

Exercise 1

Vector Operations

Q1. Write a program to create two row vectors of 10 elements each and perform the arithmetic operation as mentioned below:

1. Addition
2. Subtraction
3. Multiplication
4. Append
5. Transpose
6. Power

Logic:

Step 1: First we have to create a new script that is named as demo.m.

Step 2: Now we have to initialise two row vectors named as A and B.

Step 3: X and Y are row vectors.

Step 4: We have to use the arithmetic symbols '+', '-', '*' for addition, subtraction and multiplication.

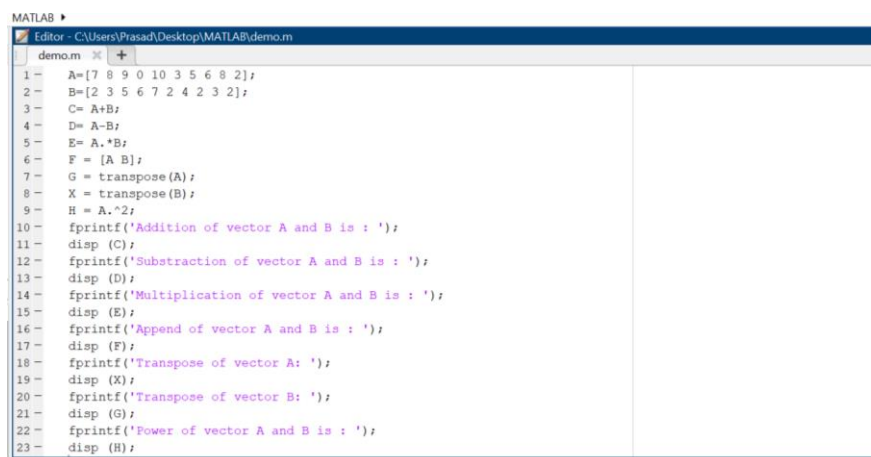
Step 5: We have to calculate power using ^ symbol.

Step 6: The results are printed using disp() function.

Step 7: The result would be printed in the command window.

Step 8: Stop

Code:



```
MATLAB
Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m
1 A=[7 8 9 0 10 3 5 6 8 2];
2 B=[2 3 5 6 7 2 4 2 3 2];
3 C= A+B;
4 D= A-B;
5 E= A.*B;
6 F = [A B];
7 G = transpose(A);
8 X = transpose(B);
9 H = A.^2;
10 fprintf('Addition of vector A and B is : ');
11 disp(C);
12 fprintf('Subtraction of vector A and B is : ');
13 disp(D);
14 fprintf('Multiplication of vector A and B is : ');
15 disp(E);
16 fprintf('Append of vector A and B is : ');
17 disp(F);
18 fprintf('Transpose of vector A : ');
19 disp(X);
20 fprintf('Transpose of vector B : ');
21 disp(G);
22 fprintf('Power of vector A and B is : ');
23 disp(H);
```

Output:

The MATLAB Editor shows a script named `demo.m` with the following code:

```
1 A=[7 8 9 0 10 3 5 6 8 2];
2 B=[2 3 5 6 7 2 4 2 3 2];
3 C=A+B;
```

The Command Window shows the output of the script:

```
>> demo
Addition of vector A and B is :    9    11    14    6    17    5    9    8    11    4
Substraction of vector A and B is :    5    5    4   -6    3    1    1    4    5    0
Multiplication of vector A and B is :   14   24   45    0   70    6   20   12   24    4
Append of vector A and B is :  Columns 1 through 19
    7    8    9    0   10    3    5    6    8    2    2    3    5    6    7    2    4    2    3
Column 20
    2
Transpose of vector A:    2
    3
    5
    6
```

The MATLAB Editor shows the same script as above.

The Command Window shows the output of the script:

```
Transpose of vector A:    2
    3
    5
    6
    7
    2
    4
    2
    3
    2
Transpose of vector B:    7
    8
    9
    0
   10
    3
    5
    6
```

The MATLAB Editor shows the same script as above.

The Command Window shows the output of the script:

```
4
2
3
2
Transpose of vector B:    7
    8
    9
    0
   10
    3
    5
    6
    8
    2
Power of vector A and B is :   49   64   81    0  100    9   25   36   64    4
```

Q2. Write a program to create two column vectors of 10 elements each and perform the arithmetic operation as mentioned below:

1. Addition

2. Subtraction

3. Multiplication

4. Append

5. Transpose

6. Power

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m.

Step 3: Now just create column vectors with the help of semi colon between the elements.

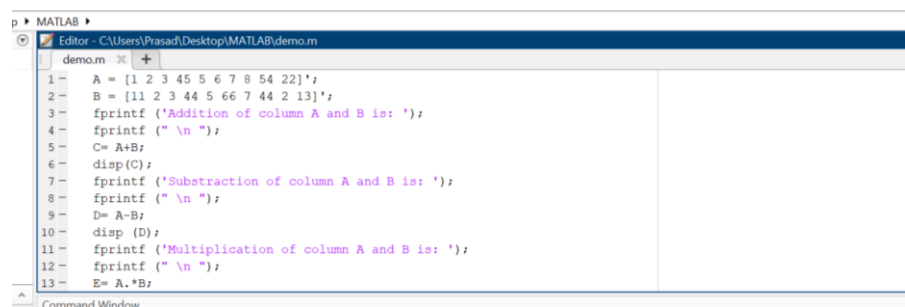
Step 4: So, we have to just use the arithmetic symbols '+', '-', '.*' for addition, subtraction and multiplication.

Step 5: We have to calculate power using ^ symbol.

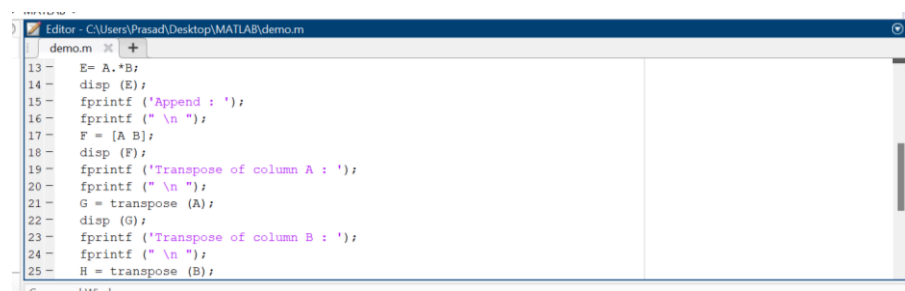
Step 6: So, the vectors (A and B) and results are printed using disp() function.

Step 7: Stop

Code:



```
1 A = [1 2 3 4 5 6 7 8 54 22]';
2 B = [11 2 3 44 5 66 7 44 2 13]';
3 fprintf('Addition of column A and B is: ');
4 fprintf("\n ");
5 C = A+B;
6 disp(C);
7 fprintf('Substraction of column A and B is: ');
8 fprintf("\n ");
9 D = A-B;
10 disp(D);
11 fprintf('Multiplication of column A and B is: ');
12 fprintf("\n ");
13 E = A.*B;
```



```
13 E = A.^B;
14 disp(E);
15 fprintf('Append : ');
16 fprintf("\n ");
17 F = [A B];
18 disp(F);
19 fprintf('Transpose of column A : ');
20 fprintf("\n ");
21 G = transpose(A);
22 disp(G);
23 fprintf('Transpose of column B : ');
24 fprintf("\n ");
25 H = transpose(B);
```

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m
23- fprintf('Transpose of column B : ');
24- fprintf("\n ");
25- H = transpose(B);
26- disp(H);
27- fprintf('Power of column A : ');
28- fprintf("\n ");
29- I = A.^2;
30- disp(I);
31- fprintf('Power of column B : ');
32- fprintf("\n ");
33- J = B.^2;
34- disp(J);

```

Output:

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m
23- fprintf('Transpose of column B : ');
24- fprintf("\n ");
Command Window
New to MATLAB? See resources for Getting Started.
>> demo
Addition of column A and B is:
12
4
6
89
10
72
14
52
56
35
Subtraction of column A and B is:
-10
0
0
1
0
-60

```

```

Command Window
New to MATLAB? See resources for Getting Started.
Subtraction of column A and B is:
-10
0
0
1
0
-60
0
-36
52
9
Multiplication of column A and B is:
11
4
9
1980
25

```

```

Command Window
New to MATLAB? See resources for Getting Started.
Multiplication of column A and B is:
11
4
9
1980
25
396
49
352
108
286
Append :
1 11
2 2
3 3
45 44
5 5

```

```

Command Window
New to MATLAB? See resources for Getting Started.

Append :
1 11
2 2
3 3
45 44
5 5
6 66
7 7
8 44
54 2
22 13

Transpose of column A :
1 2 3 45 5 6 7 8 54 22

Transpose of column B :
11 2 3 44 5 66 7 44 2 13

```

```

Command Window
New to MATLAB? See resources for Getting Started.

Transpose of column B :
11 2 3 44 5 66 7 44 2 13

Power of column A :
1
4
9
2025
25
36
49
64
2916
484

Power of column B :
121
4

```

```

Command Window
New to MATLAB? See resources for Getting Started.

25
36
49
64
2916
484

Power of column B :
121
4
9
1936
25
4356
49
1936
4
169

```

Q3. Write a program to calculate the Square and Cube of any number.

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m

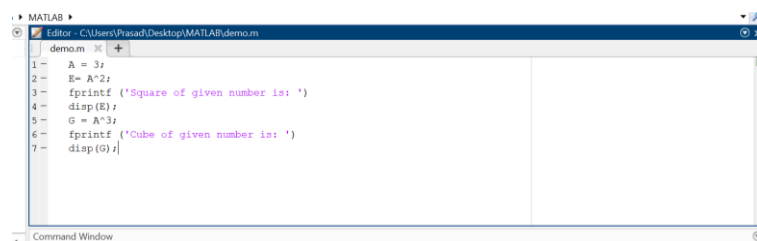
Step 3: Initialise A = 3

Step 4: So, the square and cube of the number are calculated using ^ operator.

Step 5: The number and results are printed using disp() function.

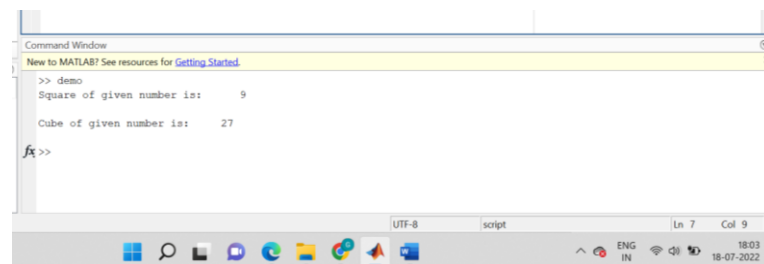
Step 6: Stop

Code:



```
demo.m
1 - A = 3;
2 - E = A^2;
3 - fprintf('Square of given number is: ')
4 - disp(E);
5 - G = A^3;
6 - fprintf('Cube of given number is: ')
7 - disp(G);
```

Output:



```
>> demo
Square of given number is:      9
Cube of given number is:     27
fx>>
```

Q4. Write a program to calculate the hypotenuse of a right-angle triangle. (Use sqrt() function)

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m.

Step 3: Now we have to just initialise the perpendicular and base of a right-angle triangle as A and B.

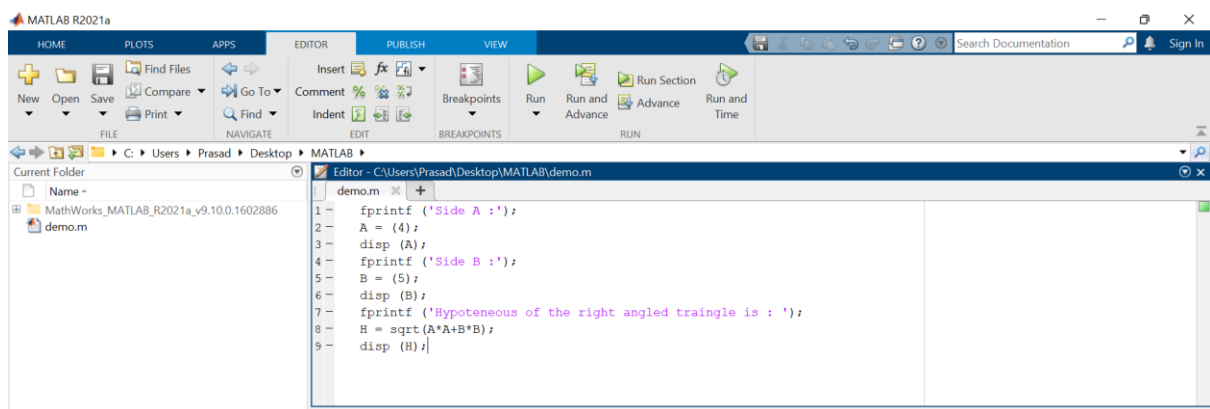
Step 4: So, the square of the A and B operator is used and sqrt() are used to calculate the square root of the squared variables and store them as Hypotenuse H .

Step 5: Also the sides and results are printed using disp() function.

Step 6: Now, the result would be printed in the command window. Comments can be added using % sign

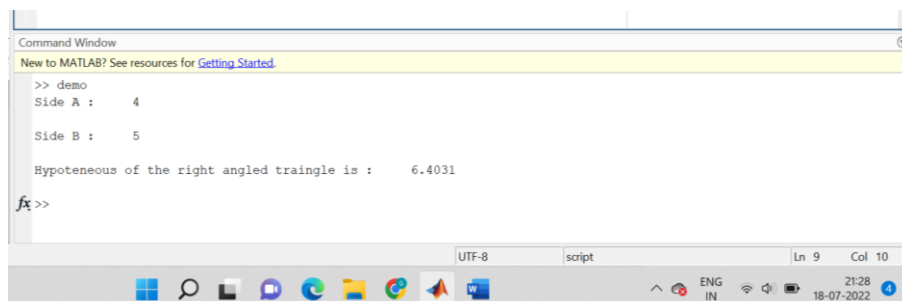
Step 7: Stop

Code:



```
1 fprintf('Side A :');
2 A = (4);
3 disp(A);
4 fprintf('Side B :');
5 B = (5);
6 disp(B);
7 fprintf('Hypoteneous of the right angled traingle is : ');
8 H = sqrt(A*A+B*B);
9 disp(H);
```

Output:



```
>> demo
Side A :    4

Side B :    5

Hypoteneous of the right angled traingle is :    6.4031

fx >>
```

Q5. Write a program to calculate the Dot Product of two vectors consisting of six elements each.

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m that would be the MATLAB editor window where the MATLAB programs can be loaded, edited and saved.

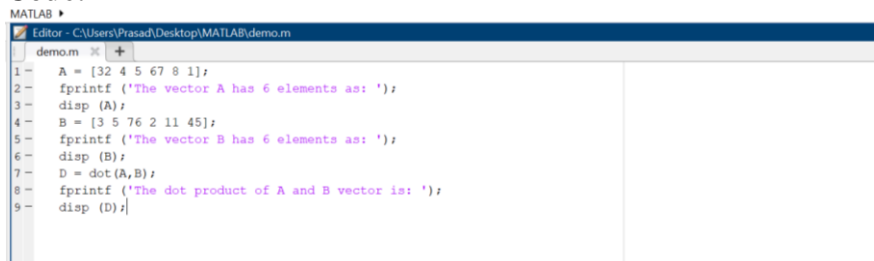
Step 3: Now we have to just initialise two vectors named as A and B each containing six elements.

Step 4: Just calculate the dot product for both vectors using dot() function.

Step 5: The results can be shown using the disp() function.


Step 6: Stop

Code:



```
MATLAB
Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m
1- A = [32 4 5 67 8 1];
2- fprintf('The vector A has 6 elements as: ');
3- disp(A);
4- B = [3 5 76 2 11 45];
5- fprintf('The vector B has 6 elements as: ');
6- disp(B);
7- D = dot(A,B);
8- fprintf('The dot product of A and B vector is: ');
9- disp(D);
```

Output:



```
Command Window
New to MATLAB? See resources for Getting Started.
>> demo
The vector A has 6 elements as:    32    4    5   67    8    1
The vector B has 6 elements as:     3    5   76    2   11   45
The dot product of A and B vector is:    763
fx>>
```


Q6. Write a program to calculate the magnitude of a vector consisting of seven elements.

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m.

Step 3: We have to now take the product of the vector with itself by just using array multiplication.

