```
close all;
%clear all;
clc;

syms xi eta;


% coordinate details - row no. = node no.
nodal_coord = [0,0;
            0.076,0;
            0.076,0.05;
            0,0.05;
            0.038,0;

            0.076,0.025;
            0.038,0.05;
            0,0.025;
            0.038,0.025;];


% element connectivity - row no. = element no.
connectivity = [1,5,9,8;
             5,2,6,9;
             9,6,3,7;
             8,9,7,4;];




n_node = size(nodal_coord,1);
n_ele = size(connectivity,1);
ndim_temp = 1;
ndim_disp = 2;
%----------material property------------------

conductance = [45, 45;]; %can add more rows for element specific property definition
E = 2e5;
alpha = 7.2e-6;
pnu = 0.3;
t = 1e-3;
%----------temp_bc---------------------------

node_temp = [1 100;
          8 100;
```

```
            4 100;
            2 0;
            6 0;
            3 0;];

temp_load = zeros((n_node*ndim_temp),1);

initial_temp = zeros((n_node*ndim_disp),1);
initial_temp_node_no = node_temp(:,1);

% assigning initial temp to all disp dofs
initial_temp([initial_temp_node_no*2-1;initial_temp_node_no*2],1) =
initial_temp([initial_temp_node_no*2-1,initial_temp_node_no*2],1) +
[node_temp(:,2);node_temp(:,2)];
%----------- nodal displacements-----------------

nodal_spc = [1 0;
            2 0;
            5 0;
            4 0;
            7 0;
            3 0;];
nodal_spc_all_dof = zeros(2*size(nodal_spc,1),2);

for i = 1:size(nodal_spc,1)
  qq = nodal_spc(i,1);
  nodal_spc_all_dof (2*i-1,1) = 2*nodal_spc(i,1)-1;
  nodal_spc_all_dof (2*i,1) = 2*nodal_spc(i,1);
  nodal_spc_all_dof (2*i-1,2) = nodal_spc(i,2);
  nodal_spc_all_dof (2*i,2) = nodal_spc(i,2);
end


%-------assigning to D matrix------------
kx = conductance(1);
ky = conductance(2);
D_temp = [kx 0; 0 ky];
D_disp = (E/(1-pnu^2))*[1, pnu, 0;
                      pnu, 1, 0;
                      0 0 (1-pnu)/2;]

N = (1/4)*[(1-xi)*(1-eta), (1+xi)*(1-eta), (1+xi)*(1+eta), (1-xi)*(1+eta)];
dNxi = [eta/4 - 1/4, 1/4 - eta/4, eta/4 + 1/4, - eta/4 - 1/4];
```

```
dNeta = [xi/4 - 1/4, - xi/4 - 1/4, xi/4 + 1/4, 1/4 - xi/4];
% dNxi = diff(N,xi);
% dNeta = diff(N,eta);


%--------------------A & G matrix -------------------
A_temp = [1 0 ; 0 1;];
A_disp = [1 0 0 0; 0 0 0 1; 0 1 1 0;];
G_temp = [dNxi; dNeta;];
G_disp = [dNxi, zeros(1,4);
        dNeta, zeros(1,4);
        zeros(1,4), dNxi;
        zeros(1,4), dNeta;];

integration_point = [-1/sqrt(3), -1/sqrt(3);
                  1/sqrt(3), -1/sqrt(3);
                  1/sqrt(3), 1/sqrt(3);
                 -1/sqrt(3), 1/sqrt(3);];

integration_weights = [1 1; 1 1; 1 1; 1 1;];

%------------------Stiffness initiation-------------
stiffness_gbl_temp = zeros(n_node*ndim_temp);


%----------------Temperature Stiffness Calculation---------------

for i = 1:n_ele
%----finding B matrix in xi-eta form
[xbar,ybar] = ele_coord(i,connectivity,nodal_coord);
