Problem A1:

a) Gerschgorin's here tells us that eigenvalues of A must either satisfy |lamda – 2| <= 1 (the sum of the absolute-values of the off-diagonals in the first/last row is 1 and the diagonal is 2), |lambda – 2| <= 2 (the sum of the absolute-values of the off-diagonals in the rest of the rows is 2). So the eigenvalues are either in the interval [1, 3] or [0, 4], as our matrix is real and symmetric it must have only real eigenvalues.

b)

Editor - C:\Users\Nwhybra\Desktop\UW AMATH Masters\AMATH 584\HW\HW8\power_iteration.m

pure_qr_no_shifts.m | test.m | RandOrthMat.m | power_iteration.m | inverse_iteration.m | +

```
function max_eig = power_iteration(A, stop)
    m = size(A, 1);
    v = rand(m, 1);
    v = v / norm(v, 2);

    error = inf;
    while error > stop
       w = A * v;
       v = w / norm(w, 2);
       max_eig = v.' * A * v;
       error = norm(A*v - max_eig*v, 2);
    end
end
```

1x1 double

	1
1	3.9190

c)

Editor - C:\Users\Nwhybra\Desktop\UW AMATH Masters\AMATH 584\HW\HW8\pure_qr_no_shifts.m

pure_qr_no_shifts.m | test.m | RandOrthMat.m | power_iteration.m | inverse_iteration.m

```matlab
function [B, Q] = pure_qr_no_shifts(A, stop)
    B = A;
    m = size(A, 1);
    % A matrix with 1's on the off-diagonals and 0's on the diagonals.
    % Used to conveniently grab the matrix entries I want to compute the
    % error.
    off_diagonal_mask = logical(ones(m, m) - eye(m));

    error = inf;
    while error > stop
        [Q, R] = qr(B);
        B = R * Q;
        % Here the error is the maximum of the off-diagonal entries.
        error = sum(max(abs(B(off_diagonal_mask))));
    end
end
```

10x10 double

	1	2	3	4	5	6	7	8	9	10
1	3.9190	-9.5441e-13	2.5982e-16	-1.8796e-16	-2.8892e-16	-1.3452e-16	-2.5568e-16	-2.5253e-17	5.7313e-17	1.8443e-17
2	-9.5467e-13	3.6825	1.2341e-16	1.5202e-16	-1.4776e-16	-2.1867e-16	7.2846e-17	-1.3462e-16	-8.3503e-17	1.9793e-16
3	0	-1.2700e-20	3.3097	1.3249e-16	1.5238e-16	1.5350e-16	5.3961e-18	1.5278e-16	-2.8109e-17	-4.7965e-16
4	0	0	-1.1176e-29	2.8308	2.7220e-16	2.9658e-16	1.3209e-16	-1.7298e-16	-1.7246e-16	1.9614e-16
5	0	0	0	-1.8971e-40	2.2846	5.9651e-17	2.5885e-16	3.2062e-16	3.1835e-16	1.4064e-16
6	0	0	0	0	-5.8826e-54	1.7154	-5.9967e-17	2.4485e-16	1.4407e-16	1.3595e-17
7	0	0	0	0	0	-3.7753e-72	1.1692	1.5039e-17	8.4684e-17	1.7691e-16
8	0	0	0	0	0	0	-3.2122e-99	0.6903	1.4902e-17	-2.3424e-16
9	0	0	0	0	0	0	0	-2.6214e-1...	0.3175	2.0518e-16
10	0	0	0	0	0	0	0	0	-2.5458e-2...	0.0810

d)

Editor - C:\Users\Nwhybra\Desktop\UW AMATH Masters\AMATH 584\HW\HW8\inverse_iteration.m

pure_qr_no_shifts.m | test.m | RandOrthMat.m | power_iteration.m | inverse_iteration.m

```
function eigen_vector = inverse_iteration(A, lambda, stop)
    m = size(A, 1);
    v = rand(m, 1);
    v = v / norm(v, 2);

    error = inf;
    I = eye(m);
    while error > stop
        w = (A - lambda * I) \ v;
        v = w / norm(w, 2);
        error = norm(A * v - lambda * v, 2);
    end

    eigen_vector = v;
end
```

10x1 double

	1
1	-0.4221
2	0.1201
3	0.3879
4	-0.2305
5	-0.3223
6	0.3223
7	0.2305
8	-0.3879
9	-0.1201
10	0.4221

Full test script:

Editor - C:\Users\Nwhybra\Desktop\UW AMATH Masters\AMATH 584\HW\HW8\test.m

pure_qr_no_shifts.m | test.m | RandOrthMat.m | power_iteration.m | inverse_iteration.m | +

```
% Construct the matrix A.
m = 10;
v = ones(10, 1);
A = diag(2 * v, 0) + diag(-1 * v(1:m-1), -1) + diag(-1 * v(1:m-1), 1);

% Computing eigen-values to check my work.
lambs = eigs(A);

% Part (b).
max_eig = power_iteration(A, 1e-6);

% Part (c).
D = pure_qr_no_shifts(A, 1e-12);

% Part(d).
fifth_eig = D(5, 5);
eigen_vector_5 = inverse_iteration(A, fifth_eig, 1e-8);
error = norm(A * eigen_vector_5 - fifth_eig * eigen_vector_5, 2);
```