MATRICES IN MATLAB

Lab Report # 02

CSE301 - L Signals & Systems Lab

Submitted by: AWAIS SADDIQUI

Registration No: 21PWCSE1993

Class Section: "A"

Submitted to:

Dr. Durr-e-Nayab

Department of Computer System Engineering

UET Peshawar

301L: Signals & Systems Lab

LAB ASSESSMENT RUBRICS

Marking Criteria	Exceeds expectation (5-4)	Meets expectation (3-2)	Does not meet expectation (1)	Score
1. Realization of Experiment	Program compiles (noerrors and no warnings). Program always works correctly and meets the specification(s). Completed between 71-100% of the requirements.	Program compiles (no errors and some warnings). Some details of the program specification are violated, program functions incorrectly for some inputs. Completed between 41-70% of the requirements.	Program fails to or compile with lots of warnings. Program only functions correctly in very limited cases or not at all. Completed less than 40% of the requirements.	30%
2. Ability to apply required code utility or data structure	Able to apply required data type or data structure and produce correct results. Familiarize and selects proper functions for simulation of given problem using software tools like MATLAB.	Able to apply required data type or data structure but does not produce correct results. Need guidance to select proper functions for simulation of given problem using software tools like MATLAB.	Unable to identify required data type or data structure. Incapable of selecting proper functions for simulation of given problem using software tools like MATLAB.	20%
3. Documentation	Clearly and effectively documented including descriptions of all variables/functions. Specific purpose is noted for each function, control structure, input requirements and output results.	Basic documentation including descriptions of all variables/functions. Specific purpose is noted for each function and control structure.	No documentation included.	10%

4. Ability to run/debug	Executes Matlab codes without errors, excellent user	Executes Matlab codes without errors. User prompts are	Does not execute Matlab codes due to errors.	20%

	prompts, good use of symbols, spacing in output. Thorough and organized testing has been completed and output from test cases is included.	understandable, minimum use of symbols or spacing in output. Some testing has been completed.	User prompts are misleading or nonexistent. No testing has been completed.	
5. Results compilation	Show processed results effectively by conducting simple computations and plotting using collected data	Show processed results effectively by conducting simple computations and plotting using collected data with minor error	Unable to show processed results effectively by conducting simple computations and plotting using collected data with minor error	10%
6. Efficiency	Excellent use of CPU and Memory.	Good but not smart use of CPU and Memory.	Inefficient use of CPU and Memory.	10%
7. Lab Performance (Team work and Lab etiquettes)	Actively engages and cooperates with other group members in an effective manner. Respectfully and carefully observes safety rules and procedures	Cooperates with other group members in a reasonable manner. Observes safety rules and procedures with minor deviation.	Distracts or discourages other group members from conducting the experiment. Disregards safety rules and procedures.	10%

Instructor:

Name: _____

Signature: _____

Signals & Systems Laboratory

MATRICES:

MATLAB works with essentially only one kind of object, a rectangular numerical matrix possibly, with complex entries. Every MATLAB variable refers to a matrix [a number is a 1 by 1 matrix]. In some situations, 1-by-1 matrices are interpreted as scalars, and matrices with only one row or one column are interpreted as vectors

Matrices in MATLAB:

MATLAB is designed to make definition of matrices and matrix manipulation as simple as possible. Matrices can be introduced into MATLAB in several different ways.

Objectives of the Lab:

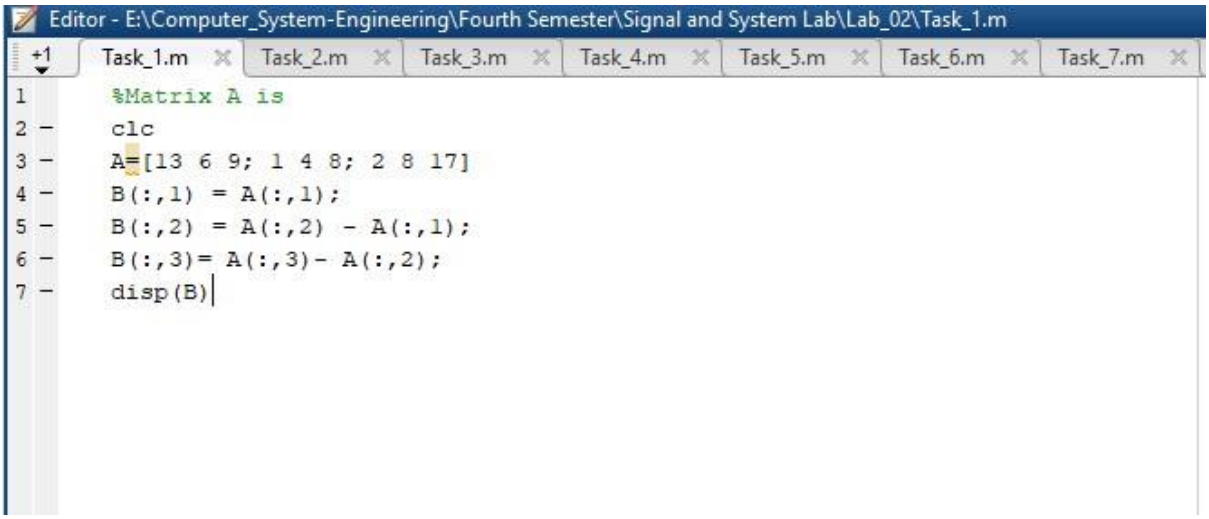
- Built in Matrix Functions
- Indexing Matrices
- Sub Matrices
- Matrix element level operations
- Round Floating-Point numbers to Integers

Task 01:

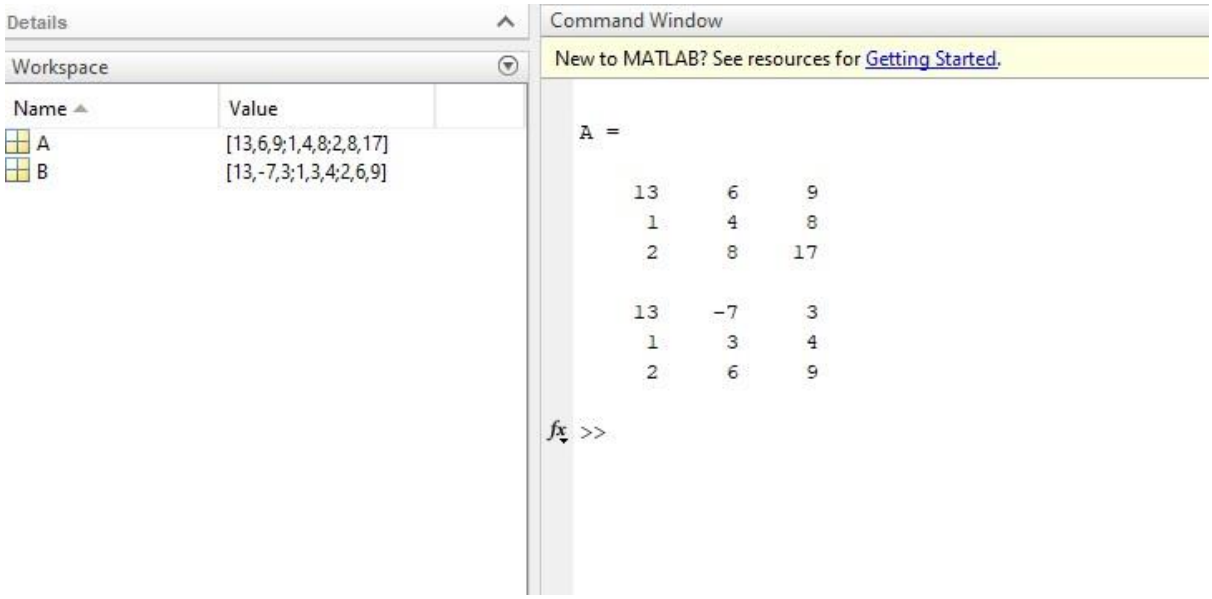
Write a program to generate a new matrix B from the matrix A given below such that each column in the new matrix except the first one is the result of subtraction of that column from the previous one i.e. 2nd new column is the result of subtraction of 2nd column and 1st column and so on. Copy the first column as it is in the new matrix.

$$\mathbf{A} = \begin{vmatrix} 13 & 6 & 9 \\ 1 & 4 & 8 \\ 2 & 8 & 17 \end{vmatrix}$$

Code:



Output:



Task 02:

Generate two 2500 sampled random discrete time signals (1 dimensional) using rand() function i.e. rand(1, 2500). Write a program to add the two such random signals together using simple vector addition.

Code:

Editor - E:\Computer_System-Engineering\Fourth Semester\Signal and System Lab\Lab_02\Task_2.m

Task_2.m | Task_3.m | Task_4.m | Task_5.m | Task_6.m | Task_7.m | Task_8.m

```
%Task-2
clear all
clc
num1=rand(1, 2500);
num2=rand(1, 2500);
result = num1+num2;
disp(result);
```

Output:

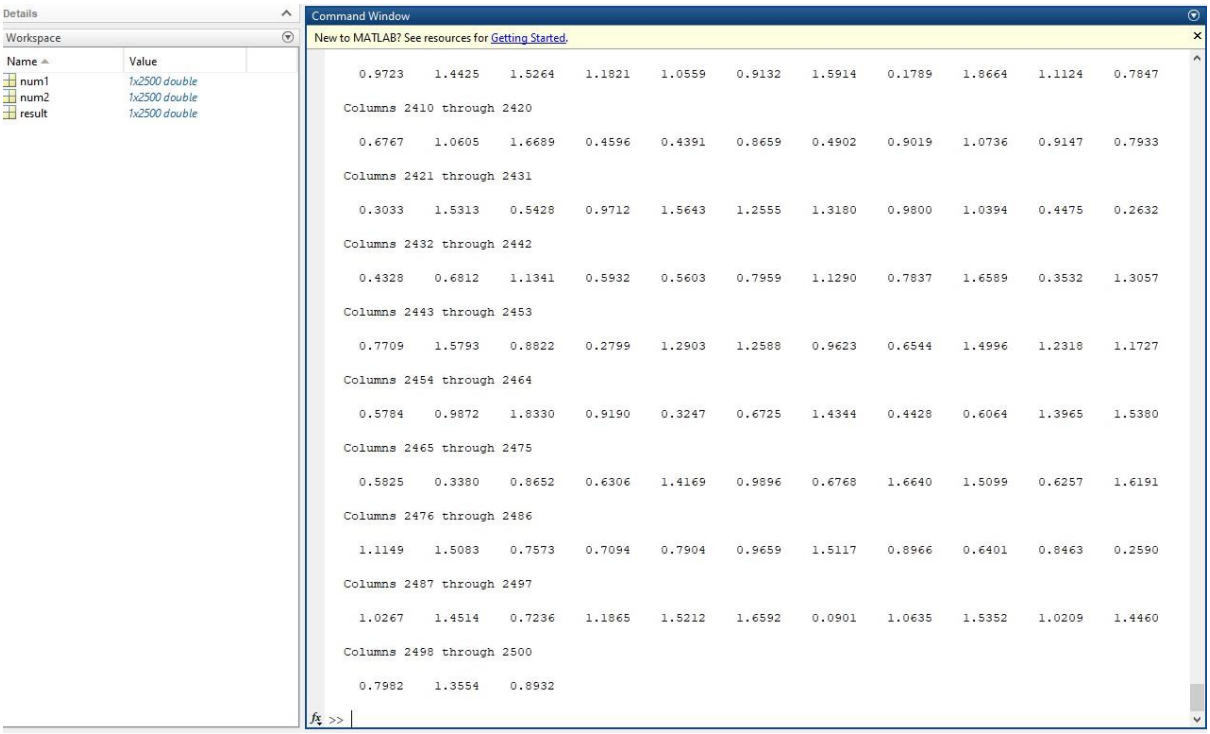
Workspace

Name	Value
num1	1x2500 double
num2	1x2500 double
result	1x2500 double

Command Window

New to MATLAB? See resources for Getting Started.