jacobian = [dNxi*xbar, dNxi*ybar; dNeta*xbar, dNeta*ybar;];
det_jacob = det(jacobian);
B_temp_local = simplify((A_temp\jacobian)*G_temp);

%-----------doing gauss integration
B_temp = gauss_integration(B_temp_local,integration_point);
curr_nodes = connectivity(i,:);
stiffness_gbl_temp([curr_nodes],[curr_nodes]) = B_temp'*D_temp*B_temp*det_jacob +
stiffness_gbl_temp([curr_nodes],[curr_nodes]);
end


%----------------Applying BC for temperature distribution---------
```

```
temp_rdof= node_temp(:,1)'; %nodes where temp is specified
temp_fdof= setdiff(1:n_node,temp_rdof);


stiff_diag = diag(stiffness_gbl_temp);
max_stiff = abs(max(stiff_diag))*1e4;

%-----adding penalty term to diagonal elements and flux vector-----

for j = 1: size(temp_rdof,2)
   gg = node_temp(j);
   hhh = node_temp(:,1) == gg;
   stiffness_gbl_temp(gg,gg) = max_stiff + stiffness_gbl_temp(gg,gg);

   temp_load(gg,1) = max_stiff*node_temp(hhh,2) + temp_load(gg,1);
end

%--------------Solving Temp eqn-----------------------
temp_global = stiffness_gbl_temp\temp_load;
temp_global_disp = zeros(n_node*ndim_disp,1);

temp_global_disp([(1:n_node)'*2-1;(1:n_node)'*2],1) = [temp_global(:,1);temp_global(:,1)];

%--------------Stress-Strain Solving------------------

stiffness_gbl_disp = zeros(n_node*ndim_disp,n_node*ndim_disp);
temp_force_global = zeros(n_node*ndim_disp,1);
temp_force = zeros(4,1);

%----------Stiffness matrix calculation----------------

for i = 1:n_ele
%----finding B matrix in xi-eta form
[xbar,ybar] = ele_coord(1,connectivity,nodal_coord);
jacobian = [dNxi*xbar, dNxi*ybar; dNeta*xbar, dNeta*ybar];
det_jacob = det(jacobian);
jj =  [inv(jacobian), zeros(2,2);zeros(2,2), inv(jacobian)];
B_disp_local = A_disp*jj*G_disp;

%-----------doing gauss integration
kk = t*B_disp_local'*D_disp*B_disp_local*det_jacob;
%gauss_integration(B_disp_local,integration_point);
curr_nodes = connectivity(i,:);
```

```
curr_nodes_disp = [connectivity(i,:)*2-1,connectivity(i,:)*2];
stiffness_gbl_disp(curr_nodes_disp,curr_nodes_disp) = gauss_integration(kk,integration_point)
+ stiffness_gbl_disp(curr_nodes_disp,curr_nodes_disp);


%-----------epsi_t
%delta_t =  temp_global_disp(curr_nodes_disp,1) - initial_temp(curr_nodes_disp,1);

N_lcl = N;
tt = N_lcl*temp_global(curr_nodes,1);
epsi_t = alpha*[tt;tt;0;];
yy = t*B_disp_local'*D_disp*epsi_t;

temp_force(curr_nodes_disp,1) = gauss_integration(yy,integration_point);
temp_force_global = temp_force_global + temp_force(:,1);
temp_force = zeros(18,1);

end


%-----------Disp BC applying-------------
disp_rdof = [nodal_spc(:,1)*2-1;nodal_spc(:,1)*2]';
disp_fdof = setdiff(1:n_node*2,disp_rdof);

stiff_diag_disp = diag(stiffness_gbl_disp);
max_stiff_disp = max(stiff_diag_disp)*1e6;

%-----------Penalty approach for displacement------

for f = 1:size(disp_rdof,2)
    dd = nodal_spc_all_dof(f,1);
    ss = nodal_spc_all_dof(f,2);
