BMATX(2, JGASH) = -CARTD(2, I)
    BMATX(3, JGASH) = -CARTD(1, I): BMATX(5, JGASH) = -SHAPE(I)
    NEXT I

    '<------ CALCULATE D*B ------------------------------------------------>
1090 FOR I = 1 TO NSTRE: FOR J = 1 TO NEDOF: DBMAT(I, J) = 0
    FOR K = 1 TO NSTRE: DBMAT(I, J) = DBMAT(I, J) + DMATX(I, K) * BMATX(K, J)
    NEXT K, J, I

    FOR ISTRE = 1 TO NSTRE
    STRSG(ISTRE) = 0
    KGASH = 0
    FOR INODE = 1 TO NNPE
    FOR IDOFN = 1 TO NPDOF
    KGASH = KGASH + 1
    STRSG(ISTRE) = STRSG(ISTRE) + DBMAT(ISTRE, KGASH) * ELDIS(IDOFN, NP(IELEM, INODE))
    NEXT IDOFN, INODE, ISTRE

    IF NTYPE = 3 THEN 1110
```

```
'       COMPUTE THE OUT OF PLANE NORMAL STRESS COMPONENT

    IF NTYPE = 1 THEN STRSG(4) = 0
    IF NTYPE = 2 THEN STRSG(4) = POISS * (STRSG(1) + STRSG(2))

'       FOR THERMAL LOADING ADD ON THE INITIAL THERMAL STRESSES

    IF ITEMP = 0 THEN 1100
    TEMP = 0
    FOR I = 1 TO NNPE: TEMP = TEMP + TEMP(NP(IELEM, I)) * SHAPE(I): NEXT I
    AL(1) = ALPHA: AL(2) = ALPHA: AL(3) = 0
    FOR I = 1 TO NSTRE: STRIN(I) = 0
    FOR J = 1 TO NSTRE
    STRIN(I) = STRIN(I) + DMATX(I, J) * AL(J)
    NEXT J, I
    FOR I = 1 TO NSTRE: STRIN(I) = -STRIN(I) * TEMP: NEXT I
    STRIN(NSTRE + 1) = 0
    FOR ISTR1 = 1 TO NSTR1
    STRSG(ISTR1) = STRSG(ISTR1) + STRIN(ISTR1)
    NEXT ISTR1

'       COMPUTE THE PRINCIPAL STRESSES

1100 XGASH = (STRSG(1) + STRSG(2)) * .5
    XGISH = (STRSG(1) - STRSG(2)) * .5
    XGESH = STRSG(3)
    XGOSH = SQR(XGISH * XGISH + XGESH * XGESH)
    STRSP(1) = XGASH + XGOSH
    STRSP(2) = XGASH - XGOSH
    IF XGISH = 0 THEN XGISH = 9.999999E-22
    STRSP(3) = ATN(XGESH / XGISH) * 28.647889757#
    IF STRSG(1) - STRSG(2) < 0 AND STRSG(3) < 0 THEN STRSP(3) = STRSP(3) - 90
    IF STRSG(1) - STRSG(2) < 0 AND STRSG(3) > 0 THEN STRSP(3) = STRSP(3) + 90
    '
    '       Print output element stresses
    '
1110 RETURN

DEFINT I-N
DEFDBL A-H, O-Z
SUB FIRST STATIC
    COLOR 12
    LOCATE 10, 25: PRINT "PLEASE ADJUST PAPER"
    LOCATE 11, 25: PRINT "Press any key to continue"
9000 K$ = INKEY$: IF K$ = "" THEN 9000
    LPRINT CHR$(27) + "C" + CHR$(66);
    LOCATE 10, 25: COLOR 28: PRINT "┌                          ┐"
    LOCATE 11, 25:        PRINT "│                          │"
    LOCATE 12, 25:        PRINT "└                          ┘"
    LOCATE 11, 29: COLOR 12: PRINT "WAIT : data being print"
    LPRINT CHR$(15);                          'set to compressed mode
    WIDTH "LPT1:", 132
END SUB

DEFINT I-N
```

```
DEFDBL A-H, O-Z
SUB HEADING STATIC
SHARED ILINE, IPAGE, F$, T$, H$
     ILINE = ILINE + 4: IPAGE = IPAGE + 1
     FOR I = 1 TO 130: LPRINT "="; : NEXT I
     LPRINT CHR$(18);                                  'release compressed mode
     LPRINT CHR$(27) + "W" + CHR$(1); "SUPATFEAP"; CHR$(27) + "W" + CHR$(0);
     LPRINT CHR$(15);
     LPRINT TAB(91); "<"; H$; "> Page"; IPAGE
     LPRINT "PROJECT  : "; T$; : LPRINT TAB(112); "FILE NAME: "; F$
     FOR I = 1 TO 130: LPRINT "="; : NEXT I: LPRINT
END SUB
```

```
DECLARE SUB FIRST ()
DECLARE SUB HEADING ()
     '*************************************************************************
     '*                                                                       *
     '*               file name "STRESS"                                      *
     '*   Subprogram to compute element stresses                              *
     '*   NTYPE = 4 ----> BRICK FINITE ELEMENT                                *
     '*                                                                       *
     '*************************************************************************
     DEFINT I-N
     DEFDBL A-H, O-Z
     GOTO 170

     '<---------------------- SUBROUTINE AREA ---------------------------->
100  COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue OR <ESC> to quit";
110  K$ = INKEY$: IF K$ = "" THEN 110
     LOCATE 25, 2: PRINT SPC(45); : RETURN

120  COLOR 14: LOCATE 23, 10: PRINT "<Return> = Accept     R = Reenter": COLOR 7
130  K$ = INKEY$: LOCATE 23, 43: IF K$ = "" THEN 130
     IF K$ = "R" OR K$ = "r" OR K$ = CHR$(13) THEN 140 ELSE BEEP: GOTO 130
140  LOCATE 23, 10: PRINT SPC(40); : RETURN

150  LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$: RETURN
160  LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$: RETURN

     '-------------------------------------------------------------------

170  DIM A!(40), A$(36), F$(10), IFILE(5)

     DEF SEG = VARSEG(A!(0))
     BLOAD "MEMO.DAT", VARPTR(A!(0))
     D$ = CHR$(A!(40)) + ":"

     OPEN "I", 1, D$ + "DIRECTOR": INPUT #1, F$: CLOSE 1

     OPEN "I", 1, D$ + F$ + ".DIR"
     FOR I = 1 TO 10: INPUT #1, F$(I): NEXT I: CLOSE 1

     FOR I = 1 TO 5: IF LEN(F$(I + 5)) > 1 THEN IFILE(I) = 1
     NEXT I

     OPEN "I", 1, D$ + F$ + ".CON"
     INPUT #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
     INPUT #1, NCASE
     INPUT #1, NUMSEG, NGAUSS, IORDER
     INPUT #1, T$
     INPUT #1, E$
     CLOSE 1

     DIM NP(NUMEL, NNPE), MATNO(NUMEL)
     DIM COORD(NUMNP * NDIME), PROPS(NMATS, 7)
     DIM NBCJ(NUMNP), KODE(NUMNP, NPDOF), TEMP(NUMNP), AL(6), OELEM(NUMEL)

     DIM ELDIS(NPDOF, NUMNP), STRSG(6), STRSP(3), STRIN(5)
```

```
OPEN "I", 1, D$ + F$ + ".BOU"
FOR I = 1 TO NUMNP
INPUT #1, NBCJ(I)
NEXT I
FOR I = 1 TO NUMNP: FOR J = 1 TO NPDOF
INPUT #1, KODE(I, J)
NEXT J, I
CLOSE 1

OPEN "I", 1, D$ + F$ + ".ELE"
FOR I = 1 TO NUMEL: FOR J = 1 TO NNPE
INPUT #1, NP(I, J)
NEXT J, I
FOR I = 1 TO NUMEL
INPUT #1, MATNO(I)
NEXT I
CLOSE 1

DEF SEG = VARSEG(COORD(1))
BLOAD D$ + F$ + ".COO", VARPTR(COORD(1))

OPEN "I", 1, D$ + F$ + ".MAT"
FOR I = 1 TO NMATS: FOR J = 1 TO 5
INPUT #1, PROPS(I, J)
NEXT J, I
CLOSE 1

'<------ Integration constants ---------------------------------------->
DIM ESTIF(24, 24), WGT(3, 3), PLACE(3, 3)
FOR I = 1 TO 3: FOR J = 1 TO 3: READ WGT(I, J): NEXT J, I
FOR I = 1 TO 3: FOR J = 1 TO 3: READ PLACE(I, J): NEXT J, I
DATA 2       , 1                  , 0.555555555555556
DATA 0       , 1                  , 0.888888888888889
DATA 0       , 0                  , 0.555555555555556
DATA 0       ,-0.577350269189626  ,-0.774596669241483
DATA 0       , 0.577350269189626  , 0
DATA 0       , 0                  , 0.774596669241483

'<------- Stress output locations ---------------------------------->
DIM STPTS(27, 3), LOCOP(9), ILOCOP(9)
FOR J = 1 TO 3: FOR I = 1 TO 8: READ STPTS(I, J): NEXT I, J
DATA  1,  1, -1, -1,  1,  1, -1, -1
DATA -1,  1,  1, -1, -1,  1,  1, -1
DATA -1, -1, -1, -1,  1,  1,  1,  1

FOR J = 1 TO 3: FOR I = 9 TO 20: READ STPTS(I, J): NEXT I, J
DATA  1,  0, -1,  0,  1,  1, -1, -1,  1,  0, -1,  0
DATA  0,  1,  0, -1, -1,  1,  1, -1,  0,  1,  0, -1
DATA -1, -1, -1, -1,  0,  0,  0,  0,  1,  1,  1,  1

FOR J = 1 TO 3: FOR I = 21 TO 27: READ STPTS(I, J): NEXT I, J
DATA  1,  0, -1,  0,  0,  0,  0
DATA  0,  1,  0, -1,  0,  0,  0
DATA  0,  0,  0,  0,  1, -1,  0

DIM DMATX(6, 6), SHAPE(8), DERIV(3, 8), GPCOD(3), ELCOD(3, 8), XJACM(3, 3)
```

```
    DIM XJACI(3, 3), CARTD(3, 8), BMATX(6, 24), DBMAT(6, 24), STRAN(5), STRES(5)
    DIM LNODE(3), PGASH(2), DGASH(2), IEQ(24), XY(5, 3)

    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN 180
    NEXT I
180 ICASE = I: CASE$ = RIGHT$(STR$(ICASE), 1)

    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1

    OPEN "R", 1, D$ + F$ + ".O" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMNP
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 1 THEN 190
    GET #1
    ELDIS(J, I) = CVD(A$)
190 NEXT J
    NEXT I
    CLOSE 1

    ';=====================================================================
200 CLS : GOSUB 150: GOSUB 160
    COLOR 11: PRINT "<<< OUTPUT ELEMENT STRESSES >>>"
    COLOR 15: PRINT "    LOAD CASE #"; CASE$; " : "; C$
    PRINT : COLOR 14
    PRINT TAB(5); "L = CHANGE LOAD CASE ("; NCASE; "CASES:";
    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN PRINT " #"; RIGHT$(STR$(I), 1);
    NEXT I: PRINT ")"
    PRINT TAB(5); "O = ON SCREEN"
    PRINT TAB(5); "H = HARD COPY"
    PRINT TAB(5); "N = NO. OF OUTPUT LOCATIONS ("; NOP; ")"
    PRINT : COLOR 13
    PRINT TAB(5); "Q -> QUIT TO ACTIVITY MENU"
    LOCATE 15, 5: PRINT "====> SELECT ?"

210 K$ = INKEY$: LOCATE 15, 20, 1, 0, 7: IF K$ = "" THEN 210
    LOCATE 15, 20: PRINT K$;

    IF K$ = "L" OR K$ = "l" THEN 220
    IF K$ = "O" OR K$ = "o" THEN 410
    IF K$ = "H" OR K$ = "h" THEN 410
    IF K$ = "N" OR K$ = "n" THEN 800
    IF K$ = "Q" OR K$ = "q" THEN RUN "A:HEAD"
    PLAY "L40 B": GOTO 210

    '<------------------------ Select load case ----------------------->
220 LOCATE 15, 1: PRINT SPC(78);
    LOCATE 15, 5: COLOR 15: INPUT "INPUT LOAD CASE NO.= ", I
    IF I < 1 OR I > 5 THEN PLAY "L40 B": GOTO 220
    IF IFILE(I) = 0 THEN PLAY "L40 B": GOTO 220
    GOSUB 120: IF K$ <> CHR$(13) THEN 220
    IF I = ICASE THEN 200
```

```
    ICASE = I: CASE$ = RIGHT$(STR$(ICASE), 1)
    OPEN "I", 1, D$ + F$ + ".LC" + CASE$
    INPUT #1, C$
    INPUT #1, IPLOD, IGRAV, IEDGE, ITEMP, IULOD
    CLOSE 1

    IF ITEMP = 0 THEN 230
    OPEN "I", 1, D$ + F$ + ".T" + CASE$
    FOR I = 1 TO NUMNP: INPUT #1, TEMP(I): NEXT I
    CLOSE 1

230 OPEN "R", 1, D$ + F$ + ".D" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMNP
    FOR J = 1 TO NPDOF
    IF KODE(I, J) = 1 THEN 240
    GET #1
    ELDIS(J, I) = CVD(A$)
240 NEXT J
    NEXT I
    CLOSE 1

    GOTO 200
    '------------------- Select element ----------------------------------
410 O$ = K$
    FOR I = 1 TO NUMEL: OELEM(I) = 0: NEXT I
    LOCATE 15, 1: PRINT SPC(30);
    COLOR 11: LOCATE 15, 1: PRINT ">ELEMENT LIST :"
    PRINT
    L$ = STRING$(78, "=")
    PRINT L$
    PRINT "    EXAMPLE:        1) ELEMENT LIST : 1 3 5 6 7 8 9    (INDIVIDUAL)"
    PRINT TAB(16); "OR   2) ELEMENT LIST : 1/5/2 6/9/1      (GENERATION)"
    PRINT TAB(16); "OR   3) ELEMENT LIST : 1 2 4 5/9/2      (COMBINATION)"
    PRINT TAB(16); "OR   3) ELEMENT LIST : A                (A = ALL)"