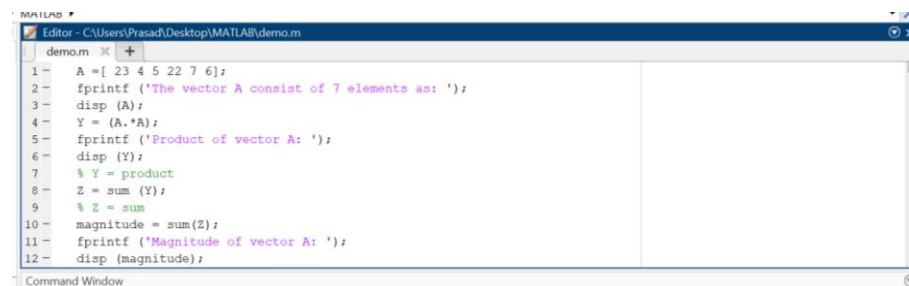
Step 4: Now take the sum function for sum of squares of elements of vector A.

Step 5: So, use the sqrt function to get the square root of the sum which is also the magnitude of the vector A.

Step 6: disp() function is used for displaying vector and results.

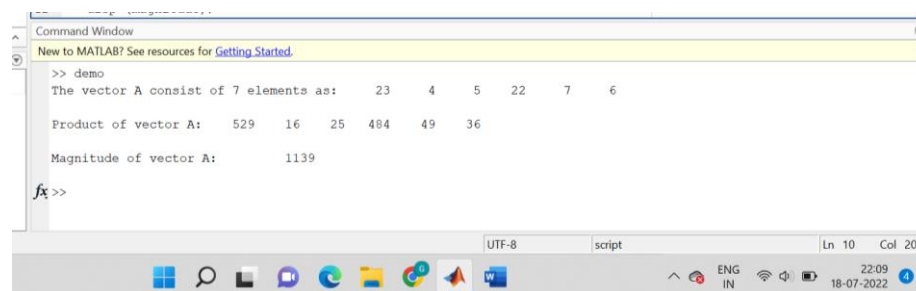
Step 7: Stop

Code:



```
1- A=[ 23 4 5 22 7 6];
2- fprintf('The vector A consist of 7 elements as: ');
3- disp(A);
4- Y=(A.*A);
5- fprintf('Product of vector A: ');
6- disp(Y);
7- % Y = product
8- Z=sum(Y);
9- % Z = sum
10- magnitude = sum(Z);
11- fprintf('Magnitude of vector A: ');
12- disp(magnitude);
```

Output:



```
>> demo
The vector A consist of 7 elements as:    23    4    5    22    7    6

Product of vector A:    529    16    25    484    49    36

Magnitude of vector A:    1139

fx>>
```

Exercise 2

Matrix Operations

Q1. Write a program to perform the scalar operations on a 4X3 matrix.

Logic:

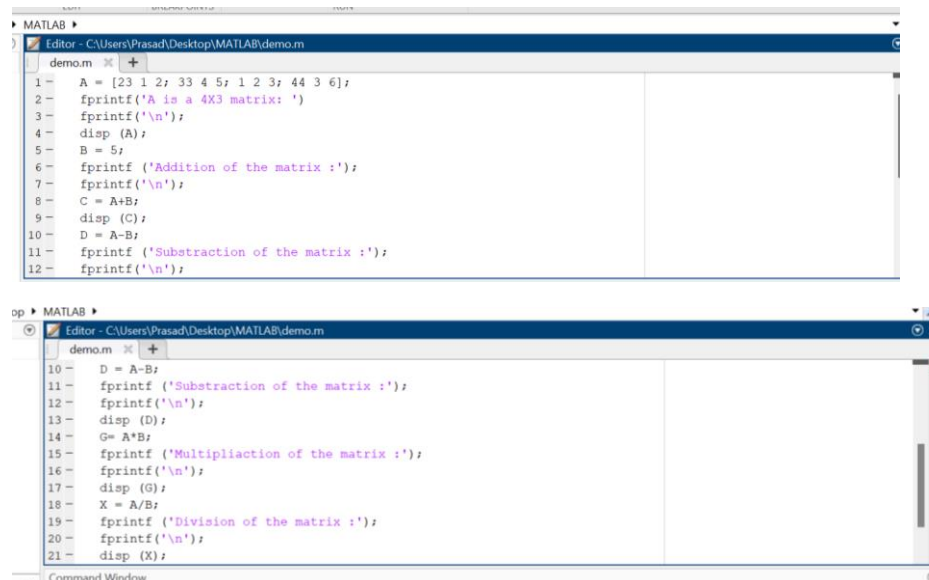
Step 1: Start

Step 2: First we have to create a new script named as demo.m

Step 3: Now just create a matrix and initialize a variable.

Step 4: So, the scalar operations are performed on matrix such as- Division is performed using ' / ' Operator. Multiplication is performed using '*' operator. Addition is performed using '+' operator. Subtraction is performed using '-' operator.

Step 5: Stop

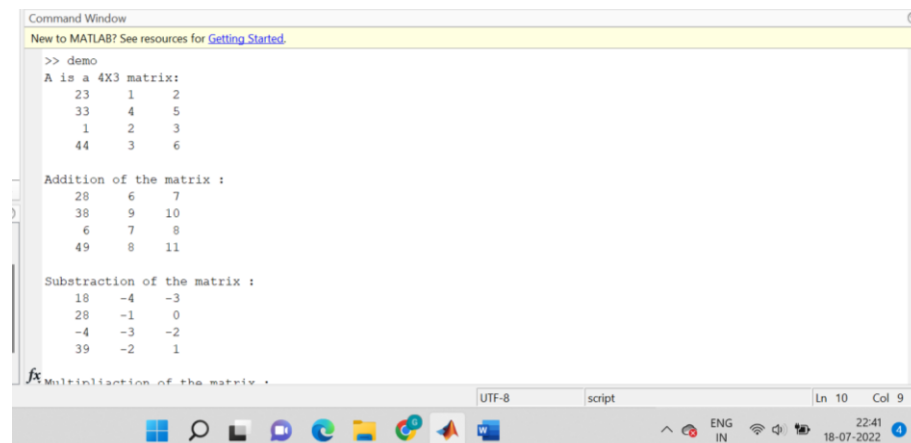
Code:


```

1- A = [23 1 2; 33 4 5; 1 2 3; 44 3 6];
2- fprintf('A is a 4X3 matrix: ');
3- fprintf('\n');
4- disp(A);
5- B = 5;
6- fprintf('Addition of the matrix :');
7- fprintf('\n');
8- C = A+B;
9- disp(C);
10- D = A-B;
11- fprintf('Subtraction of the matrix :');
12- fprintf('\n');

10- D = A-B;
11- fprintf('Subtraction of the matrix :');
12- fprintf('\n');
13- disp(D);
14- G= A*B;
15- fprintf('Multipliacion of the matrix :');
16- fprintf('\n');
17- disp(G);
18- X = A/B;
19- fprintf('Division of the matrix :');
20- fprintf('\n');
21- disp(X);

```

Output:


```

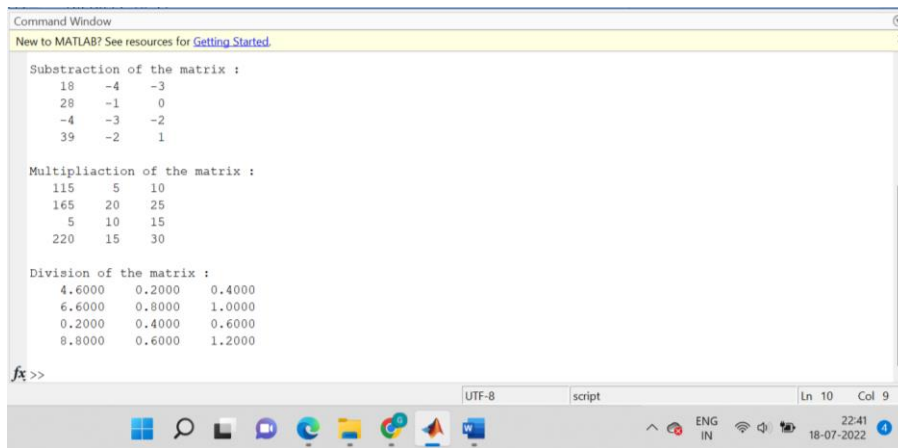
>> demo
A is a 4X3 matrix:
    23     1     2
    33     4     5
     1     2     3
    44     3     6

Addition of the matrix :
    28     6     7
    38     9    10
     6     7     8
    49     8    11

Subtraction of the matrix :
    18    -4    -3
    28    -1     0
    -4    -3    -2
    39    -2     1

Multipliacion of the matrix :

```



The image shows a screenshot of the MATLAB Command Window. At the top, there is a yellow banner that says "New to MATLAB? See resources for [Getting Started](#)". Below this, the window displays the results of three matrix operations. First, "Subtraction of the matrix :" followed by a 4x3 matrix: $\begin{bmatrix} 18 & -4 & -3 \\ 28 & -1 & 0 \\ -4 & -3 & -2 \\ 39 & -2 & 1 \end{bmatrix}$. Next, "Multipliacion of the matrix :" followed by a 4x3 matrix: $\begin{bmatrix} 115 & 5 & 10 \\ 165 & 20 & 25 \\ 5 & 10 & 15 \\ 220 & 15 & 30 \end{bmatrix}$. Finally, "Division of the matrix :" followed by a 4x3 matrix of decimal values: $\begin{bmatrix} 4.6000 & 0.2000 & 0.4000 \\ 6.6000 & 0.8000 & 1.0000 \\ 0.2000 & 0.4000 & 0.6000 \\ 8.8000 & 0.6000 & 1.2000 \end{bmatrix}$. The Command Window prompt "fx >>" is visible at the bottom left. The status bar at the bottom shows "UTF-8", "script", "Ln 10", "Col 9", and the system clock "22:41 18-07-2022".

```
Command Window
New to MATLAB? See resources for Getting Started

Subtraction of the matrix :
18  -4  -3
28  -1   0
-4  -3  -2
39  -2   1

Multipliacion of the matrix :
115   5  10
165  20  25
  5  10  15
220  15  30

Division of the matrix :
4.6000  0.2000  0.4000
6.6000  0.8000  1.0000
0.2000  0.4000  0.6000
8.8000  0.6000  1.2000

fx >>
```

Q2. Write a program to perform**1. Matrix multiplication (Direct)****Logic:**

Step 1: Start

Step 2: First we have to create a new script named as demo.m

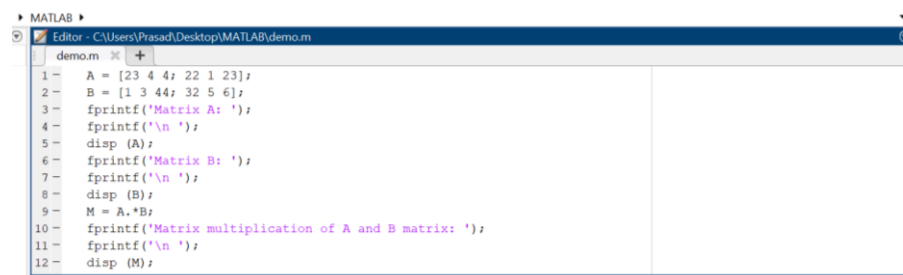
Step 3: Now just make 2 matrices.

Step 4: We have to use '*' symbol for direct multiplication.

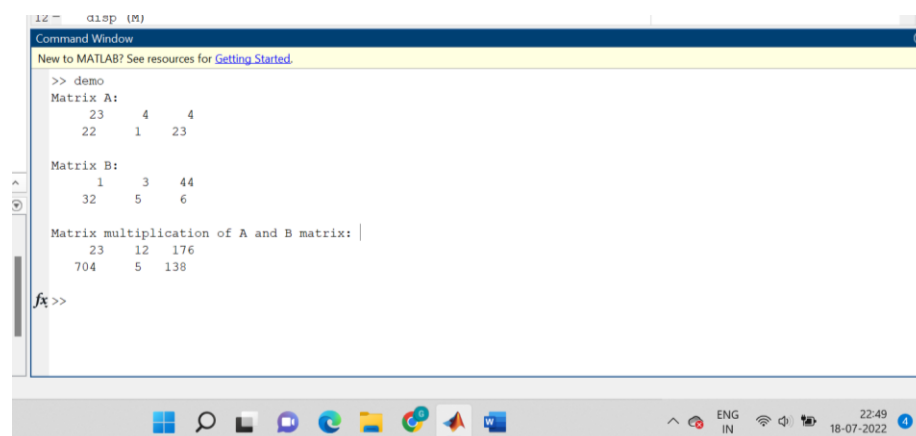
Step 5: The result would be printed in the command window.

Step 6: Use disp() function for displaying result.

Step 7: Stop

Code:

```
1 A = [23 4 4; 22 1 23];
2 B = [1 3 44; 32 5 6];
3 fprintf('Matrix A: ');
4 fprintf('\n ');
5 disp(A);
6 fprintf('Matrix B: ');
7 fprintf('\n ');
8 disp(B);
9 M = A.*B;
10 fprintf('Matrix multiplication of A and B matrix: ');
11 fprintf('\n ');
12 disp(M);
```

Output:

```
>> demo
Matrix A:
    23     4     4
    22     1    23

Matrix B:
     1     3    44
    32     5     6

Matrix multiplication of A and B matrix:
    23    12    176
    704     5    138
```

2. Prod ()**Logic:**

Step 1: Start

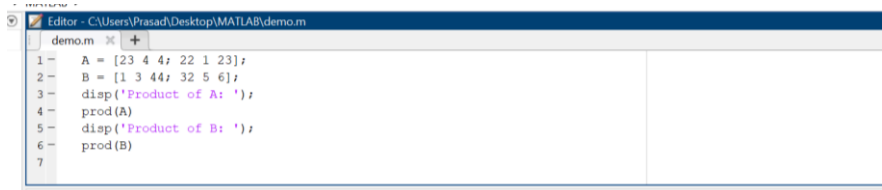
Step 2: First we have to create a new script named as demo.m.

Step 3: Now just create 2 matrices.

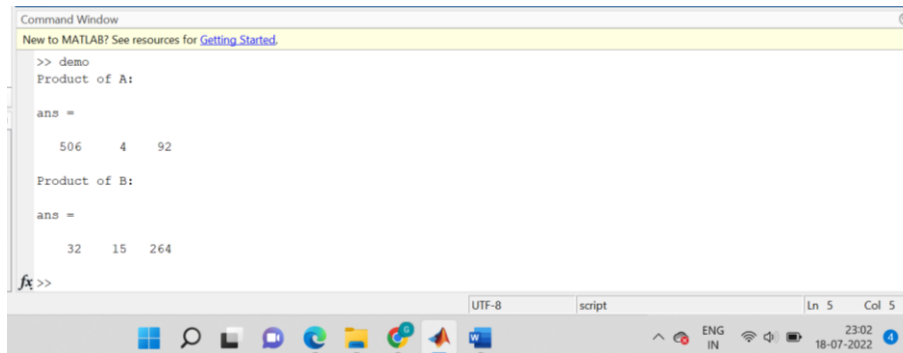
Step 4: So, use the Prod() to calculate the product of each column.

Step 5: Result would be printed in the command window. Use disp() for displaying result.

Step 6: Stop

Code:

```
demo.m
1 A = [23 4 4; 22 1 23];
2 B = [1 3 44; 32 5 6];
3 disp('Product of A: ');
4 prod(A)
5 disp('Product of B: ');
6 prod(B)
7
```

Output:

```
Command Window
New to MATLAB? See resources for Getting Started.
>> demo
Product of A:
ans =
    506     4     92
Product of B:
ans =
    32    15    264
fx>>
```

Q3. Write a program to concatenate two 4X4 matrices vertically and horizontally.

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m.

Step 3: Now just create a two 4X4 matrices.

Step 4: So, we have to concatenate horizontally so we have to use square brackets and a comma to separate the variables.

Step 5: Now, to concatenate vertically use square brackets and a semi colon to separate the variables.

Step 6: We know that the matrix and results are printed using disp() function.

