```
% Jacob Sayono
% 505368811

% MAE C163B
% Project 2

close; clear; clc;

% lengths (m)
a1 = 0.325;
a2 = 0.225;
d1 = 0.416;
d4 = 0.093;

% key positions
home = [0, -1, 0, .325;
        1, 0, 0, .225;
        0, 0, 1, .203;
        0, 0, 0, 1];
feeder = [0, -1, 0, .325;
          1, 0, 0, .225;
          0, 0, 1, .180;
          0, 0, 0, 1];
goal1 = [1, 0, 0, .280;
         0, 1, 0, .240;
         0, 0, 1, .180;
         0, 0, 0, 1];
goal2 = [0, -1, 0, .280;
         1, 0, 0, .330;
         0, 0, 1, .180;
         0, 0, 0, 1];
goal3 = [-1, 0, 0, .370;
         -1, 0, 0, .240;
          0, 0, 1, .180;
          0, 0, 0, 1];
goal4 = [0, 1, 0, .370;
        -1, 0, 0, .240;
         0, 0, 1, .180;
         0, 0, 0, 1];

% via points
a = [0, -1, 0, .325;
     1, 0, 0, .225;
     0, 0, 1, .190;
     0, 0, 0, 1];
b = [1, 0, 0, .280;
     0, 1, 0, .240;
     0, 0, 1, .190;
     0, 0, 0, 1];
c = [0, -1, 0, .280;
     1, 0, 0, .330;
     0, 0, 1, .190;
     0, 0, 0, 1];
d = [-1, 0, 0, .370;
     0, -1, 0, .330;
     0, 0, 1, .190;
     0, 0, 0, 1];
e = [0, 1, 0, .370;
     -1, 0, 0, .240;
     0, 0, 1, .190;
     0, 0, 0, 1];

% forward kinematics
T = FK(0, 0, .100, 0)

% inverse kinematics
[t1, t2, d3, t4] = IK(a)
```

```matlab
% visualize robot
L1 = Revolute('alpha', 0, 'a', 0, 'd', d1, 'qlim', [-170 170]*pi/180);
L2 = Revolute('alpha', 0, 'a', a1, 'd', 0, 'qlim', [-145 145]*pi/180);
L3 = Prismatic('alpha', pi, 'a', a2, 'theta', 0, 'qlim', [0 .15]);
L4 = Revolute('alpha', 0, 'a', 0, 'd', d4, 'qlim', [-360 360]*pi/180);
tool = transl(0, 0, 0);
PCB_Builder = SerialLink([L1 L2 L3 L4], 'name', 'PCB_Builder', 'tool', tool);

% convert waypoints into joint space values and interpolate

% forward kinematics function
function T = FK(theta1, theta2, d3, theta4)
    a1 = 0.325;
    a2 = 0.225;
    d1 = 0.416;
    d4 = 0.093;

    T01 = mat_from_DH(0, 0, d1, theta1);
    T12 = mat_from_DH(0, a1, 0, theta2 + 90);
    T23 = mat_from_DH(0, a2, -d3, 0);
    T34 = mat_from_DH(0, 0, -d4, theta4);

    T = T01*T12*T23*T34;
end

% matrix function given DH parameters
function matrix = mat_from_DH(alpha_iminus1, a_iminus1, d_i, theta_i)
    matrix = [cosd(theta_i) -sind(theta_i) 0 a_iminus1;
              sind(theta_i)*cosd(alpha_iminus1) cosd(theta_i)*cosd(alpha_iminus1) -sind(alpha_iminus1) -sind(alpha_iminus1)*d_i;
              sind(theta_i)*sind(alpha_iminus1) cosd(theta_i)*sind(alpha_iminus1) cosd(alpha_iminus1) cosd(alpha_iminus1)*d_i;
              0 0 0 1];
end

% inverse kinematics function
function [theta1, theta2, d3, theta4] = IK(T)
    [x, y, z, theta] = mat_to_pos(T);

    a1 = 0.325;
    a2 = 0.225;
    d3_min = 0;
    d3_max = 0.150;
    d3_val = .323-z;

    c_theta_2 = (x^2+y^2-a1^2-a2^2)/(2*a1*a2);
    s_theta_2_pos = sqrt(1-c_theta_2^2);
    s_theta_2_neg = -sqrt(1-c_theta_2^2);

    theta_2_val1 = atan2(s_theta_2_pos,c_theta_2);
    theta_2_val2 = atan2(s_theta_2_neg,c_theta_2);

    L3_1 = a1+cos(theta_2_val1)*a2;
    L3_2 = a1+cos(theta_2_val2)*a2;
    L4_1 = sin(theta_2_val1)*a2;
    L4_2 = sin(theta_2_val2)*a2;

    theta_1_val1 = atan2(y,x) - atan2(L4_1,L3_1);
    theta_1_val2 = atan2(y,x) - atan2(L4_2,L3_2);

    theta_4_val1 = theta - pi/4 - theta_1_val1 - theta_2_val1;
    theta_4_val2 = theta - pi/4 - theta_1_val2 - theta_2_val2;

    if (d3_val < d3_min || d3_val > d3_max)
        d3 = -1;
    else
        d3 = d3_val;
    end

    if (check_angles(theta_1_val1, theta_2_val1, theta_4_val1) == 1)
```

```
        theta1 = theta_1_val1;
        theta2 = theta_2_val1;
        theta4 = theta_4_val1;
        return
    elseif (check_angles(theta_1_val2, theta_2_val2, theta_4_val2) == 1)
        theta1 = theta_1_val2;
        theta2 = theta_2_val2;
        theta4 = theta_4_val2;
        return
    else
        theta1 = -1;
        theta2 = -1;
        theta4 = -1;
        return
    end
end

% check angles function
function check = check_angles(t1_test, t2_test, t4_test)
    t1_min = -170;
    t1_max = 170;
    t2_min = -145;
    t2_max = 145;
    t4_min = -360;
    t4_max = 360;

    check = -1;
    if (t1_test < t1_min || t1_test > t1_max)
        return
    elseif (t2_test < t2_min || t2_test > t2_max)
        return
    elseif (t4_test < t4_min || t4_test > t4_max)
        return
    else
        check = 1;
        return
    end
end

% euler angles function
function [x, y, z, theta] = mat_to_pos(T)
    x = T(1,4);
    y = T(2,4);
    z = T(3,4);
    eul_angles = rotm2eul(T(1:3,1:3));
    theta = eul_angles(1);
end
```

```
T =

         0   -1.0000         0    0.3250
    1.0000         0         0    0.2250
         0         0    1.0000    0.2230
         0         0         0    1.0000
```

```
Undefined function 'rotm2eul' for input arguments of type 'double'.

Error in proj2>mat_to_pos (line 181)
    eul_angles = rotm2eul(T(1:3,1:3));

Error in proj2>IK (line 104)
    [x, y, z, theta] = mat_to_pos(T);

Error in proj2 (line 67)
[t1, t2, d3, t4] = IK(a)
```