```
%% Problem 2
close all;
clear all;
clc;
%% While Loop
% Kmax = 0;
% Tmax = 0;
% while Tmax<30 % Tmax<30 or K<50
% Ph = randi([-10,10],3,4)
%% Problem 2a K>50
% Ph = [-8,-7,-5,0;-6,-10,1,1;-7,3,4,-1];

%% Problem 2a T>30
% Ph = [8,6,4,1;3,6,4,8;7,3,-4,-9];

%% User Input
fprintf('Input the P0, P1, P0dot_bar, P1dot_bar as [#;#;#] for each when prompt and this function will plot the Hermite Curve.\n')
fprintf('or\n')
fprintf('Input the Ph as a 3x4 matrix when prompt and this function will plot the Hermite Curve.\n\n')
fprintf('Do you want to input as vectors (type "1") or matrix (type "2").\n')
type = input('Input Choice Here ("1" or "2" only):');
typeif = num2str(type);

if strcmp(typeif,'1')
    fprintf('\nType in a 3x1 matrix for each positon or tangent vector.\n')
    P0_bar = input('P0_bar (3by1 matrix) as [#;#;#]:');            % 3by1 matrix
    P1_bar = input('P1_bar (3by1 matrix) as [#;#;#]:');            % 3by1 matrix
    P0dot_bar = input('P0dot_bar (3by1 matrix) as [#;#;#]:');      % 3by1 matrix
    P1dot_bar = input('P1dot_bar (3by1 matrix) as [#;#;#]:');      % 3by1 matrix
        Ph = [P0_bar P1_bar P0dot_bar P1dot_bar];
end
if strcmp(typeif,'2')
    fprintf('\nType in a 3x4 matrix.\n')
    Ph = input('Ph :');
        P0_bar = Ph(:,1);       % 3by1 matrix
        P1_bar = Ph(:,2);       % 3by1 matrix
        P0dot_bar = Ph(:,3);    % 3by1 matrix
        P1dot_bar = Ph(:,4);    % 3by1 matrix
end

%% Output
P0_bar = Ph(:,1);       % 3by1 matrix
P1_bar = Ph(:,2);      % 3by1 matrix
P0dot_bar = Ph(:,3);   % 3by1 matrix
P1dot_bar = Ph(:,4); % 3by1 matrix
% Hermite Curve Define in term of u
syms u
Mh = [2 -3 0 1;...
      -2 3 0 0;...
```

```
      1 -2 1 0;...
      1 -1 0 0];
U = [u^3; u^2; u; 1];
Bh = Mh*U;              % Hermite Curve
pu = Ph*Bh;             % P(u) function base off Hermite Curve

% Constant K (Curvature) equation in term of u
pt = pu;
pt_t = diff(pt,u);
pt_tt = diff(pt_t,u);
pt_ttt = diff(pt_tt,u);
K = norm(cross(pt_t,pt_tt))/norm(pt_t)^3;
% Constant T (Torsion) equation in term of u
T = dot(pt_t,cross(pt_tt,pt_ttt))/norm(cross(pt_t,pt_tt))^2;
%%%%%%%%%%%%
t = 0:.01:1; % linspace(0,1,100);
% v = linspace(-10,10,100);
% [t,t] = meshgrid(t,t);
pusub = zeros(3,length(t));
for i = 1:length(t)
pusub(:,i) = subs(pu, u, t(i));
end
x = pusub(1,:);
y = pusub(2,:);
z = pusub(3,:);

Kvalue = zeros(length(t));
Tvalue = zeros(length(t));

%% Finding max Curvature K and Torsion T
for i = 1:length(t)
    Kvalue(i) = subs(K,u,t(i));
    Tvalue(i) = subs(T,u,t(i));
% Positon for K value
end
Kvalue = abs(Kvalue);
Tvalue = abs(Tvalue);
[Kmax, Klocation] = max(Kvalue);
[Tmax, Tlocation] = max(Tvalue);

% Max K and T.
Kmax = Kmax(1);
Klocation_at_u = t(Klocation(1));
Tmax = Tmax(1);
Tlocation_at_u = t(Tlocation(1));
% end

% K and T location
maxKlocation = subs(pu, u, Klocation_at_u);
maxKlocation = double(maxKlocation);
maxTlocation = subs(pu, u, Tlocation_at_u);
```

```
maxTlocation = double(maxTlocation);

%% Plot

Hermite = plot3(x,y,z);
grid on
hold all

% Tangent Lines
P0dot_bar_mag = norm(P0dot_bar);
P1dot_bar_mag = norm(P1dot_bar);
P0dot_bar_normalize = P0dot_bar/P0dot_bar_mag;
P1dot_bar_normalize = P1dot_bar/P1dot_bar_mag;
tangentP0 = quiver3(P0_bar(1),P0_bar(2),P0_bar(3),P0dot_bar_normalize(1),↙
P0dot_bar_normalize(2),P0dot_bar_normalize(3),'r');
tangentP1 = quiver3(P1_bar(1),P1_bar(2),P1_bar(3),P1dot_bar_normalize(1),↙
P1dot_bar_normalize(2),P1dot_bar_normalize(3),'g');

%% P0 and P1 Plot
p0pt = scatter3(P0_bar(1),P0_bar(2),P0_bar(3),'*r');
p1pt = scatter3(P1_bar(1),P1_bar(2),P1_bar(3),'*g');
Kmaxplot = scatter3(maxKlocation(1,1),maxKlocation(2,1),maxKlocation(3,1),'c','filled');
Tmaxplot = scatter3(maxTlocation(1,1),maxTlocation(2,1),maxTlocation(3,1),'r','filled');

title('Hermite Plot')
xlabel('x'); ylabel('y'); zlabel('z')
legend([Hermite,p0pt,p1pt,tangentP0,tangentP1,Kmaxplot,Tmaxplot],'Hermite↙
Curve','P_0','P_1','P_0 Tangent','P_1 Tangent','Max K','Max T');
% axis equal

fprintf('\nThis program output are Ph and numerical values upon calling. Then plot the↙
Hermite Curve.\n\n')
fprintf('Ph =\n'); disp(Ph)
fprintf('Max Curvature:'); disp(Kmax)
fprintf('Max Curvature location u ='); disp(Klocation_at_u)
fprintf('Max Curvature x,y,z position =\n'); disp(maxKlocation)

fprintf('Max Torsion:'); disp(Tmax)
fprintf('Max Torsion location u ='); disp(Tlocation_at_u)
fprintf('Max Torsion x,y,z position =\n'); disp(maxTlocation)

fprintf('Check graph in plot.\n');
fprintf('For Numerical Values of type in desire values base off WorkSpace:\n')
```