```
function config = RRR_inverse_2D(Link_Lengths, pose_end)  % question (b)
x = pose_end(1); % assign x
y = pose_end(2); % assign y
phi = pose_end(3); % assign phi
l1 = Link_Lengths(1); % assign link 1
l2 = Link_Lengths(2); % assign link 2
l3 = Link_Lengths(3); % assign link 3
% calculate cosine of theta2 using I.P.K
cosTheta2 = (x^2+y^2+l3^2-2*l3*(x*cos(phi)+y*sin(phi))-l1^2-l2^2)/(2*l1*l2);
if (-1 <= cosTheta2)&&(cosTheta2 <= 1)  % check if point is accessible
    % calculate sine of theta2 for the first solution using I.P.K
    sinTheta2_a = sqrt(1-cosTheta2^2);
    % calculate sine of theta2 for the second solution using I.P.K
    sinTheta2_b = -1*sinTheta2_a;
    theta2_a = atan2(sinTheta2_a, cosTheta2); % calculate theta 2 for the first solution
    theta2_b = atan2(sinTheta2_b, cosTheta2); % calculate theta 2 for the second solution
    % calculate theta 1 for the first solution
    theta1_a = atan2(y-l3*sin(phi),x-l3*cos(phi)) - atan2(l2*sinTheta2_a, l1+l2*cosTheta2);
    % calculate theta 1 for the second solution
    theta1_b = atan2(y-l3*sin(phi),x-l3*cos(phi)) - atan2(l2*sinTheta2_b, l1+l2*cosTheta2);
    % calculate theta 3 for the first solution
    theta3_a = phi - theta1_a - theta2_a;
    % calculate theta 3 for the second solution
    theta3_b = phi - theta1_b - theta2_b;
    % assign the solution 1 to the config matrix (1st column vector)
    config(:,1) = [theta1_a; theta2_a; theta3_a];
    % assign the solution 2 to the config matrix (2nd column vector)
    config(:,2) = [theta1_b; theta2_b; theta3_b];
else
    config = "point cannot be reached!";

end
end
```

```
Not enough input arguments.

Error in RRR_inverse_2D (line 2)
x = pose_end(1); % assign x
```

Published with MATLAB® R2018b