    stiffness_gbl_disp(dd,dd) = max_stiff_disp + stiffness_gbl_disp(dd,dd);
    temp_force_global(f,1) = max_stiff_disp*ss + temp_force_global(f,1);
end

%-----------finding global disp vector-----------
%disp_global_fdof = stiffness_gbl_disp([disp_fdof],[disp_fdof])\temp_force_global([disp_fdof],1)
disp_global = stiffness_gbl_disp\temp_force_global;



%-----------------Stress calculation---------------
```

```
stress_ele_xx = zeros(n_ele,1);
stress_ele_yy = zeros(n_ele,1);
stress_ele_xy = zeros(n_ele,1);

for i = 1:n_ele
xi=0;
eta=0;
%calculating B matrix
[xbar,ybar] = ele_coord(1,connectivity,nodal_coord);
jacobian = subs([dNxi*xbar, dNxi*ybar; dNeta*xbar, dNeta*ybar]);
det_jacob = det(jacobian);
jj =  [inv(jacobian), zeros(2,2);zeros(2,2), inv(jacobian)];
B_stress = subs(A_disp*jj*G_disp);


% epsi t calculation
N_lcl_stress = subs(N);
curr_nodes_stress = connectivity(i,:);
tt = double(N_lcl_stress*temp_global(curr_nodes_stress,1));

curr_nodes_dof = [connectivity(i,:)*2-1,connectivity(i,:)*2];

epsi_t_stress = alpha*[tt;tt;0;];


%extracting disp for curr nodes

disp_curr_stress = disp_global(curr_nodes_dof,1);
sigma_lcl = double(D_disp*(B_stress*disp_curr_stress-epsi_t_stress));

stress_ele_xx(i) = sigma_lcl(1,1);
stress_ele_yy(i) = sigma_lcl(2,1);
stress_ele_xy(i) = sigma_lcl(3,1);

end

%--------averaging stress for each node for plotting
stress_node_xx = zeros(n_node,1);
stress_node_yy = zeros(n_node,1);
stress_node_xy = zeros(n_node,1);

%----finding no. of elements connecting to that same particular node
node_count = zeros(n_node,1);
```

```
for i = 1:n_ele
    %getting nodes for every element
    node_curr = [connectivity(i,:)];
    %adding stress to nodes
    stress_node_xx(node_curr) = stress_ele_xx(i) + stress_node_xx(node_curr);
    stress_node_yy(node_curr) = stress_ele_yy(i) + stress_node_yy(node_curr);
    stress_node_xy(node_curr) = stress_ele_xy(i) + stress_node_xy(node_curr);
    node_count(node_curr) = node_count(node_curr) + 1;
end

stress_node_xx = stress_node_xx./node_count;
stress_node_yy = stress_node_yy./node_count;
stress_node_xy = stress_node_xy./node_count;





%---------------Displaying Results--------------------

%---to fnf mesh spacing
vv = max(nodal_coord(:,1));
vvv = diff(sort(nodal_coord(:,1)));
vvvv = vvv(vvv~=0);
le_x = min(vvvv);
bb = max(nodal_coord(:,2));
bbb = diff(sort(nodal_coord(:,2)));
bbbb = bbb(bbb~=0);
le_y = min(bbbb);

ndiv_x = (vv/le_x)+1;
ndiv_y = (bb/le_y)+1;


temp_distrib = zeros(ndiv_x,ndiv_y);
x_disp_distrib = zeros(ndiv_x,ndiv_y);
y_disp_distrib = zeros(ndiv_x,ndiv_y);


for ir = 1:9
    ycord = ndiv_y - nodal_coord(ir,2)/le_y;
    xcord = nodal_coord(ir,1)/le_x+1;
    row = find(nodal_coord(:,1) == nodal_coord(ir,1) & nodal_coord(:,2) == nodal_coord(ir,2));
    temp_distrib(ycord,xcord) = temp_global(row);
    x_disp_distrib(ycord,xcord) = disp_global(2*row-1);
