```
function meshgen
%*********************************************************************
%                                                                    *
%                                                                    *
%   FINITE ELEMENT    MESH GENERATOR                                 *
%                                                                    *
%   THIS PROGRAM GENERATES FINTIE ELEMENT MESHES for EITHER          *
%   3 NODE OR 6 NODE TRIANGULAR ELEMENTS                             *
%                                                                    *
%                                                                    *
%   NNPE          = NUMBER OF NODES PER ELEMENT (MUST BE 3 OR 6)     *
%   NUMLPS        = NUMBER OF LOOPS:BE USED for CURRENT MESH         *
%   NDIV(I,J)     = NUMBER OF DIVISIONS ON SIDE J OF LOOP I          *
%                     J=1 REFERS:SIDE DEFINED BY FIRST THREE         *
%                         (XCOR,YCOR). WILL AUTOMATICALLY            *
%                         DESIGNATE SIDE 3 AS HAVING THE SAME        *
%                         NUMBER OF DIVISIONS                        *
%                                                                    *
%                     J=2 REFER:SIDE DEFINED BY (XCOR,YCOR)          *
%                         2, 3 AND 4. WILL DESIGNATE SIDE 4 AS       *
%                         HAVING THE SAME NUMBER OF DIVISIONS        *
%                                                                    *
%   JOIN(I,J,K)   = CURRENT LOOP (LOPP I) SIDE K IS JOINED:          *
%                   :SIDE JOIN(I,J,2) OF LOOP (I,J,1)                *
%   XCOR(I,J)     = THE 8 X-COORDINATES OF LOOP I                    *
%   YCOR(I,J)     = THE 8 Y-COORDINATES OF LOOP I                    *
%                                                                    *
% DIMENSIONS OF THE ARRAYS USED IN THIS PROGRAM                      *
%                                                                    *
% MXEE = NUMBER OF ELEMENTS                                          *
% MXLL = NUMBER OF LOOPS                                             *
% MXNN = NUMBER OF NODAL POINTS                                      *
% MXSS = MAXIMUM NUMBER OF NODES ALONG ANY LOOP SIDE                 *
%               MXSS MUST BE EQUAL OR GREATER THAN:                  *
%                  NDIV + 1    for NNPE = 3                          *
%                2*NDIV +1     for NNPE = 6                          *
%                                                                    *
%*********************************************************************

clear all
clc

mxee = 2000;
mxnn = 2200;
mxss = 99;
mxll = 10;

xcor(1:mxll,1:8) = 0;
ycor(1:mxll,1:8) = 0;
xord(1:mxnn) = 0;
yord(1:mxnn) = 0;
rn(1:8) = 0;

npn(1:mxnn)           = 0;
np(1:mxee,1:6)        = 0;
ndiv(1:mxll,1:4)      = 0;
joinn(1:mxll,1:4,1:2) = 0;
lnr(1:4,1:mxss)       = 0;
lnp(1:6,1:4,1:mxss)   = 0;

irz    = 0;
contr  = 1;
rcheck = 0;
```

```
disp('PROGRAM MESHGEN')

%==========================
%=  DEFINITE  FILE NAMES  =
%==========================
fileID = -1;
errmsg = '';
while fileID < 0
    disp(errmsg);
    file = input('Open file: ', 's');
    filein  = strcat(file,'.min');
    [fileID,errmsg] = fopen(filein);
end

%  READ DATA FROM FILE
A = fscanf(fileID,'%d %d',2);
nnpe  = A(1);
numlps= A(2);

X = ['Number of nodes/element : ',num2str(nnpe)];
disp(X)
X = ['Number of loops inmesh  : ',num2str(numlps)];
disp(X)

if (nnpe ~= 3) && (nnpe ~= 6)