Step 7: Stop

Code:

```
demo.m
1- A = [21 44 22 70; 85 3 2 2; 9 14 32 16; 55 31 21 5];
2- B = [21 4 2 70; 5 3 2 2; 9 14 32 16; 5 3 2 5];
3- fprintf("4 x 4 matrix A: \n");
4- disp(A);
5- fprintf("4 x 4 matrix B: \n");
6- disp(B);
7- %horizontal concat
8- H = [A B];
9- fprintf(" Horizontal Concat : \n");
10- disp(H);
11- %vertical concat
12- V = [A; B];
13- fprintf(" Vertical Concat: \n");
14- disp(V);
```

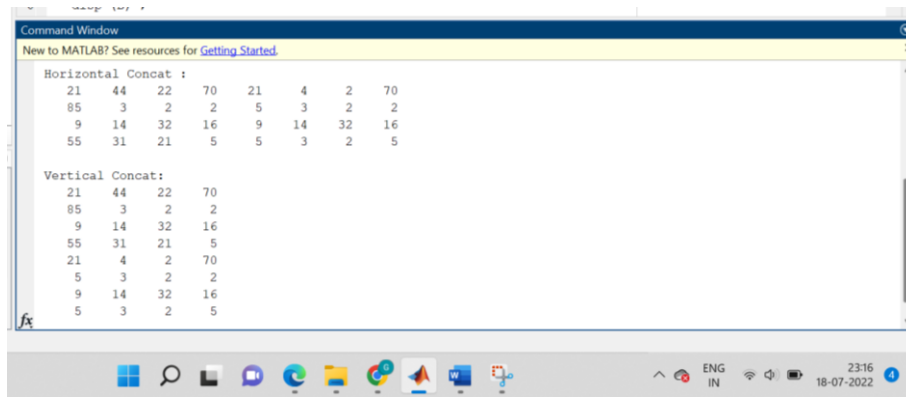
Output:

```
6- disp(B);
Command Window
New to MATLAB? See resources for Getting Started.
>> demo
4 x 4 matrix A:
21    44    22    70
85     3     2     2
 9    14    32    16
55    31    21     5

4 x 4 matrix B:
21     4     2    70
 5     3     2     2
 9    14    32    16
 5     3     2     5

Horizontal Concat :
21    44    22    70    21     4     2    70
85     3     2     2     5     3     2     2
 9    14    32    16     9    14    32    16
55    31    21     5     5     3     2     5

UTF-8    script    Ln 10 Col 11
23:16
18-07-2022
```



The image shows a MATLAB Command Window with a blue title bar. A yellow banner at the top says "New to MATLAB? See resources for [Getting Started](#)." Below this, the text "Horizontal Concat:" is followed by a 4x8 matrix of numbers. Then, the text "Vertical Concat:" is followed by a 12x4 matrix of numbers. The window is part of a desktop environment with a taskbar at the bottom showing various application icons and system status icons on the right.

```
Command Window
New to MATLAB? See resources for Getting Started.

Horizontal Concat :
21 44 22 70 21 4 2 70
85 3 2 2 5 3 2 2
9 14 32 16 9 14 32 16
55 31 21 5 5 3 2 5

Vertical Concat:
21 44 22 70
85 3 2 2
9 14 32 16
55 31 21 5
21 4 2 70
5 3 2 2
9 14 32 16
5 3 2 5
```

Q4. Write a program to calculate the determinant and inverse of the 3X3 matrix.**Logic:**

Step 1: Start

Step 2: First we have to create a new script named as demo.m.

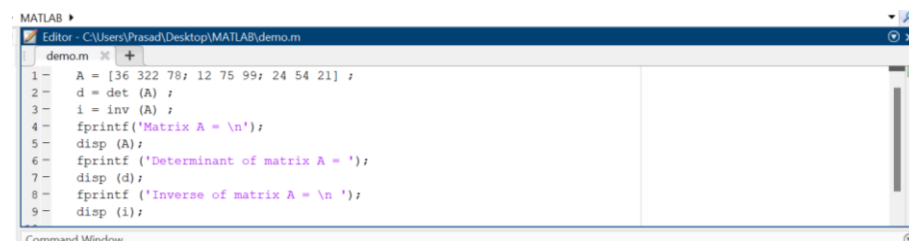
Step 3: Now we just have to create a 3 x 3 matrix A.

Step 4: The $d = \det(A)$ that will return the determinant of square matrix A.

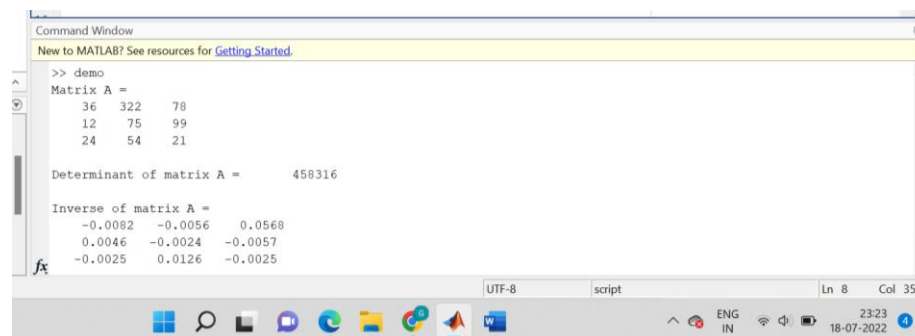
Step 5: We have $i = \text{inv}(A)$ that will just compute the inverse of square matrix A.

Step 6: We know that the matrix and results are printed using `disp()` function.

Step 7: Stop

Code:

```
1- A = [36 322 78; 12 75 99; 24 54 21];
2- d = det(A);
3- i = inv(A);
4- fprintf('Matrix A = \n');
5- disp(A);
6- fprintf('Determinant of matrix A = ');
7- disp(d);
8- fprintf('Inverse of matrix A = \n ');
9- disp(i);
```

Output:

```
>> demo
Matrix A =
    36    322    78
    12     75    99
    24     54    21

Determinant of matrix A =      458316

Inverse of matrix A =
   -0.0082   -0.0056    0.0568
    0.0046   -0.0024   -0.0057
   -0.0025    0.0126   -0.0025
```


Exercise 3

Arrays Operations (2-D and Multi-Dimensional)

Q1. Write a program to create an array of size 5X10 to perform the following functions: (Use new variables to store the values for each sub-question).

Logic:

Step 1: Start

Step 2: We have to first create a new script named as demo.m

Step 3: So, first to access the columns from 5 to 10, ':' is used. It's a range operator.

Step 4: Now if we want to access the rows from 2 to 5, ':' is used. It's a range operator.

Step 5: Now if we want to access any one number within an array, specify the row and column.

Step 6: Then to access the last row, enter the row number and just leave the column part empty.

Step 7: If we want to access the 6th column, enter the column number and now just you have to leave the row part empty.

Step 8: Also, if we want to choose the even rows in the array, enter 2:2:end so this simply means that the range will start from index 2 with a gap of 2.

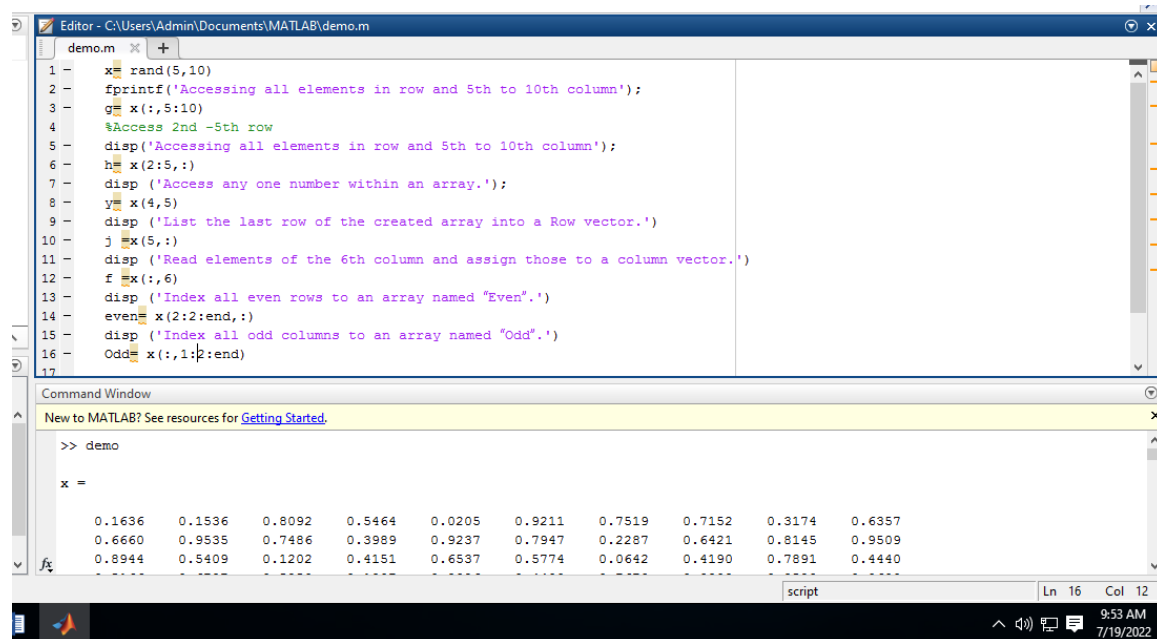
Step 9: Also, if want to choose the odd columns in the array, we have to just enter 1:2:end so this simply means that the range will start from index 1 with a gap of 2.

Step 10: Now your result would be printed in the command window and the comments can be added using % sign if you want to.

Step 11: Stop

1. Access 5th -10th column.

Code:



```

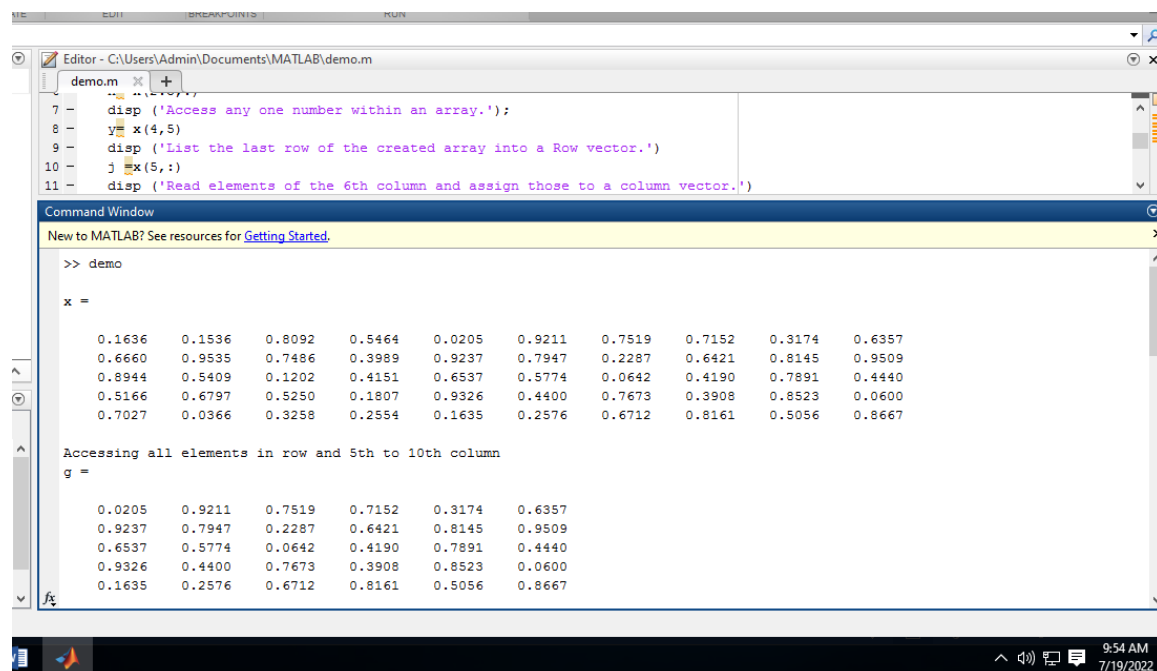
demo.m
1 - x = rand(5,10)
2 - fprintf('Accessing all elements in row and 5th to 10th column');
3 - g = x(:,5:10)
4 - %Access 2nd -5th row
5 - disp('Accessing all elements in row and 5th to 10th column');
6 - h = x(2:5,:)
7 - disp('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp('Read elements of the 6th column and assign those to a column vector. ')
12 - f = x(:,6)
13 - disp('Index all even rows to an array named "Even".')
14 - even = x(2:2:end,:)
15 - disp('Index all odd columns to an array named "Odd".')
16 - Odd = x(:,1:2:end)
17
Command Window
New to MATLAB? See resources for Getting Started.
>> demo

x =

    0.1636    0.1536    0.8092    0.5464    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440

```

Output:



```

demo.m
7 - disp('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp('Read elements of the 6th column and assign those to a column vector. ')
12
Command Window
New to MATLAB? See resources for Getting Started.
>> demo

x =

    0.1636    0.1536    0.8092    0.5464    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

Accessing all elements in row and 5th to 10th column
g =

    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

```

2. Access 2nd -5th row.

Code:

The screenshot shows the MATLAB Editor with a file named 'demo.m' and the Command Window displaying the execution results.

Editor - demo.m:

```

7 - disp('Access any one number within an array. ');
8 - y = x(4,5);
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:);
11 - disp('Read elements of the 6th column and assign those to a column vector. ')

```

Command Window:

```

>> demo

x =

    0.1636    0.1536    0.8092    0.5464    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

Accessing all elements in row and 5th to 10th column
g =

    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

```

Output:

The screenshot shows the MATLAB Command Window displaying the output of the script.

```

h =

    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

Access any one number within an array.

y =

    0.9326

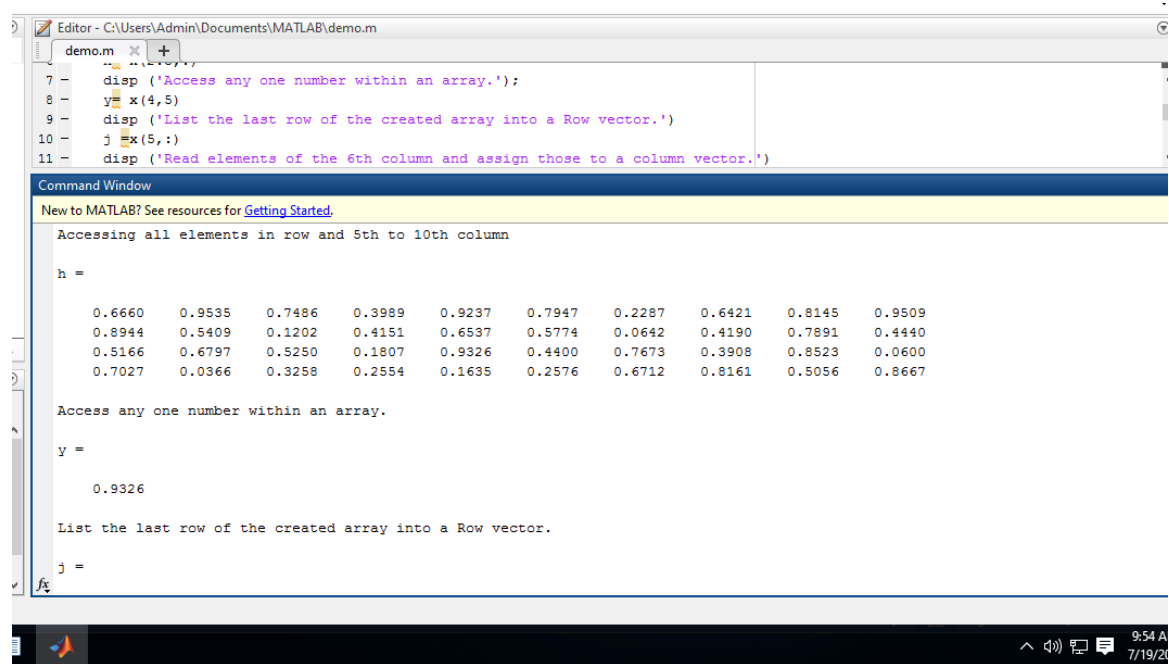
List the last row of the created array into a Row vector.

j =

```

3. Access any one number within an array.

Code:



The image shows a MATLAB Editor window with a file named 'demo.m' and a Command Window below it. The code in the editor performs the following steps:

```

7 - disp ('Access any one number within an array. ');
8 - y = x(4,5);
9 - disp ('List the last row of the created array into a Row vector. ');
10 - j = x(5,:);
11 - disp ('Read elements of the 6th column and assign those to a column vector. ');

