Q3

Part (a):

Iterations: 3301,

Optimal solution = [-0.0067; 0.0557; -0.0525; -0.1147; 0.1255]

Part (b):

Iterations: 3732,

Optimal solution = [-0.0056; 0.0452; -0.0365; -0.1090; 0.1119]

Part (c):

Iterations: 1271,

Optimal solution = [-0.0067; 0.0554; -0.0522; -0.1146; 0.1252]

Main code:

```
%% Initialize
A = hilb(5);
x = [1;2;3;4;5];
epsilon = 1e-4;

%% Part a
[x_opt1, val_opt1, iter1] = gm_backtrack(A, x, 1, 0.5, 0.5, epsilon);

%% Part b
[x_opt2, val_opt2, iter2] = gm_backtrack(A, x, 1, 0.1, 0.5, epsilon);

%% Part c
[x_opt3, val_opt3, iter3] = gm_exact(A, x, epsilon);
```

gm_backtrack:

```
function [x_opt, val_opt, iter] = gm_backtrack(A, x_init, s, alpha, beta, epsilon)

    x = x_init;
    f = x.'*A*x;
    grad = 2*A*x;
    iter=0;

    while (norm(grad)>epsilon)
        iter=iter+1;
        d = -grad;
        t=s;
            while (f - ((x + t*d).'*(A)*(x + t*d)) < -alpha*t*grad.'*d)
                t=beta*t;
            end
        x = x + t*d; % update solution
        f = x.'*A*x; % new value
        grad = 2*A*x; % new gradient
        fprintf("Iteration: %3d, Value: %2.6f, Gradient Norm: %2.6f \n", iter, f,
norm(grad));
```

```
    end

    x_opt = x;
    val_opt = f;

end
```

gm_exact:

```
function [x_opt, val_opt, iter] = gm_exact(A, x_init, epsilon)

    % f = xT A x, grad = 2 Ax

    x = x_init;
    grad = 2*A*x;
    iter = 0;

    while (norm(grad) > epsilon)
        iter = iter + 1;
        d = -grad/norm(grad); % compute optimal direction
        t = -(d.'*grad)/(2*d.'*A*d); % compute optimal stepsize
        x = x + t*d; % update solution

        grad = 2*A*x; % new gradient
        f = x.'*A*x; % new value
        fprintf("Iteration: %3d, Value: %2.6f, Gradient Norm: %2.6f \n", iter, f,
norm(grad));
    end

    x_opt = x;
    val_opt = f;

end
```