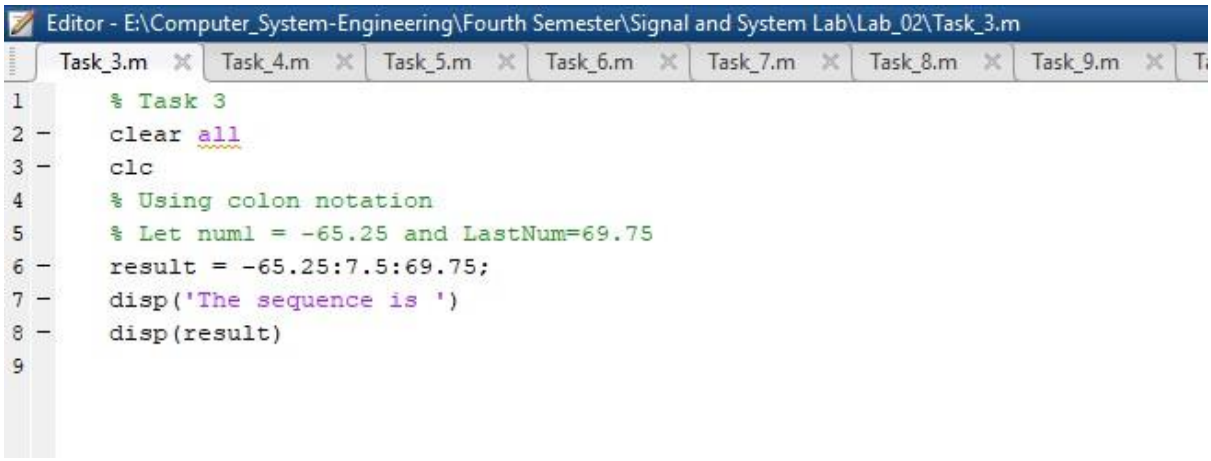
```
Columns 1 through 11

   1.6705    1.5766    0.6506    1.2122    1.3363    0.4792    0.8462    1.4347    1.8005    1.8637    1.0966

Columns 12 through 22

   1.7860    0.9585    0.4885    0.8877    0.4026    0.4446    1.3398    1.1333    1.5008    1.5819    0.3342

Columns 23 through 33

   1.1872    1.7935    1.0192    0.8959    1.2509    1.2489    1.0398    0.8669    1.3340    0.4822    0.7505

Columns 34 through 44

   0.9959    0.1806    1.1033    1.1418    0.9047    1.8279    0.5035    0.8762    1.1277    1.2334    1.6560

Columns 45 through 55

   0.6534    0.9879    0.9330    0.8758    0.7949    0.8221    1.1644    0.9129    1.5167    0.8743    0.9918

Columns 56 through 66

   1.4364    1.0994    0.7343    1.5658    0.8686    1.6477    0.7373    0.5201    1.3220    1.1220    1.4867

Columns 67 through 77

   1.2722    0.7460    0.7377    0.6909    1.0849    0.6832    0.8245    0.8523    1.8872    0.4454    0.2322

Columns 78 through 88

   1.1373    0.8630    0.4822    1.1666    0.9713    1.4651    0.6451    1.2698    0.8793    1.3424    1.4214

Columns 89 through 99

   1.0285    1.0012    0.2156    0.8059    0.7726    1.4296    1.7914    0.2143    1.5409    0.5009    0.8473

Columns 100 through 110

   1.1728    0.2120    1.3402    1.2544    0.8500    0.9721    1.2034    1.0526    1.4533    0.7388    1.0314
```


Details
Workspace
Name
Value
num1
1x2500 double
num2
1x2500 double
result
1x2500 double
Command Window
New to MATLAB? See resources for Getting Started.
0.9723 1.4425 1.5264 1.1821 1.0559 0.9132 1.5914 0.1789 1.8664 1.1124 0.7847
Columns 2410 through 2420
0.6767 1.0605 1.6689 0.4596 0.4391 0.8659 0.4902 0.9019 1.0736 0.9147 0.7933
Columns 2421 through 2431
0.3033 1.5313 0.5428 0.9712 1.5643 1.2555 1.3180 0.9800 1.0394 0.4475 0.2632
Columns 2432 through 2442
0.4328 0.6812 1.1341 0.5932 0.5603 0.7959 1.1290 0.7837 1.6589 0.3532 1.3057
Columns 2443 through 2453
0.7709 1.5793 0.8822 0.2799 1.2903 1.2588 0.9623 0.6544 1.4996 1.2318 1.1727
Columns 2454 through 2464
0.5784 0.9872 1.8330 0.9190 0.3247 0.6725 1.4344 0.4428 0.6064 1.3965 1.5380
Columns 2465 through 2475
0.5825 0.3380 0.8652 0.6306 1.4169 0.9896 0.6768 1.6640 1.5099 0.6257 1.6191
Columns 2476 through 2486
1.1149 1.5083 0.7573 0.7094 0.7904 0.9659 1.5117 0.8966 0.6401 0.8463 0.2590
Columns 2487 through 2497
1.0267 1.4514 0.7236 1.1865 1.5212 1.6592 0.0901 1.0635 1.5352 1.0209 1.4460
Columns 2498 through 2500
0.7982 1.3554 0.8932
>>