    LOCATE 15, 18: INPUT "", A$

    GOSUB 120: IF K$ <> CHR$(13) THEN 410

    IF A$ = "A" OR A$ = "a" THEN A$ = "1/" + RIGHT$(STR$(NUMEL), LEN(STR$(NUMEL)) - 1) + "/1"
    L = LEN(A$)
    IL = 0

420 IBGN$ = "": IEND$ = "": ISTEP$ = ""
430 IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 460
    IF IL$ <> "/" THEN IBGN$ = IBGN$ + IL$: GOTO 430
440 IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 460
    IF IL$ <> "/" THEN IEND$ = IEND$ + IL$: GOTO 440
450 IL = IL + 1: IL$ = MID$(A$, IL, 1): IF IL > L OR IL$ = " " THEN 460
    IF IL$ <> "/" THEN ISTEP$ = ISTEP$ + IL$: GOTO 450
460 GOSUB 470
    IF IL < L THEN 420 ELSE GOTO 480

470 IBGN = VAL(IBGN$): IEND = VAL(IEND$): ISTEP = VAL(ISTEP$)
    IF IBGN < 1 OR IBGN > NUMEL THEN RETURN
```

```
    IF IEND > NUMEL THEN IEND = NUMEL
    IF ISTEP < 1 THEN OELEM(IBGN) = 1: RETURN
    FOR K = IBGN TO IEND STEP ISTEP
    OELEM(K) = 1: NEXT K
    RETURN

480 IF O$ = "O" OR K$ = "o" THEN 490
    IF O$ = "H" OR K$ = "h" THEN 590
    '<---------------------- Output on screen ---------------------------->
490 CLS : GOSUB 150: GOSUB 160
    COLOR 11: LOCATE 25, 2: PRINT "Press any key to continue OR <ESC> to quit";
    LOCATE 1, 1: COLOR 11: PRINT "<<< OUTPUT ELEMENT STRESSES >>>"
    COLOR 15: PRINT "    LOAD CASE #"; CASE$; " : "; C$
    COLOR 13

    PRINT "LOC.    X-STRESS    Y-STRESS    Z-STRESS   XY-STRESS   YZ-STRESS   ZX-STRESS"
    PRINT STRING$(80, "-")

    NSTRE = 6
    NEDOF = NNPE * NPDOF

    VIEW PRINT 6 TO 24

    '     Loop over each element

    FOR IELEM = 1 TO NUMEL
    IF OELEM(IELEM) = 0 THEN 570
    COLOR 14
    PRINT "Element no. "; IELEM
    PRINT "-----------"
    COLOR 15

    GOSUB 850                          'element coordinates
    GOSUB 900                          'comput D matrix

    '     Enter loop over each sampling point

    FOR L = 1 TO NOP
    GOSUB 1000                                  'compute stress
    PRINT USING "##  "; M;
    FOR I = 1 TO 6: PRINT USING " #.#####^^^^"; STRSG(I); : NEXT I
    PRINT
    NEXT L

560 K$ = INKEY$: IF K$ = "" THEN 560
    IF K$ = CHR$(27) THEN VIEW PRINT 1 TO 24: GOTO 580
570 NEXT IELEM
    VIEW PRINT 1 TO 24
580 GOTO 200

    '<--------------------- OUTPUT ON HARD COPY ------------------------->
590 CLS : GOSUB 150: GOSUB 160: COLOR 11
    NSTRE = 6
    NEDOF = NNPE * NPDOF
    PRINT "┌──────────────────────────────────┐"
    PRINT "│ HARD COPY for ELEMENT STRESSES   │"
```

```
    PRINT "└──────────────────────────────┘"
    PRINT
    H$ = "STRESS"
    IF IPAGE = 0 THEN CALL FIRST: CALL HEADING
    IF ILINE < 56 THEN 600
    LPRINT CHR$(12);
    ILINE = 0
    CALL HEADING

600 GOSUB 670

    '      Loop over each element

    FOR IELEM = 1 TO NUMEL
    IF OELEM(IELEM) = 0 THEN 660
    IF (60 - ILINE) >= (NOP + 1) THEN 610
    LPRINT CHR$(12); : ILINE = 0: CALL HEADING: GOSUB 670          'page up
610 LPRINT "Element "; IELEM: ILINE = ILINE + 1
    GOSUB 850                          'element coordinates
    GOSUB 900                          'comput D matrix

    '      Enter loop over each sampling point

    FOR L = 1 TO NOP
    GOSUB 1000                                    'compute stress
    IF ILINE < 60 THEN 620
    LPRINT CHR$(12); : ILINE = 0: CALL HEADING: GOSUB 670          'page up
620 ILINE = ILINE + 1
    LPRINT USING "  ##"; M;
    FOR I = 1 TO 6: LPRINT USING "    #.#####^^^^"; STRSG(I); : NEXT I
    LPRINT
    NEXT L
    LPRINT : ILINE = ILINE + 1
660 NEXT IELEM
    GOTO 200

    '           Print title
670 LPRINT
    LPRINT "** ELEMENT STRESS **"
    LPRINT "LOAD CASE #"; CASE$; " : "; C$
    PRINT "LOC.       X-STRESS       Y-STRESS       Z-STRESS      XY-STRESS      YZ-STRESS      ZX-STRESS"
    LPRINT STRING$(94, "-")
700 ILINE = ILINE + 5: RETURN

    '<---------------------- OUTPUT LOCATIONS ---------------------------->
800 CLS : GOSUB 150: GOSUB 160
    COLOR 11: PRINT "<<< OUTPUT LOCATIONS >>>"
    PRINT : COLOR 15
    PRINT "Maximum number of location = 9"
    PRINT "Location number is 1 to 27"
    PRINT : PRINT : COLOR 14
    PRINT "NUMBER OF OUTPUT LOCATIONS ="; NOP
    COLOR 15: PRINT STRING$(80, "="): COLOR 12
    PRINT "           ";
    FOR I = 1 TO 9: PRINT "   Loc"; USING "#"; I; : NEXT I: PRINT
    PRINT "LOC. LIST :";
```

```
    FOR I = 1 TO NOP: PRINT USING "#######"; LOCOP(I);
    NEXT I
    COLOR 15: PRINT STRING$(80, "=")
    GOSUB 120: IF K$ = CHR$(13) THEN 200

810 LOCATE 9, 30: INPUT "", NOP
    IF NOP > 9 THEN PLAY "L40 8": GOTO 810

    LOCATE 12, 10: PRINT SPC(65);
    FOR I = 1 TO NOP
820 LOCATE 12, 8 + I * 7: INPUT "", LOCOP(I)
    IF LOCOP(I) < 0 OR LOCOP(I) > 27 THEN 820
    NEXT I
    GOTO 800

    '<-------------------- SET UP ELEMENT COORD. -------------------------->
850 FOR I = 1 TO NDIME: FOR J = 1 TO NNPE
    ELCOD(I, J) = COORD((NP(IELEM, J) - 1) * NDIME + I)
    NEXT J, I
    RETURN

    '<---------------------- COMPUTE D MATRIX ----------------------------->
900 LPROP = MATNO(IELEM)
    YOUNG = PROPS(LPROP, 1)
    POISS = PROPS(LPROP, 2)
    THICK = PROPS(LPROP, 3)
    ALPHA = PROPS(LPROP, 5)
    FOR I = 1 TO NSTRE: FOR J = 1 TO NSTRE: DMATX(I, J) = 0: NEXT J, I

    '------------------ D MATRIX FOR BRICK ELEMENT ------------------------
    C = YOUNG * (1 - POISS) / ((1 + POISS) * (1 - 2 * POISS))
    DMATX(1, 1) = C: DMATX(1, 2) = C * POISS / (1 - POISS)
    DMATX(1, 3) = DMATX(1, 2): DMATX(2, 2) = C
    DMATX(2, 3) = DMATX(1, 2): DMATX(2, 1) = DMATX(1, 2)
    DMATX(3, 1) = DMATX(1, 3): DMATX(3, 2) = DMATX(2, 3): DMATX(3, 3) = C
    DMATX(4, 4) = C * (1 - 2 * POISS) / (2 * (1 - POISS))
    DMATX(5, 5) = DMATX(4, 4): DMATX(6, 6) = DMATX(4, 4)

    RETURN

    '======================== COMPUTE STRESS ================================
1000 M = LOCOP(L)
    R = STPTS(M, 1): S = STPTS(M, 2): T = STPTS(M, 3)

    '------------ SHAPE FUNCTION & DERIVATIVE FOR BRICK ELEMENT -------------
    ST = S * T: RT = R * T: RS = R * S: RST = R * S * T

    SHAPE(1) = (1 - T - S + ST + R - RT - RS + RST) / 8
    SHAPE(2) = (1 - T + S - ST + R - RT + RS - RST) / 8
    SHAPE(3) = (1 - T + S - ST - R + RT - RS + RST) / 8
    SHAPE(4) = (1 - T - S + ST - R + RT + RS - RST) / 8
    SHAPE(5) = (1 + T - S - ST + R + RT - RS - RST) / 8
    SHAPE(6) = (1 + T + S + ST + R + RT + RS + RST) / 8
    SHAPE(7) = (1 + T + S + ST - R - RT - RS - RST) / 8
    SHAPE(8) = (1 + T - S - ST - R - RT + RS + RST) / 8
```

```
    DERIV(1, 1) = (1 - T - S + ST) / 8: DERIV(2, 1) = (-1 + T - R + RT) / 8
    DERIV(1, 2) = (1 - T + S - ST) / 8: DERIV(2, 2) = (1 - T + R - RT) / 8
    DERIV(1, 3) = (-1 + T - S + ST) / 8: DERIV(2, 3) = (1 - T - R + RT) / 8
    DERIV(1, 4) = (-1 + T + S - ST) / 8: DERIV(2, 4) = (-1 + T + R - RT) / 8
    DERIV(1, 5) = (1 + T - S - ST) / 8: DERIV(2, 5) = (-1 - T - R - RT) / 8
    DERIV(1, 6) = (1 + T + S + ST) / 8: DERIV(2, 6) = (1 + T + R + RT) / 8
    DERIV(1, 7) = (-1 - T - S - ST) / 8: DERIV(2, 7) = (1 + T - R - RT) / 8
    DERIV(1, 8) = (-1 - T + S + ST) / 8: DERIV(2, 8) = (-1 - T + R + RT) / 8

    DERIV(3, 1) = (-1 + S - R + RS) / 8
    DERIV(3, 2) = (-1 - S - R - RS) / 8
    DERIV(3, 3) = (-1 - S + R + RS) / 8
    DERIV(3, 4) = (-1 + S + R - RS) / 8
    DERIV(3, 5) = (1 - S + R - RS) / 8
    DERIV(3, 6) = (1 + S + R + RS) / 8
    DERIV(3, 7) = (1 + S - R - RS) / 8
    DERIV(3, 8) = (1 - S - R + RS) / 8

    '<---------------- CALCULATE COORDINATE OF SAMPLING POINT-------->
1030 FOR I = 1 TO NDIME: GPCOD(I) = 0
    FOR J = 1 TO NNPE: GPCOD(I) = GPCOD(I) + ELCOD(I, J) * SHAPE(J): NEXT J, I

    '<---------------- CREATE JACOBIAN MATRIX XJACM ----------------->
    FOR I = 1 TO NDIME: FOR J = 1 TO NDIME: XJACM(I, J) = 0
    FOR K = 1 TO NNPE: XJACM(I, J) = XJACM(I, J) + DERIV(I, K) * ELCOD(J, K)
    NEXT K, J, I

    '-------- Calculate determinant & inverse of jacobian matrix (3x3)-------
1040 DJACB = XJACM(1, 1) * (XJACM(2, 2) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 2))
    DJACB = DJACB - XJACM(1, 2) * (XJACM(2, 1) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 1))
    DJACB = DJACB + XJACM(1, 3) * (XJACM(2, 1) * XJACM(3, 2) - XJACM(2, 2) * XJACM(3, 1))
    IF DJACB <= 0 THEN PRINT "ZERO OR NEGATIVE ": STOP

    XJACI(1, 1) = (XJACM(2, 2) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 2)) / DJACB
    XJACI(1, 2) = -(XJACM(1, 2) * XJACM(3, 3) - XJACM(1, 3) * XJACM(3, 2)) / DJACB
    XJACI(1, 3) = (XJACM(1, 2) * XJACM(2, 3) - XJACM(1, 3) * XJACM(2, 2)) / DJACB
    XJACI(2, 1) = -(XJACM(2, 1) * XJACM(3, 3) - XJACM(2, 3) * XJACM(3, 1)) / DJACB
    XJACI(2, 2) = (XJACM(1, 1) * XJACM(3, 3) - XJACM(1, 3) * XJACM(3, 1)) / DJACB
    XJACI(2, 3) = -(XJACM(1, 1) * XJACM(2, 3) - XJACM(1, 3) * XJACM(2, 1)) / DJACB
    XJACI(3, 1) = (XJACM(2, 1) * XJACM(3, 2) - XJACM(2, 2) * XJACM(3, 1)) / DJACB
    XJACI(3, 2) = -(XJACM(1, 1) * XJACM(3, 2) - XJACM(1, 2) * XJACM(3, 1)) / DJACB
    XJACI(3, 3) = (XJACM(1, 1) * XJACM(2, 2) - XJACM(1, 2) * XJACM(2, 1)) / DJACB

    '<-------- CALCULATE CARTESIANS DERIVATIVES ---------------------------->
1050 FOR I = 1 TO NDIME: FOR J = 1 TO NNPE: CARTD(I, J) = 0
    FOR K = 1 TO NDIME
    CARTD(I, J) = CARTD(I, J) + XJACI(I, K) * DERIV(K, J)
    NEXT K, J, I

    '<------ EVALUATE THE B AND DB MATRICES ------------------------------->
    FOR I = 1 TO NSTRE: FOR J = 1 TO NEDOF: BMATX(I, J) = 0: NEXT J, I
    ON NTYPE GOTO 1060, 1060, 1070, 1080

    '<------ B MATRIX FOR PLANE STRESS , PLANE STRAIN --------------------->
1060 NGASH = 0
    FOR I = 1 TO NNPE: MGASH = NGASH + 1: NGASH = MGASH + 1
```