    disp('NNPE <> 3 And NNPE <> ');
    exit;
end
if numlps > mxll
    disp('NUMLPS > maxloops');
    exit;
end

for i = 1:numlps
    A = fscanf(fileID,'%d %d',2);
    ndiv(i, 1) = A(1);
    ndiv(i, 2) = A(2);

    A = fscanf(fileID,'%d %d %d %d %d %d %d %d ',8);
    for j = 1:4
        for k = 1:2
            joinn(i, j, k) = A(2*(j-1)+k);
        end %end %Next k
    end %Next j

    for j = 1:8
        A = fscanf(fileID,'%f %f',2);
        xcor(i, j) = A(1);
        ycor(i, j) = A(2);
    end %Next j
end %Next i

fclose(fileID);


npd3  = nnpe/3;
dfact = 1;

for i = 1:numlps
    ndiv(i, 3) = ndiv(i, 1);
    ndiv(i, 4) = ndiv(i, 2);
end %Next i
```

```
% CALCULATE NUMBER OF NODAL POINTS
numnp = 0;
numel = 0;

for i = 1:numlps

    i7 = (npd3 * ndiv(i, 1) + 1) * (npd3 * ndiv(i, 2) + 1);
    numel = numel + 2 * ndiv(i, 1) * ndiv(i, 2);
    numnp = numnp + i7;
    j1 = 4;

    for j = 1:4
        if joinn(i, j, 1) ~= 0
            numnp = numnp - (npd3 * ndiv(i, j) + 1);
            if joinn(i, j1, 1) ~= 0
                numnp = numnp + 1;
            end %if
            j1 = j;
        else
            j1 = j;
        end %if
    end %Next j

end %Next i

inp = 0;
iel = 0;

    for i = 1:numlps

        i1 = ndiv(i, 1);
        i2 = npd3 * i1 + 1;

        i4 = ndiv(i, 2);
        i5 = npd3 * i4 + 1;

        i7 = i2 * i5;

%   CALCULATE SIDE ARRAYS

        jend = i2;

        j2 = i7 - i2;
        j4 = i2 + 1;

        for j = 1:jend

            j4 = j4 - 1;
            lnr(1, j) = j;
            lnr(3, j4) = j2 + j;
        end %Next j

        jend = i5;
        j4 = i5 + 1;

        for j = 1:jend
            j4 = j4 - 1;
            lnr(2, j) = j * i2;
            lnr(4, j4) = (j - 1) * i2 + 1;
        end %Next j
```

```
%   CALCULATE NPN ARRAY

        jend = i7;
        npn(1:jend) = 0;

        for j = 1:4

            if joinn(i, j, 1) ~= 0
                j1 = joinn(i, j, 1);
                j2 = joinn(i, j, 2);

                kend = npd3 * ndiv(i, j) + 1;
                k2 = kend + 1;

                for k = 1:kend
                    k1 = lnr(j, k);
                    k2 = k2 - 1;
                    npn(k1) = lnp(j1, j2, k2);
                end %Next k
            end %if

        end %Next j

        jend = i7;

        for j = 1:jend
            if npn(j) == 0
                inp = inp + 1;
                npn(j) = inp;
            end %if
        end %Next j

%   form LNP ARRAY

        jend = i2;

        for j = 1:jend
            j1 = lnr(1, j);
            j3 = lnr(3, j);
            lnp(i, 1, j) = npn(j1);
            lnp(i, 3, j) = npn(j3);
        end %Next j

        jend = i5;

        for j = 1:jend
            j2 = lnr(2, j);
            j4 = lnr(4, j);
            lnp(i, 2, j) = npn(j2);
            lnp(i, 4, j) = npn(j4);
        end %Next j

%   CALCULATE NODAL POINT COORDINATES

        r = npd3 * i1;
        dx = 1 / r;
        r = npd3 * i4;
        dy = 1 / r;

        jend = i5;
        kend = i2;

        k1 = 0;
```

```
        for j = 1:jend

            r = j - 1;
            ry = r * dy;

            for k = 1:kend

                r = k - 1;
                rx = r * dx;
                rn(1) = 1 * (1 - rx) * (1 - ry) * (1 - 2 * rx - 2 * ry);
                rn(2) = 4 * (rx) * (1 - rx) * (1 - ry);
                rn(3) = -1 * (rx) * (1 - ry) * (1 - 2 * rx + 2 * ry);
                rn(4) = 4 * (rx) * (ry) * (1 - ry);
                rn(5) = -1 * (rx) * (ry) * (3 - 2 * rx - 2 * ry);
                rn(6) = 4 * (rx) * (1 - rx) * (ry);