```

The Command Window displays the following output:

```

New to MATLAB? See resources for Getting Started.

Accessing all elements in row and 5th to 10th column

h =

    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

Access any one number within an array.

y =

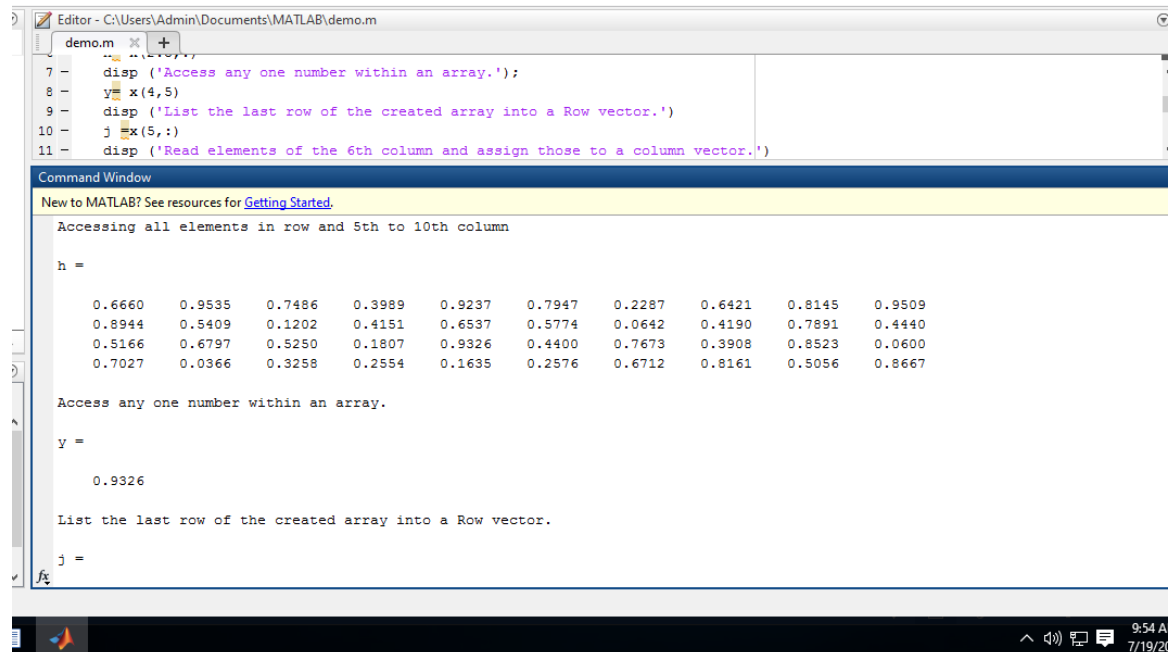
    0.9326

List the last row of the created array into a Row vector.

j =

```

Output:



The image shows a MATLAB Editor window with a file named 'demo.m' and a Command Window below it. The code in the editor performs the following steps:

```

7 - disp ('Access any one number within an array. ');
8 - y = x(4,5);
9 - disp ('List the last row of the created array into a Row vector. ');
10 - j = x(5,:);
11 - disp ('Read elements of the 6th column and assign those to a column vector. ');

```

The Command Window displays the following output:

```

New to MATLAB? See resources for Getting Started.

Accessing all elements in row and 5th to 10th column

h =

    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

Access any one number within an array.

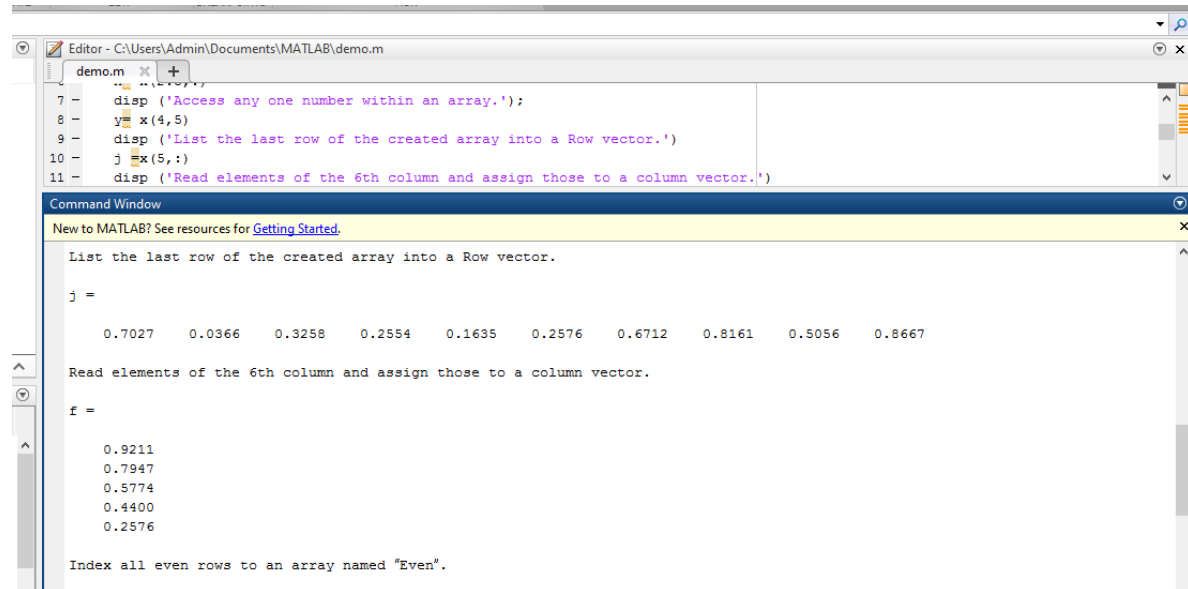
y =

    0.9326

List the last row of the created array into a Row vector.

j =

```

4. List the last row of the created array into a Row vector.**Code:**

The image shows the MATLAB Editor and Command Window. The Editor displays the following code in `demo.m`:

```
7 - disp('Access any one number within an array.');
```

```
8 - y = x(4,5)
```

```
9 - disp('List the last row of the created array into a Row vector.')
```

```
10 - j = x(5,:)
```

```
11 - disp('Read elements of the 6th column and assign those to a column vector.')
```

The Command Window shows the execution of the code:

```
List the last row of the created array into a Row vector.
```

```
j =
```

```
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667
```

```
Read elements of the 6th column and assign those to a column vector.
```

```
f =
```

```
    0.9211
```

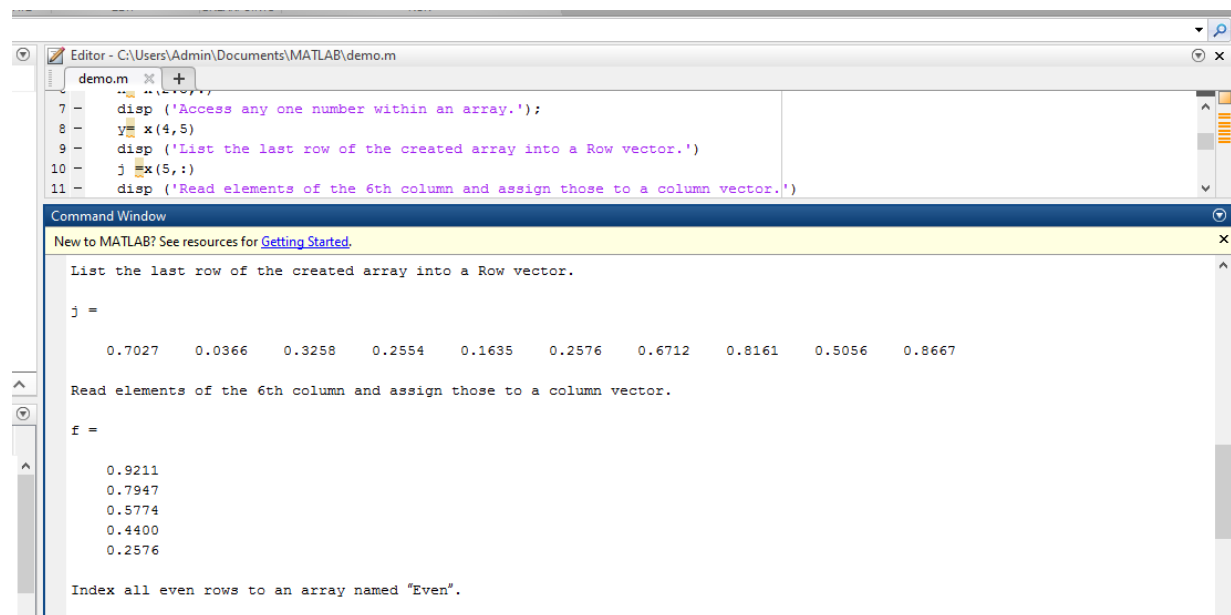
```
    0.7947
```

```
    0.5774
```

```
    0.4400
```

```
    0.2576
```

```
Index all even rows to an array named "Even".
```

Output:

The image shows the MATLAB Editor and Command Window. The Editor displays the following code in `demo.m`:

```
7 - disp('Access any one number within an array.');
```

```
8 - y = x(4,5)
```

```
9 - disp('List the last row of the created array into a Row vector.')
```

```
10 - j = x(5,:)
```

```
11 - disp('Read elements of the 6th column and assign those to a column vector.')
```

The Command Window shows the execution of the code:

```
List the last row of the created array into a Row vector.
```

```
j =
```

```
    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667
```

```
Read elements of the 6th column and assign those to a column vector.
```

```
f =
```

```
    0.9211
```

```
    0.7947
```

```
    0.5774
```

```
    0.4400
```

```
    0.2576
```

```
Index all even rows to an array named "Even".
```

5. Read elements of the 6th column and assign those to a column vector.

Code:

The screenshot shows the MATLAB Editor with a script named 'demo.m' and the Command Window displaying the output of the script. The script contains the following code:

```

7 - disp('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp('Read elements of the 6th column and assign those to a column vector. ')

```

The Command Window output is as follows:

```

New to MATLAB? See resources for Getting Started.

Index all even rows to an array named "Even".

even =

    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600

Index all odd columns to an array named "Odd".

Odd =

    0.1636    0.8092    0.0205    0.7519    0.3174
    0.6660    0.7486    0.9237    0.2287    0.8145
    0.8944    0.1202    0.6537    0.0642    0.7891
    0.5166    0.5250    0.9326    0.7673    0.8523
    0.7027    0.3258    0.1635    0.6712    0.5056

```

Output:

The screenshot shows the MATLAB Editor with the same script 'demo.m' and the Command Window displaying the output of the script. The script contains the following code:

```

7 - disp('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp('Read elements of the 6th column and assign those to a column vector. ')

```

The Command Window output is as follows:

```

New to MATLAB? See resources for Getting Started.

List the last row of the created array into a Row vector.

j =

    0.7027    0.0366    0.3258    0.2554    0.1635    0.2576    0.6712    0.8161    0.5056    0.8667

Read elements of the 6th column and assign those to a column vector.

f =

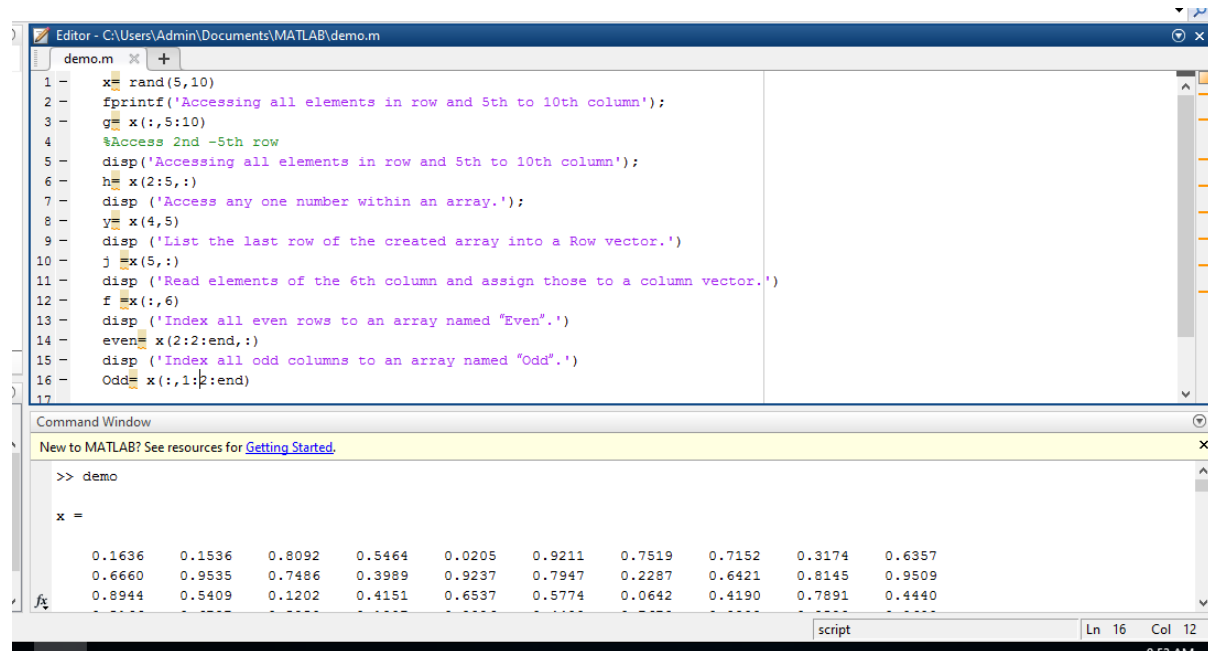
    0.9211
    0.7947
    0.5774
    0.4400
    0.2576

Index all even rows to an array named "Even".

```

6. Index all even rows to an array named "Even".

Code:



```

demo.m
1 - x = rand(5,10)
2 - fprintf('Accessing all elements in row and 5th to 10th column');
3 - g = x(:,5:10)
4 - %Access 2nd -5th row
5 - disp('Accessing all elements in row and 5th to 10th column');
6 - h = x(2:5,:)
7 - disp ('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp ('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp ('Read elements of the 6th column and assign those to a column vector. ')
12 - f = x(:,6)
13 - disp ('Index all even rows to an array named "Even".')
14 - even = x(2:2:end,:)
15 - disp ('Index all odd columns to an array named "Odd".')
16 - Odd = x(:,1:2:end)
17

```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```

>> demo

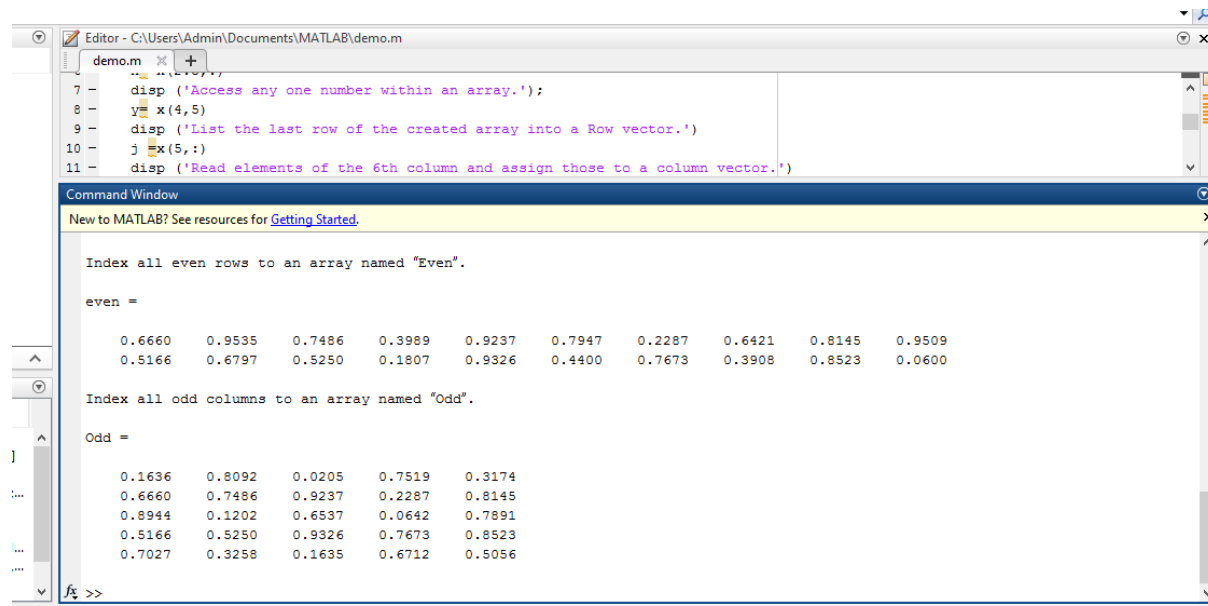
x =

    0.1636    0.1536    0.8092    0.5464    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440

```

script Ln 16 Col 12

Output:



```

demo.m
7 - disp ('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp ('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp ('Read elements of the 6th column and assign those to a column vector. ')

```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```

Index all even rows to an array named "Even".

even =

    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600

Index all odd columns to an array named "Odd".

Odd =

    0.1636    0.8092    0.0205    0.7519    0.3174
    0.6660    0.7486    0.9237    0.2287    0.8145
    0.8944    0.1202    0.6537    0.0642    0.7891
    0.5166    0.5250    0.9326    0.7673    0.8523
    0.7027    0.3258    0.1635    0.6712    0.5056

```

f6 >>

7. Index all odd columns to an array named "Odd".

Code:

```

1 - x = rand(5,10)
2 - fprintf('Accessing all elements in row and 5th to 10th column');
3 - g = x(:,5:10)
4 - %Access 2nd -5th row
5 - disp('Accessing all elements in row and 5th to 10th column');
6 - h = x(2:5,:)
7 - disp('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp('Read elements of the 6th column and assign those to a column vector. ')
12 - f = x(:,6)
13 - disp('Index all even rows to an array named "Even".')
14 - even = x(2:2:end,:)
15 - disp('Index all odd columns to an array named "Odd".')
16 - Odd = x(:,1:2:end)
17

```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```

>> demo

x =

    0.1636    0.1536    0.8092    0.5464    0.0205    0.9211    0.7519    0.7152    0.3174    0.6357
    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.8944    0.5409    0.1202    0.4151    0.6537    0.5774    0.0642    0.4190    0.7891    0.4440

```

script Ln 16 Col 12

Output:

```

7 - disp('Access any one number within an array. ');
8 - y = x(4,5)
9 - disp('List the last row of the created array into a Row vector. ')
10 - j = x(5,:)
11 - disp('Read elements of the 6th column and assign those to a column vector. ')

```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```

Index all even rows to an array named "Even".

even =

    0.6660    0.9535    0.7486    0.3989    0.9237    0.7947    0.2287    0.6421    0.8145    0.9509
    0.5166    0.6797    0.5250    0.1807    0.9326    0.4400    0.7673    0.3908    0.8523    0.0600

Index all odd columns to an array named "Odd".

Odd =

    0.1636    0.8092    0.0205    0.7519    0.3174
    0.6660    0.7486    0.9237    0.2287    0.8145
    0.8944    0.1202    0.6537    0.0642    0.7891
    0.5166    0.5250    0.9326    0.7673    0.8523
    0.7027    0.3258    0.1635    0.6712    0.5056

```

ft >>

Q2. Write a program to read all the elements of the first seven columns from the previously created array (5X10 array created in the previous question), to perform the following operations: (Use new variables to store the values for each sub-question).