```
    BMATX(1, MGASH) = CARTD(1, I): BMATX(1, NGASH) = 0
    BMATX(2, MGASH) = 0: BMATX(2, NGASH) = CARTD(2, I)
    BMATX(3, MGASH) = CARTD(2, I): BMATX(3, NGASH) = CARTD(1, I)
    NEXT I
    GOTO 1090

    '<------ B MATRIX FOR PLATE BENDING ---------------------------------->
1070 JGASH = 0
    FOR I = 1 TO NNPE
    IGASH = JGASH + 1
    BMATX(4, IGASH) = CARTD(1, I): BMATX(5, IGASH) = CARTD(2, I)
    IGASH = IGASH + 1: JGASH = IGASH + 1
    BMATX(1, IGASH) = -CARTD(1, I): BMATX(3, IGASH) = -CARTD(2, I)
    BMATX(4, IGASH) = -SHAPE(I): BMATX(2, JGASH) = -CARTD(2, I)
    BMATX(3, JGASH) = -CARTD(1, I): BMATX(5, JGASH) = -SHAPE(I)
    NEXT I
    GOTO 1090

    '------------------- B MATRIX FOR BRICK ELEMENT -------------------------
1080 NGASH = 0
    FOR I = 1 TO NNPE
    LGASH = NGASH + 1: MGASH = LGASH + 1: NGASH = MGASH + 1
    BMATX(1, LGASH) = CARTD(1, I): BMATX(1, MGASH) = 0
    BMATX(1, NGASH) = 0
    BMATX(2, LGASH) = 0: BMATX(2, MGASH) = CARTD(2, I)
    BMATX(2, NGASH) = 0
    BMATX(3, LGASH) = 0: BMATX(3, MGASH) = 0
    BMATX(3, NGASH) = CARTD(3, I)
    BMATX(4, LGASH) = CARTD(2, I): BMATX(4, MGASH) = CARTD(1, I)
    BMATX(4, NGASH) = 0
    BMATX(5, LGASH) = 0: BMATX(5, MGASH) = CARTD(3, I)
    BMATX(5, NGASH) = CARTD(2, I)
    BMATX(6, LGASH) = CARTD(3, I): BMATX(3, MGASH) = 0
    BMATX(6, NGASH) = CARTD(1, I)
    NEXT I

    '<------ CALCULATE D*B ---------------------------------------->
1090 FOR I = 1 TO NSTRE: FOR J = 1 TO NEDOF: DBMAT(I, J) = 0
    FOR K = 1 TO NSTRE: DBMAT(I, J) = DBMAT(I, J) + DMATX(I, K) * BMATX(K, J)
    NEXT K, J, I

    FOR ISTRE = 1 TO NSTRE
    STRSG(ISTRE) = 0
    KGASH = 0
    FOR INODE = 1 TO NNPE
    FOR IDOFN = 1 TO NPDOF
    KGASH = KGASH + 1
    STRSG(ISTRE) = STRSG(ISTRE) + DBMAT(ISTRE, KGASH) * ELDIS(IDOFN, NP(IELEM, INODE))
    NEXT IDOFN, INODE, ISTRE

RETURN

    IF NTYPE = 3 THEN 1110

    '      COMPUTE THE OUT OF PLANE NORMAL STRESS COMPONENT
```

```
    IF NTYPE = 1 THEN STRSG(4) = 0
    IF NTYPE = 2 THEN STRSG(4) = POISS * (STRSG(1) + STRSG(2))

    '       FOR THERMAL LOADING ADD ON THE INITIAL THERMAL STRESSES

    IF ITEMP = 0 THEN 1100
    TEMP = 0
    FOR I = 1 TO NNPE: TEMP = TEMP + TEMP(NP(IELEM, I)) * SHAPE(I): NEXT I
    AL(1) = ALPHA: AL(2) = ALPHA: AL(3) = 0
    FOR I = 1 TO NSTRE: STRIN(I) = 0
    FOR J = 1 TO NSTRE
    STRIN(I) = STRIN(I) + DMATX(I, J) * AL(J)
    NEXT J, I
    FOR I = 1 TO NSTRE: STRIN(I) = -STRIN(I) * TEMP: NEXT I
    STRIN(NSTRE + 1) = 0
    FOR ISTR1 = 1 TO NSTR1
    STRSG(ISTR1) = STRSG(ISTR1) + STRIN(ISTR1)
    NEXT ISTR1

    '       COMPUTE THE PRINCIPAL STRESSES

1100 XGASH = (STRSG(1) + STRSG(2)) * .5
    XGISH = (STRSG(1) - STRSG(2)) * .5
    XGESH = STRSG(3)
    XGOSH = SQR(XGISH * XGISH + XGESH * XGESH)
    STRSP(1) = XGASH + XGOSH
    STRSP(2) = XGASH - XGOSH
    IF XGISH = 0 THEN XGISH = 9.999999E-22
    STRSP(3) = ATN(XGESH / XGISH) * 28.647889757#
    IF STRSG(1) - STRSG(2) < 0 AND STRSG(3) < 0 THEN STRSP(3) = STRSP(3) - 90
    IF STRSG(1) - STRSG(2) < 0 AND STRSG(3) > 0 THEN STRSP(3) = STRSP(3) + 90
    '
    '       Print output element stresses
    '
1110 RETURN

DEFINT I-N
DEFDBL A-H, O-Z
SUB FIRST STATIC
    COLOR 12
    LOCATE 10, 25: PRINT "PLEASE ADJUST PAPER"
    LOCATE 11, 25: PRINT "Press any key to continue"
9000 K$ = INKEY$: IF K$ = "" THEN 9000
    LPRINT CHR$(27) + "C" + CHR$(66);
    LOCATE 10, 25: COLOR 28: PRINT "┌──────────────────────────┐"
    LOCATE 11, 25:           PRINT "│                          │"
    LOCATE 12, 25:           PRINT "└──────────────────────────┘"
    LOCATE 11, 29: COLOR 12: PRINT "WAIT : data being print"
    LPRINT CHR$(15);                          'set to compressed mode
    WIDTH "LPT1:", 132
END SUB

DEFINT I-N
DEFDBL A-H, O-Z
SUB HEADING STATIC
SHARED ILINE, IPAGE, F$, T$, H$
```

```
    ILINE = ILINE + 4: IPAGE = IPAGE + 1
    FOR I = 1 TO 130: LPRINT "="; : NEXT I
    LPRINT CHR$(18);                                'release compressed mode
    LPRINT CHR$(27) + "W" + CHR$(1); "SUPATFEAP"; CHR$(27) + "W" + CHR$(0);
    LPRINT CHR$(15);
    LPRINT TAB(91); "<"; H$; "> Page"; IPAGE
    LPRINT "PROJECT  : "; T$; : LPRINT TAB(112); "FILE NAME: "; F$
    FOR I = 1 TO 130: LPRINT "="; : NEXT I: LPRINT
END SUB
```

```
DECLARE SUB CHECK ()
DECLARE SUB MYKEY (KP%)
DECLARE SUB STATUS ()
DECLARE SUB MENU ()
DECLARE SUB DEC28IN (N%, B$)
    '*************************************************************************
    '*                                                                       *
    '*                    file name "PL"                                     *
    '*                                                                       *
    '*      PLOT OF 2-D & 3-D  STRUCTURE                                     *
    '*                                                                       *
    '*************************************************************************
    DEFINT I-N

    COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
    COMMON NCASE, NUMSEG, NGAUSS, IORDER, T$, E$, MAXVOL, F$()
    COMMON IOPTION

    '-------------------------------------------------------------------------
    DIM NUM$(10), NO(16), NFLAG(16)

    L$ = STRING$(79, "=")
    NODE$ = "BU2 R4 D4 L8 U4 R8"
    NUM$(0) = "BR5 U4 R5 D4 L5 BR5"
    NUM$(1) = "BR5 U4 BD4"
    NUM$(2) = "BR5 BU4 R5 D2 L5 D2 R5"
    NUM$(3) = "BR5 BU4 R5 D2 L3 BR3 D2 L5 BR5"
    NUM$(4) = "BR5 BU4 D3 R5 BL1 BU1 D2 BR1"
    NUM$(5) = "BR5 BU4 BR5 L5 D2 R5 D2 L5 BR5"
    NUM$(6) = "BR5 BU4 BR5 L5 D4 R5 U2 L5 BR5 BD2"
    NUM$(7) = "BR5 BU4 R5 D4"
    NUM$(8) = "BR5 U4 R5 D4 L5 U2 R5 BD2"
    NUM$(9) = "BR5 R5 U4 L5 D2 R5 BD2"

    TX = 0: TY = 0                          'translation on screen
    PI = 3.141593
    SCF = 2.25                              'screen factor

    '-------------------------------------------------------------------------

    DIM CO#(NUMNP * NDIME), NP(NUMEL, NNPE)
    DIM X(NUMNP), Y(NUMNP), Z(NUMNP)
    DIM X1(NUMNP), Y1(NUMNP), Z1(NUMNP)

    DEF SEG = VARSEG(CO#(1))
    BLOAD D$ + F$ + ".COO", VARPTR(CO#(1))

    OPEN "I", 1, D$ + F$ + ".ELE"
    FOR I = 1 TO NUMEL
    FOR J = 1 TO NNPE
    INPUT #1, NP(I, J)
    NEXT J, I
    CLOSE 1

    IF NDIME = 3 THEN DATA$ = "3D8.DAT": GOTO 10
    IF NNPE = 4 THEN DATA$ = "2D4.DAT": GOTO 10
```

```
    IF NNPE = 8 THEN DATA$ = "2D8.DAT": GOTO 10
    CLS : PRINT "ELEMENT TYPE NOT ACCEPT": END
10  OPEN "I", 1, DATA$
    INPUT #1, NVEC
    FOR I = 1 TO NVEC: INPUT #1, NO(I): NEXT I
    FOR I = 1 TO NVEC: INPUT #1, NFLAG(I): NEXT I
    CLOSE 1

    '=====================================================================

    CLS : SCREEN 2

    SPEED = .5
    MO = 1: KP = 1
    NSCREEN = 1
    ZFACT = 1                                   'zoom factor
    NODEN = 1                                   'plot node number
    IF NDIME = 2 THEN THETA = -90
    CALL MENU
    CALL STATUS
    GOSUB 1210                                  'readjust & scaling
    GOSUB 2310                                  'check boundary of graphic
    GOSUB 1300                                  'rotate
    GOSUB 1000                                  'PLOT CUBE

    GOSUB 500
20  K$ = INKEY$: IF K$ = "" THEN 20
    IF ASC(K$) = 0 THEN 30
    IF K$ = "V" OR K$ = "v" THEN KP = 1: GOSUB 500: GOTO 20
    IF K$ = "Z" OR K$ = "z" THEN KP = 2: GOSUB 500: GOTO 20
    IF K$ = "T" OR K$ = "t" THEN KP = 3: GOSUB 500: GOTO 20
    IF K$ = "N" OR K$ = "n" THEN KP = 4: GOSUB 500: GOTO 20
    IF K$ = "D" OR K$ = "d" THEN KP = 5: GOSUB 500: GOTO 20
    IF K$ = "P" OR K$ = "p" THEN KP = 6: GOSUB 500: GOTO 20
    IF K$ = "Q" OR K$ = "q" THEN KP = 7: GOSUB 500: GOTO 20

    IF K$ = CHR$(13) THEN 40
    BEEP: GOTO 20
30  KP = ASC(MID$(K$, 2))
    IF KP = 72 THEN KP = MO - 1: GOSUB 500: GOTO 20          'up
    IF KP = 75 THEN KP = MO - 1: GOSUB 500: GOTO 20          'left
    IF KP = 77 THEN KP = MO + 1: GOSUB 500: GOTO 20          'right
    IF KP = 80 THEN KP = MO + 1: GOSUB 500: GOTO 20          'down
    BEEP: GOTO 20

40  ON MO GOSUB 220, 290, 370, 460, 470, 480, 490
    GOTO 20
    '-------------------EDIT VPOINT MODE ---------------------------------
220 LOCATE 15, 70: PRINT ">": LOCATE 16, 70: PRINT ">"
230 CALL MYKEY(KP): ON KP GOSUB 240, 250, 260, 270, 280
    IF KP <> 6 THEN 230
    LOCATE 15, 70: PRINT " ": LOCATE 16, 70: PRINT " "
    GOTO 20

240 PHI = PHI - SPEED: CALL CHECK
    LOCATE 15, 76: PRINT USING "###.#"; PHI: RETURN
```

```
250  THETA = THETA - SPEED: CALL CHECK
     LOCATE 16, 76: PRINT USING "###.#"; THETA: RETURN
260  THETA = THETA + SPEED: CALL CHECK
     LOCATE 16, 76: PRINT USING "###.#"; THETA: RETURN
270  PHI = PHI + SPEED: CALL CHECK
     LOCATE 15, 76: PRINT USING "###.#"; PHI: RETURN
280  THETA = 0: PHI = 0
     LOCATE 15, 76: PRINT USING "###.#"; PHI
     LOCATE 16, 76: PRINT USING "###.#"; THETA: RETURN
     '------------------- ZOOM MODE ---------------------------------------
290  LOCATE 20, 70: PRINT ">"
300  CALL MYKEY(KP): ON KP GOSUB 310, 320, 330, 340, 350
     IF KP <> 6 THEN 300
     LOCATE 20, 70: PRINT " "
     GOTO 20

310  ZFACT = ZFACT + SPEED / 100: GOSUB 360: RETURN
320  ZFACT = ZFACT - SPEED / 100: GOSUB 360: RETURN
330  ZFACT = ZFACT + SPEED / 100: GOSUB 360: RETURN
340  ZFACT = ZFACT - SPEED / 100: GOSUB 360: RETURN
350  ZFACT = 1: GOSUB 360: RETURN
360   LOCATE 20, 76: PRINT USING "###.#"; ZFACT: RETURN
     '------------------ EDIT TRANSLATION MODE  ---------------------------
370   LOCATE 18, 70: PRINT ">": LOCATE 19, 70: PRINT ">"
380  CALL MYKEY(KP): ON KP GOSUB 390, 400, 410, 420, 430
     IF KP <> 6 THEN 380
     LOCATE 18, 70: PRINT " ": LOCATE 19, 70: PRINT " "
     GOTO 20
     RETURN

