NAME: Jakub Slowinski

STUDENT NUMBER: 16319781

Please indicate your answers by entering the option ((i), (ii), (iii) or (iv)) where asked. You should append the completed document as a pdf with your type written worked solutions and upload to Blackboard by Friday 22nd of February 2019.

Q 2.31

Part (a):

    (i) 4
    (ii) 13
    (iii) 26
    (iv) 18

Your Answer: (ii)

Part (b):

    (i)   0
    (ii)   12
    (iii)   7
    (iv)   4

Your Answer: (i)

Q 3.2

Part (a):

    (i)   0.1241
    (ii)   **0.8125**
    (iii)   0.074995
    (iv)   0.003462

Your Answer: (ii)

Part (b):

    (i)   0.72481
    (ii)   **0.85261**
    (iii)   0.62849
    (iv)   0.17238

Your Answer: (ii)

Part (c):

    (i)   0.65782
    (ii)   0.59371
    (iii)   0.45802
    (iv)   **0.85261**

Your Answer: (iv)

Q 4.24

(i)   Inverse(a)=

-0.7143	**0.0**	**1.4286**
0.2571	**0.1000**	**0.2857**
-0.2286	**-0.2000**	**0.8571**

Inverse(b)=

1.6667	**2.8889**	**-2.2222**	**1.0000**
0.0	**0.3333**	**-0.3333**	**0.0**
-0.3333	**-0.4444**	**0.1111**	**0.0**
1.5000	**2.0000**	**-1.5000**	**0.5000**

(ii)

Inverse(a)=

0.7243	0.0	1.3286
1.2571	0.1000	0.2757
-0.2386	-0.2010	0.9571

Inverse(b)=

1.6677	2.9889	3.2222	1.01700
0.3433	-0.3433	0.3333	0.00371
-0.3433	-0.2879	0.2111	0.0
1.2400	2.0120	-1.5783	0.5600

(iii)

Inverse(a)=

0.7143	0.003	2.3276
1.2671	0.1100	0.3759
-0.2486	-0.2110	0.9771

Inverse(b)=

1.6877	3.9789	3.2002	2.01800
0.3533	-0.4433	0.3333	0.02371
-0.3443	-0.2999	0.3121	0.0382
1.2420	3.0130	-1.5733	0.5610

(iv)

Inverse(a)=

0.8343	1.01	1.3336
2.2572	0.1003	0.3857
-0.2486	-0.2110	0.9671

Inverse(b)=

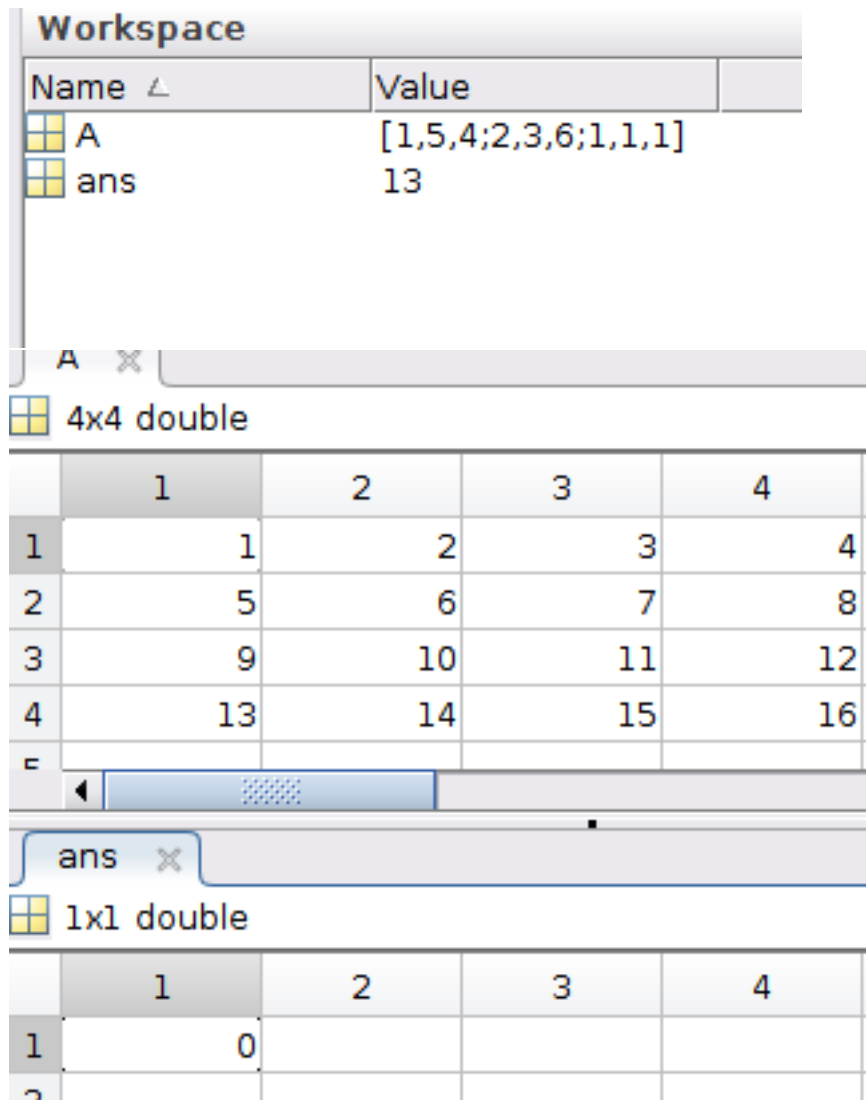|			
1.6777	4.9889	3.2232	1.11700
0.3443	-0.3443	0.3233	0.07371
-0.3443	-0.2979	0.3211	0.07800
1.2480	2.1220	-1.5883	0.5621

