```
function ret=iPUMA(nx, ny, nz, ox, oy, oz, ax, ay, az, Px, Py, Pz)
%input: end-effector position
%ouput: joint angles at each frames

%initial condition: iPUMA(1,0,0,0,1,0,0,0,1,411.50,139.70,1160.10)
format compact
format short
%DH parameter
A =  [-90 0 90 -90 90 0] ; %twist angle
r = [0 431.80 -20.32 0 0 0]; %offset as to xn
d = [671.83 139.70 0 431.80 0 56.50]; %offset as to z(n-1)
%DH model
T0_6 = [nx ox ax Px; ny oy ay Py; nz oz az Pz; 0 0 0 1];
% Joint 5 position
P = [Px-56.50*ax; Py-56.50*ay; Pz-56.50*az];

%Determining Joint angles for frames 1,2,and 3
C1 = sqrt(P(1)^2+P(2)^2);
C2 = P(3)-d(1);
C3 = sqrt(C1^2+C2^2);
C4 = sqrt(r(3)^2+d(4)^2);
D1 = d(2)/C1;
D2 = (C3^2+r(2)^2-C4^2)/(2*r(2)*C3);
D3 = (r(2)^2+C4^2-C3^2)/(2*r(2)*C4);

a1 = atan2d(D1,sqrt(abs(1-D1^2)));
a2 = atan2d(sqrt(abs(1-D2^2)),D2);
b = atan2d(sqrt(abs(1-D3^2)),D3);
p1 = atan2d(P(2),P(1));
p2 = atan2d(C2,C1);
%Joint angles: theta_1, theta_2, and theta_3
J = [p1-a1 round(a2-p2) round(b-90)];

%Apply forward kinematics at first three joints
T = [];
for n = 1:3
    matT = [cosd(J(n)) -sind(J(n))*cosd(A(n)) ...
            sind(J(n))*sind(A(n)) r(n)*cosd(J(n));
            sind(J(n)) cosd(J(n))*cosd(A(n)) ...
            -cosd(J(n))*sind(A(n)) r(n)*sind(J(n));
            0 sind(A(n)) cosd(A(n)) d(n);
            0 0 0 1];
        T = [T; {matT}];
end
T0_3 = T{1}*T{2}*T{3};
T3_6 = inv(T0_3)*T0_6;

%Joint angle: theta_4, theta_5, and theta_6
J4 = round(atan2d(T3_6(2,3),T3_6(1,3)));
J5 = round(atan2d(sqrt(abs(1-T3_6(3,3)^2)),T3_6(3,3)));
```

```
J6 = atan2d(T3_6(3,2),-T3_6(3,1));

J = [J J4 J5 J6];
%Plotting the result
if J(1,1) >= -160 && J(1,1) <= 160 && J(1,2)>= -225 ...
        && J(1,2) <= 45 && J(1,3) >= -45 && J(1,3) <= 225 ...
        && J(1,4) >= -110 && J(1,4) <= 170 && J(1,5) >= -100 ...
        && J(1,5) <= 100 && J(1,6) >= -266 && J(1,6) <= 266
    T = [];
    for n = 1:6
        matT = [cosd(J(n)) -sind(J(n))*cosd(A(n)) ...
            sind(J(n))*sind(A(n)) r(n)*cosd(J(n));
            sind(J(n)) cosd(J(n))*cosd(A(n)) ...
            -cosd(J(n))*sind(A(n)) r(n)*sind(J(n));
            0 sind(A(n)) cosd(A(n)) d(n);
            0 0 0 1];
        T = [T; {matT}];
     ret=1;
    end

else
    ret=-1;
end
```