LABORATORY ASSIGNMENT

After reviewing Chapter of the lecture notes write a MATLAB program that calculates axial displacements and forces in a bar.

The bar is fixed on the left side, subjected to a continuous load q and a concentrated force P at the right end. Use linear shape functions and the CALFEM instructions *assem* and *solveq*.
Determine the elements of the stiffness matrix and load vector using symbolic operations with *diff* and *int*.
Calculate natural frequencies and mode shapes using the *eigen* procedure.

In the report, submit to the instructor:

- the program code, and
- a brief report containing:
  - plots of displacements, axial force,
  - the first four mode shapes and corresponding frequencies,
  - and a convergence plot of the displacement at the end of the bar as a function of the number of degrees of freedom.

For a selected finite element, use the solution vector from the program to extract the corresponding degrees of freedom and write down the formula for the approximate solution in that element **by hand**.

Make sure the input data and results are realistic.

Algorithm:

1. After declaring ***syms x*,** define the input data: ***q*** (as a function of x), ***E***, ***A***, ***L***, ***P***.
2. Define the number of elements ***Nel*** and compute:
   - the number of degrees of freedom ***Ndof***, and
   - the vector of node coordinates ***coord***.

3. Initialize the global stiffness matrix ***KG***, mass matrix ***MG***, and the global load vector ***fG*** as zero.

4. Insert the value of the concentrated force ***P*** into the appropriate entry of the vector ***fG***.

5. Loop over elements (***iel = 1:Nel***)

   a. Select node coordinates ***x1*** and ***x2***
   b. Define linear shape functions ***f1*** and ***f2***, and compute their derivatives (symbolically)
   c. Form matrices ***B*** and ***N***
   d. Compute the element stiffness matrix ***Kel*** and load vector ***Pel*** (using symbolic integration)
   e. Use the ***assem*** function to assemble:

   [KG, fG] = assem([iel, iel, iel+1], KG, Kel, fG, Pel);

   f. Compute the element mass matrix ***Mel*** (symbolically)
   g. Use assem again to update the mass matrix:

   MG = assem([iel, iel, iel+1], MG, Mel);

6. Create a boundary condition vector ***bc*** of size 1×2: the number of the known DOF (displacement) and its value.

7. Solve the system of equations with the kinematic conditions using ***solveq***:

   u = solveq(KG, fG, bc);

8. Use the ***plot*** command to visualize the displacement approximation.

9. Loop over elements (***i = 1:Nel***):

   a. Select node coordinates ***x1*** and ***x2***
   b. ***Extract*** DOF values from vector u
   c. Compute derivatives of the shape functions
   d. Write the formula for axial force ***S*** in the element
   e. Use ***fplot*** to visualize the force approximation:

   fplot(S, [x1, x2]), hold on

10. Use the ***eigen*** procedure to compute frequencies and mode shapes:

    [La, Egv] = eigen(KG, MG, bc(:,1));

11. Plot the first four mode shapes. Example for the first one:

Freq = sqrt(La)/(2*pi);

figure(3)

plot(coord, Egv(:,1)), grid on

MATLAB Code Skeleton (to copy and complete):

```
% AXIAL LOADING OF A BAR
clear;
syms x

%% INPUT DATA
E =
A =
rho =
L =
q =
P =

%% DISCRETIZATION
Nel =
Ndof = Nel + 1;
coord = linspace(0, L, Ndof);
```

```
%% INITIALIZATION
KG = zeros(Ndof, Ndof);
MG =
fG =

fG(end) =          % Apply concentrated force

%% ELEMENT LOOP
for iel = 1:Nel
  x1 = coord(iel);
  x2 =
  phi1 = (x - x2) / (x1 - x2);
  phi2 =
  N = [phi1, phi2];
  B =
  Kel =
  Pel =
  [KG, fG] = assem([iel, iel, iel+1], KG, Kel, fG, Pel);
  Mel =
  MG =
end

%% SOLVE SYSTEM
bc =
u = solveq(KG, fG, bc);
```

```
figure(1)
plot
xlabel('x'); ylabel('Displacement'); grid on

%% AXIAL FORCE
figure(2), hold on
for iel = 1:Nel
   x1 = coord(iel);
   x2 =
   u1 = u(iel);
   u2 =
   dphi1 = 1 / (x1 - x2);
   dphi2 =
   S =
   fplot(S, [x1, x2]), hold on
end
xlabel('x'); ylabel('Axial Force'); grid on

%% NATURAL FREQUENCIES & MODES
[La, Egv] = eigen(KG, MG, bc(:,1));
Freq = sqrt(La) / (2 * pi);

figure(3)
plot(coord, Egv(:,1)), grid on
xlabel('x'); ylabel('1st Mode Shape')
```