```
%AE 510 Class
%Author: Your instructor

clear
close all
clc

%%%%%%%%%%%%PREPROCESSING%%%%%%%%%%%
%coordinate matrix [x,y] for each node

A = 0.0001;
alpha = 5E-6;
D_T = 100;
co = [1, 1;
      1, 0;
      0, 0;
      0, 1];

E = 70E9;

%element-node connectivity matrix (and area for each truss in column 3)
e = [4  1   A;
     2  4   2*A;
     1  2   A;
     3  2   A];

Nel = size(e,1);%number of elements
Nnodes = size(co,1); %number of nodes
nne = 2; %number of nodes per element
dof = 2; %degree of freedom per node

%%%%%%%%%%%%PREPROCESSING END%%%%%%%%%%%

%%%Generic block: Initializes global stiffness matrix 'K' and force vector 'F'
K = zeros(Nnodes*dof,Nnodes*dof);
F = zeros(Nnodes*dof,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%Assemble Global system - generic FE code for 2D and 3D trusses
for A = 1:Nel

    n = (co(e(A,2),:) - co(e(A,1),:));   %n = [x2-x1;y2-y1]

    n_2 = co(e(A,2),:);
    n_1 = co(e(A,1),:);

    L = norm(n); %length of truss

    n = n./L; %n = [sin,cos]
    Area = e(A,3); %area

    k11 = (E*Area/L)*(n'*n);%k matrix part = EA/L*[c^2 cs;sc s^2]
```

```
    %local stiffness matrix and force vector
    localstiffness = [k11 -k11;-k11 k11];    %full local stiffness matrix
    localforce = zeros(nne*dof,1);%external forces are added at the end, so
 leave as zeros. If temp changes, modify for thermal expansion
    localforce = localforce + ((E.*Area*alpha*D_T).*[-n./L n./L])';
    %DONT TOUCH BELOW BLOCK!! Assembles the global stiffness matrix, Generic
 block which works for any element
    for B = 1: nne
        for i = 1: dof
            nK1 = (e(A, B)-1)*dof+i;
            nKe1 = (B-1)*dof+i;
            F(nK1) = F(nK1) + localforce(nKe1);
            for C = 1: nne
                for j = 1: dof
                    nK2 = (e(A, C)-1)*dof+j;
                    nKe2 = (C-1)*dof+j;
                    K(nK1, nK2) = K(nK1, nK2) + localstiffness(nKe1, nKe2);
                end
            end
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

end

K_copy = K;
%%%%%%%%%%%%%%%%%%%%%BOUNDARY CONDITIONS%%%%%%%%%%%%%%%%%%%%%%%%%%

%external forces
P = 100;
F(3) = F(3) + P.*cosd(60);
F(4) = F(4) - P.*sind(60);
%Apply displacement BC by eliminating rows and columns of nodes 3-4
 (corresponding to
%degrees of freedom 5 to 8) - alternative (and more generic method) is the
 penalty approach, or
%static condensation approach - see later class notes

deletedofs = [5:8];
K(deletedofs,:) = [];
K(:,deletedofs) = [];
F(deletedofs,:) = [];

%%%%%%%%%%%%%%%%%%%%%BOUNDARY CONDITIONS END%%%%%%%%%%%%%%%%%%%%%%%%%%


%solve for displacement unknowns (uk)
uk = K\F

%expand u to include deleted displacement bcs
u = ones(Nnodes*dof,1);
u(deletedofs) = 0;
```

```
I = find(u == 1);
u(I) = uk;

u_copy = [uk;zeros(4,1)];
%%%%%%%%%%%%%POSTPROCESSING%%%%%%%%%%%%%%%%%%%%%

%%%Step 6:Postprocess results
for i = 1:Nel

    %get data about truss i
    n = (co(e(i,2),:) - co(e(i,1),:));
    L = norm(n);
    n = n./L;

    Area = e(i,3);
    n1 = e(i,1);n2 = e(i,2);%global numbers for node 1 and 2 of truss i
    d = [u(n1*dof-1) u(n1*dof) u(n2*dof-1) u(n2*dof)]';%displacements of the
 two nodes
    sigma(i) = E*([-n./L n./L]*d) - E*Area*alpha*D_T;%stress formula, If temp
 changes, modify for thermal expansion
end
sigma
% Area = [1E-4,2E-4,1E-4,1E-4];
% sigma.*Area
%
% K_copy(5:8,:)*u_copy
```

```
uk =

   1.0e-03 *

    0.5000
    0.2702
    0.4948
   -0.2298


sigma =

   1.0e+07 *

    3.4996    2.5354    3.4996    3.4630
```

Published with MATLAB® R2022b