390  TY = TY + SPEED: GOSUB 450: RETURN
400  TX = TX + SPEED: GOSUB 440: RETURN
410  TX = TX - SPEED: GOSUB 440: RETURN
420  TY = TY - SPEED: GOSUB 450: RETURN
430  TX = 0: TY = 0: GOSUB 440: GOSUB 450: RETURN
440  LOCATE 18, 76: PRINT USING "###.#"; TX: RETURN
450  LOCATE 19, 76: PRINT USING "###.#"; TY: RETURN
     '---------------------- NODE NUMBER (ON/OFF)----------------------------
460  IF NODEN = 0 THEN NODEN = 1: LOCATE 22, 70: PRINT "   (ON) ": RETURN
     NODEN = 0: LOCATE 22, 70: PRINT "   (OFF)": RETURN
     '----------------------- DRAW ON SCREEN --------------------------------
470  GOSUB 1300: GOSUB 1000: RETURN
     '----------------------- PRPLOT ----------------------------------------
480  GOTO 5000: RETURN
     '------------------------ QUIT -----------------------------------------
490  RUN "HEAD"

500  IF KP < 1 THEN KP = 7
     IF KP > 7 THEN KP = 1
     LOCATE MO, 70: PRINT "  "
     LOCATE KP, 70: PRINT "->"
     MO = KP
     RETURN

     '---------- Recalculate of nodal coord. for SCREEN COORDINATES ---------
1000 FOR I = 1 TO NUMNP
```

```
    XV = ZFACT * S * (X(I) - XMIN) + TX
    YV = ZFACT * S * (YMIN - Y(I)) + 199 + TY
    XS = SCF * XV + XVMIN + LXOFF
    YS = YV - YVMIN - LYOFF
    X1(I) = XS: Y1(I) = YS
    NEXT I
    '------------ Plot of total elements ----------------------------------
    CLS 1
    FOR I = 1 TO NUMEL
    FOR J = 1 TO NVEC
    NODE = NP(I, NO(J))
    XS = X1(NODE): YS = Y1(NODE)
    IF NFLAG(J) = 1 THEN LINE -(XS, YS) ELSE PSET (XS, YS)
    NEXT J, I
    IF NODEN = 1 THEN GOSUB 1010
    RETURN

    '------------------PLOT NODE NUMBER ----------------------------------
1010 FOR I = 1 TO NUMNP
    PSET (X1(I), Y1(I))
    DRAW "BU2 BR4"
    NUMBER$ = STR$(I)
    FOR J = 2 TO LEN(NUMBER$)
    K = VAL(MID$(NUMBER$, J, 1))
    DRAW NUM$(K)
    NEXT J
    NEXT I
    RETURN
    '---------------- SCALING & CENTERING ------------------------------
1100 B = 0: C = 0: LXOFF = 0: LYOFF = 0
    SX = ((XVMAX - XVMIN) / SCF) / (XMAX - XMIN): SY = (YVMAX - YVMIN) / (YMAX - YMIN)
    S = SY
    IF SX < S THEN S = SX: LYOFF = .5 * (YVMAX - YVMIN - (YMAX - YMIN) * S)
    IF S = SY THEN LXOFF = .5 * (XVMAX - XVMIN - (XMAX - XMIN) * S * SCF)
    RETURN
    '----------- MAX VALUE OF ORIGINAL DATA & CENTERING -------------------
1210 XMIN = CO#(1): XMAX = XMIN
    YMIN = CO#(2): YMAX = YMIN
    IF NDIME = 3 THEN ZMIN = CO#(3): ZMAX = ZMIN
    FOR I = 1 TO NUMNP
    X = CO#((I - 1) * NDIME + 1): Y = CO#((I - 1) * NDIME + 2)
    IF NDIME = 3 THEN Z = CO#((I - 1) * NDIME + 3)
    IF X < XMIN THEN XMIN = X ELSE IF X > XMAX THEN XMAX = X
    IF Y < YMIN THEN YMIN = Y ELSE IF Y > YMAX THEN YMAX = Y
    IF Z < ZMIN THEN ZMIN = Z ELSE IF Z > ZMAX THEN ZMAX = Z
    NEXT I
    XRANGE = XMAX - XMIN: YRANGE = YMAX - YMIN: ZRANGE = ZMAX - ZMIN
    FOR I = 1 TO NUMNP
    CO#((I - 1) * NDIME + 1) = CO#((I - 1) * NDIME + 1) - .5 * XRANGE
    CO#((I - 1) * NDIME + 2) = CO#((I - 1) * NDIME + 2) - .5 * YRANGE
    IF NDIME = 3 THEN CO#((I - 1) * NDIME + 3) = CO#((I - 1) * NDIME + 3) - .5 * ZRANGE
    NEXT I
    XMAX = .5 * XRANGE: YMAX = .5 * YRANGE: ZMAX = .5 * ZRANGE
    XMAX = SQR(XMAX * XMAX + YMAX * YMAX + ZMAX * ZMAX): XMIN = -XMAX
    YMAX = XMAX: YMIN = XMIN
    GOSUB 1100
```

```
    RETURN
    '---------------- VPOINT CALCULATION ---------------------------------
1300 IF THETA > 360 THEN THETA = THETA - 360
    IF THETA < -180 THEN THETA = 360 + THETA
    IF PHI > 360 THEN PHI = PHI - 360
    IF PHI < -180 THEN PHI = 360 + PHI
    SN1 = SIN(THETA * PI / 180): CN1 = COS(THETA * PI / 180)
    SN2 = SIN(PHI * PI / 180): CN2 = COS(PHI * PI / 180)
    FOR I = 1 TO NUMNP
    X = CO#((I - 1) * NDIME + 1): Y = CO#((I - 1) * NDIME + 2)
    IF NDIME = 3 THEN Z = CO#((I - 1) * NDIME + 3)
    X(I) = -X * SN1 + Y * CN1
    Y(I) = -X * CN1 * CN2 - Y * SN1 * CN2 + Z * SN2
    NEXT I
    RETURN

2300 IF NSCREEN = 0 THEN NSCREEN = 1 ELSE NSCREEN = 0
2310 IF NSCREEN = 0 THEN 2320                    'if full screen
    XVMIN = 0: XVMAX = 549                       'effective window
    YVMIN = 0: YVMAX = 194                       'relative to world coord
    LINE (XVMAX + 1, 199 - (YVMAX + 5) - 1)-(XVMAX + 1, 199 - YVMIN + 1)'separate screen
    GOTO 2330
2320 XVMIN = 0: XVMAX = 639                      'effective window
    YVMIN = 0: YVMAX = 194                       'relative to world coord
2330 VIEW SCREEN (XVMIN, 199 - (YVMAX + 5))-(XVMAX, 199 - YVMIN) 'effect. screen
    GOSUB 1100
    RETURN

    '--------------------- HARDCOPY GRAPHIC ROUTINE -----------------------
5000 XVMIN = 0: XVMAX = 639                      'effective dot
    YVMIN = 0: YVMAX = 639                       'relative to world coord
    SCF = 1.1063                                 'scale factor for printer x&y
    GOSUB 1100                                   'scale
    GOSUB 1300                                   'vpoint calculation
    CALL P(27): CALL P(51): CALL P(24)           'set line spacing

    FOR I = 1 TO NUMNP
    XV = S * (X(I) - XMIN)
    YV = S * (YMIN - Y(I)) + 639
    XS = SCF * XV + XVMIN + LXOFF
    YS = YV - YVMIN - LYOFF
    X1(I) = XS: Y1(I) = YS
    NEXT I

    MAXP = 25600
    DIM NPRINT(MAXP), NH(2)                      'print mtx & help vec.
    NPRINT(0) = 128
    DEF SEG = VARSEG(NPRINT(0))
    XADD = VARPTR(NPRINT(0))
    '------------ Plot of total elements ---------------------------------
    FOR I = 1 TO NUMEL
    FOR J = 1 TO NVEC
    IVEC = IVEC + 1: LOCATE 1, 1: PRINT IVEC; "VEC"
    NODE = NP(I, NO(J))
    NXS1 = NXS2: NYS1 = NYS2
    NXS2 = X1(NODE): NYS2 = Y1(NODE)
```

```
    IF NFLAG(J) = 0 THEN 5020
    NXLEN = NXS2 - NXS1
    NYLEN = NYS2 - NYS1
    IF NXLEN = 0 THEN 5010

    MSTEP = SGN(NXLEN)          'compute for any dot
    NXLEN = ABS(NXLEN)
    SLOPE = NYLEN / NXLEN
    FOR L = 0 TO (NXLEN - 1)
    NY = NYS1 + SLOPE * L
    NX = NXS1 + L * MSTEP
    GOSUB 5100          'PSET (NX,NY)
    NEXT L
    GOTO 5020

5010 MSTEP = SGN(NYLEN)         'compute for vertical dot
    NYLEN = ABS(NYLEN)
    FOR L = 0 TO (NYLEN - 1)
    NY = NYS1 + MSTEP * L
    NX = NXS1
    GOSUB 5100          'PSET (NX,NY)
    NEXT L

5020 NEXT J
    NEXT I

    IF NODEN = 1 THEN GOSUB 5300

    FOR X = 1 TO 80
    CALL P(27)
    CALL P(42)                  '"*"
    CALL P(4)                   'set bit image dot density = 640
    CALL P(128)                 'set 640 dot printed
    CALL P(2)                   '
    FOR J = 1 TO 640
    XOFF = (X - 1) * 640 + J - 1
    N = PEEK(XADD + XOFF): CALL P(N)
    NEXT J
    LPRINT CHR$(10);
    NEXT X

    RUN "HEAD"
    '---------------- PUT POINT IN ADDRESS XADD+XOFF----------------------
    '          start nrow = 0:ncol = 0:XOFF = 0
5100 NROW = NY \ 8: XROW = NROW
    XOFF = (XROW - 1) * 640 + NX
    N = PEEK(XADD + XOFF)
    CALL DEC2BIN(N, B$)
    NYY = NY - NROW * 8 + 1
    IF MID$(B$, NYY, 1) = "1" THEN 5200
    N = N + 2 ^ (8 - NYY)
    POKE (XADD + XOFF), N
5200 RETURN
    '---------------- MARK DOT OF NODE NUMBER ----------------------------
5300 DIM NUM(9, 5)
    FOR I = 0 TO 9: FOR J = 1 TO 5
```

```
    READ NUM(I, J)
    NEXT J, I
    DATA 0,62,34,62,0
    DATA 0,34,62,2,0
    DATA 0,46,42,58,0
    DATA 0,42,42,62,0
    DATA 0,56,8,62,0
    DATA 0,58,42,46,0
    DATA 0,62,42,14,0
    DATA 0,32,32,62,0
    DATA 0,62,42,62,0
    DATA 0,58,42,62,0
    FOR I = 1 TO NUMNP
    LOCATE 2, 1: PRINT I; "NODE"
    NX = X1(I): NY = Y1(I)
    NROW = NY \ 8: XROW = NROW
    XOFF = (XROW - 1) * 640 + NX + 4
    NUMBER$ = STR$(I)
    FOR J = 2 TO LEN(NUMBER$)
    K = VAL(MID$(NUMBER$, J, 1))
    FOR L = 1 TO 5
    XOFF = XOFF + 1
    NPVAL = NUM(K, L)
    N = PEEK(XADD + XOFF)
    CALL DEC2BIN(N, B$)
    IF LEFT$(B$, 1) = "1" THEN NPVAL = NPVAL + 128
    POKE (XADD + XOFF), NPVAL
    NEXT L, J, I
    RETURN

DEFINT I-N
SUB CHECK STATIC
SHARED THETA, PHI
    IF THETA > 360 THEN THETA = THETA - 360
    IF THETA < -180 THEN THETA = 360 + THETA
    IF PHI > 360 THEN PHI = PHI - 360
    IF PHI < -180 THEN PHI = 360 + PHI
END SUB

DEFINT I-N
SUB DEC2BIN (N, B$) STATIC
    DEFINT I-N
    B$ = "": J = N
    FOR I = 1 TO 8
    M = 2 ^ (8 - I): K = J - M
    IF K >= 0 THEN B$ = B$ + "1": J = K: GOTO 9999
    B$ = B$ + "0"
9999 NEXT I
END SUB

DEFINT I-N
SUB MENU STATIC
    LOCATE 1, 70: PRINT "  VPOINT"
    LOCATE 2, 70: PRINT "  ZOOM"
    LOCATE 3, 70: PRINT "  TRANSL."
    LOCATE 4, 70: PRINT "  NODE NO."
```

```
    LOCATE 5, 70: PRINT "  DRAW"
    LOCATE 6, 70: PRINT "  PLOT"
    LOCATE 7, 70: PRINT "  QUIT"
END SUB

DEFINT I-N
SUB MYKEY (KP) STATIC
SHARED SPEED
8000 K$ = INKEY$: IF K$ = "" THEN 8000
     IF ASC(K$) > 0 THEN 8010
     KP = ASC(MID$(K$, 2))
     IF KP = 73 THEN SPEED = 1: GOTO 8000                    'page up
     IF KP = 81 THEN SPEED = .5: GOTO 8000                   'page down
     IF KP = 72 THEN KP = 1: EXIT SUB                            'up
     IF KP = 75 THEN KP = 2: EXIT SUB                            'left
     IF KP = 77 THEN KP = 3: EXIT SUB                            'right
     IF KP = 80 THEN KP = 4: EXIT SUB                            'down
     IF KP = 71 THEN KP = 5: EXIT SUB                            'Home
     GOTO 8000
8010 IF K$ <> CHR$(13) THEN BEEP: GOTO 8000
     KP = 6
END SUB

DEFINT I-N
SUB STATUS STATIC
SHARED PHI, THETA, TX, TY, NODEN, ZFACT
     LOCATE 14, 70: PRINT "  STATUS"
     LOCATE 15, 70: PRINT "  V = "; USING "###.#"; PHI
     LOCATE 16, 70: PRINT "  H = "; USING "###.#"; THETA
     LOCATE 18, 70: PRINT "  TX= "; USING "###.#"; TX
     LOCATE 19, 70: PRINT "  TY= "; USING "###.#"; TY
     LOCATE 20, 70: PRINT "  Z = "; USING "###.#"; ZFACT
     LOCATE 21, 70: PRINT "  NODE NO."
     IF NODEN = 1 THEN LOCATE 22, 70: PRINT "    (ON)"
     IF NODEN = 0 THEN LOCATE 22, 70: PRINT "    (OFF)"
END SUB
```