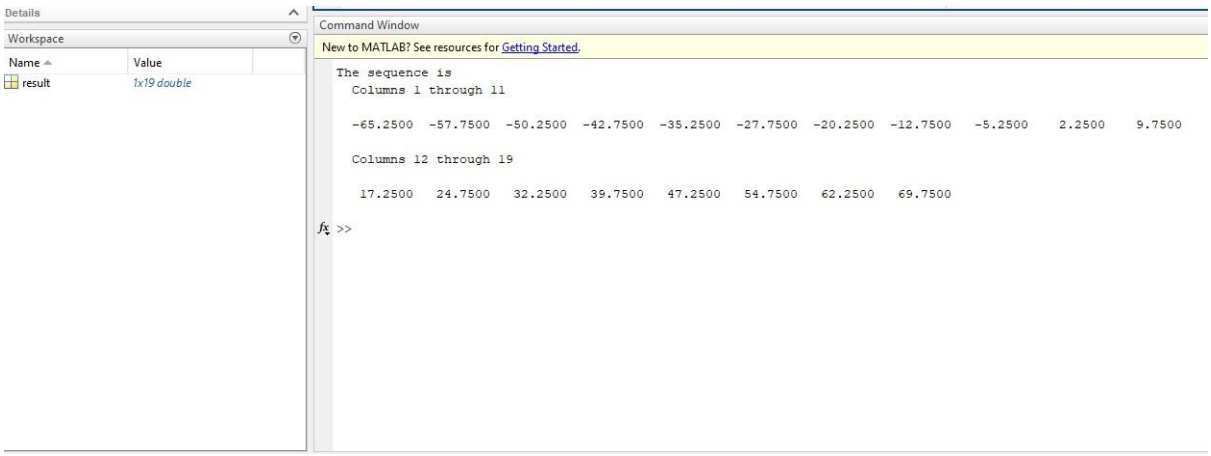
Task 03:

Using colon notation, generate the following sequence: -65.25, -57.75, -50.25. . . . . ., 54.75,62.25, 69.75

Code:



Output:



Task 04:

Given the matrices:

A=[-12,34,61,-9;65,78,90,12; 14,78,45,12; 60,25,3,8]

B= [34,67,08,09; 12, -91,12,9; 89, -8,0,02; 16,09,23,67]

Find the following:

1) Array addition; store the result in matrix C

2) Array subtraction; store the result in matrix D

3) Array multiplication using .* operator; store the result in matrix E

4) Array division using ./ operator; store the result in matrix F

5) Array exponentiation using .^ operator; store the result in matrix G

6) Take sin of A and store the result in H, Take sqrt of B and store the result in I. Find H*I and store the result in J.

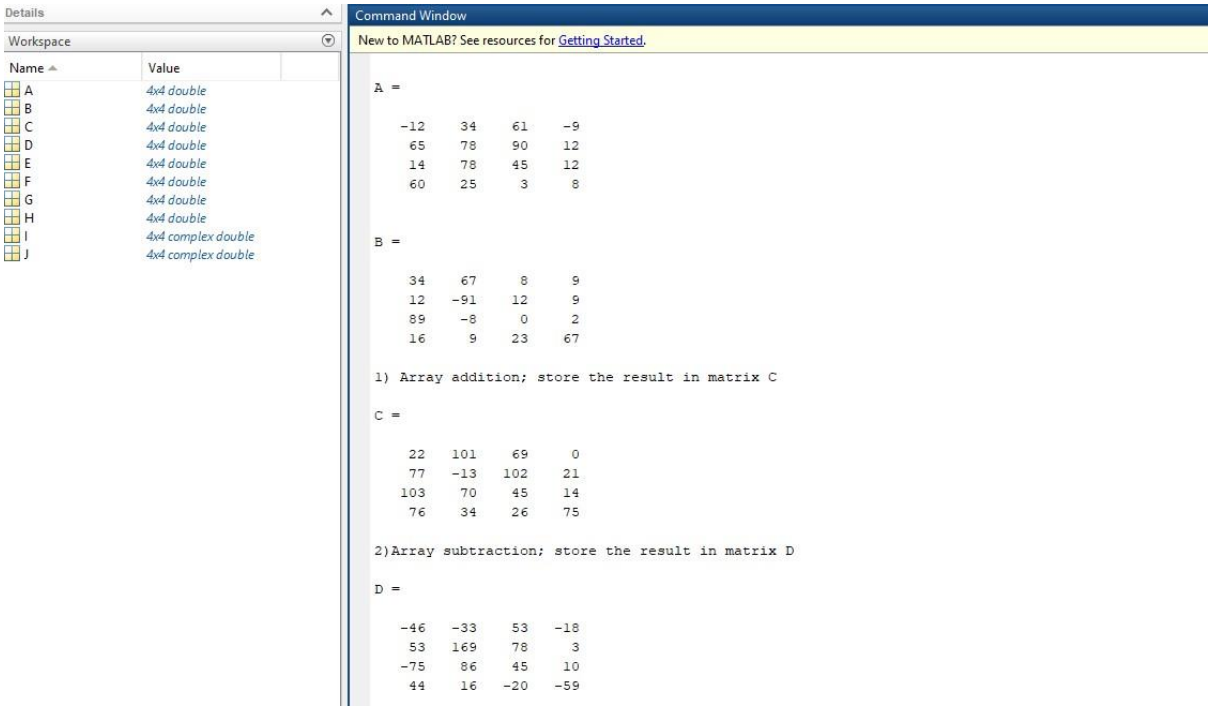
Code:

Editor - E:\Computer_System-Engineering\Fourth Semester\Signal and System Lab\Lab_02\Task_4.m