Logic:

Step 1: Start

Step 2: First we have to create a new script named as demo.m

Step 3: To find reciprocal, the function reci() is used.

Step 4: So, basically to find sum of 3rd column and 6th column we have to first access the columns and then find sum of the entire column using sum() and then add the sums of 2 columns.

Step 5: Now if you want to access any 4 columns and store it in a new variable we have to use reshape() to reshape the matrix.

Step 6: So, the Sqrt() is used to find square root of an array.

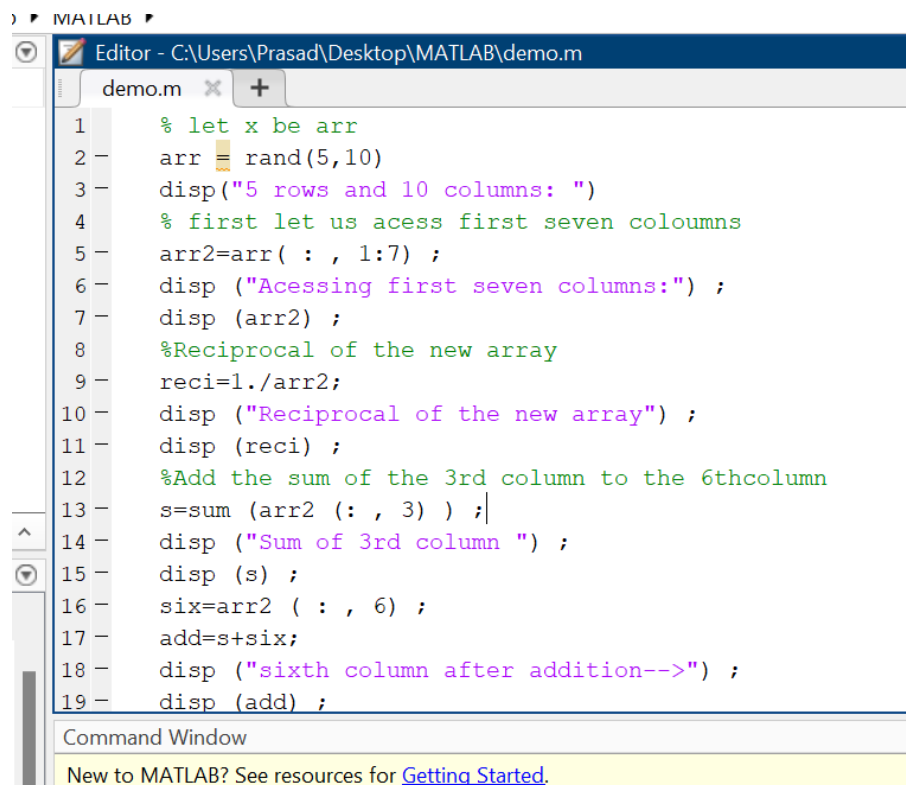
Step 7: We use ^ is used to find square of an array.

Step 8: Your result would be printed in the command window and the comments can be added using % sign if you wish to.

Step 9: Stop.

1. Reciprocal of the new array.

Code:



```
1 % let x be arr
2 arr = rand(5,10)
3 disp("5 rows and 10 columns: ")
4 % first let us access first seven columns
5 arr2=arr( : , 1:7) ;
6 disp ("Accessing first seven columns:") ;
7 disp (arr2) ;
8 %Reciprocal of the new array
9 reci=1./arr2;
10 disp ("Reciprocal of the new array") ;
11 disp (reci) ;
12 %Add the sum of the 3rd column to the 6thcolumn
13 s=sum (arr2 ( : , 3) ) ;
14 disp ("Sum of 3rd column ") ;
15 disp (s) ;
16 six=arr2 ( : , 6) ;
17 add=s+six;
18 disp ("sixth column after addition-->") ;
19 disp (add) ;
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

Output:

```

1 % let x be arr
Command Window
New to MATLAB? See resources for Getting Started.
>> demo

arr =

    0.3015    0.6665    0.0326    0.3689    0.6448    0.1206    0.2518    0.9827    0.9063    0.0225
    0.7011    0.1781    0.5612    0.4607    0.3763    0.5895    0.2904    0.7302    0.8797    0.4253
    0.6663    0.1280    0.8819    0.9816    0.1909    0.2262    0.6171    0.3439    0.8178    0.3127
    0.5391    0.9991    0.6692    0.1564    0.4283    0.3846    0.2653    0.5841    0.2607    0.1615
    0.6981    0.1711    0.1904    0.8555    0.4820    0.5830    0.8244    0.1078    0.5944    0.1788

5 rows and 10 columns:
Accessing first seven columns:
    0.3015    0.6665    0.0326    0.3689    0.6448    0.1206    0.2518
    0.7011    0.1781    0.5612    0.4607    0.3763    0.5895    0.2904
    0.6663    0.1280    0.8819    0.9816    0.1909    0.2262    0.6171
    0.5391    0.9991    0.6692    0.1564    0.4283    0.3846    0.2653
    0.6981    0.1711    0.1904    0.8555    0.4820    0.5830    0.8244

Reciprocal of the new array
    3.3172    1.5003    30.6741    2.7106    1.5510    8.2911    3.9713
fx 1.4263    5.6138    1.7819    2.1705    2.6577    1.6963    3.4430

```

```

1 % let x be arr
Command Window
New to MATLAB? See resources for Getting Started.
Reciprocal of the new array
    3.3172    1.5003    30.6741    2.7106    1.5510    8.2911    3.9713
    1.4263    5.6138    1.7819    2.1705    2.6577    1.6963    3.4430
    1.5007    7.8116    1.1340    1.0187    5.2377    4.4211    1.6205
    1.8549    1.0009    1.4944    6.3937    2.3351    2.6000    3.7696
    1.4324    5.8438    5.2512    1.1689    2.0746    1.7153    1.2130

Sum of 3rd column
    2.3353

sixth column after addition-->
    2.4559
    2.9248
    2.5615
    2.7199
    2.9183

Reshaped:
    0.6665    0.1711    0.6692    0.9816    0.3763
    0.1781    0.0326    0.1904    0.1564    0.1909
fx 0.1280    0.5612    0.3689    0.8555    0.4283

```

2. Add the sum of the 3rd column to the 6th column.

Code:

```

MATLAB
Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m
1 % let x be arr
2 arr = rand(5,10)
3 disp("5 rows and 10 columns: ")
4 % first let us access first seven columns
5 arr2=arr( : , 1:7) ;
6 disp ("Accessing first seven columns:") ;
7 disp (arr2) ;
8 %Reciprocal of the new array
9 reci=1./arr2;
10 disp ("Reciprocal of the new array") ;
11 disp (reci) ;
12 %Add the sum of the 3rd column to the 6thcolumn
13 s=sum (arr2 ( : , 3) ) ;
14 disp ("Sum of 3rd column ") ;
15 disp (s) ;
16 six=arr2 ( : , 6) ;
17 add=s+six;
18 disp ("sixth column after addition-->") ;
19 disp (add) ;
Command Window

```

Output:

```

1 % let x be arr
Command Window
New to MATLAB? See resources for Getting Started.
Reciprocal of the new array
3.3172    1.5003    30.6741    2.7106    1.5510    8.2911    3.9713
1.4263    5.6138    1.7819    2.1705    2.6577    1.6963    3.4430
1.5007    7.8116    1.1340    1.0187    5.2377    4.4211    1.6205
1.8549    1.0009    1.4944    6.3937    2.3351    2.6000    3.7696
1.4324    5.8438    5.2512    1.1689    2.0746    1.7153    1.2130

Sum of 3rd column
2.3353

sixth column after addition-->
2.4559
2.9248
2.5615
2.7199
2.9183

Reshaped:
0.6665    0.1711    0.6692    0.9816    0.3763
0.1781    0.0326    0.1904    0.1564    0.1909
fx 0.1280    0.5612    0.3689    0.8555    0.4283

```

3. Access any of the four columns and reshape it to the 4X5 matrix.

Code:

```

MATLAB ▶
Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m x +
1 % let x be arr
2 arr = rand(5,10)
3 disp("5 rows and 10 columns: ")
4 % first let us access first seven columns
5 arr2=arr( : , 1:7) ;
6 disp ("Accessing first seven columns:") ;
7 disp (arr2) ;
8 %Reciprocal of the new array
9 reci=1./arr2;
10 disp ("Reciprocal of the new array") ;
11 disp (reci) ;
12 %Add the sum of the 3rd column to the 6thcolumn
13 s=sum (arr2 ( : , 3) ) ;
14 disp ("Sum of 3rd column ") ;
15 disp (s) ;
16 six=arr2 ( : , 6) ;
17 add=s+six;
18 disp ("sixth column after addition:") ;
19 disp (add) ;

Command Window
New to MATLAB? See resources for Getting Started.

0.4874    0.0293    0.0363    0.7319    0.2323    0.3399    0.6796

fx >>
UTF-8 script

```

Output:

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m x +
1 % let x be arr

Command Window
New to MATLAB? See resources for Getting Started.

sixth column after addition-->
    2.4559
    2.9248
    2.5615
    2.7199
    2.9183

Reshaped:
    0.6665    0.1711    0.6692    0.9816    0.3763
    0.1781    0.0326    0.1904    0.1564    0.1909
    0.1280    0.5612    0.3689    0.8555    0.4283
    0.9991    0.8819    0.4607    0.6448    0.4820

Square root of the new array:
    0.5490    0.8164    0.1806    0.6074    0.8030    0.3473    0.5018
    0.8373    0.4221    0.7491    0.6788    0.6134    0.7678    0.5389
    0.8163    0.3578    0.9391    0.9908    0.4369    0.4756    0.7856
    0.7343    0.9995    0.8180    0.3955    0.6544    0.6202    0.5151
    0.8355    0.4137    0.4364    0.9249    0.6943    0.7635    0.9080

fx

```

4. Square root of the new array.

Code:

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m x +
14 - disp ("Sum of 3rd column ") ;
15 - disp (s) ;
16 - six=arr2 ( : , 6) ;
17 - add=s+six;
18 - disp ("sixth column after addition:") ;
19 - disp (add) ;
20 %access and reshape in 4x5
21 - arr3=arr2 ( : , 2:5) ;
22 - res=reshape (arr3, 4, 5) ;
23 - disp ("Reshaped:") ;
24 - disp (res) ;
25 %Square root of the new array
26 - sq=sqrt (arr2) ;
27 - disp ("Square root of the new array:") ;
28 - disp (sq) ;
29 %Square of the new array
30 - new= arr2.^2;
31 - disp ("Square of the new array:") ;
32 - disp (new) ;

Command Window
New to MATLAB? See resources for Getting Started.

0.4874    0.0293    0.0363    0.7319    0.2323    0.3399    0.6796

```

Output:

```

EDIT BREAKPOINTS RUN
MATLAB ▶
Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m x +
1 % let x be arr

Command Window
New to MATLAB? See resources for Getting Started.

Reshaped:
0.6665    0.1711    0.6692    0.9816    0.3763
0.1781    0.0326    0.1904    0.1564    0.1909
0.1280    0.5612    0.3689    0.8555    0.4283
0.9991    0.8819    0.4607    0.6448    0.4820

Square root of the new array:
0.5490    0.8164    0.1806    0.6074    0.8030    0.3473    0.5018
0.8373    0.4221    0.7491    0.6788    0.6134    0.7678    0.5389
0.8163    0.3578    0.9391    0.9908    0.4369    0.4756    0.7856
0.7343    0.9995    0.8180    0.3955    0.6544    0.6202    0.5151
0.8355    0.4137    0.4364    0.9249    0.6943    0.7635    0.9080

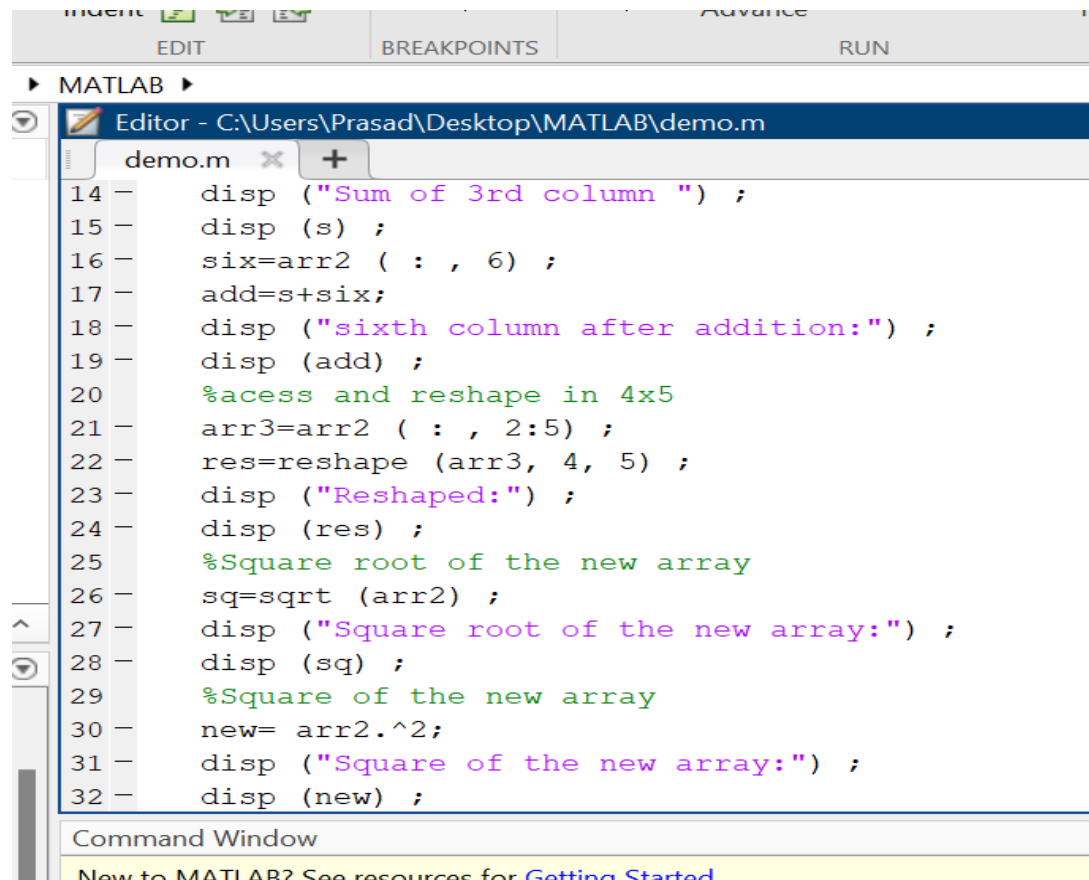
Square of the new array:
0.0909    0.4443    0.0011    0.1361    0.4157    0.0145    0.0634
0.4915    0.0317    0.3149    0.2123    0.1416    0.3475    0.0844
0.4440    0.0164    0.7777    0.9636    0.0365    0.0512    0.3808
0.2907    0.9982    0.4478    0.0245    0.1834    0.1479    0.0704
0.4874    0.0293    0.0363    0.7319    0.2323    0.3399    0.6796

fx >>

```

5. Square of the new array.

Code:



```

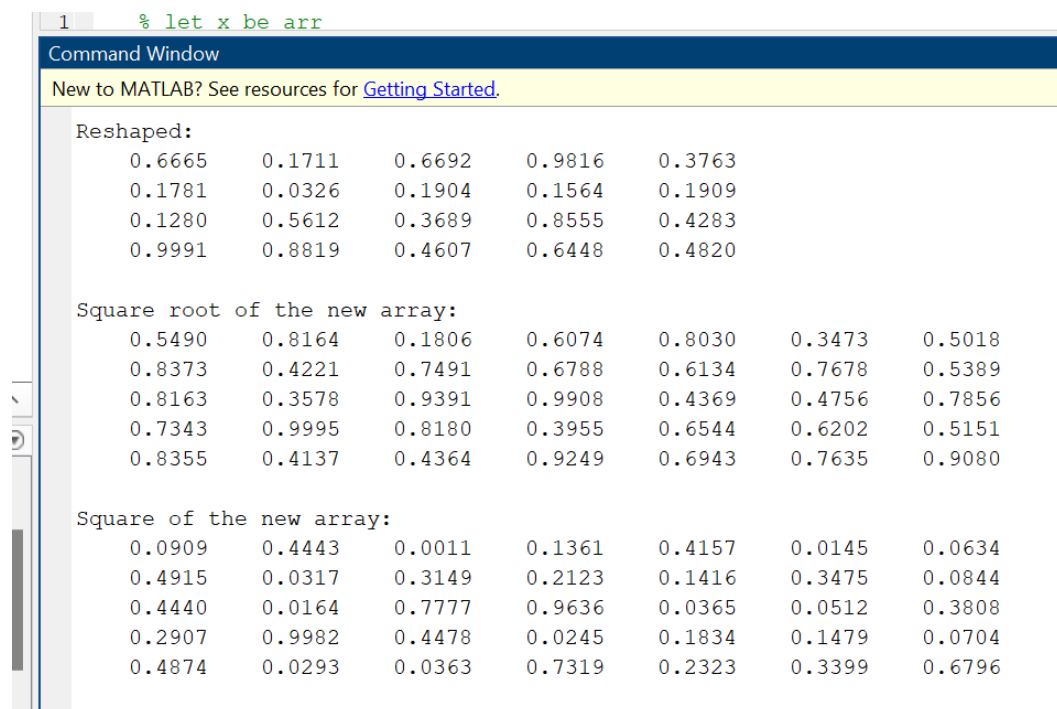
14 disp ("Sum of 3rd column ") ;
15 disp (s) ;
16 six=arr2 ( : , 6) ;
17 add=s+six;
18 disp ("sixth column after addition:") ;
19 disp (add) ;
20 %access and reshape in 4x5
21 arr3=arr2 ( : , 2:5) ;
22 res=reshape (arr3, 4, 5) ;
23 disp ("Reshaped:") ;
24 disp (res) ;
25 %Square root of the new array
26 sq=sqrt (arr2) ;
27 disp ("Square root of the new array:") ;
28 disp (sq) ;
29 %Square of the new array
30 new= arr2.^2;
31 disp ("Square of the new array:") ;
32 disp (new) ;

```

Command Window

New to MATLAB? See resources for [Getting Started](#)

Output:



```

1 % let x be arr

```

Command Window

New to MATLAB? See resources for [Getting Started](#).

Reshaped:

0.6665	0.1711	0.6692	0.9816	0.3763
0.1781	0.0326	0.1904	0.1564	0.1909
0.1280	0.5612	0.3689	0.8555	0.4283
0.9991	0.8819	0.4607	0.6448	0.4820

Square root of the new array:

0.5490	0.8164	0.1806	0.6074	0.8030	0.3473	0.5018
0.8373	0.4221	0.7491	0.6788	0.6134	0.7678	0.5389
0.8163	0.3578	0.9391	0.9908	0.4369	0.4756	0.7856
0.7343	0.9995	0.8180	0.3955	0.6544	0.6202	0.5151
0.8355	0.4137	0.4364	0.9249	0.6943	0.7635	0.9080

Square of the new array:

0.0909	0.4443	0.0011	0.1361	0.4157	0.0145	0.0634
0.4915	0.0317	0.3149	0.2123	0.1416	0.3475	0.0844
0.4440	0.0164	0.7777	0.9636	0.0365	0.0512	0.3808
0.2907	0.9982	0.4478	0.0245	0.1834	0.1479	0.0704
0.4874	0.0293	0.0363	0.7319	0.2323	0.3399	0.6796

Q3. Write a program to raise the power of the 1st column to the 4th column in a 3X6 array. (Use numbers 1-10)

Logic:

Step 1: Start

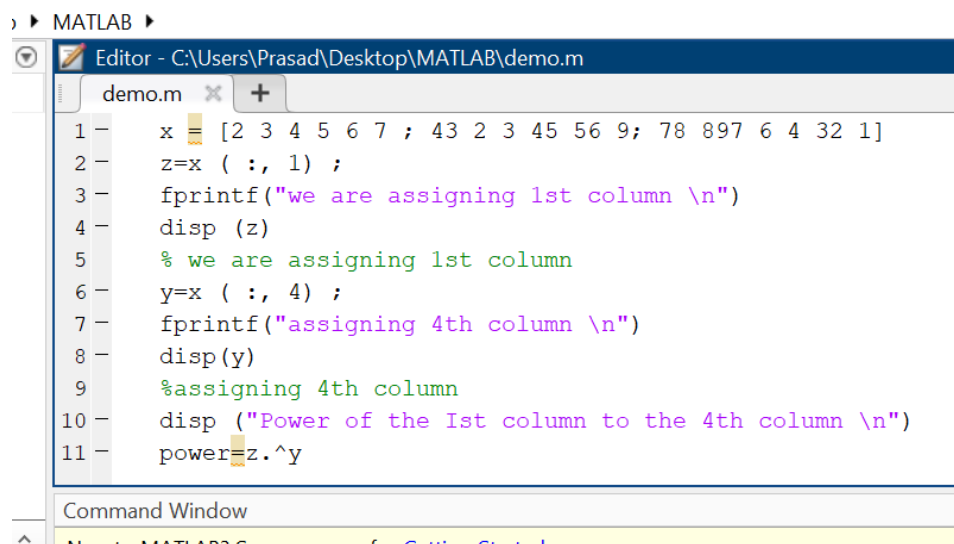
Step 2: First we have to create a new script named as demo.m.

Step 3: Now we have to create a 3X6 matrix. Access the 1st and 4th column.

Step 4: So just use the . ^ is used to raise 1st to the 4th column.

Step 5: Stop

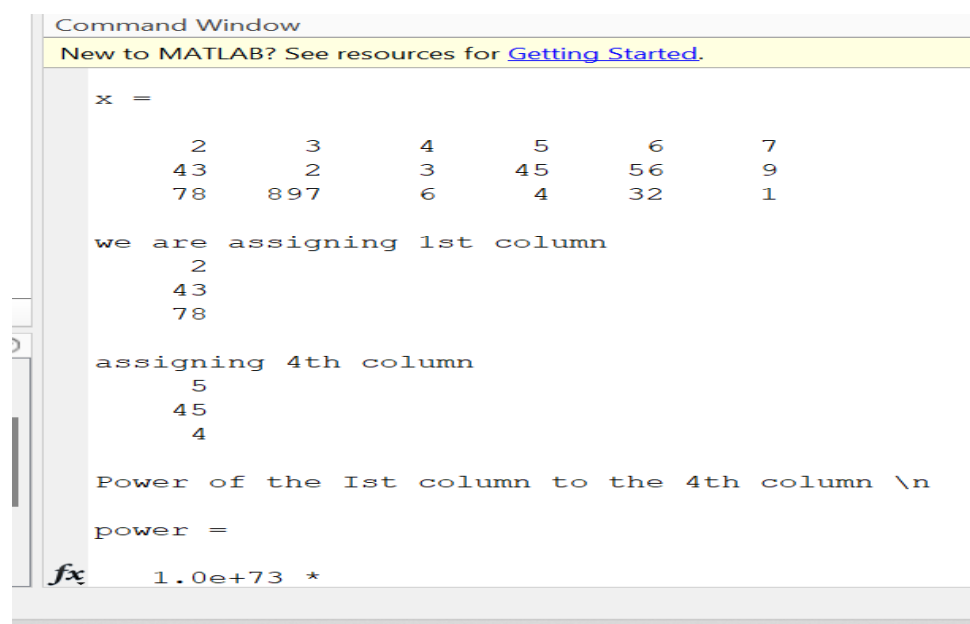
Code:



```

1 - x = [2 3 4 5 6 7 ; 43 2 3 45 56 9; 78 897 6 4 32 1]
2 - z=x ( :, 1) ;
3 - fprintf("we are assigning 1st column \n")
4 - disp (z)
5 - % we are assigning 1st column
6 - y=x ( :, 4) ;
7 - fprintf("assigning 4th column \n")
8 - disp(y)
9 - %assigning 4th column
10 - disp ("Power of the 1st column to the 4th column \n")
11 - power=z.^y
  
```

Output:



```

Command Window
New to MATLAB? See resources for Getting Started.

x =

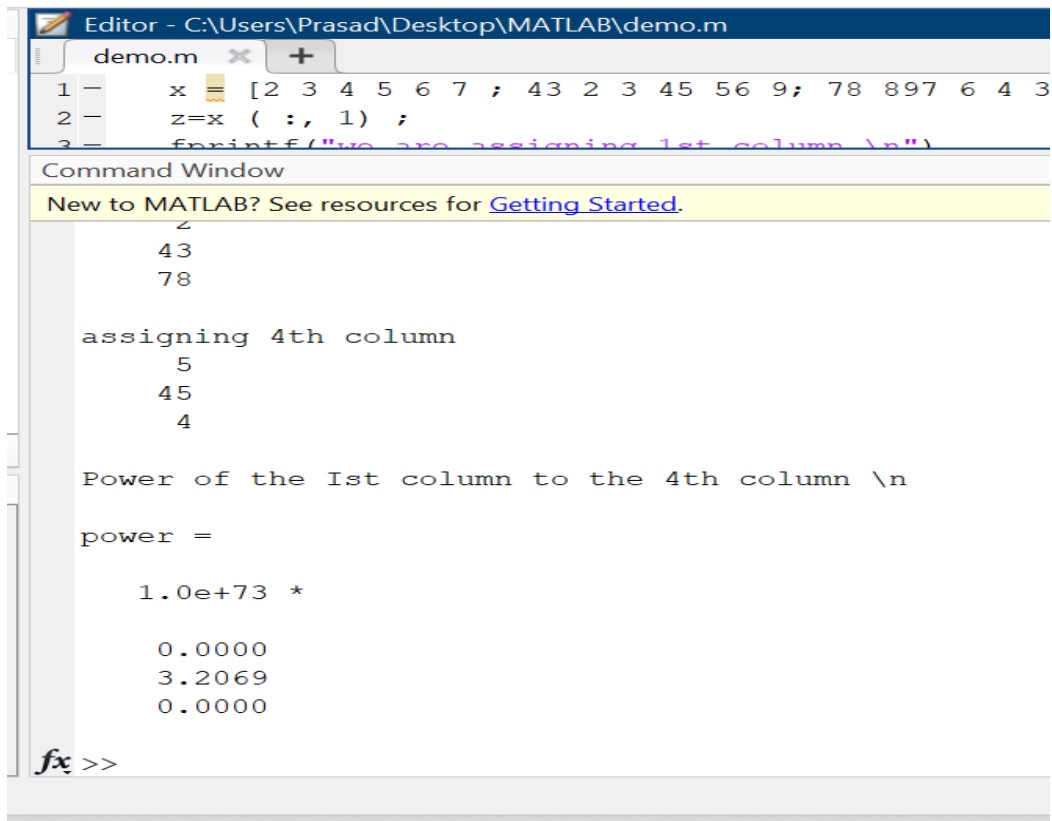
     2     3     4     5     6     7
    43     2     3    45    56     9
    78   897     6     4    32     1

we are assigning 1st column
     2
    43
    78

assigning 4th column
     5
    45
     4

Power of the 1st column to the 4th column \n
power =

1.0e+73 *
  
```



The image shows a MATLAB Editor window with a file named 'demo.m' open. The editor contains three lines of code: a matrix assignment for 'x', a column extraction for 'z', and a formatted print statement. Below the editor is the Command Window, which displays the output of the script. The output includes the first three columns of the matrix 'x', the 4th column of 'x', and the result of raising the first column of 'x' to the power of the 4th column of 'x'.

```
Editor - C:\Users\Prasad\Desktop\MATLAB\demo.m
demo.m
1 x = [2 3 4 5 6 7 ; 43 2 3 45 56 9; 78 897 6 4 3]
2 z=x ( :, 1) ;
3 fprintf("we are assigning 1st column \n")

Command Window
New to MATLAB? See resources for Getting Started.

2
43
78

assigning 4th column
5
45
4

Power of the 1st column to the 4th column \n

power =

1.0e+73 *

0.0000
3.2069
0.0000

fx >>
```


Q4. Write a program to calculate the sum of each column of an array of size 5X5 and assign those values to the last row of the array and display the size of the array.

Logic:

Step 1: Start

Step 2: So first we have to create a new script named as demoo.m

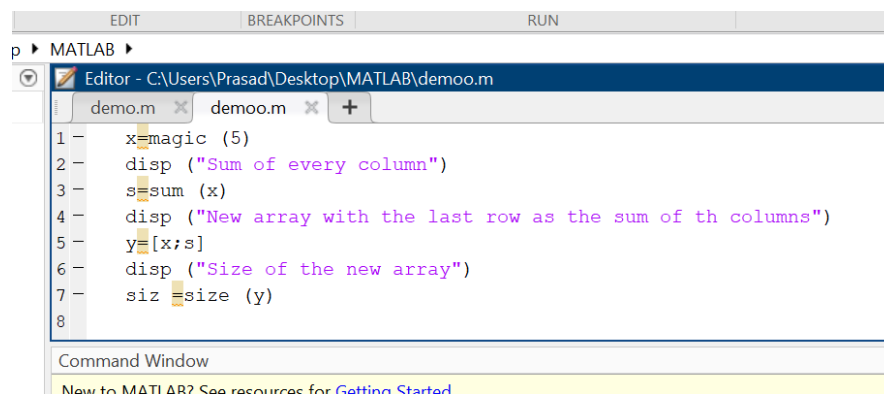
Step 3: Now we have to create a 5X5 matrix also we have to use sum() to calculate the sum of every column and assign it to a variable.

Step 4: Now we just have to create a new matrix and append the sum to it.

Step 5: So, the result would be printed in the command window.

Step 6: Stop

Code:

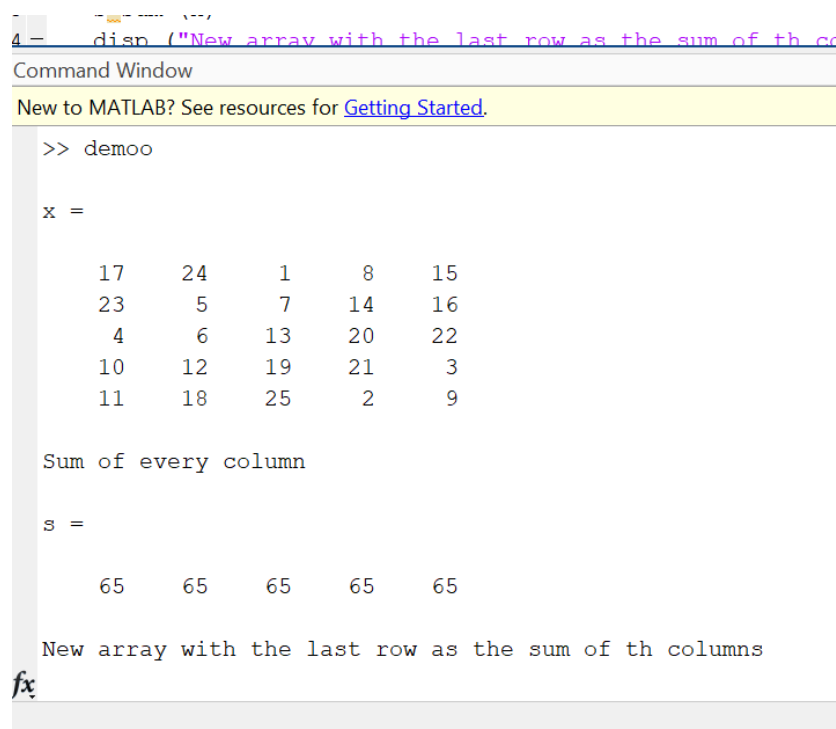


```

1 - x=magic (5)
2 - disp ("Sum of every column")
3 - s=sum (x)
4 - disp ("New array with the last row as the sum of th columns")
5 - y=[x;s]
6 - disp ("Size of the new array")
7 - siz=size (y)
8

```

Output:



```

4 - disp ("New array with the last row as the sum of th columns")
Command Window
New to MATLAB? See resources for Getting Started.