```
'*************************************************************************
'*                      file name "SOLVER"                               *
'*                                                                       *
'*   OBJECTS:-assemble element stiffness matrix                          *
'*           -reduce element stiffness matrix in segment                 *
'*           -assemble element load vector & reduce                      *
'*           -backsub for solutions (nodal displacements)                *
'*                                                                       *
'*************************************************************************
DEFINT I-N
DEFDBL A-H, O-Z

COMMON D$, F$, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
COMMON NCASE, NUMSEG, NGAUSS, KORDER, T$, E$, MAXVOL, F$()
COMMON IOPTION
COMMON LSTMAX, NEQMAX, KMAX
'      Maximum nodes repersented in segment
'      Maximum equations repersented in segment
'      Maximum usage storage
COMMON WAVECOUNT, STIFFCOUNT, FORCECOUNT, SKCOUNT
COMMON NPDOF(), IORDER()

DIM IFILE(5)
DD$ = D$

'                Dimension array
DIM IDIAG1(NEQMAX), IDIAG2(NEQMAX), MOVE(NEQMAX), IEQS(NEQMAX)
DIM NPR(NUMNP)
DIM IELE(NUMEL), NPT(NUMEL, NNPE)
DIM LNEQ(NEQMAX)

DIM SK(MAXVOL), ST(24, 24), SL(24), IIDOF(8)
SKCOUNT = 0

STARTtime! = TIMER
GOTO 120

100  LOCATE 1, 65: COLOR 14: PRINT "DATE :"; : COLOR 15: PRINT DATE$: RETURN
110  LOCATE 2, 65: COLOR 14: PRINT "TIME :"; : COLOR 15: PRINT TIME$: RETURN

'*************************************************************************
'*              MAIN PROGRAM "SOLVER"                                    *
'*               file name "SOLVER"                                      *
'*************************************************************************

120  CLS : GOSUB 100: GOSUB 110
     COLOR 31, 13
     LOCATE 1, 5: PRINT "                RUNNING                ": COLOR 15, 13
     LOCATE 2, 5: PRINT "                                       "
     LOCATE 3, 5: PRINT "ASSEMBLY & SOLVING STIFFNESS EQUATIONS": COLOR 7, 0

     NUMEQT = 0
     FOR I = 1 TO NUMNP: NUMEQT = NUMEQT + NPDOF(I): NEXT I
     DIM B(NUMEQT)

     FOR I = 1 TO 5: IF LEN(F$(I + 5)) > 1 THEN IFILE(I) = 1
```

```
    NEXT I
    PRINT : COLOR 11
    PRINT TAB(5); "Number of segments ..... ="; NUMSEG
    PRINT TAB(5); "Number of load cases ..... ="; NCASE; "CASES"; " (CASE";
    FOR I = 1 TO 5: IF IFILE(I) = 1 THEN PRINT " #"; RIGHT$(STR$(I), 1);
    NEXT I: PRINT ")"
    PRINT TAB(5); "Number of equations ..... ="; NUMEQT

    '<------- part of assemble element stiffness & forward reduction ----->

    ISEG = 0: IELEM = 0: IEQ = 0

130 ISEG = ISEG + 1: ISEG$ = RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1)
    OPEN "B", 3, DD$ + "GST." + ISEG$
    GOSUB 730                       'read wave data record
    GOSUB 410                       'sliding matrix [A]

    COLOR 14
    LOCATE 9, 1: PRINT "<<< ASSEMBLY OF STIFFNESS SEGMENT #"; ISEG$; " >>>"
    COLOR 15: PRINT TAB(5); "Elements ..."
    FOR I = 1 TO IELEX
    IELEM = IELEM + 1
    LOCATE 10, 17: PRINT IELEM;

    I1DOF(1) = 1: IF NPDOF(NPT(I, 1)) = 0 THEN I1DOF(1) = 0
    FOR J = 2 TO NNPE
    ISUM = 0
    FOR K = 1 TO (J - 1)
    ISUM = ISUM + NPDOF(NPT(I, K))
    NEXT K
    I1DOF(J) = ISUM + 1
    IF ISUM = 0 AND NPDOF(NPT(I, J)) = 0 THEN I1DOF(J) = 0
    NEXT J

    NUMEQ = 0
    FOR J = 1 TO NNPE
    NUMEQ = NUMEQ + NPDOF(NPT(I, J))
    NEXT J

    FOR J = 1 TO NUMEQ
    FOR K = J TO NUMEQ
    GET #3, , ST(J, K)
    ST(K, J) = ST(J, K)
    NEXT K, J

    '         ASSEMBLE ELEMENT STIFFNESS

    FOR J = 1 TO NNPE
    NPJ = NPR(NPT(I, J))
    FOR K = 1 TO NNPE
    NPK = NPR(NPT(I, K))
    IF NPJ > NPK THEN 150
    JD = NPDOF(NPT(I, J))
    KD = NPDOF(NPT(I, K))
    FOR I1 = 1 TO JD
    IROW = NPJ + I1 - 1
```

```
    FOR I2 = 1 TO KD
    JCOL = NPK + I2 - 1
    IF IROW > JCOL THEN 140
    ILOC = IDIAG2(JCOL) - (JCOL - IROW)
    JI = I1DOF(J) + I1 - 1: KI = I1DOF(K) + I2 - 1
    SK(ILOC) = SK(ILOC) + ST(JI, KI)
140 NEXT I2, I1
150 NEXT K
    NEXT J

    NEXT I

    CLOSE #3
    KILL DD$ + "GST." + ISEG$

    DEF FNILOC (I1, J1) = IDIAG2(J1) - (J1 - I1)

    '       FORWARD REDUCTION (REDUCE TO UPPER TRIANGULAR)

    IEND = NEQ + 1
    FOR J = NEQ TO 2 STEP -1
    IHIGHT = IDIAG2(J) - IDIAG2(J - 1)
    IBGN = J - IHIGHT + 1
    IF IBGN >= IEND THEN 160
    FOR I = IBGN TO IEND - 1
    LNEQ(I) = J
    NEXT I
    IEND = IBGN
160 NEXT J


    LOCATE 11, 1: COLOR 14
    PRINT "<<< FORWARD REDUCTION OF STIFFNESS SEGMENT #"; ISEG$; " >>>"
    COLOR 15: PRINT TAB(5); "Equations .."
    NN = ICOMP
    FOR J = 1 TO NN
    IEQ = IEQ + 1: LOCATE 12, 17: PRINT IEQ;
    JJ = J + 1
    FOR I = JJ TO LNEQ(J)              'NEQ
    IH = IDIAG2(I) - IDIAG2(I - 1)
    IF IH < (I - J + 1) THEN 180
    LJI = FNILOC(J, I)
    LJJ = FNILOC(J, J)
    C = -SK(LJI) / SK(LJJ)
    FOR K = I TO LNEQ(I)               'NEQ
    KH = IDIAG2(K) - IDIAG2(K - 1)
    IF KH < (K - J + 1) THEN 170
    LIK = FNILOC(I, K)
    LJK = FNILOC(J, K)
    SK(LIK) = SK(LIK) + SK(LJK) * C
170 NEXT K
180 NEXT I
    NEXT J

    IF NUMSEG = 1 THEN 190
```

```
    GOSUB 740                       'save ass & red stiffness

    IF ISEG < NUMSEG THEN 130

    '----------------------- PART OF LOAD ASSEMBLING ---------------------
190 FOR ICASE = 1 TO 5
    IF IFILE(ICASE) = 0 THEN 280
    CASE$ = RIGHT$(STR$(ICASE), 1)
    COLOR 14: LOCATE 13, 1: PRINT "<<< ASSEMBLY OF LOAD CASE #"; CASE$; " >>>"
    COLOR 15: PRINT TAB(5); "Elements ..."

    FOR I = 1 TO NUMEQT: B(I) = 0: NEXT I

    OPEN "I", 3, DD$ + "LOAD." + CASE$
    ISEG = 0: IELEM = 0: IEQ = 0

    '---------------------------------------------------------------------
200 ISEG = ISEG + 1: ISEG$ = RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1)
    IF NUMSEG = 1 THEN 210
    GOSUB 730                                   'read wave data

    FOR I = 1 TO NEQ                            'for JUSTFY parameter
    IDIAG2(I) = IDIAG2(I) + JUSTFY
    NEXT I

    GOSUB 760                                   'load ass & red stiff

    IEND = NEQ + 1                              'last equation
    FOR J = NEQ TO 2 STEP -1
    IHIGHT = IDIAG2(J) - IDIAG2(J - 1)
    IBGN = J - IHIGHT + 1
    IF IBGN >= IEND THEN 205
    FOR I = IBGN TO IEND - 1
    LNEQ(I) = J
    NEXT I
    IEND = IBGN
205 NEXT J


210 FOR I = 1 TO IELEX
    IELEM = IELEM + 1
    LOCATE 14, 17: PRINT IELEM;

    I1DOF(1) = 1: IF NPDOF(NPT(I, 1)) = 0 THEN I1DOF(1) = 0
    FOR J = 2 TO NNPE
    ISUM = 0
    FOR K = 1 TO (J - 1)
    ISUM = ISUM + NPDOF(NPT(I, K))
    NEXT K
    I1DOF(J) = ISUM + 1
    IF ISUM = 0 AND NPDOF(NPT(I, J)) = 0 THEN I1DOF(J) = 0
220 NEXT J

    NUMEQ = 0
    FOR J = 1 TO NNPE
    NUMEQ = NUMEQ + NPDOF(NPT(I, J))
```

```
    NEXT J

    FOR J = 1 TO NUMEQ: INPUT #3, SL(J): NEXT J

    '                       ASSEMBLE ELEMENT LOAD

    FOR J = 1 TO NNPE
    NPJ = NPR(NPT(I, J))
    JD = NPDOF(NPT(I, J))
    FOR K = 1 TO JD
    JLOC = NPJ + K - 1
    ILOC = I1DOF(J) + K - 1
    B(IEQS(JLOC)) = B(IEQS(JLOC)) + SL(ILOC)
    NEXT K, J

    NEXT I

    '                       REDUCE RIGHT HAND SIDE

    COLOR 14
    LOCATE 15, 1: PRINT "<<< FORWARD REDUCTION OF LOAD CASE #"; CASE$; " >>>"
    COLOR 15: PRINT TAB(5); "Equations .."
    NN = ICOMP
    FOR J = 1 TO NN
    IEQ = IEQ + 1: LOCATE 16, 17: PRINT IEQ;
    JJ = J + 1
    FOR I = JJ TO LNEQ(J)                    'NEQ
    IH = IDIAG2(I) - IDIAG2(I - 1)
    IF IH < (I - J + 1) THEN 230
    LJI = FNILOC(J, I)
    LJJ = FNILOC(J, J)
    C = -SK(LJI) / SK(LJJ)
    B(IEQS(I)) = B(IEQS(I)) + B(IEQS(J)) * C
230 NEXT I
    NEXT J

    IF NUMSEG = 1 THEN 240
    IF ISEG = NUMSEG THEN 240
    GOTO 200

240 CLOSE 3
    KILL DD$ + "LOAD." + CASE$

    '------------------ BACK-SUBSTITUTION PART ------------------------

    COLOR 14: LOCATE 17, 1: PRINT "<<< SOLUTIONS OF LOAD CASE #"; CASE$; " >>>"
    COLOR 15: PRINT TAB(5); "Equations .."
    IEQ = IEQ + 1

    IF ISEG = NUMSEG THEN 260

250 GOSUB 730                                'read wave data

    FOR I = 1 TO NEQ                         'for JUSTFY parameter
    IDIAG2(I) = IDIAG2(I) + JUSTFY
    NEXT I
```

```
    GOSUB 760                                   'load ass & red stiffness

    IEND = NEQ + 1                              'last equation
    FOR J = NEQ TO 2 STEP -1
    IHIGHT = IDIAG2(J) - IDIAG2(J - 1)
    IBGN = J - IHIGHT + 1
    IF IBGN >= IEND THEN 255
    FOR I = IBGN TO IEND - 1
    LNEQ(I) = J
    NEXT I
    IEND = IBGN
255 NEXT J


    '               BACK-SUBSTITUTION

260 IBGN = NEQ - ICOMP + 1
    FOR II = IBGN TO NEQ
    IEQ = IEQ - 1: LOCATE 18, 17: PRINT IEQ;
    I = NEQ - II + 1
    I1 = I + 1
    FOR J = I1 TO LNEQ(I)                       'NEQ
    JH = IDIAG2(J) - IDIAG2(J - 1)
    IF JH < J - I + 1 THEN 270
    LIJ = FNILOC(I, J)
    B(IEQS(I)) = B(IEQS(I)) - SK(LIJ) * B(IEQS(J))
270 NEXT J
    LII = FNILOC(I, I)
    B(IEQS(I)) = B(IEQS(I)) / SK(LII)
    NEXT II

    ISEG = ISEG - 1: ISEG$ = RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1)
    IF ISEG > 0 THEN 250

    '    SAVE SOLUTION

    OPEN "R", 1, D$ + F$ + ".D" + CASE$, 8
    FIELD #1, 8 AS A$
    FOR I = 1 TO NUMEQT
    S$ = MKD$(B(I))
    LSET A$ = S$
    PUT #1, I
    NEXT I
    CLOSE 1

280 NEXT ICASE

    FINISHtime! = TIMER
    TOTALtime! = FINISHtime! - STARTtime!
    F$(16) = STR$(TOTALtime!)
    OPEN "O", 1, D$ + F$ + ".DIR": FOR I = 1 TO 16: WRITE #1, F$(I): NEXT I: CLOSE 1

    OPEN "O", 1, D$ + F$ + ".CON"
    WRITE #1, NUMNP, NUMEL, NNPE, NPDOF, NDIME, NMATS, NTYPE
```

```
    WRITE #1, NCASE
    WRITE #1, NUMSEG, NGAUSS, IORDER
    WRITE #1, T$
    WRITE #1, E$
    CLOSE 1

    FOR ISEG = 1 TO NUMSEG
    ISEG$ = RIGHT$(STR$(ISEG), LEN(STR$(ISEG)) - 1)
    KILL DD$ + "WVE." + ISEG$: IF NUMSEG = 1 THEN 290
    KILL DD$ + "AST." + ISEG$
290 NEXT ISEG