Your Answer: (i)

- Q2.31

Matlab Code

```matlab
1  function D = Determinant(A)
2  [n, m]=size(A);   % n= no. of rows in A, m= no. of
        columns in A
3
4  if n ~= m      % check if A is square
5      D ='The matrix must be square';
6
7  elseif n > 4 % check if bigger than 4x4
8      D ='Matrix too big';
9
10 elseif n == 2 % 2x2 matrix determinant
11         D = A(1,1)*A(2,2) - A(1,2)*A(2,1);
12
13 elseif n == 3 % 4x4 matrix
14     % 3 new matrixes, ready for recursive call
15     M1 = [A(2,2),A(2,3);A(3,2),A(3,3)];
16     M2 = [A(2,1),A(2,3);A(3,1),A(3,3)];
17     M3 = [A(2,1),A(2,2);A(3,1),A(3,2)];
18     % recursive call below to 2x2 matrix
19         D = A(1,1)*Determinant(M1) -  A(1,2)*
                Determinant(M2) +  A(1,3)*Determinant(M3)
                ;
20 elseif n == 4  %4x4 matrix
21     % 4 matrixes
22     M1 = [A(2,2),A(2,3),A(2,4); A(3,2),A(3,3),A(3,4);
            A(4,2),A(4,3),A(4,4)];
23     M2 = [A(2,1),A(2,3),A(2,4); A(3,1),A(3,3),A(3,4);
            A(4,1),A(4,3),A(4,4)];
24     M3 = [A(2,1),A(2,2),A(2,4); A(3,1),A(3,2),A(3,4);
            A(4,1),A(4,2),A(4,4)];
25     M4 = [A(2,1),A(2,2),A(2,3); A(3,1),A(3,2),A(3,3);
            A(4,1),A(4,2),A(4,3)];
26     % calling recurisvely the function with 4, 3x3
            matrixes
27         D = A(1,1)*Determinant(M1)-A(1,2)*Determinant
                (M2) + A(1,3)*Determinant(M3)-A(1,4)*
                Determinant(M4);
28 end
```

Workspace

Name △	Value
A	[1,5,4;2,3,6;1,1,1]
ans	13

A ✕

⊞ 4x4 double

	1	2	3	4
1	1	2	3	4
2	5	6	7	8
3	9	10	11	12
4	13	14	15	16
5				

ans ✕

⊞ 1x1 double

	1	2	3	4
1	0			
2				

- Q3.2
  root of f(x) = x - $2e^{-x}$

  (a) Bisection method
  $$a = 0, b = 1$$
  $$f(0) = 0 - 2e^0 = -2$$
  $$f(1) = 1 - 2e^{-1} = 0.2642$$

2

First iteration:
$$x_1 = \frac{a+b}{2} = \frac{0+1}{2} = 0.5$$
$$f(x_1) = 0.5 - 2e^{-0.5} = -0.7$$

as this is a negative number and you need a negative value and positive value between a and b; a now equals c_1
Second iteration:
$$a = 0.5, b = 1$$
$$x_2 = \frac{0.5+1}{2} = 0.75$$
$$f(x_2) = 0.75 - 2e^{-0.75} = -0.1947$$

as this is a negative number, it replaces the previous negative number, a
Third iteration:
$$a = 0.75, b = 1$$
$$x_3 = \frac{0.75+1}{2} = 0.875$$
$$f(x_3) = 0.875 - 2e^{-0.875} = 0.04$$

as this is a positive number, this is the new b
Final iteration:
$$a = 0.75, b = 0.875$$
$$x_4 = \frac{0.75 + 0.875}{2} = 0.8125$$

(b) Secant method
$$x_1 = 0, x_2 = 1$$
$$f(x_1) = 0 - 2e^0 = -2$$
$$f(x_2) = 1 - 2e^{-1} = 0.2642$$