>> demoo

x =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9

Sum of every column

s =

    65    65    65    65    65

New array with the last row as the sum of th columns
fx

```

```
14 % Create a new array with the last row as the sum of the columns
Command Window
New to MATLAB? See resources for Getting Started.
New array with the last row as the sum of the columns

y =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
    65    65    65    65    65

Size of the new array

siz =

     6     5

fx >>
```

Q5. Write a program to use the “Sort” function to sort the array of size 4X4. Sort row-wise and column-wise.

Logic:

Step 1: Start

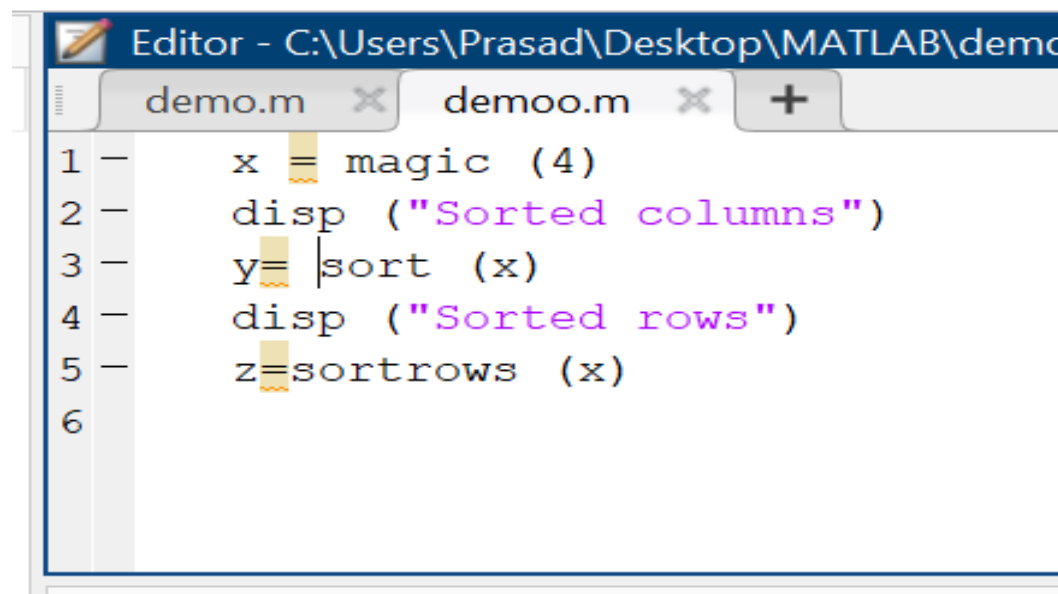
Step 2: First we have to Create a new script named as demoo.m

Step 3: Now, just create a 5X5 matrix and we have to use sort() that sorts the columns and sortrows() sorts the rows.

Step 4: Stop

Code:

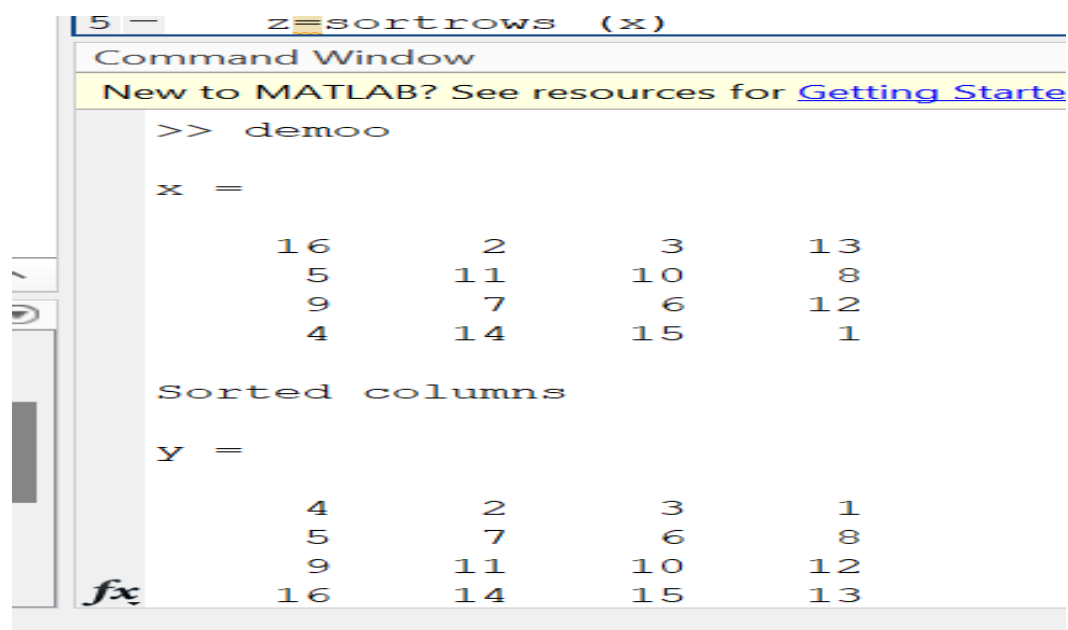
MATLAB ▶



The image shows the MATLAB Editor window with a script named demoo.m. The script contains the following code:

```
1 - x = magic (4)
2 - disp ("Sorted columns")
3 - y= sort (x)
4 - disp ("Sorted rows")
5 - z=sortrows (x)
6
```

Output:



The image shows the MATLAB Command Window with the following output:

```
>> demoo

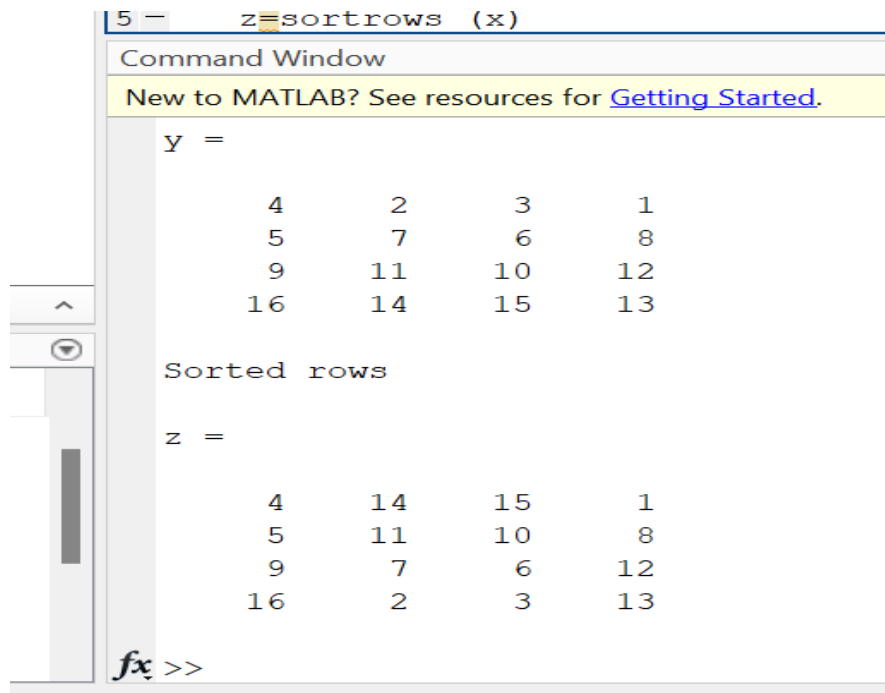
x =

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

Sorted columns

y =

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



The screenshot shows the MATLAB Command Window with the command `z=sortrows (x)` entered. The output displays two matrices: `y` and `z`. Matrix `y` is the original data, and matrix `z` is the result of sorting the rows of `x` based on the first column in ascending order. The Command Window also includes a message for new users: "New to MATLAB? See resources for [Getting Started](#)."

```
5 - z=sortrows (x)
Command Window
New to MATLAB? See resources for Getting Started.

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

Sorted rows

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

fx >>
```

Q6. Write a program to use the “Circshift” function to move the array elements in a circular pattern and bring it back to its normal position.

Logic:

Step 1: Start

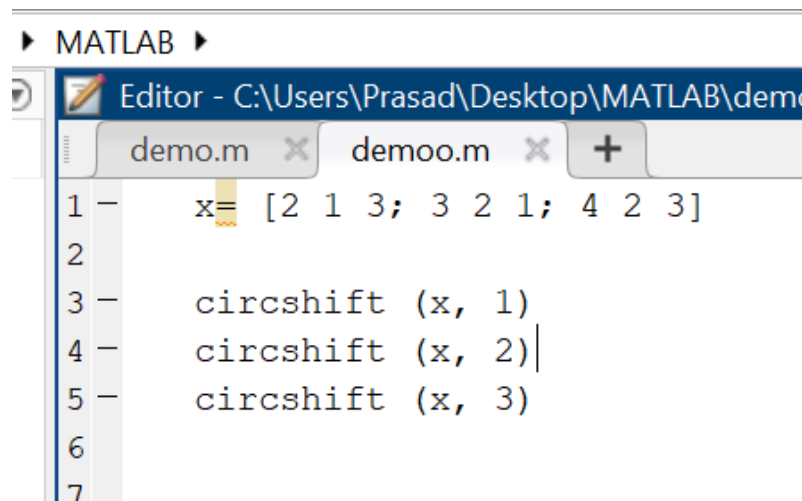
Step 2: So first we have to create a new script named as demoo.m

Step 3: So, now we have to create a matrix and use Circshift() that shifts the rows.

Step 4: So, we have to now see that it moves the last row to the top and moves the other rows down by 1 index.

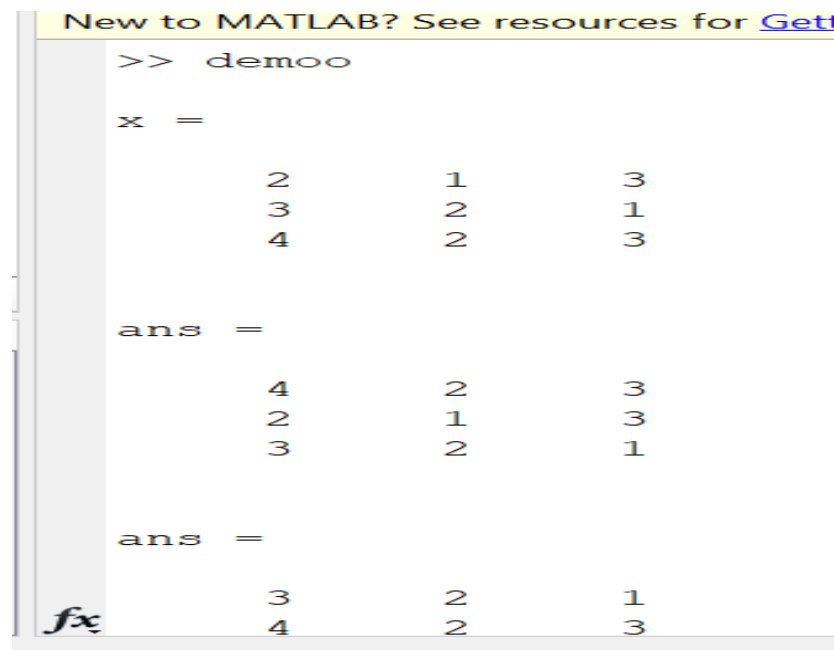
Step 5: Stop

Code:



```
MATLAB ▶
Editor - C:\Users\Prasad\Desktop\MATLAB\demo
demo.m x demoo.m x +
1 - x = [2 1 3; 3 2 1; 4 2 3]
2
3 - circshift (x, 1)
4 - circshift (x, 2)
5 - circshift (x, 3)
6
7
```

Output:



```
New to MATLAB? See resources for Gett
>> demoo

x =

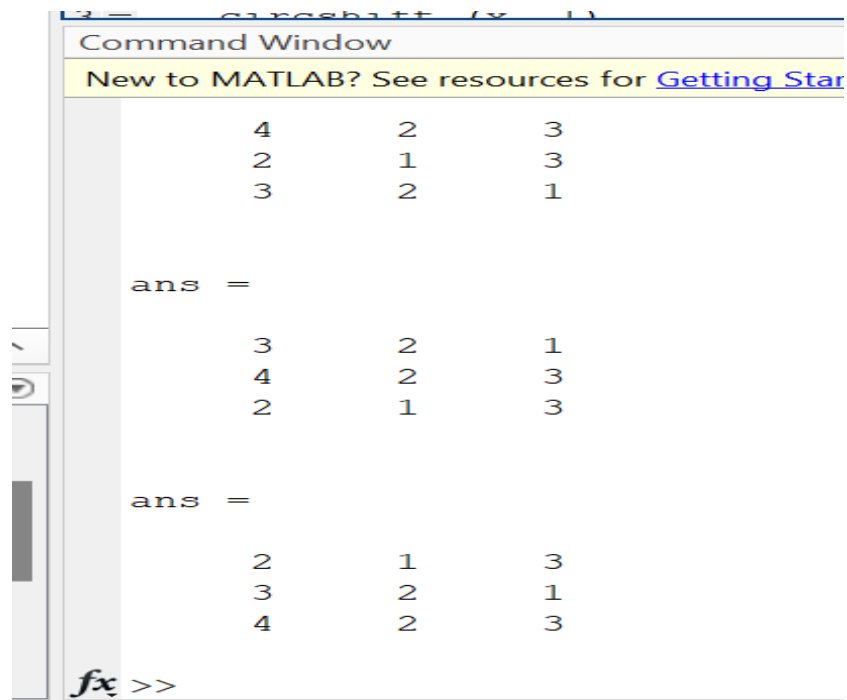
     2     1     3
     3     2     1
     4     2     3

ans =

     4     2     3
     2     1     3
     3     2     1

ans =

     3     2     1
     4     2     3
```



The screenshot shows the MATLAB Command Window. At the top, a command prompt shows `B = circshift(X,1)`. Below it, a message says "New to MATLAB? See resources for [Getting Started](#)". The main display area shows the result of the operation: a 3x3 matrix of numbers. The first matrix is the input `X`, and the second matrix is the output `ans`, which is `X` shifted one row down. The third matrix is the result of `ans` shifted one row down, with the bottom row wrapped to the top.

```
B = circshift(X,1)

Command Window

New to MATLAB? See resources for Getting Started

      4      2      3
      2      1      3
      3      2      1

ans =

      3      2      1
      4      2      3
      2      1      3

ans =

      2      1      3
      3      2      1
      4      2      3

fx >>
```

Q7. Write a program to display a maximum and minimum number within the created array of any size.

Logic:

Step 1: Start

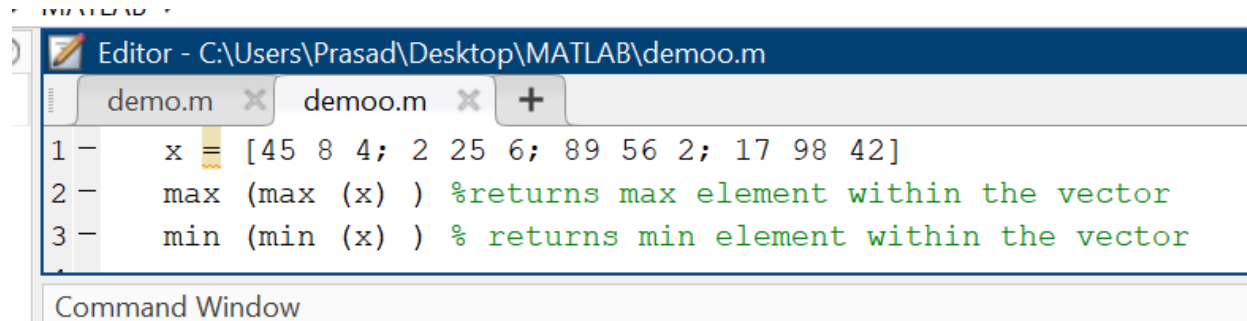
Step 2: So, first we have to create a new script named as demoo.m.