                rn(7) = -1 * (1 - rx) * (ry) * (1 + 2 * rx - 2 * ry);
                rn(8) = 4 * (1 - rx) * (ry) * (1 - ry);
                k1 = k1 + 1;
                k2 = npn(k1);
                xord(k2) = 0;
                yord(k2) = 0;

                for l = 1:8
                    xord(k2) = xord(k2) + rn(l) * xcor(i, l);
                    yord(k2) = yord(k2) + rn(l) * ycor(i, l);
                end %Next l
            end %Next k
        end %Next j

%   CALCULATION OF NP ARRAY

        if npd3 ~= 1

            jend = i4;
            kend = i1;

            for j = 1:jend
                for k = 1:kend
                    iel = iel + 2;
                    n1 = iel - 1;
                    n2 = iel;

                    k1 = (j - 1) * 2 * i2 + (k - 1) * 2 + 1;
                    k2 = k1 + 1;
                    k3 = k1 + 2;
                    k4 = k1 + i2;
                    k5 = k4 + 1;
                    k6 = k4 + 2;
                    k7 = k1 + 2 * i2;
                    k8 = k7 + 1;
                    k9 = k7 + 2;

                    k1 = npn(k1);
                    k2 = npn(k2);
                    k3 = npn(k3);
                    k4 = npn(k4);
                    k5 = npn(k5);
                    k6 = npn(k6);
                    k7 = npn(k7);
                    k8 = npn(k8);
                    k9 = npn(k9);
```

```
                d1 = (xord(k9) - xord(k1)) ^ 2 + (yord(k9) - yord(k1)) ^ 2;
                d2 = (xord(k7) - xord(k3)) ^ 2 + (yord(k7) - yord(k3)) ^ 2;
                d1 = d1 + (0.0001) * d1 * dfact;

                if d2 < d1

                    np(n1, 1) = k1;
                    np(n1, 2) = k2;
                    np(n1, 3) = k3;
                    np(n1, 4) = k5;
                    np(n1, 5) = k7;
                    np(n1, 6) = k4;

                    np(n2, 1) = k3;
                    np(n2, 2) = k6;
                    np(n2, 3) = k9;
                    np(n2, 4) = k8;
                    np(n2, 5) = k7;
                    np(n2, 6) = k5;

                else

                    np(n1, 1) = k1;
                    np(n1, 2) = k5;
                    np(n1, 3) = k9;
                    np(n1, 4) = k8;
                    np(n1, 5) = k7;
                    np(n1, 6) = k4;

                    np(n2, 1) = k1;
                    np(n2, 2) = k2;
                    np(n2, 3) = k3;
                    np(n2, 4) = k6;
                    np(n2, 5) = k9;
                    np(n2, 6) = k5;

                end %if

            end %Next k
        end %Next j

    else

        jend = i4;
        kend = i1;

        for j = 1:jend
            for k = 1:kend

                iel = iel + 2;
                n1 = iel - 1;
                n2 = iel;

                k1 = (j - 1) * i2 + (k - 1) + 1;
                k2 = k1 + 1;
                k3 = k1 + i2;
                k4 = k3 + 1;

                k1 = npn(k1);
                k2 = npn(k2);
                k3 = npn(k3);
                k4 = npn(k4);
```

```
                    d1 = (xord(k4) - xord(k1)) ^ 2 + (yord(k4) - yord(k1)) ^ 2;
                    d2 = (xord(k3) - xord(k2)) ^ 2 + (yord(k3) - yord(k2)) ^ 2;
                    d1 = d1 + (0.0001) * d1 * dfact;
                    dfact = -dfact;

                    if d2 < d1

                        np(n1, 1) = k1;
                        np(n1, 2) = k2;
                        np(n1, 3) = k3;

                        np(n2, 1) = k2;
                        np(n2, 2) = k4;
                        np(n2, 3) = k3;