Task_4.m | Task_5.m | Task_6.m | Task_7.m | Task_8.m | Task_9.m | Task_10.m

```
% Task-4
clear all
clc
A=[-12,34,61,-9;65,78,90,12; 14,78,45,12; 60,25,3,8]
B=[34,67,8,9; 12,-91,12,9; 89,-8,0,2; 16,9,23,67]
disp('1) Array addition; store the result in matrix C ')
C=A+B
disp('2)Array subtraction; store the result in matrix D ')
D=A-B
disp('3)Array multiplication using .* operator; store the result in matrix E')
E=A.*B
disp('4)Array division using ./ operator; store the result in matrix F')
F=A./B
disp('5)Array exponentiation using .^ operator; store the result in matrix G')
G=A.^B
disp('6)Take sin of A and store the result in H, Take sqrt of B and store the result in I. ')
disp('Find H*I and store the result in J.')
H=sin(A);
I= sqrt(B);
J=H*I
```

Output:


Details
Workspace
Name
Value
A 4x4 double
B 4x4 double
C 4x4 double
D 4x4 double
E 4x4 double
F 4x4 double
G 4x4 double
H 4x4 double
I 4x4 complex double
J 4x4 complex double
Command Window
New to MATLAB? See resources for Getting Started.
A =
-12 34 61 -9
65 78 90 12
14 78 45 12
60 25 3 8
B =
34 67 8 9
12 -91 12 9
89 -8 0 2
16 9 23 67
1) Array addition; store the result in matrix C
C =
22 101 69 0
77 -13 102 21
103 70 45 14
76 34 26 75
2)Array subtraction; store the result in matrix D
D =
-46 -33 53 -18
53 169 78 3
-75 86 45 10
44 16 -20 -59

Command Window

New to MATLAB? See resources for Getting Started.

```
3)Array multiplication using .* operator; store the result in matrix E

E =

        -408        2278         488         -81
         780       -7098        1080         108
        1246        -624           0          24
         960         225          69         536

4)Array division using ./ operator; store the result in matrix F

F =

   -0.3529    0.5075    7.6250   -1.0000
    5.4167   -0.8571    7.5000    1.3333
    0.1573   -9.7500       Inf    6.0000
    3.7500    2.7778    0.1304    0.1194

5)Array exponentiation using .^ operator; store the result in matrix G

G =

  1.0e+102 *

    0.0000    4.0653    0.0000   -0.0000
    0.0000    0.0000    0.0000    0.0000
    1.0125    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000

6)Take sin of A and store the result in H, Take sqrt of B and store the result in I.
Find H*I and store the result in J.

J =

  -5.8013 + 0.0000i   3.1557 + 2.3145i   1.3740 + 0.0000i  -1.5427 + 0.0000i
  12.8893 + 0.0000i   5.1582 + 7.4316i   1.5458 + 0.0000i   0.8947 + 0.0000i
  13.4378 + 0.0000i   6.4988 + 7.3098i   2.0090 + 0.0000i   1.3251 + 0.0000i
   3.0529 + 0.0000i   0.4731 - 0.8634i   3.4242 + 0.0000i   6.9863 + 0.0000i

fx >>
```

Task 05:

A=[3 7 -4 12; -5 9 10 2; 6 13 8 11; 15 5 4 1]

Find the following:

1) Create 4x3 array B consisting of all elements in the second through fourth columns of A

2) Create 3x4 array C consisting of all elements in the second through fourth rows of A

3) Create 2x3 array D consisting of all elements in the first two rows and the last three .

Code:

Editor - E:\Computer_System-Engineering\Fourth Semester\Signal and System Lab\Lab_02\Task_5.m

Task_5.m | Task_6.m | Task_7.m | Task_8.m | Task_9.m | Task_10.m

```
% Task -5
clear all
clc
% 1) Create 4x3 array B
A = [3 7 -4 12; -5 9 10 2; 6 13 8 11; 15 5 4 1]
B = [A(:,4) A(:,4) A(:,4)]
% 2) Create 3x4 array C
C = [A(4,:); A(4,:); A(4,:)]
% 3) Create 2x3 array D
%D= [A(1:2,2),A(1:2,3),A(1:2,4)];
D = [A(1:2,2), A(1:2,3), A(1:2,4)]

```

Output:

Details

Workspace

Name	Value
A	4x4 double
B	4x3 double
C	3x4 double
D	[7,-4,12;9,10,2]

Command Window

New to MATLAB? See resources for Getting Started.

```
A =

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


B =

    12    12    12
     2     2     2
    11    11    11
     1     1     1


C =

    15     5     4     1
    15     5     4     1
    15     5     4     1


D =

     7    -4    12
     9    10     2
```

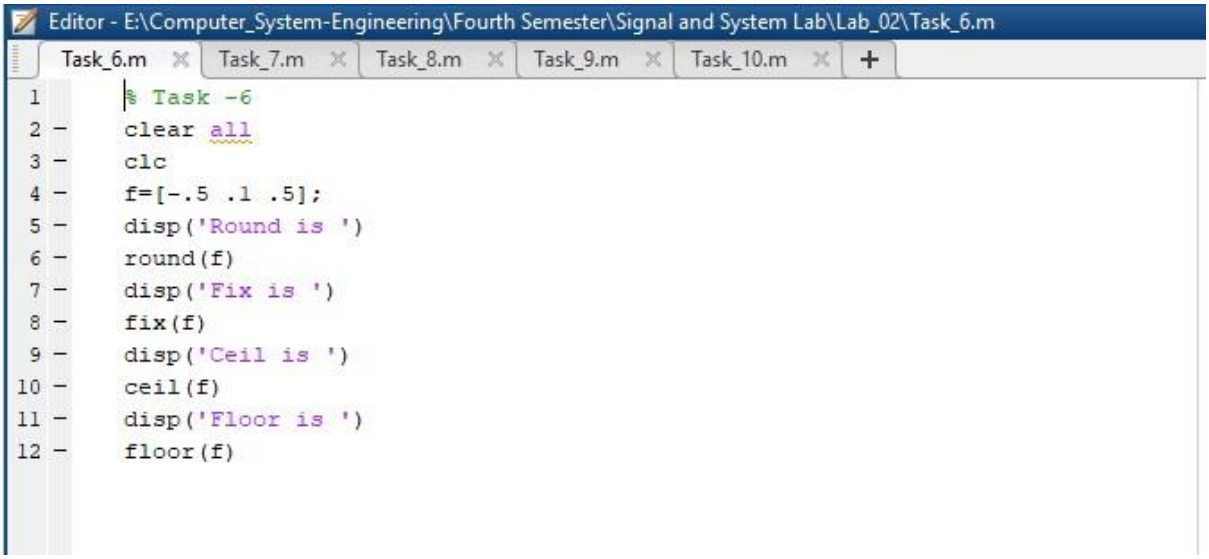
Task 06:

MATLAB has functions to round floating point numbers to integers. These are round, fix, ceil, and floor. Test how these functions work. Determine the output of the following:

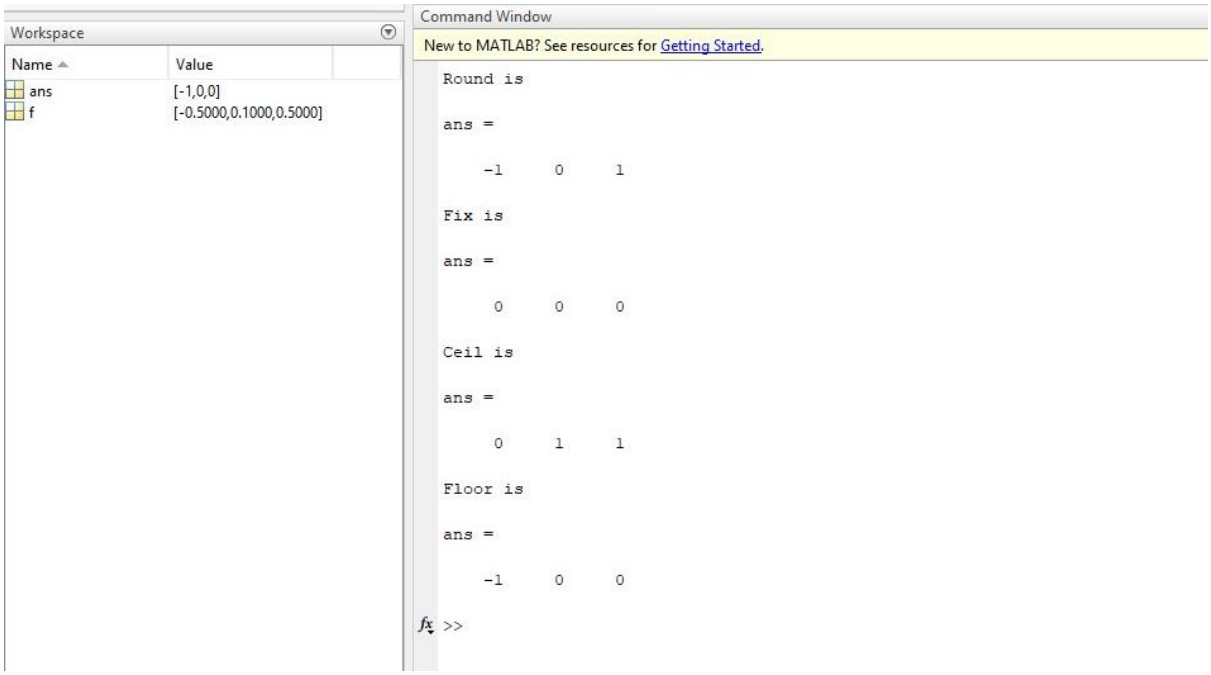
>> f = [-.5 .1 .5];

1) round(f) 2) fix(f) 3) ceil(f) 4) floor(f)

Code:



Output:



Task 07:

A = [-3 5; 4 8]

1) Column-wise sum of all elements of A . 2) Column-wise product of all elements of A .

3) Length of matrix A . 4) Size of matrix A

Code:

Editor - E:\Computer_System-Engineering\Fourth Semester\Signal and System Lab\Lab_02\Task_7.m

Task_7.m | Task_8.m | Task_9.m | Task_10.m

```
% Task -7
clear all
clc
A = [-3 5; 4 8]
disp('1) Column-wise sum is ')
sum(A,2)
disp('2) Column-wise product is ')
prod(A,2)
disp('3) Length of Matrix A is ')
length(A)
disp('4) Size of Matrix A is ')
size(A)
```

Output:

Details

Workspace

Name	Value
A	[-3,5;4,8]
ans	[2,2]

Command Window

New to MATLAB? See resources for Getting Started.

```
A =

    -3     5
     4     8

1) Column-wise sum is

ans =

     2
    12

2) Column-wise product is

ans =

   -15
    32

3) Length of Matrix A is

ans =

     2

4) Size of Matrix A is

ans =

     2     2
```

Task 08:

The end command is used to access the last row or column of a matrix. Use the end command to delete and update the last row and column.

Matrix **A = [3 23 34 12 34 5 56 23; 12 34 34 32 23 23 45 1; 67 23 2 4 4 5 6 456; 4 5 1 1 2 34 45 56; 67 67 45 67 78 7 8 5; 6 35 5 3 5 56 7 8]**

