```matlab
function J = computeCostMulti(X, y, theta)
%COMPUTECOSTMULTI Compute cost for linear regression with multiple variables
%   J = COMPUTECOSTMULTI(X, y, theta) computes the cost of using theta as the
%   parameter for linear regression to fit the data points in X and y

% Initialize some useful values
m = length(y); % number of training examples

% You need to return the following variables correctly
J = 0;

% ====================== YOUR CODE HERE ======================
% Instructions: Compute the cost of a particular choice of theta
%               You should set J to the cost.

h=X*theta; % funcion de hipotesis
e=h.-y; % error de cada prediccion
e_cuadrado=e.^2; % cuadrado de los errores
suma_e_cuadrado=sum(e_cuadrado); % suma de los cuadrados

J=1/(2*m)*suma_e_cuadrado % Cost Function

% =========================================================================

function [theta, J_history] = gradientDescentMulti(X, y, theta, alpha, num_iters)
%GRADIENTDESCENTMULTI Performs gradient descent to learn theta
%   theta = GRADIENTDESCENTMULTI(x, y, theta, alpha, num_iters) updates theta by
%   taking num_iters gradient steps with learning rate alpha

% Initialize some useful values
m = length(y); % number of training examples
J_history = zeros(num_iters, 1);

for iteration = 1:num_iters
    % Perform a single gradient step on the parameter vector theta.

    % we minimize the value of J(theta) by changing the values of the
    % vector theta NOT changing X or y

    % alpha = learning rate as a single number

    % hypothesis = mx1 column vector
    % X = mxn matrix
    % theta = nx1 column vector
    hypothesis = X * theta;

    % errors = mx1 column vector
    % y = mx1 column vector
    errors = hypothesis .- y;

    newDecrement = (alpha * (1/m) * errors' * X);

    theta = theta - newDecrement';

    % Save the cost J in every iteration
    J_history(iteration) = computeCostMulti(X, y, theta);

end

end
```

```matlab
function [theta] = normalEqn(X, y)
%NORMALEQN Computes the closed-form solution to linear regression
%   NORMALEQN(X,y) computes the closed-form solution to linear
%   regression using the normal equations.

theta = zeros(size(X, 2), 1);

% ====================== YOUR CODE HERE ======================
% Instructions: Complete the code to compute the closed form solution
%               to linear regression and put the result in theta.
%

% ---------------------- Sample Solution ----------------------

theta = pinv(X' * X) * X' * y;


% -------------------------------------------------------------


% ============================================================

end
```