```matlab
% CODE FOR QUADRATIC ELEMENTS %

function Project_01_Quadratic
L=0.5;                                  % Rod Length %
Le=zeros(5,1);                          % Element Size Matrix %
d=zeros(5,1);                           % L2 Error Values Matrix %
e=zeros(5,1);                           % Energy Error Values Matrix %
nn=[11,41,161,641,2565];                % No. of nodes %
x=zeros(5,1);
y1=zeros(5,1);                          % Extra Matrices used for Storage %
y2=zeros(5,1);
for i=1:5
    Le(i)=L/(nn(i)-1);                  % Le = Element Length %
    [d(i),e(i)]=solve_fem(nn(i));       % Solves model for given set of values %
    x(i)=log10(Le(i));
    y1(i)=log10(d(i));
    y2(i)=log10(e(i));
end
plot(x,y1,'ro-');                       % L2 Error Graph Plotting %
xlabel('Element Size');
ylabel('L2 Error');
title('Qudratic Element L2 Error');
figure;
plot(x,y2,'bo-');                       % Energy Error Graph Plotting %
xlabel('Element Size');
ylabel('Energy Error');
title('Quadratic Element Energy Error');

Convergence_L2=(y1(4)-y1(3))/(x(4)-x(3))% Calculates rate of Convergence for ↙
displacement%
Convergence_EE=(y2(4)-y2(3))/(x(4)-x(3))% Calculates rate of Convergence for energy%

end

function [L2,EE]=solve_fem(nn)

% Inputs %
N=nn;                                               % No of Nodes %
L=0.5;                                              % Length of Rod %
ne=(N-1)/2;                                         % No. of Elements %

% Connectivity and Nodes %
conn=[1:2:N-2;2:2:N-1;3:2:N];
nodes=[0:(L/(N-1)):L];
end_nodes=[1:2:N-2];                                % End Nodes Matrix %

% Stiffness Matrix Calculation %
tic;
Ae=(pi/4)*0.01*0.01*70e9;                           % Area * Elastic Modulus %
Le=L/ne;                                            % Le = Element Length %
K=zeros(N);
k=(Ae/(6*Le))*[14,-16,2;-16,32,-16;2,-16,14];

for c = conn
        K(c,c) = K(c,c) + k;
```

```
end
toc;

% Calculation of Body Force on every element %
tic;                                                     % Time Start %
f=zeros(N,1);
for c=conn
    Xe=nodes(c);
    le=Xe(2)-Xe(1);
for q=[-1,1]/sqrt(3)
    n=[0.5*(q-1.0)*q,(1.0-q*q),0.5*(q+1.0)*q];
    x=n*Xe';
    f(c)=f(c)+n'*2000*(sin(10*pi*x))*le;
end
end
f(N)=f(N)-((pi/4)*0.01*0.01*1000000);                    % Applying Traction Condition %
f(1)=0;                                                  % Body force=0 at x=0 %

% Boundary Conditions %
K(1,2)=0;
K(1,3)=0;
K(2,1)=0;
K(3,1)=0;

% Solution %
u=zeros(N,1);
u=K\f;
toc;                                                     % Stop Time %

% Calculation of new node positions %
new_nodes=zeros(N,1);
for i=1:N
    new_nodes(i,1)=nodes(i)+u(i,1);
end
figure;
plot(nodes,u,'.b');                                     % Plotting graph for Displacement ↙
Field %
xlabel('X distance');
ylabel('Displacement U');
title('Displacement Field against X Distance');
legend('Displacement Field 100');

% Elements lengths and strain calculations %

new_length=zeros(ne,1);
for i=1:2:N-2
    new_length((i+1)/2,1)=new_nodes(i+2,1)-new_nodes(i,1); % New Length of Elements %
    strain((i+1)/2,1)=((new_length((i+1)/2,1)-Le)/Le);
end
format long;

figure;
plot(end_nodes,strain,'.b');                  % Plotting Graphs for Strain Variation %
title('Strain Field against X Distance');
legend('Strain Field FEM');
```

```
xlabel('X distance');
ylabel('Strain Value');

% Calculation of L2 and Energy Error Norm %

L2_error=0;
EE_error=0;
E=70e9;
A=(pi/4)*0.01*0.01;
for c=conn
    Xe=nodes(c);
    le=Xe(2)-Xe(1);
    w=[5/9 8/9 5/9];                             % Weight Functions for 3 point Gauss↙
Quadrature %
    mm=1;
    for q=[-1,0,1]/sqrt(3/5)                     % 3 point Gauss Quadrature %
        n=[0.5*(q-1.0)*q,(1.0-q*q),0.5*(q+1.0)*q];% Shape Function %
        B=(2/le)*[(2*q)-1,-2*q,(2*q)+1];         % Derivative of the Shape Function %
        x=n*Xe';
        u_h=n*u(c);
        strain_h=B*u(c);
        exact_u=((20*sin(10*pi*x))/(E*A*(pi^2))+(((200/(pi*A*E))-(1e6/E))*x));
        exact_strain=(((200*cos(10*pi*x))/(E*A*pi))+((200/(pi*A*E))-(1e6/E)));
        L2_error = L2_error + ((exact_u-u_h)^2)*(le/2)*w(mm);
        EE_error = EE_error + (0.5*70e9*(exact_strain-strain_h)^2)*(2/le)*w(mm);
        mm=mm+1;
    end
end

L2=sqrt(L2_error);                    % Final Value of L2_Error %
EE=sqrt(EE_error);                    % Final Value of Energy Error %

% Graph for Plotting of Solution %
uu=zeros(5,1);
ee=zeros(5,1);
i=1;
for c=conn
    % for Linear Elements, c is 2 by 1, for quadratic, c is 3 by 1 %
    xe = nodes(c);
    for p=linspace(-1,1,10)
        shape=[0.5*(p-1.0)*p,(1.0-p*p),0.5*(p+1.0)*p];
        B=(2/le)*[(2*q)-1,-2*q,(2*q)+1];
        uu(i)=shape*u(c);                 % Displacement Values %
        ee(i)=B*u(c);                     % Strain Values %
        i=i+1;
    end
end
figure;
plot(p,uu,'ro');
figure;
plot(p,ee,'bo');
end
```