```
function[M, K, lambda] = MIE597VP2_NgoNumerical(n, L, P)
    %Richie Ngo MIE 597V 27413591 Project 2
    %This function is supposed to solve Project 2 for the cantilever
 beam
    %using the given information and through the Galerkin method. This
 is
    %the numerical method.
    E = 195*10^9;       %Young's Modulus (Pa)
    d = .01;            %Diameter (m)
    I = pi*(d/2)^4/2;   %Area moment of inertia (m^4)
    cross_A = pi*(d/2)^2;       %Cross-sectional area (m^2)
    rho = 8000;         %Density (kg/m^3)
    m = rho*cross_A*L;  %Mass (kg)
    zeta = .01;
    beta = [1.875 4.694 7.855 10.996 14.137]';  %Beta for cantilever
 and n <= 5
    if n > 5
        for j = 6:n
            beta(j) = (2*j - 1)*pi/(2*L);   %Beta for cantilever and n
 > 5
        end
    end
    A = zeros(n);
    B = A;
    G = A;
    for i = 1:n
        for j = 1:n
            sigma = (sinh(beta(j)*L) - sin(beta(j)*L))/...  %Sigma
                (cosh(beta(j)*L) + cos(beta(j)*L));
            phi_phi = @(x, beta1, beta2, sigma) (cosh(beta1*x) -
 cos(beta1*x) -...  %Phi*phi
                sigma*(sinh(beta1*x) - sin(beta1*x))).*...
                (cosh(beta2*x) - cos(beta2*x) -...
                sigma*(sinh(beta2*x) - sin(beta2*x)));
            phi_phi_2 =@(x, beta1, beta2, sigma) (cosh(beta1*x) -
 cos(beta1*x) -...  %Phi*phi^(2)
                sigma*(sinh(beta1*x) - sin(beta1*x))).*...
                (cosh(beta2*x) + cos(beta2*x) -...
                sigma*(sinh(beta2*x) + sin(beta2*x)))*beta2^2;
            phi_phi_4 =@(x, beta1, beta2, sigma) (cosh(beta1*x) -
 cos(beta1*x) -...  %Phi*phi^(4)
                sigma*(sinh(beta1*x) - sin(beta1*x))).*...
                (cosh(beta2*x) - cos(beta2*x) -...
                sigma*(sin(beta2*x) - sinh(beta2*x)))*beta2^4;
            %Integrates to find matrices
            A(i,j) = integral(@(x)phi_phi_4(x, beta(i), beta(j),
 sigma), 0, L);
            B(i,j) = integral(@(x)phi_phi_2(x, beta(i), beta(j),
 sigma), 0, L);
            G(i,j) = integral(@(x)phi_phi(x, beta(i), beta(j), sigma),
 0, L);
        end
    end
    K = E*I*A + P*B;    %Stiffness matrix
    M = m*G;            %Mass matrix
    D = inv(M)*K;
    lambda = sqrt(eig(D));
end
```

```
M =

    0.6283    0.0374
    3.0396    0.6283


K =

   1.0e+06 *

    0.0052    2.4106
    0.0257   -1.0086


lambda =

   1.0e+03 *

   0.0918 + 0.0000i
   0.0000 + 5.3223i
```

Published with MATLAB® R2015b