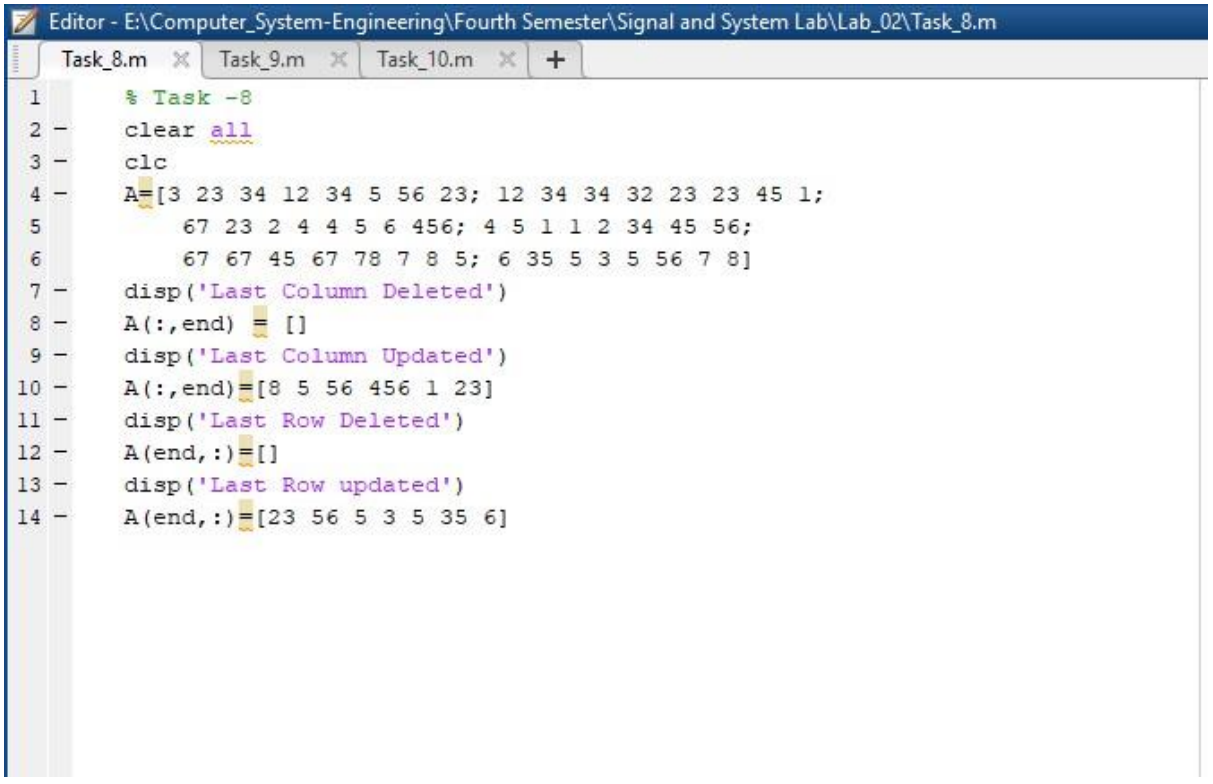
Hint:

For deleting a column use A(3 , :)=[ ];

For deleting last column use A(: , end)=[ ];

and vice versa.

Code:



Output:

Command Window

New to MATLAB? See resources for Getting Started.

```
A =

     3    23    34    12    34     5    56    23
    12    34    34    32    23    23    45     1
    67    23     2     4     4     5     6   456
     4     5     1     1     2    34    45    56
    67    67    45    67    78     7     8     5
     6    35     5     3     5    56     7     8

Last Column Deleted

A =

     3    23    34    12    34     5    56
    12    34    34    32    23    23    45
    67    23     2     4     4     5     6
     4     5     1     1     2    34    45
    67    67    45    67    78     7     8
     6    35     5     3     5    56     7

Last Column Updated

A =

     3    23    34    12    34     5     8
    12    34    34    32    23    23     5
    67    23     2     4     4     5    56
     4     5     1     1     2    34   456
    67    67    45    67    78     7     1
     6    35     5     3     5    56    23
```

Workspace

Name	Value
A	5x7 double

```
Last Row Deleted

A =

     3    23    34    12    34     5     8
    12    34    34    32    23    23     5
    67    23     2     4     4     5    56
     4     5     1     1     2    34   456
    67    67    45    67    78     7     1

Last Row updated

A =

     3    23    34    12    34     5     8
    12    34    34    32    23    23     5
    67    23     2     4     4     5    56
     4     5     1     1     2    34   456
    23    56     5     3     5    35     6

fx >>
```

Task 09:

Try the following commands in MATLAB and comment on them:

A(3,end)

(ii) A(:)

(iii) A(: , end)

(iv) Y = linspace(20,100)

(v) Y = linspace(20,100,50)

Output:

(i) A(3,end) =This command accesses the element in the third row and last column of the matrix A. The keyword 'end' is used to refer to the last index in a matrix.

(ii) A(:) =This command reshapes the matrix A into a single column vector, where each element of A is listed in column-major order.

(iii) A(: , end) = This command accesses all the elements in the last column of the matrix A.

(iv) Y = linspace(20,100) =This command generates a row vector Y with 100 equally spaced points between 20 and 100, inclusive. The default number of points is 100, so the resulting vector has 100 elements.

(v) Y = linspace(20,100,50) =This command generates a row vector Y with 50 equally spaced points between 20 and 100, inclusive. The resulting vector has 50 elements. The third argument specifies the number of points to generate.

Workspace

Name	Value
A	6x8 double
ans	[23;1;456;56;5;8]
Y	1x50 double

Command Window

New to MATLAB? See resources for Getting Started.

```
A =

     3    23    34    12    34     5    56    23
    12    34    34    32    23    23    45     1
    67    23     2     4     4     5     6   456
     4     5     1     1     2    34    45    56
    67    67    45    67    78     7     8     5
     6    35     5     3     5    56     7     8

A(3,end) mean Extract the third through the last elements in Last coloumn

ans =

   456

A(:) mean reshapes all elements of A into a single column vector

ans =

     3
    12
    67
     4
    67
     6
    23
    34
    23
     5
    67
    35
    34
    34
     2
     1
    45
     5
    12
    32
     4
     1
    67
```