    ERASE NPDOF, IORDER
    PLAY "L10 BGBGBGBGBGBGBGBGBG"
    CHAIN "HEAD"


    '========================== SUBROUTINE AREA ============================

    '------------ Subroutine slide matrix [SK] -------------------------

410 ID11 = IDIAG1(1)
    SK(ID11) = 0
    FOR I = 2 TO MOVEX
    I2 = MOVE(I)
    JBGN = IDIAG1(I - 1) + 1
    JEND = IDIAG1(I)
    IDIAG = IDIAG2(I2)
    FOR J = JBGN TO JEND
    J1 = I - JEND + J
    J2 = MOVE(J1)
    IF J2 = 0 THEN 420
    J3 = IDIAG - (I2 - J2)
    IF J3 = J THEN 430
    SK(J3) = SK(J)
420 SK(J) = 0
430 NEXT J
    NEXT I

    IF JUSTFY = 0 THEN RETURN
    IBGN = IDIAG2(1)
    IEND = IDIAG2(NEQ)
    I1 = IEND
    I2 = IEND + JUSTFY
    FOR I = IBGN TO IEND
    SK(I2) = SK(I1)
    SK(I1) = 0
    I2 = I2 - 1
    I1 = I1 - 1
    NEXT I
    FOR I = 1 TO NEQ
    IDIAG2(I) = IDIAG2(I) + JUSTFY
    NEXT I
    RETURN

    '----------------------- DISK SEGMENT INPUT -------------------------
```

```
730  OPEN "B", 2, DD$ + "WVE." + ISEG$
     GET #2, , ISEG
     GET #2, , IELEX
     GET #2, , NEQ
     GET #2, , ICOMP
     GET #2, , MOVEX
     GET #2, , JUSTFY
     FOR I = 1 TO MOVEX: GET #2, , IDIAG1(I): NEXT I
     FOR I = 1 TO NEQ: GET #2, , IDIAG2(I): NEXT I
     FOR I = 1 TO MOVEX: GET #2, , MOVE(I): NEXT I
     FOR I = 1 TO NEQ: GET #2, , IEQS(I): NEXT I
     FOR I = 1 TO NUMNP: GET #2, , NPR(I): NEXT I
     FOR I = 1 TO IELEX: GET #2, , IELE(I): NEXT I
     FOR I = 1 TO IELEX
     FOR J = 1 TO NNPE
     GET #2, , NPT(I, J)
     NEXT J, I
     CLOSE 2
     RETURN

     '--------- SAVE FULLY ASSEMBLED & REDUCED STIFFNESS MTX ----------------
740  OPEN "B", 2, DD$ + "AST." + ISEG$
     SKCOUNT = SKCOUNT + 1
     PUT #2, , SK(IDIAG2(1))
     FOR J = 2 TO NEQ
     JH = IDIAG2(J) - IDIAG2(J - 1)
     IBGN = J - JH + 1
     IF IBGN > ICOMP THEN 750
     IEND = ICOMP: IF J < ICOMP THEN IEND = J
     FOR I = IBGN TO IEND
     LIJ = IDIAG2(J) - (J - I)
     SKCOUNT = SKCOUNT + 1
LOCATE 12, 30: PRINT SKCOUNT;
     PUT #2, , SK(LIJ)
     NEXT I
750  NEXT J
     CLOSE #2
     RETURN

     '--------- LOAD FULLY ASSEMBLED & REDUCED STIFFNESS MTX ----------------
760  OPEN "B", 2, DD$ + "AST." + ISEG$
     GET #2, , SK(IDIAG2(1))
     FOR J = 2 TO NEQ
     JH = IDIAG2(J) - IDIAG2(J - 1)
     IBGN = J - JH + 1
     IF IBGN > ICOMP THEN 770
     IEND = ICOMP: IF J < ICOMP THEN IEND = J
     FOR I = IBGN TO IEND
     LIJ = IDIAG2(J) - (J - I)
     GET #2, , SK(LIJ)
     NEXT I
770  NEXT J
     CLOSE #2
     RETURN
```

```
; FILE NAME "P.ASM"
; assembly program for line printer works as LPRINT CHR$(INTEGER%)
; CALL P(INTEGER%)
; INTEGER% = any integer value <= 255 because of low ax is al
cseg   segment para public 'code'
public P
P proc far
    assume cs:cseg,ds:nothing,ss:nothing,es:nothing
    push bp
    mov bp,sp
    mov si,[bp+6]
    mov ax,[si]      ;load called value in al (low value of ax <= 255

    call print      ;hardcopy of al value

    pop bp           ; return to caller
    ret 2
P endp
;
print   proc near
        push    dx
        mov     an,00
        mov     dx,00
        int     17h
        pop     dx
        ret
print   endp
cseg ends
    end


A>
```

2D4.DAT

```
5   1   2   3   4   1   0   1   1   1   1
```

2D8.DAT

```
9   1   2   3   4   5   6   7   8   1
0   1   1   1   1   1   1   1   1
```

3D8.DAT

```
16  1   2   3   4   1   5   6   7   8
5   4   8   3   7   2   6   0   1   1
1   1   1   1   1   1   1   0   1   0
1   0   1
```

ภาคผนวก ข

วิธีการป้อนข้อมูล

ข.1 ความนำ

โปรแกรมที่เขียนขึ้นในงานวิจัยนี้ เขียนขึ้นเพื่อใช้กับเครื่องไมโครคอมพิวเตอร์ขนาด 16 บิท เช่น IBM PC หรือคล้ายกัน เป็นต้น โดยเขียนโปรแกรมเป็นภาษาไมโครซอฟท์ควิกเบสิค (Microsoft QuickBASIC 4.0) แปลเป็นภาษาเครื่องเพื่อความสะดวกและรวดเร็วยิ่งขึ้นในการทำงาน และได้ปรับปรุงโปรแกรมในการป้อนข้อมูลและแสดงผลเพื่อให้ผู้ที่สนใจสามารถศึกษาและใช้โปรแกรมนี้ได้สะดวก

อนึ่งโปรแกรมนี้ ผู้ใช้สามารถกำหนดการใช้เนื้อที่หน่วยความจำหลักในส่วนที่เก็บค่าสัมประสิทธิ์สติฟเนสเมตริกซ์ได้ (มากที่สุดเท่ากับ 32767 หน่วย) ทำให้มีความยืดหยุ่นในการใช้งานพอสมควร เช่น ในกรณีที่ใช้เครื่องไมโครคอมพิวเตอร์ที่มีหน่วยความจำหลักน้อย หรือมีเนื้อที่หน่วยความจำหลักเหลือน้อยเนื่องจากใช้หน่วยความจำหลักร่วมกับโปรแกรมอื่น เป็นต้น สำหรับหน่วยความจำสำรองนั้น โปรแกรมนี้ต้องการใช้ดิสค์ไดร์ฟสองตัวในกรณีที่ใช้แผ่นจานแม่เหล็กชนิดอ่อนขนาด 5.25 นิ้ว การเลือกการทำงานที่ต้องการโดยกดแป้นตัวอักษรภาษาอังกฤษตัวแรกซึ่งได้ขยายความหมายไว้ในบรรทัดเดียวกันเมื่อโปรแกรมมีคำสั่งให้เลือกการทำงาน ดังนี้

```
===> SELECT ?
```

และภายหลังจากการป้อนข้อมูลทุกครั้งจะมีคำถามว่ายอมรับหรือไม่ ดังนี้

```
<Return> = Accept      R = Reenter
```

กด <Return> หมายถึงยอมรับข้อมูลที่ได้ป้อนไว้หลังสุด

กด R หมายถึงไม่ยอมรับข้อมูลที่ได้ป้อนไว้หลังสุด

เมื่อเริ่มการใช้งาน โปรแกรมย่อยที่จะต้องเรียกคือ โปรแกรมย่อย HELLO จากนั้นในโปรแกรมจะทำการเรียกใช้โปรแกรมย่อยอื่น ๆ ได้เองอย่างต่อเนื่องตามความต้องการของการทำงาน เมื่อเรียกโปรแกรม HELLO จะปรากฏ USER MENU เริ่มแรกคือ

<<< USER MENU >>> S = START SUPATFEAP

H = HARDWARE CONFIGURATION

I = INFORMATION

E = EXIT

====> SELECT ?

S หมายถึง เริ่มเข้าสู่ส่วนควบคุมการทำงานหลักของโปรแกรม

H หมายถึง กำหนดชื่อดิสค์ไดรฟ์ที่จะใช้เป็นที่สำหรับเก็บข้อมูล

I หมายถึง เป็นข้อมูลโดยย่อของโปรแกรมนี้

E หมายถึง เลิกการทำงาน

ภายหลังที่กดแป้น S แล้ว (ต้องไม่ลืมที่จะตรวจดูชื่อของดิสค์ไดรฟ์สำหรับเก็บข้อมูลให้ถูกต้องเสียก่อน) จะปรากฏ ACTIVITY MENU ซึ่งควบคุมการทำงานหลักของโปรแกรมทั้งหมดเป็นหมวดใหญ่ ๆ รวม 4 หมวด เมื่อเลิกจากการทำงานในหมวดต่าง ๆ แล้ว ก็จะกลับมาเข้าสู่ ACTIVITY MENU นี้ ซึ่งมีหมวดการทำงานดังนี้คือ

ACTIVITY MENU : D = DATA MODE

S = SOLUTION MODE

R = RESULT MODE

G = GRAPHIC MODE

Q -> QUIT TO USER MENU

E -> EXIT TO SYSTEM

===> SELECT ?

D หมายถึง หมวดการสร้างหรือแก้ไขข้อมูล

S หมายถึง หมวดทำการวิเคราะห์หาค่าการเคลื่อนที่ที่ชิ้ว

R หมายถึง หมวดผลลัพธ์แสดงค่าการเคลื่อนที่ที่ชิ้ว และคำนวณหาหน่วยแรงที่เกิดขึ้นในชิ้นส่วน

G หมายถึง หมวดกราฟิกแสดงรูปร่างของโครงสร้าง

ข.2 การสร้างหรือแก้ไขข้อมูล

การเข้าสู่หมวดนี้ทำได้โดยการกดแป้น D จาก ACTIVITY MENU การทำงานของโปรแกรมภายใต้หมวดนี้จะเป็นการเตรียมข้อมูลทั้งหมดของโครงสร้างที่จะทำการวิเคราะห์ อันได้แก่ จำนวนชิ้ว หมายเลขของชิ้ว พิกัดของชิ้ว สภาพเงื่อนไขที่ชิ้ว จำนวนชิ้นส่วน สภาพการเชื่อมโยงของแต่ละชิ้นส่วนเข้าด้วยกัน คุณสมบัติของแต่ละชิ้นส่วน และน้ำหนักกระทำที่กระทำต่อโครงสร้าง ข้อมูลดังกล่าวสามารถแสดงบนจอภาพหรือออกทางเครื่องพิมพ์ ข้อมูลเหล่านี้จะถูกบันทึกไว้ในแผ่นจานแม่เหล็กเพื่อสามารถเรียกใช้ได้อีกเมื่อต้องการ ลำดับการสร้างหรือแก้ไขข้อมูลมีดังนี้

1. การกำหนดชื่อแฟ้มข้อมูล เป็นการกำหนดชื่อแฟ้มข้อมูลให้โปรแกรมทราบว่าต้องการที่จะทำการวิเคราะห์แฟ้มข้อมูลของโครงสร้างใด โดยกำหนดได้ 3 กรณีคือ

```
CURRENT PROJECT MASTER FILENAME ===> PSS1
OPTIONS :       C  = CONTINUE
                N  = NEW PROJECT
                E  = OTHER EXISTING PROJECT
                Q -> QUIT TO ACTIVITY MENU
                ====> SELECT ?
```

C หมายถึง เลือกใช้แฟ้มข้อมูลอันเดิม

N หมายถึง เลือกสร้างแฟ้มข้อมูลใหม่

E หมายถึง เลือกแฟ้มข้อมูลอื่นซึ่งได้สร้างและเก็บไว้แล้วในแผ่นจานแม่เหล็ก

ข้อมูลที่ต้องป้อนสำหรับการระบุแฟ้มข้อมูลมีดังนี้