First iteration

$$x_3 = x_2 - f(x_2) * \frac{x_2 - x_1}{f(x_2) - f(x_1)}$$

$$x_3 = 1 - 0.2642 * \frac{1-0}{0.2642 + 2} = 0.8516$$

$$f(x_3) = 0.8833 - 2e^{-0.8833} = 0.0565$$

3

Second iteration

$$x_4 = x_3 - f(x_3) * \frac{x_3 - x_2}{f(x_3) - f(x_2)}$$

$$x_4 = 0.8833 - 0.0565 * \frac{0.8833 - 1}{0.0565 - 0.2642} = 0.8516$$

$$f(x_4) = 0.8516 - 2e^{-0.8516} = -0.0019$$

Third iteration

$$x_5 = x_4 - f(x_4) * \frac{x_4 - x_3}{f(x_4) - f(x_3)}$$

$$x_5 = 0.8516 + 0.0019 * \frac{0.8516 - 0.8833}{-0.019 - 0.0565} = 0.8529$$

$$f(x_5) = 0.8529 - 2e^{-0.8529} = 0.00054$$

Final iteration

$$x_6 = x_5 - f(x_5) * \frac{x_5 - x_4}{f(x_5) - f(x_4)}$$

$$x_6 = 0.8529 - 0.00054 * \frac{0.8529 - 0.8516}{0.00054 + 0.0019} = 0.85261$$

(c) Newton's method

$$f(x) = x - 2e^{-x}, x_1 = 1$$

$$f'(x) = 2e^{-x} + 1$$

$$f(x_1) = 1 - 2e^{-1} = 0.2642$$

$$f'(x_1) = 2e^{-1} + 1 = 1.7358$$

First iteration

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

$$x_2 = 1 - \frac{0.2642}{1.7358} = 0.8478$$

$$f(x_2) = 0.8478 - 2e^{-0.8478} = -0.0089$$

$$f'(x_2) = 2e^{-0.8478} + 1 = 1.8567$$

Second iteration

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)}$$

$$x_3 = 0.8478 - \frac{-0.0089}{1.8467} = 0.8526$$

4

$$f(x_2) = 0.8526 - 2e^{-0.8526} = -0.0001$$
$$f'(x_2) = 2e^{-0.8526} + 1 = 1.8526$$

Third iteration
$$x_4 = x_3 - \frac{f(x_3)}{f'(x_3)}$$
$$x_4 = 0.8526 - \frac{-0.00001}{1.8526} = 0.8526$$
$$f(x_4) = 0.8526 - 2e^{-0.8526} = -0.0001$$
$$f'(x_4) = 2e^{0.8526} + 1 = 1.8567$$

Final iteration
$$x_5 = 0.8526 - \frac{-0.00001}{1.8526} = 0.8256$$

- Q4.24

Matlab Code

```
1  function Ainv = Inverse (A)
2  [n, m]=size(A);  % n= no. of rows in A, m= no. of
        columns in A
3  if n ~= m      % check if A is a square matrix
4      Ainv ='The matrix must be square';
5      return
6  end
7  if n == 0      % check if A is empty
8      Ainv ='Matrix cant be empty';
9      return
10 end
11
12 Ainv = eye(n); % set up identity matrix
13 for r = 1 : n
14     for c = r : n
15         if A(c,r) ~= 0 % cannot operate a division on
                a zero
16             t = 1/A(r,r); % left val of row being
                    operated on
17             for i = 1 : n
18                 %getting A(r,k) to be 1
19                 A(r,i) = t * A(r,i);
20                 %repeat calculation on identity
                        matrix
```

```matlab
21                          Ainv(r,i) = t * Ainv(r,i);
22                  end
23                  %looking for not diagonal elements
24                  for i = 1 : n
25                      if i ~= r % if count doesnt equal row
26                          % for rows that
27                          t = -A(i,r);
28                          for j = 1 : n
29                              % subtracting to get the
                                  element to be zero
30                              A(i,j) = A(i,j) + t * A(r,j);
31                              Ainv(i,j) = Ainv(i,j) + t *
                                  Ainv(r,j);
32                          end
33                      end
34                  end
35              end
36              break
37          end
38  end
```

A — 3x3 double

	1	2	3
1	-1	2	1
2	2	2	-4
3	0.2000	1	0.5000
4			
5			
6			

ans — 3x3 double

	1	2	3	4
1	-0.7143	5.5511e-...	1.4286	
2	0.2571	0.1000	0.2857	
3	-0.2286	-0.2000	0.8571	
4				
5				
6				

A — 4x4 double

	1	2	3	4
1	-1	-2	1	2
2	1	1	-4	-2
3	1	-2	-4	-2
4	2	-4	1	-2
5				
6				

ans — 4x4 double

	1	2	3	4
1	1.6667	2.8889	-2.2222	1
2	0	0.3333	-0.3333	0
3	-0.3333	-0.4444	0.1111	0
4	1.5000	2	-1.5000	0.5000
5				
6				