                    else

                        np(n1, 1) = k1;
                        np(n1, 2) = k4;
                        np(n1, 3) = k3;

                        np(n2, 1) = k1;
                        np(n2, 2) = k2;
                        np(n2, 3) = k4;

                    end %if

                end %Next k
            end %Next j

        end %if

    end %Next i

    ib = 0;

    for i = 1:numel
        for j = 1:nnpe
            for k = j:nnpe

                k1 = abs(np(i, j) - np(i, k));
                if k1 >= ib
                    ib = k1 + 1;
                end

            end %Next k
        end %Next j
    end %Next i

%  OUTPUT OF DATA
    fileout = strcat(file,'.mout');
    fileID = fopen(fileout,'w');

    fprintf(fileID,'%10i %10i %10i %10i %10i\r\n', numnp, numel, ib, irz, nnpe);

    for i = 1:numnp
        fprintf(fileID,'%10i %15.5f %15.5f\r\n', i, xord(i), yord(i));
    end % Next i
```

```
    for i = 1:numel
        if nnpe == 3
            fprintf(fileID,'%10i %10i %10i %10i\r\n', i, np(i, 1), np(i, 2), np(i, 3));
        else
            fprintf(fileID,'%10i %10i %10i %10i %10i %10i %10i\r\n', i, np(i, 1),
                                    np(i, 2), np(i, 3), np(i, 4), np(i, 5), np(i, 6));
        end %if
    end %Next i

    fclose(fileID);

    X = ['MESHGEN Complete - band width = ',num2str(ib)];
    disp(X)

%----------------------------------------------------------------------
%    Plot mesh generated by the MESHGEN
%----------------------------------------------------------------------

    xmin = min(xord);
    xmax = max(xord);
    ymin = min(yord);
    ymax = max(yord);

    xdiff = xmax - xmin;
    ydiff = ymax - ymin;

    offset = 0.05;

    xleft   = xmin - offset * xdiff;
    xright  = xmax + offset * xdiff;
    ybottom = ymin - offset * ydiff;
    ytop    = ymax + offset * ydiff;
    width   = xright - xleft;
    height  = ytop - ybottom;
    box     = max( width, height);

    pscale = 1;

    rectangle('Position',[xleft ybottom box box ])
    set(gca,'fontsize',10)
    hold on

    print_nodes    = input('Print node numbers (y/n)  : ', 's');
    print_elements = input('Print elment numbers (y/n): ', 's');

    % Print mesh, node numbers, and element numbers
    offset = 0.05;

    for i = 1:numel
        for j = 1:nnpe
            xp(j) = (xord(np(i, j)) - xmin) * pscale;
            yp(j) = (yord(np(i, j)) - ymin) * pscale;
        end %Next j

        sumx = 0;
        sumy = 0;

        xnp_max = max(xp(1:nnpe));
        xnp_min = min(xp(1:nnpe));
        ynp_max = max(yp(1:nnpe));
        ynp_min = min(yp(1:nnpe));
        xplus = xnp_max-xnp_min;
        yplus = ynp_max-ynp_min;
```

```
        np_off = max(xplus, yplus)/10;


        for j = 1:nnpe
            j1 = j;
            j2 = j + 1;
            if j2 > nnpe
                j2 = 1;
            end

            plot([xp(j1) xp(j2)],[yp(j1) yp(j2)],'-ok','MarkerSize',2);

            if (print_nodes(1) == 'y')
                text(xp(j2)+np_off,yp(j2)+np_off,num2str(np(i,j2)),'FontSize',6);
            end
            hold on

            sumx = sumx + xord(np(i, j));
            sumy = sumy + yord(np(i, j));

        end %for j = 1:nnpe

        if (print_elements(1) == 'y')
            xnp = ((sumx / nnpe) - xmin) * pscale;
            ynp = ((sumy / nnpe) - ymin) * pscale;
            text(xnp,ynp,num2str(i),'FontSize',6);
            hold on
        end %if (print_elements(1) == 'y')

    end %for i = 1:numel

end %function meshgen
```