```
=============================================================================
MASTER FILENAME : pss1
PROJECT TITLE   : PLANE STRESS Q4-100N-56E
ENGINEER        : MR.SUPAT UTHATIWAT
AUTHORITY       : MR.SUPAT UTHAIWAT
=============================================================================
```

MASTER FILENAME หมายถึง ชื่อแฟ้มข้อมูล (F$)

PROJECT TITLE หมายถึง ชื่อหัวข้อเรื่อง (T$)

ENGINEER หมายถึง ชื่อผู้ที่ทำการวิเคราะห์ (E$)

ภายหลังจากที่กำหนดชื่อแฟ้มข้อมูลที่ต้องการแล้วจึงทำการกำหนดชนิดของชิ้นส่วน ดังนี้

1...PLANE STRESS ELEMENT (4 NODE)

2... (8 NODE)

3...PLANE STRAIN ELEMENT (4 NODE)

4... (8 NODE)

5...PLATE BENDING ELEMENT (4 NODE)

6... (8 NODE)

7...BRICK ELEMENT (8 NODE)

หมายเหตุ กรณีที่ใช้แฟ้มข้อมูลซึ่งได้สร้างและเก็บไว้แล้วนั้น จะไม่สามารถเลือกชนิดของชิ้นส่วนได้อีก

เมื่อกำหนดข้อมูลข้างต้นเรียบร้อยแล้ว ขั้นตอนต่อไปจะเป็นการสร้างหรือแก้ไขในรายละเอียดของโครงสร้างโดยจะปรากฏ DATA MENU ซึ่งสามารถเลือกที่จะสร้างหรือแก้ไขข้อมูลในกรณีย่อย 3 กรณีได้แก่ ข้อมูลเกี่ยวกับชั้ว ข้อมูลเกี่ยวกับชิ้นส่วน ข้อมูลเกี่ยวกับน้ำหนักกระทำ การเลือกดังกล่าวสามารถทำโดยกดแป้นตัวอักษรที่อยู่ข้างหน้า

<<< DATA MENU >>>

N = NODE DATA

E = ELEMENT DATA

L = LOAD DATA

Q -> QUIT TO ACTIVITY MENU

N หมายถึง ป้อนข้อมูลเกี่ยวกับชั้ว

E หมายถึง ป้อนข้อมูลเกี่ยวกับชิ้นส่วน

L หมายถึง ป้อนข้อมูลเกี่ยวกับน้ำหนักกระทำ

2. ป้อนข้อมูลรายละเอียดของโครงสร้างเกี่ยวกับชั้ว ด้วยการกดแป้น N จาก DATA MENU

<<<<< NODE DATA >>>>

N = NO.OF NODES .....[100 NODES]

C = COORDINATE DATA

B = BOUNDARY DATA

O = OUTPUT

Q -> QUIT TO DATA MENU

N หมายถึง ป้อนจำนวนของชิ้วทั้งหมด

C หมายถึง ป้อนโคออร์ดิเนทของชิ้วแต่ละชิ้ว

B หมายถึง ป้อนสภาพเงื่อนไขที่ชิ้ว

O หมายถึง แสดงข้อมูลเกี่ยวกับชิ้ว

Q หมายถึง ออกจากการสร้างหรือแก้ไขข้อมูลเกี่ยวกับชิ้ว

หมายเหตุ ถ้าระดับชั้นความเสรีในทิศทางใดเป็นอิสระ จะป้อนสภาพเงื่อนไขในทิศทางนั้นเป็น O

ถ้าระดับชั้นความเสรีในทิศทางใดถูกบังคับ จะป้อนสภาพเงื่อนไขในทิศทางนั้นเป็น 1

3. ป้อนข้อมูลรายละเอียดของโครงสร้างเกี่ยวกับชิ้นส่วน

<<<< ELEMENT DATA >>>>

N = NO.OF ELEMENTS .....[56 ELEMENTS]

S = NO.OF MATERIAL SETS ...[1 SETS]

E = ELEMENT CONNECTIVITY

M = MATERIAL PROPERTY

O = OUTPUT

Q -> QUIT TO DATA MENU

N หมายถึง ป้อนจำนวนชิ้นส่วนทั้งหมด

S หมายถึง ป้อนจำนวนชุดของคุณสมบัติของชิ้นส่วนที่แตกต่างกัน

E หมายถึง ป้อนสภาพการเชื่อมโยงของแต่ละชิ้นส่วนเข้าด้วยกัน

M หมายถึง ป้อนคุณสมบัติของชิ้นส่วนแต่ละชุด

O หมายถึง แสดงข้อมูลเกี่ยวกับชิ้นส่วน

Q หมายถึง ออกจากการสร้างหรือแก้ไขข้อมูลเกี่ยวกับชิ้นส่วน

หมายเหตุ การป้อนสภาพการเชื่อมโยงของแต่ละชิ้นส่วน ให้ป้อนหมายเลขชิ้วเรียงตามทิศทางทวนเข็มนาฬิกาดังในรูปที่ 7.1

4. ป้อนข้อมูลรายละเอียดของโครงสร้างเกี่ยวกับน้ำหนักกระทำ สำหรับน้ำหนักกระทำนั้น สามารถกำหนดชุดของน้ำหนักกระทำได้ทั้งหมด 5 ชุด(ชุดที่ 1-5) แต่ละชุดอาจประกอบด้วยน้ำหนักกระทำชนิดเดียวหรือหลายชนิดรวมกัน

4.1 การสร้าง แก้ไข หรือยกเลิกชุดของน้ำหนักกระทำ

<<< LOAD CASE MENU >>>

E = EDIT LOAD CASE

D = DELETE LOAD CASE

Q -> QUIT TO DATA MENU

E หมายถึง สร้างหรือแก้ไขชุดของน้ำหนักกระทำ

D หมายถึง ยกเลิกชุดของน้ำหนักกระทำ

4.2 ป้อนข้อมูลรายละเอียดของน้ำหนักกระทำแต่ละชนิดของน้ำหนักกระทำชุดหนึ่ง ๆ ด้วยการกดแป้น E จาก LOAD CASE MENU

<<< L O A D D A T A >>>

N = NODAL LOAD

G = GRAVITY LOADING

D = DISTRIBUTED EDGE LOAD

T = THERMAL LOADING

Q -> QUIT TO LOAD CASE MENU

N หมายถึง มีน้ำหนักกระทำที่ชิ้วของชิ้นส่วนในทิศทาง x และ y

G หมายถึง มีน้ำหนักกระทำเนื่องจากน้ำหนักตัวเองของชิ้นส่วน

D หมายถึง มีน้ำหนักแผ่สม่ำเสมอ

T หมายถึง มีความเครียดเริ่มแรกอันเนื่องมาจากการเปลี่ยนแปลงของอุณหภูมิ

Q หมายถึง สิ้นสุดการสร้างข้อมูลของน้ำหนักกระทำชุดนี้

หมายเหตุ - เครื่องหมายของน้ำหนักกระทำที่ชิ้ว (ดังในรูปที่ 7.2)

ในแนวแกน x และ y มีค่าเป็นบวก

ในทิศทางตรงกันข้ามกับแนวแกน x และ y มีค่าเป็นลบ

- เครื่องหมายของน้ำหนักแผ่สม่ำเสมอ

มีค่าเป็นบวก ตามทิศทางดังรูปที่ 7.3 และ 7.4

มีค่าเป็นลบ ตามทิศทางตรงกันข้ามกับรูปที่ 7.3 และ 7.4

- อุณหภูมิที่เปลี่ยนไปเทียบกับอุณหภูมิอ้างอิงที่ 0 องศา

ข.3 <u>การวิเคราะห์</u>

การเข้าสู่หมวดนี้ทำได้โดยการกดแป้น S จาก ACTIVITY MENU การทำงานของโปรแกรมภายใต้หมวดนี้จะเป็นการวิเคราะห์หาค่าการเคลื่อนที่ที่ขั้วของโครงสร้างเนื่องจากน้ำหนักกระทำแต่ละชุดที่ได้เตรียมไว้ในหมวดการสร้างหรือแก้ไขข้อมูล

< < < SOLUTION NODE > > >

N = Number of Gauss point in each direction (2)

O = Order of element as input

E = EXECUTION

M = MAX. VOLUME OF CORE STORAGE (7000)

N หมายถึง จุดอ้างอิงโดยวิธีของเกาส์

O หมายถึง การเลือกเรียงลำดับของชิ้นส่วน

E หมายถึง เริ่มทำการวิเคราะห์ตามค่าพารามิเตอร์ที่ตั้งไว้

M หมายถึง เลือกกำหนดการใช้เนื้อที่หน่วยความจำหลักในส่วนที่เก็บค่าสัมประสิทธิ์สติฟเนสเมตริกซ์ได้ (มากที่สุดเท่ากับ 32767 หน่วย)

ภายหลังจากที่ได้กำหนดค่าพารามิเตอร์แล้ว กด E เพื่อให้โปรแกรมเริ่มทำการวิเคราะห์ ซึ่งในการวิเคราะห์นี้สามารถเลือกเป้าหมายการวิเคราะห์ได้ 2 กรณีดังนี้คือ

W หมายถึง วิเคราะห์เพื่อหาค่าพารามิเตอร์ต่าง ๆ ที่ใช้ในวิธีสกายลายน์ฟรอนทัล

C หมายถึง วิเคราะห์เพื่อหาค่าการเคลื่อนที่ที่ขั้ว

และโปรแกรมจะกลับเข้าสู่ ACTIVITY MENU

ข.4 <u>การแสดงผลลัพธ์</u>

การเข้าสู่หมวดแสดงผลลัพธ์สามารถกระทำได้โดยการกดแป้น R จาก ACTIVITY MENU ซึ่งมี 2 กรณีคือ แสดงค่าการเคลื่อนที่ที่ขั้ว และแสดงค่าความเค้นที่เกิดขึ้นที่ชิ้นส่วน

<<< RESULT MODE >>>

D = DISPLACEMENTS

S = STRESSES

Q -> QUIT TO ACTIVITY MENU

D หมายถึง แสดงค่าการเคลื่อนที่ที่ขั้ว

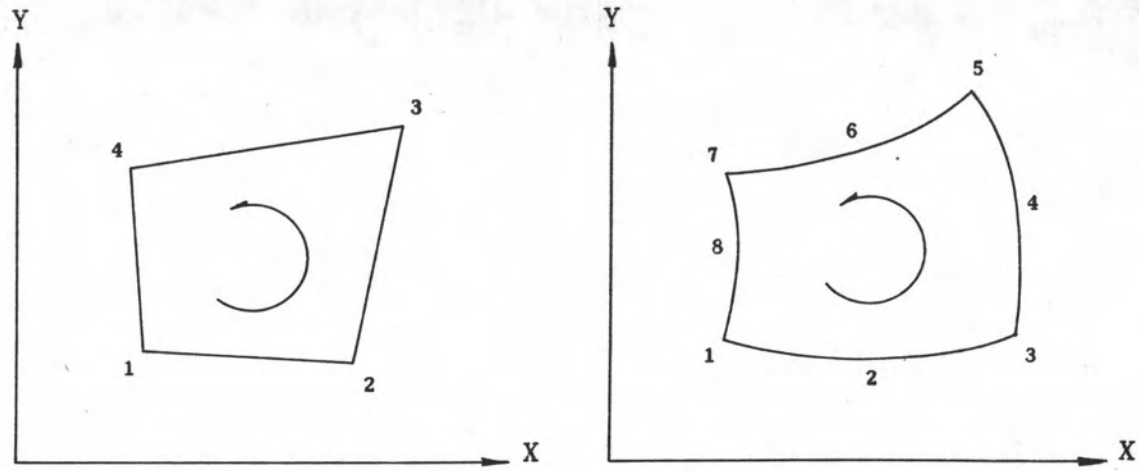
S หมายถึง แสดงค่าความเค้นที่เกิดขึ้น

หมายเหตุ เครื่องหมายค่าการเคลื่อนที่ที่ขั้ว (ดังในรูปที่ 7.2)

มีค่าเป็น บวก แสดงว่ามีการเคลื่อนที่ในแนวแกน x หรือ y

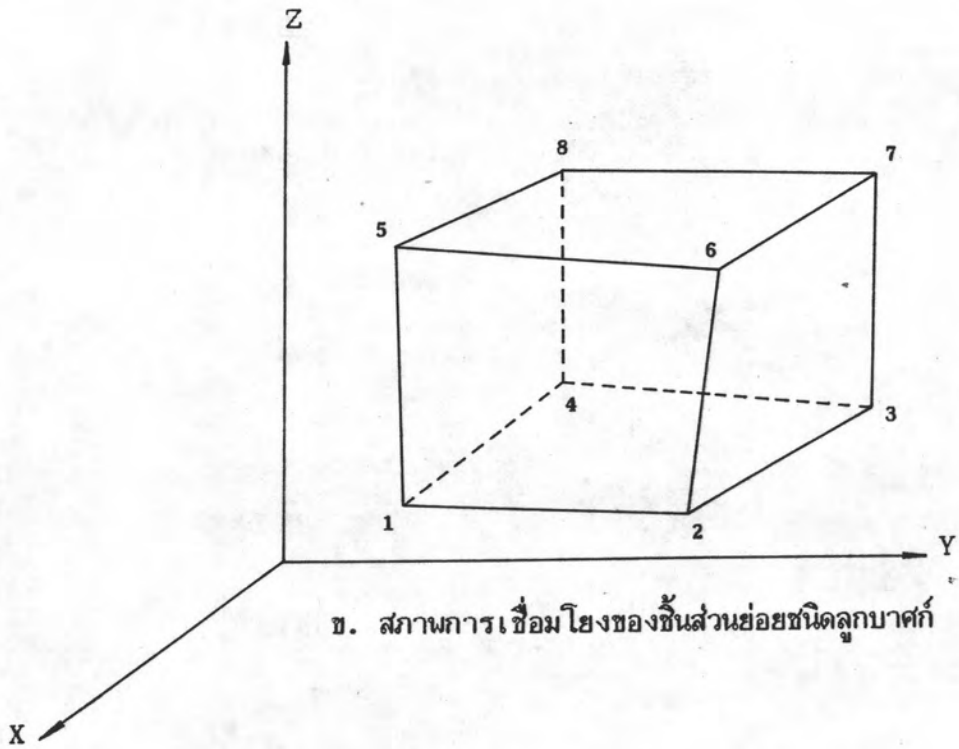
มีค่าเป็น ลบ แสดงว่ามีการเคลื่อนที่ในแนวตรงข้ามกับแกน x หรือ y

ตำแหน่งที่แสดงค่าความเค้นของชิ้นส่วนชนิดลูกบาศก์แสดงในรูปที่ 7.5



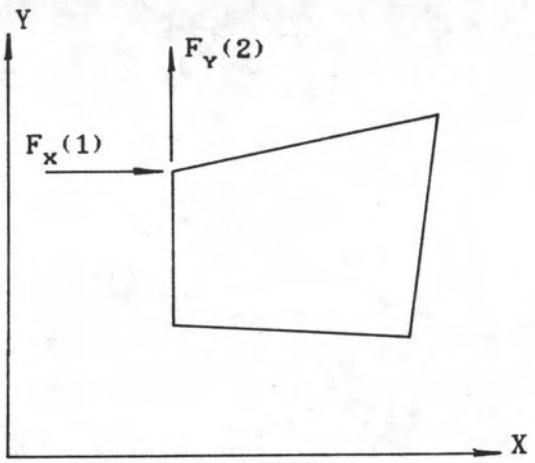
ชิ้นส่วนย่อยไอโซพาราเมตริกเชิงเส้น ชิ้นส่วนย่อยไอโซพาราเมตริกกำลังสอง

ก. สภาพการเชื่อมโยงชิ้นส่วนย่อยชนิดความเค้นในระนาบ
ชนิดความเครียดในระนาบ และชนิดแผ่นบางรับแรงดัด

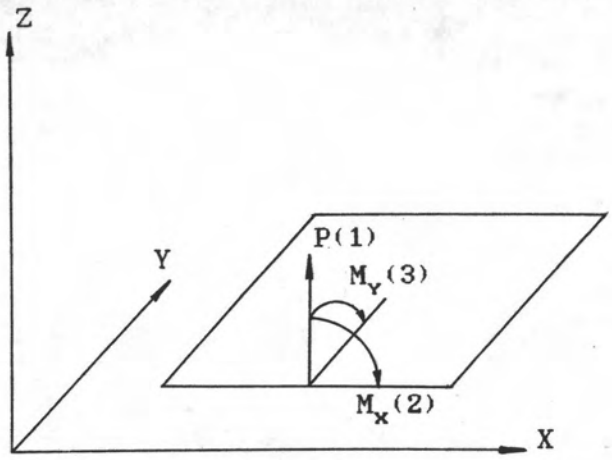


ข. สภาพการเชื่อมโยงของชิ้นส่วนย่อยชนิดลูกบาศก์

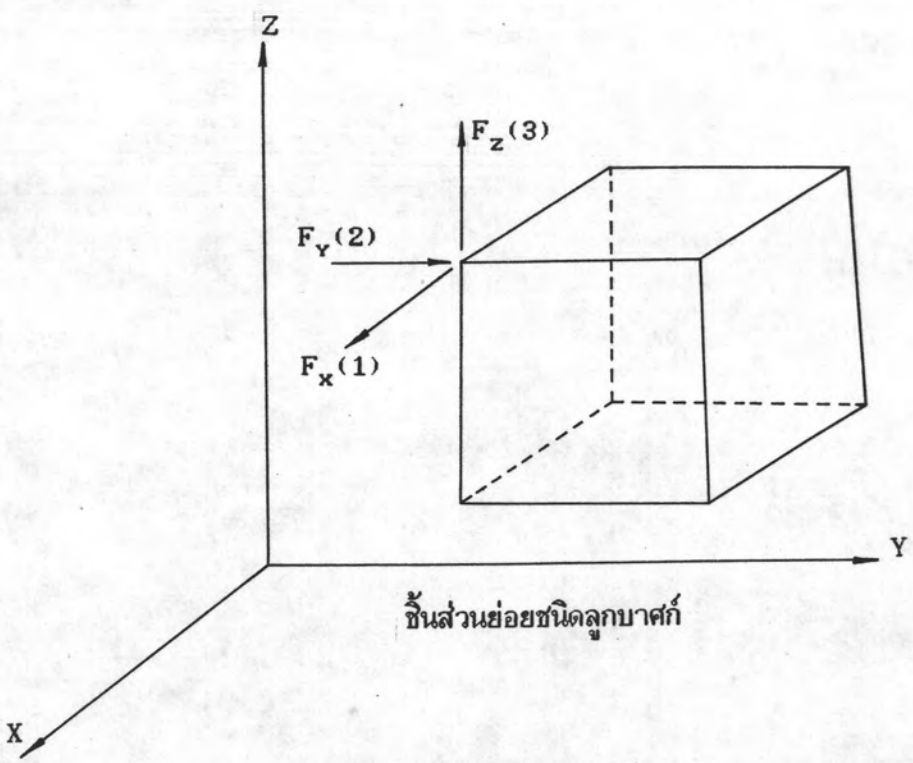
รูปที่ 7.1 แสดงสภาพการเชื่อมโยงของชิ้นส่วน



ชิ้นส่วนย่อยชนิดความเค้นในระนาบ
และชนิดความเครียดในระนาบ



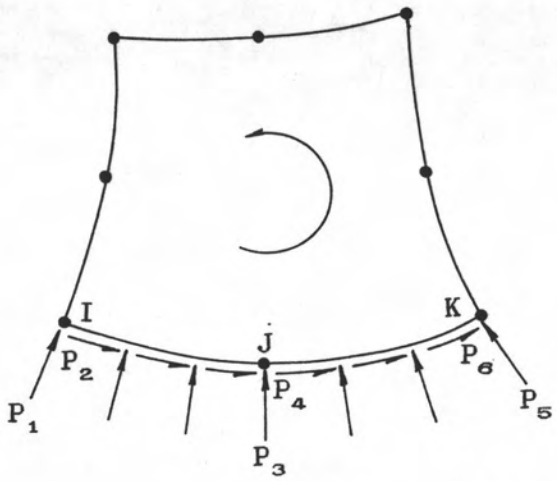
ชิ้นส่วนย่อยชนิดแผ่นบางรับแรงดัด

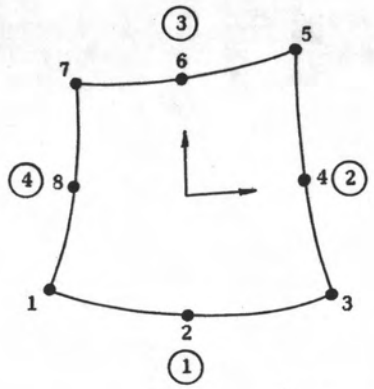


ชิ้นส่วนย่อยชนิดลูกบาศก์

หมายเหตุ ตัวเลขใน ( ) หมายถึงลำดับค่าระดับชั้นความเสรี

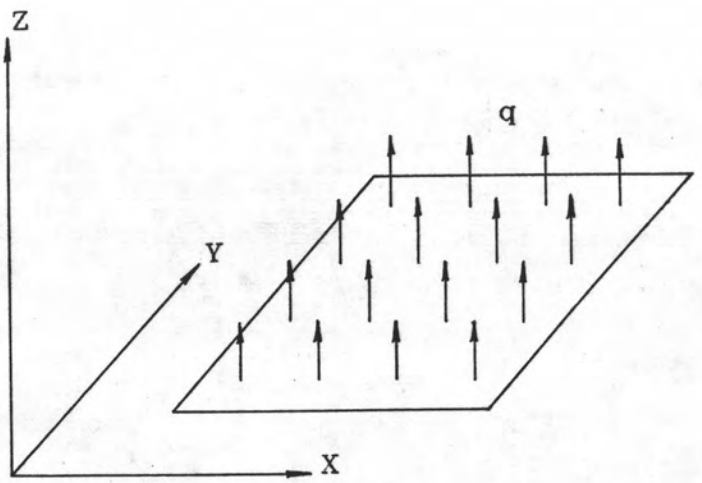
รูปที่ 7.2 แสดงทิศทางสำหรับค่าที่เป็นบวกของน้ำหนักกระทำที่ขั้วและการเคลื่อนที่ที่ขั้ว





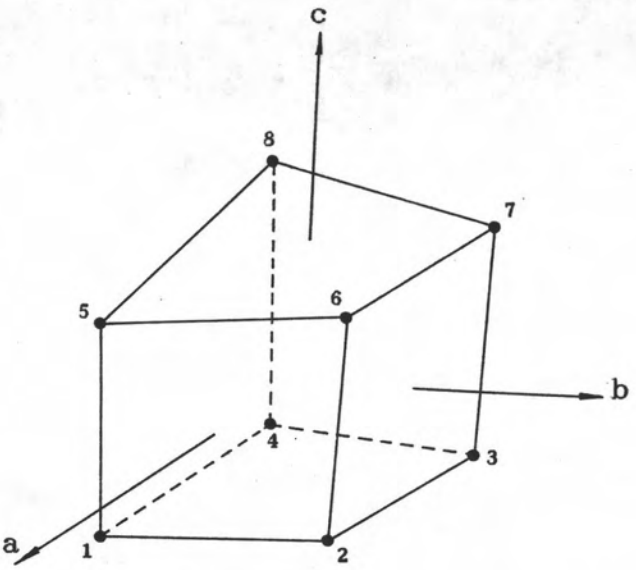
○ หมายเลขประจำด้านของชิ้นส่วนย่อย (ISIDE)

ก. ชิ้นส่วนย่อยชนิดความเค้นในระนาบและชนิดความเครียดในระนาบ



ข. ชิ้นส่วนย่อยชนิดแผ่นบางรับแรงดัด

รูปที่ 7.3 แสดงทิศทางสำหรับค่าที่เป็นบวกของน้ำหนักแผ่สม่ำเสมอ



ด้านที่ 1 คือด้านที่อยู่ในแนวแกน + a

2 คือด้านที่อยู่ในแนวแกน - a

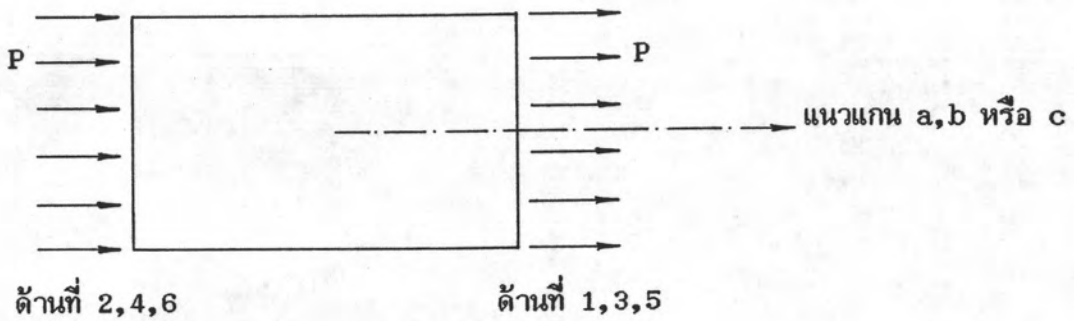
3 คือด้านที่อยู่ในแนวแกน + b

4 คือด้านที่อยู่ในแนวแกน - b

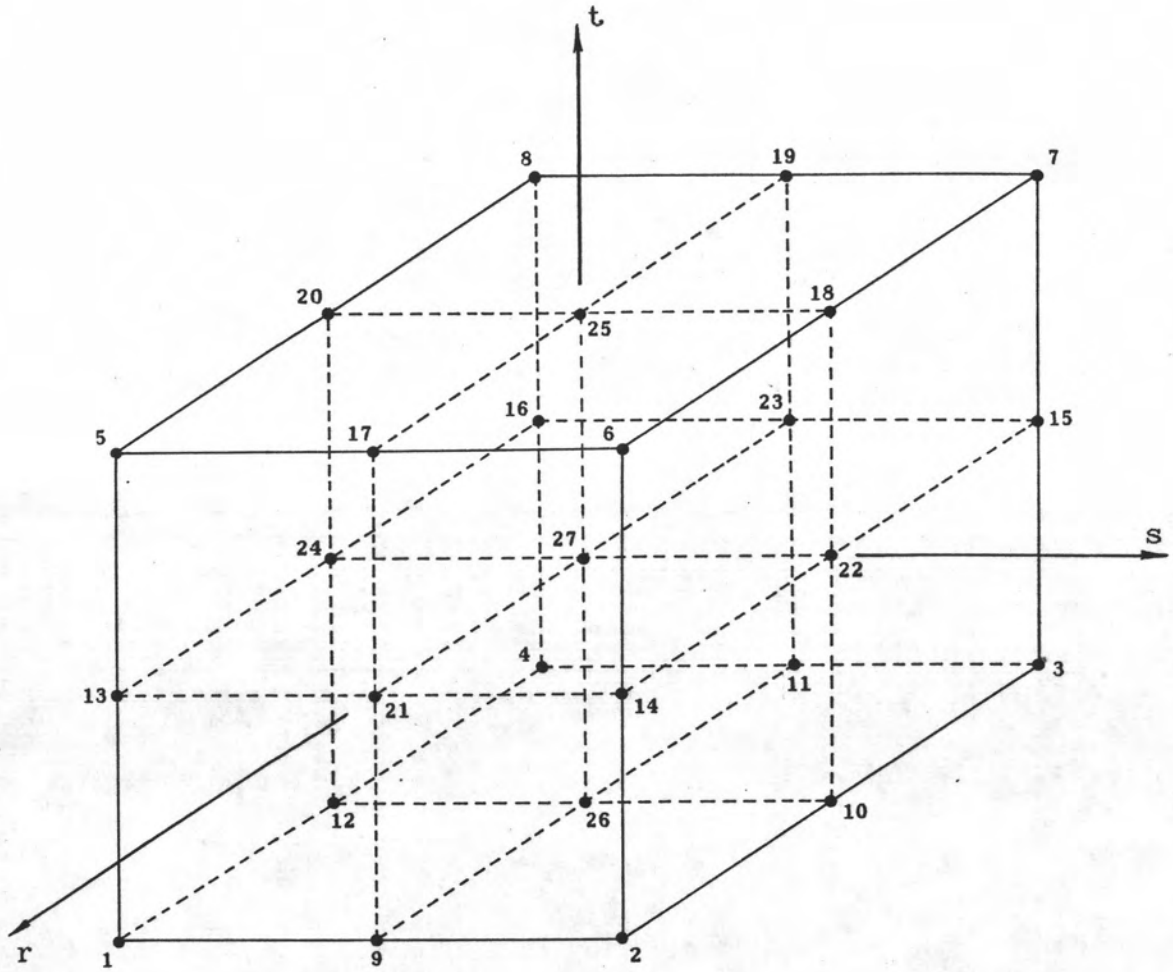
5 คือด้านที่อยู่ในแนวแกน + c

6 คือด้านที่อยู่ในแนวแกน - c

ชิ้นส่วนย่อยชนิดลูกบาศก์



รูปที่ 7.4 แสดงทิศทางสำหรับค่าที่เป็นบวกของน้ำหนักแผ่สม่ำเสมอ



รูปที่ 7.5 แสดงตำแหน่งความเค้นที่เกิดขึ้นของชิ้นส่วนย่อยชนิดลูกบาศก์

(STRESS OUTPUT LOCATIONS)

ประวัติ

นายสุภัทร์ อุทัยวัฒน์ เกิดเมื่อวันที่ 19 ธันวาคม พ.ศ. 2502 ที่จังหวัดตรัง สำเร็จการศึกษาวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมโยธา จาก สถาบันเทคโนโลยีพระจอมเกล้าธนบุรี ในเดือนมีนาคม พ.ศ. 2525 ทำงานครั้งแรกด้านการก่อสร้างอาคารกับ บริษัทโมดูล่าร์ จำกัด เดือนมิถุนายน พ.ศ.2525 และ บริษัทคอนติเนนตัลคอนสตรัคชั่น จำกัด เมื่อเดือนกันยายน พ.ศ. 2525 เข้าศึกษาหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิศวกรรมโครงสร้าง ภาควิชาวิศวกรรมโยธา คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2527