    x_disp_distrib(ycord,xcord) = disp_global(2*row);

end



%displaying temp distrib in matrix form
disp('Temperature Distribution');
disp(temp_distrib);
%creating meshgrid for contour plotting
[x1,x2] = meshgrid(0:le_x:vv,0:le_y:bb);

%plotting in temp contours
figure;
contourf (x1,x2,temp_distrib,100,'LineColor','none');
hh = colorbar;
xlabel('x-axis (cm)'); ylabel('y-axis (cm)');
title('Temperature Distribution');
ylabel(hh,'Temperature (°C)');
pbaspect([le_x le_y 1]);

% %plotting in x disp contours
% figure;
% contourf (x1,x2,x_disp_distrib,100,'LineColor','none');
% hh = colorbar;
% xlabel('x-axis (cm)'); ylabel('y-axis (cm)');
% title('Displacement in x-axis');
% ylabel(hh,'Displacement (cm)');
% pbaspect([le_x le_y 1]);
%
% %plotting in y disp contours
% figure;
% contourf (x1,x2,y_disp_distrib,100,'LineColor','none');
% hh = colorbar;
% xlabel('x-axis (cm)'); ylabel('y-axis (cm)');
% title('Displacement in y-axis');
% ylabel(hh,'Displacement (cm)');
% pbaspect([le_x le_y 1]);
%
```

```
%-----------Displaying Stress Distribution
x_min = min(nodal_coord(:,1));
x_max = max(nodal_coord(:,1));

y_min = min(nodal_coord(:,2));
y_max = max(nodal_coord(:,2));


[x_grid,y_grid] = meshgrid(x_min:le_x:x_max,y_min:le_y:y_max);
stress_grid_xx_node =
griddata(nodal_coord(:,1),nodal_coord(:,2),stress_node_xx,x_grid,y_grid,'cubic');
stress_grid_yy_node =
griddata(nodal_coord(:,1),nodal_coord(:,2),stress_node_yy,x_grid,y_grid,'cubic');
stress_grid_xy_node =
griddata(nodal_coord(:,1),nodal_coord(:,2),stress_node_xy,x_grid,y_grid,'cubic');

%-------Stress Contours Plotting----------
figure;
contourf(x_grid,y_grid,stress_grid_xx_node,100,'LineColor','none');
hh = colorbar;
xlabel('x-axis (cm)'); ylabel('y-axis (cm)');
title('Stress-xx Dsitribution');
ylabel(hh,'Stress (N/m_{2}');
pbaspect([le_x le_y 1]);


figure;
contourf(x_grid,y_grid,stress_grid_yy_node,100,'LineColor','none');
hh = colorbar;
xlabel('x-axis (cm)'); ylabel('y-axis (cm)');
title('Stress-yy Dsitribution');
ylabel(hh,'Stress (N/m_{2}');
pbaspect([le_x le_y 1]);


figure;
contourf(x_grid,y_grid,stress_grid_xy_node,100,'LineColor','none');
hh = colorbar;
xlabel('x-axis (cm)'); ylabel('y-axis (cm)');
title('Stress-xy Dsitribution');
ylabel(hh,'Stress (N/m_{2}');
pbaspect([le_x le_y 1]);
```

```
%---------Function to retrieve coordinates to current element
function [xbar,ybar] = ele_coord(n,connectivity,nodal_coord)
xbar = zeros(4,1);
ybar = zeros(4,1);
node_list = connectivity(n,:);
for c = 1:size(node_list,2)
    xbar(c) = nodal_coord(node_list(c),1);
    ybar(c) = nodal_coord(node_list(c),2);
end
end

%------------Function performs gauss intergation of given matrix
function [B_lol] = gauss_integration(lol,integration_point)
syms xi eta

B_lol = zeros(size(lol));

for pp = 1:size(integration_point,1)
    B_lol = B_lol + subs(lol,{xi, eta}, integration_point(pp,:));
end
end
```