Question 3

```
function main
   clear all;
   close all;
   clc;

   n = 200;
   rng(0); % seed random number generator
   x = rand(n,1);
   z = zeros(n,1);
   k = n*0.4;
   rp = randperm(n);
   outlier_subset = rp(1:k);
   z(outlier_subset)=1; % outliers
   y = (1-z).*(10*x + 5 + randn(n,1)) + z.*(20 - 20*x + 10*randn(n,1));
   % plot data and true line
   scatter(x,y,'b')
   hold on
   t = 0:0.01:1;
   plot(t,10*t+5,'k')
   % add your code for ordinary least squares below
   [w_ols, b_ols] = OLS(x, y);
   plot(t, w_ols*t + b_ols, 'g--');
   % add your code for the robust regression MM algorithm below
   [w_rob, b_rob] = roblr(x, y);

   plot(t, w_rob*t + b_rob, 'r:')
   legend('data','true line','least squares','robust')
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
function [w,b] = wls(xtr, ytr,c)
   %Function to implement weighted linear regression
   %xtr is the input feature vector, ytr is the output vector and c is the
   % vector of weights
   xtr = [ones(size(xtr, 1),1) xtr];
   theta = (xtr' * diag(c) * xtr) \ (xtr' * diag(c) * ytr);
   b = theta(1);
   w = theta(2:end);
end

function [ w_rob, b_rob ] = roblr( x, y )
%Function to implement robust linear regression
%   x is the input feature vector and y is the output vector
```

```
    [n, p] = size(x);
    xtr = [ones(n, 1) x];
    theta = zeros(p+1, 1);
    for iter = 1:50
        r = y - xtr * theta;
        repeat_term = (1+r.^2).^0.5;
        rho_old = repeat_term - 1;
        obj_old = sum(rho_old)/n;
        weight_vector = r./(repeat_term);
        [w_new, b_new] = wls(x, y, weight_vector);
        theta = [b_new w_new]';
        r_new = y - xtr *theta;
        repeat_term_new = (1+r_new.^2).^0.5;
        obj_new = sum(repeat_term_new - 1)/n;
              if abs(obj_new - obj_old) < 1e-6
           break
        end
    end
    w_rob = w_new;
    b_rob = b_new;
end

function [ w, b ] = OLS( xtr, ytr)
    %implements ordinary linear regression here
    %xtr is the input feature vector and ytr is the input feature vector
    xtr = [ones(size(xtr, 1),1) xtr];
    theta = (xtr' * xtr) \ (xtr' * ytr);
    b = theta(1);
    w = theta(2:end);
end
```