Command Window

New to MATLAB? See resources for Getting Started.

```
     1
   456
    56
     5
     8

A(:,end) Gives Last Column of A

ans =

    23
     1
   456
    56
     5
     8

linspace(20,100) generate a row vector Y containing 100 equally spaced values between 20 and 100

Y =

  Columns 1 through 11

   20.0000   20.8081   21.6162   22.4242   23.2323   24.0404   24.8485   25.6566   26.4646   27.2727   28.0808

  Columns 12 through 22

   28.8889   29.6970   30.5051   31.3131   32.1212   32.9293   33.7374   34.5455   35.3535   36.1616   36.9697

  Columns 23 through 33

   37.7778   38.5859   39.3939   40.2020   41.0101   41.8182   42.6263   43.4343   44.2424   45.0505   45.8586

  Columns 34 through 44

   46.6667   47.4747   48.2828   49.0909   49.8990   50.7071   51.5152   52.3232   53.1313   53.9394   54.7475

  Columns 45 through 55

   55.5556   56.3636   57.1717   57.9798   58.7879   59.5960   60.4040   61.2121   62.0202   62.8283   63.6364
```

Command Window

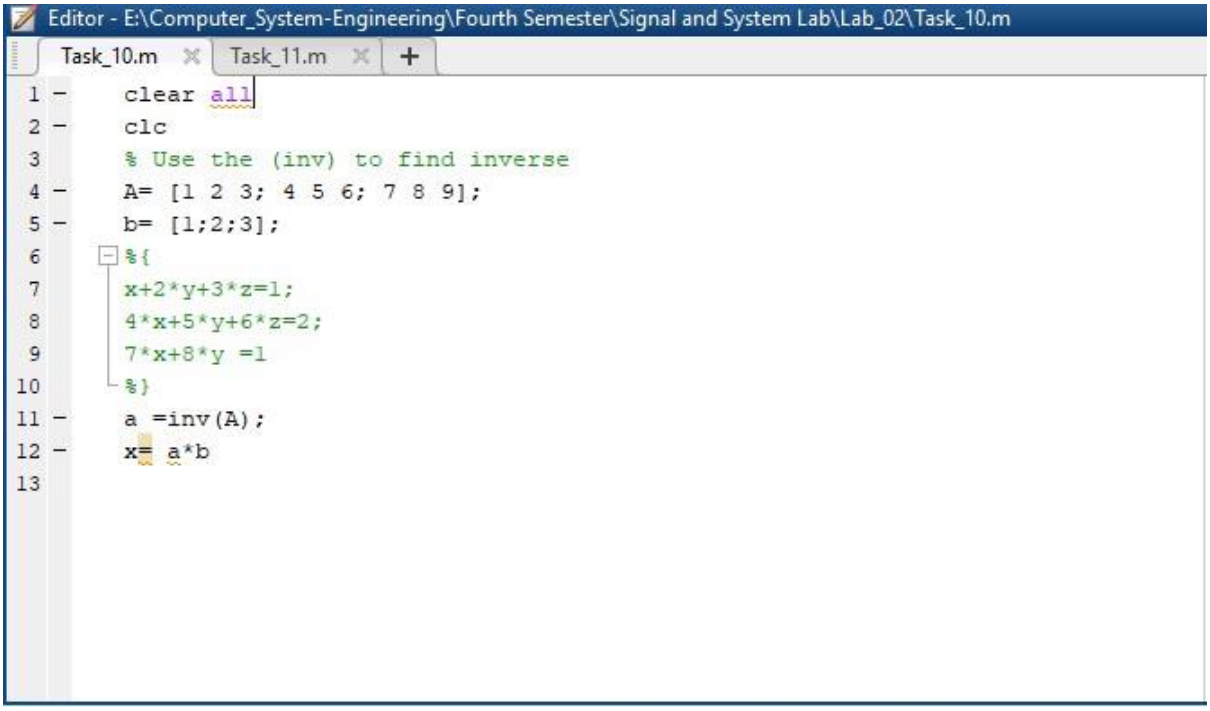
New to MATLAB? See resources for Getting Started.

```
    73.3333   74.1414   74.9495   75.7576   76.5657   77.3737   78.1818   78.9899   79.7980   80.6061   81.4141

  Columns 78 through 88

    82.2222   83.0303   83.8384   84.6465   85.4545   86.2626   87.0707   87.8788   88.6869   89.4949   90.3030

  Columns 89 through 99

    91.1111   91.9192   92.7273   93.5354   94.3434   95.1515   95.9596   96.7677   97.5758   98.3838   99.1919

  Column 100

   100.0000

linspace(20,100,50) generate a row vector containing 50 equally spaced values

Y =

  Columns 1 through 11

    20.0000   21.6327   23.2653   24.8980   26.5306   28.1633   29.7959   31.4286   33.0612   34.6939   36.3265

  Columns 12 through 22

    37.9592   39.5918   41.2245   42.8571   44.4898   46.1224   47.7551   49.3878   51.0204   52.6531   54.2857

  Columns 23 through 33

    55.9184   57.5510   59.1837   60.8163   62.4490   64.0816   65.7143   67.3469   68.9796   70.6122   72.2449

  Columns 34 through 44

    73.8776   75.5102   77.1429   78.7755   80.4082   82.0408   83.6735   85.3061   86.9388   88.5714   90.2041

  Columns 45 through 50

    91.8367   93.4694   95.1020   96.7347   98.3673  100.0000

fx >>
```

Task 10:

Use the inverse (inv(A)) function to find the inverse of A for finding the unknowns for the Linear equation.

Code:



Output:

Task 11:

Solve Task 10 by taking the equations from user.

Hint: Take the matrix A and b from user.

Code:

Editor - E:\Computer_System-Engineering\Fourth Semester\Signal and System Lab\Lab_02\Task_11.m

Task_11.m

```
clear all
clc
% Task -11
A =  input('Enter 3 by 3 matrix = ');
b= input('Enter 1 by 3 matrix = ');
inverse = inv(A);
disp('inverse of A')
x = inverse*b
```

Output: