```
function [M, K, Fqz] = assemble_bar_fem(N, L, E, A, rho, qz)
%
% Function to assemble FEM matrices of 1D tension/torsion problem
% using a uniform mesh.
%
% Synopsis:
%     [M, K, F]  =   assemble_bar_fem(N, L, E, A, rho, qz)
%
% Input:
%     N          =   Number of Nodes (number of elements is N-1)
%     L          =   Length of bar
%     E          =   Young's (tension) or Shear Modulus (torsion)
%     A          =   X-section Area (tension) or polar moment (torsion)
%     rho        =   Mass density per unit length (tension)
%                    or centrodial moment of inertia (torsion) per unit length
%     qz         =   Function handle to distributed force per unit length
%                    or torque per unit length qz(z) as @(z)()
%
% Output:
%     M          =   NxN lumped global mass matrix
%     K          =   NxN global stiffness matrix
%     Fqz        =   Nx1 global force vector
%
%
% By: Ryan S. Elliott -- Jan. 2015
%

N = 5; % number of nodes according to hw4 prob5
L = 1.6; % meter
rho = 1;
E = 70e9;
A = 2e-4;
M = zeros(N,N);
K = zeros(N,N);
Fqz = zeros(N,1);
qz = @(z) 100*(1 - (2*z/L -1)^2);
% Use evenly spaced nodes for N-1 identical elements

% length of each element
len = L/(N-1);

% local lumped mass matrix (same for all elements)
m = (len*rho/2.0) * [[1.0, 0.0];[0.0, 1.0]];

% local stiffness matrix (same for all elements)
k = (E*A/len) * [[1.0, -1.0];[-1.0, 1.0]];

for i = 1:(N-1)
  % compute elment i local force vector using 5 point Gaussian Quadrature
  ff = quadrature(i, len, qz);

  % Set global connectivity for element
```

```
  G1 = i;
  G2 = i+1;

  % Define Range variable for element
  Range = [G1, G2];

  % add element i contribution to global mass matrix
  M(Range, Range) = M(Range, Range) + m;

  % add element i contribution to global stiffness matrix
  K(Range, Range) = K(Range, Range) + k;

  % add element i contribution to global force vector
  Fqz(Range) = Fqz(Range) + ff;
end;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
function ff = quadrature(i, len, f)
%   function to integrate using 5 point GQ the force vector on element

% shape functions
N1 = @(s)((1.0-s)/2.0);
N2 = @(s)((s+1.0)/2.0);

% local nodal locations
lz(1) = (i-1)*len;
lz(2) = (i)*len;

% Mapping from master element coordinate (s) to global coordinate (z)
z = @(s)((lz(1)+lz(2))/2.0 + s*(lz(2)-lz(1))/2.0);
% Jacobian of mapping
jac = (lz(2)-lz(1))/2.0;

% GQ points
s(1) = -1.0;
s(2) = -sqrt(3.0/7.0);
s(3) = 0.0;
s(4) = sqrt(3.0/7.0);
s(5) = 1.0;
% GQ weights
w(1) = 1.0/10.0;
w(2) = 49.0/90.0;
w(3) = 32.0/45.0;
w(4) = 49.0/90.0;
w(5) = 1.0/10.0;
% compute function values at quadrature points
gq(1) = z(s(1));
gq(2) = z(s(2));
gq(3) = z(s(3));
gq(4) = z(s(4));
```

```
gq(5) = z(s(5));

ff = zeros(2,1);
for i = 1:5
  ff(1) = ff(1) + jac*w(i)*f(gq(i))*N1(s(i));
  ff(2) = ff(2) + jac*w(i)*f(gq(i))*N2(s(i));
end;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
```

```
ans =

    0.2000         0         0         0         0
         0    0.4000         0         0         0
         0         0    0.4000         0         0
         0         0         0    0.4000         0
         0         0         0         0    0.2000
```

Published with MATLAB® R2023b