Step 3: So, the Max() and min() will give the entire row with maximum and minimum elements.

Step 4: Also, the Max(max()) and min(min()) will give the maximum and minimum element within the vector.

Step 5: Stop

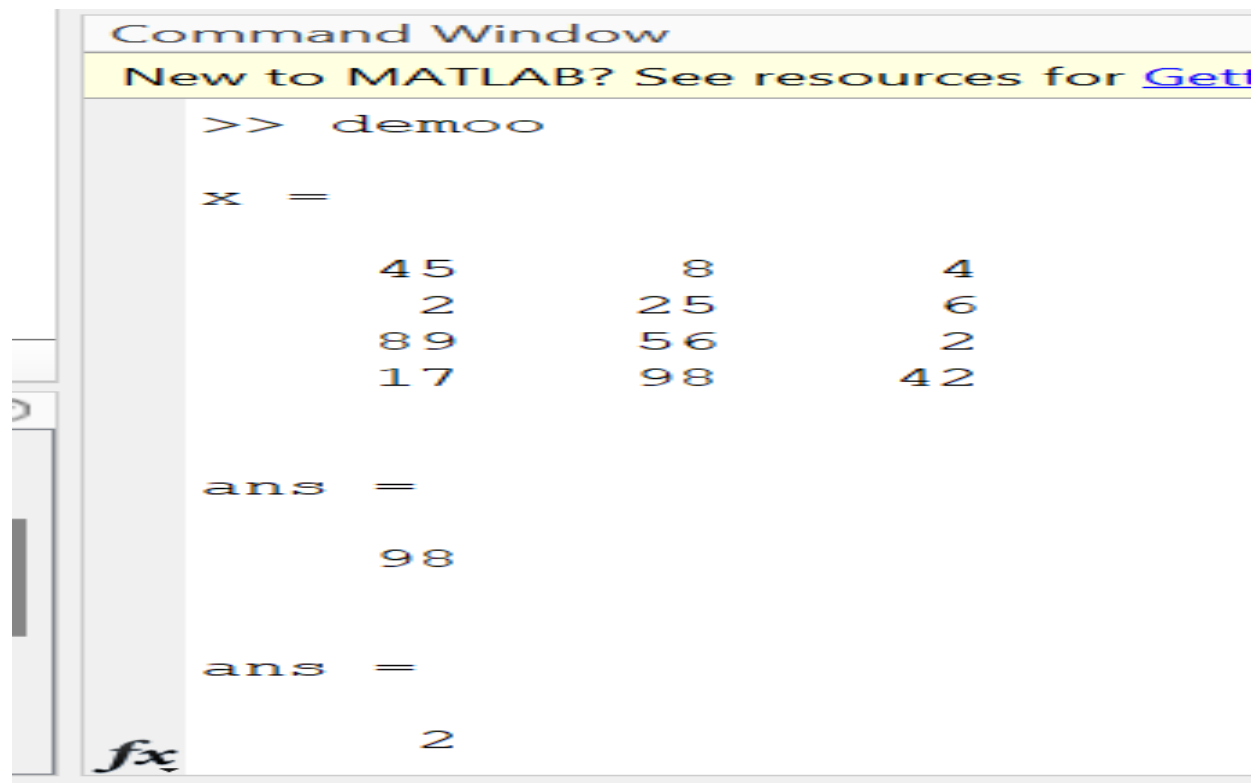
Code:



The screenshot shows the MATLAB Editor window with a script named demoo.m. The script contains three lines of code: a 4x3 matrix x is defined, the maximum element is found using max(max(x)), and the minimum element is found using min(min(x)).

```
1 - x = [45 8 4; 2 25 6; 89 56 2; 17 98 42]
2 - max (max (x) ) %returns max element within the vector
3 - min (min (x) ) % returns min element within the vector
```

Output:



The screenshot shows the MATLAB Command Window with the output of the demoo script. The matrix x is displayed, followed by the maximum element (98) and the minimum element (2).

```
>> demoo

x =

    45     8     4
     2    25     6
    89    56     2
    17    98    42

ans =

    98

ans =

     2
```

Q8. Write a program to create a random array of 10 rows and 5 columns to display elements less than 0.89.

Logic:

Step 1: Start

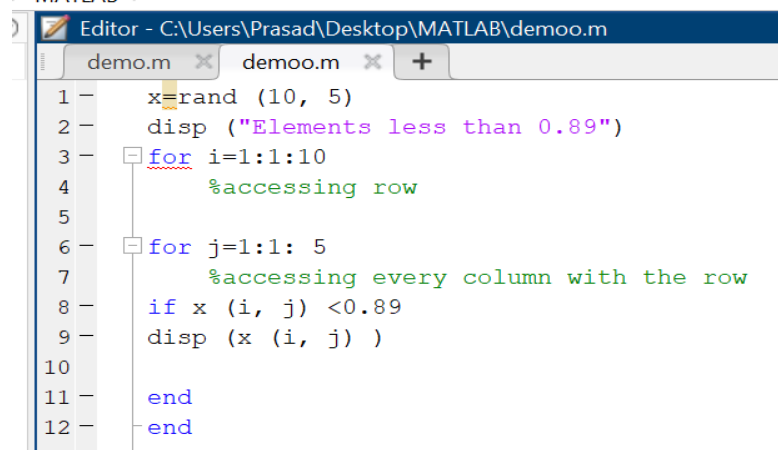
Step 2: First we have to create a new script named as demoo.m

Step 3: Now we have to first create a matrix with rand() to create a matrix with random functions.

Step 4: After that we have to find elements less than 0.89, access the row and then each column with the row using for loop so, then use conditional operator to verify if the element is less than 0.89.

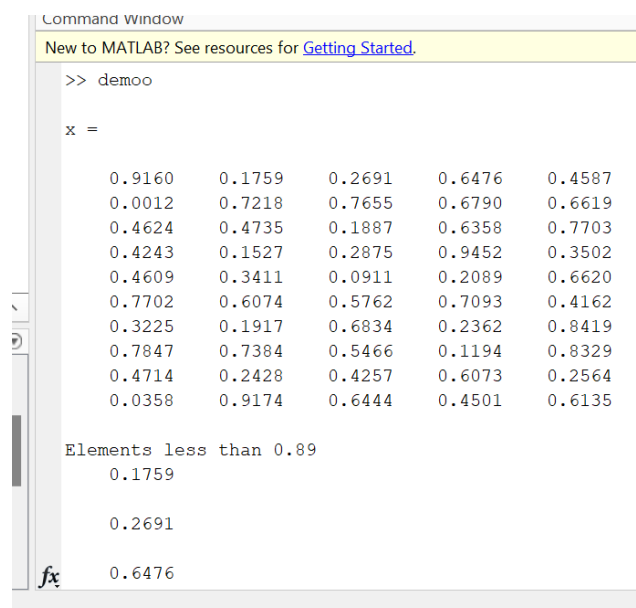
Step 5: Stop

Code:



```
Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1 - x=rand (10, 5)
2 - disp ("Elements less than 0.89")
3 - for i=1:1:10
4 -     %accessing row
5 -
6 - for j=1:1: 5
7 -     %accessing every column with the row
8 -     if x (i, j) <0.89
9 -     disp (x (i, j) )
10
11 - end
12 - end
```

Output:



```
Command Window
New to MATLAB? See resources for Getting Started.
>> demoo

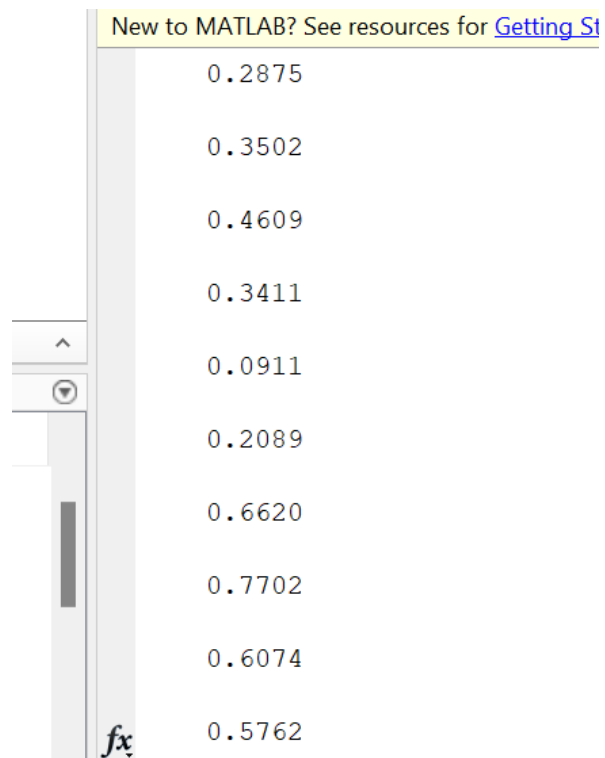
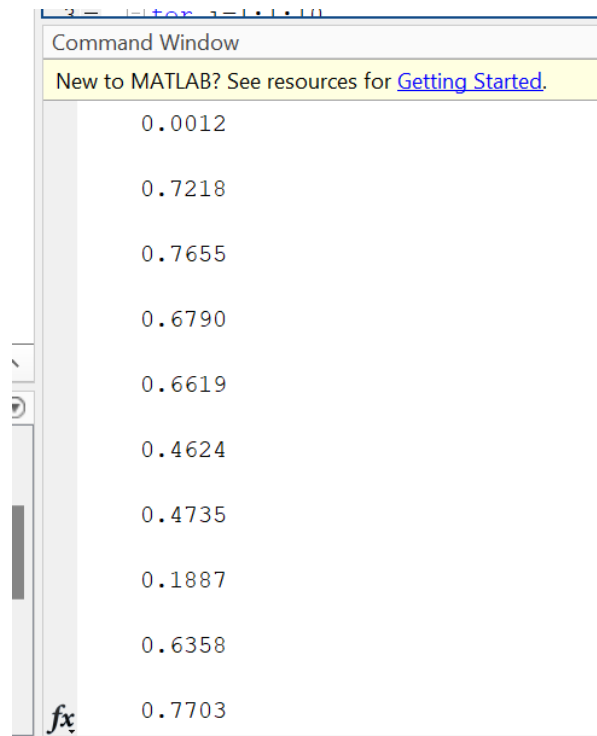
x =

    0.9160    0.1759    0.2691    0.6476    0.4587
    0.0012    0.7218    0.7655    0.6790    0.6619
    0.4624    0.4735    0.1887    0.6358    0.7703
    0.4243    0.1527    0.2875    0.9452    0.3502
    0.4609    0.3411    0.0911    0.2089    0.6620
    0.7702    0.6074    0.5762    0.7093    0.4162
    0.3225    0.1917    0.6834    0.2362    0.8419
    0.7847    0.7384    0.5466    0.1194    0.8329
    0.4714    0.2428    0.4257    0.6073    0.2564
    0.0358    0.9174    0.6444    0.4501    0.6135

Elements less than 0.89
    0.1759

    0.2691

fx    0.6476
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

0.4714

0.2428

0.4257

0.6073

0.2564

0.0358

0.6444

0.4501

0.6135

fx >>

Q9. Write a program to create a 2D array with elements (7 10 5; 4 1 2; 3 6 9) to perform the following operations:

Logic:

Step 1: Start

Step 2: First we have to create a new script named demoo.m

Step 3: So, the (end, end, 2)=0 would create a new array in the 2nd dimension.

Step 4: After that the Cat(3,x,...) will create a new array in the 3rd dimension.

Step 5: Now the x(1,2,3) will give the element in the 1st row and 2nd column of the 3rd dimension.

Step 6: So, after that the x(2,3,3) will give the element in the 2nd row and 3rd column of the 3rd dimension.

Step 7: Here, the x(3,:,2) will give the element in the 3rd row of the 2nd dimension.

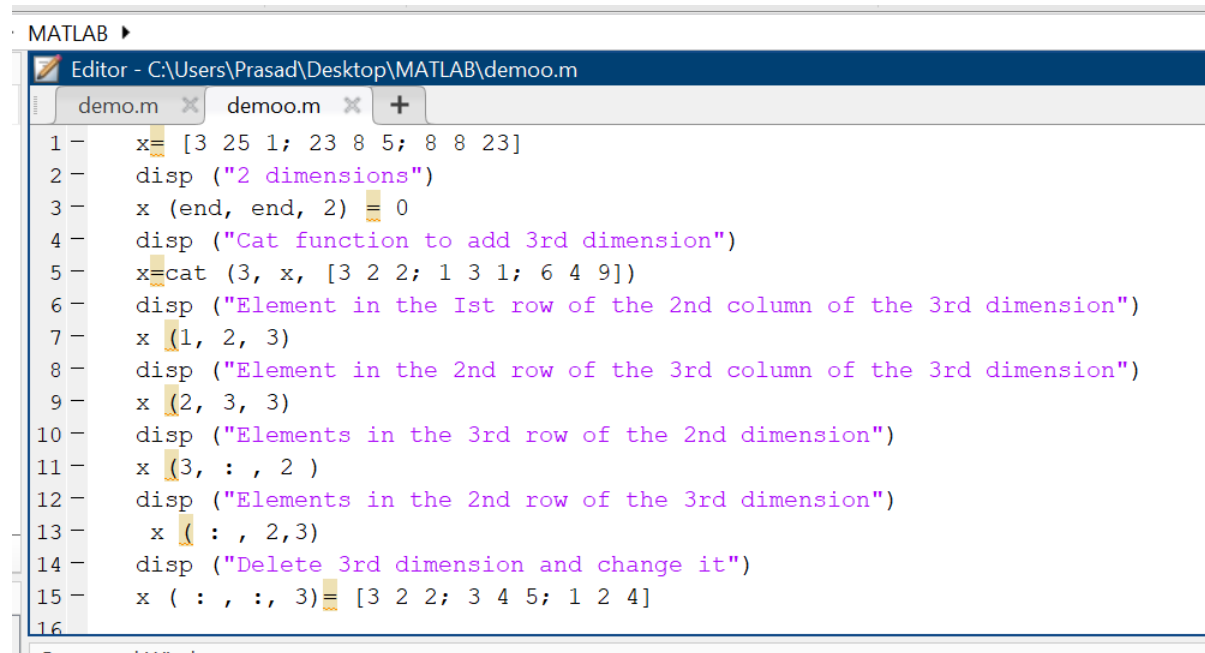
Step 8: After step 7 the x(:,2,3) will give the element in the 2nd column of the 3rd dimension.

Step 9: We have to now assign a new array to the 3rd dimension.

Step 10: Stop

1. Add the 2nd dimension.

Code:



```

MATLAB ▶
Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1- x = [3 25 1; 23 8 5; 8 8 23]
2- disp ("2 dimensions")
3- x (end, end, 2) = 0
4- disp ("Cat function to add 3rd dimension")
5- x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6- disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7- x (1, 2, 3)
8- disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9- x (2, 3, 3)
10- disp ("Elements in the 3rd row of the 2nd dimension")
11- x (3, :, 2)
12- disp ("Elements in the 2nd row of the 3rd dimension")
13- x (:, 2, 3)
14- disp ("Delete 3rd dimension and change it")
15- x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]
16-

```

Output:

```

Command Window
New to MATLAB? See resources for Getting Started.

>> demoo

x =

     3    25     1
    23     8     5
     8     8    23

2 dimensions

x(:,:,1) =

     3    25     1
    23     8     5
     8     8    23

x(:,:,2) =

```

```

Command Window
New to MATLAB? See resources for Getting Started.

     8     8    23

2 dimensions

x(:,:,1) =

     3    25     1
    23     8     5
     8     8    23

x(:,:,2) =

     0     0     0
     0     0     0
     0     0     0

Cat function to add 3rd dimension

```

2. Use the “Cat” function to add the 3rd dimension.

Code:

```

MATLAB ▶
Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1 x = [3 25 1; 23 8 5; 8 8 23]
2 disp ("2 dimensions")
3 x (end, end, 2) = 0
4 disp ("Cat function to add 3rd dimension")
5 x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6 disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7 x (1, 2, 3)
8 disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9 x (2, 3, 3)
10 disp ("Elements in the 3rd row of the 2nd dimension")
11 x (3, :, 2)
12 disp ("Elements in the 2nd row of the 3rd dimension")
13 x (:, 2, 3)
14 disp ("Delete 3rd dimension and change it")
15 x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]
16

```

Output:

```

Command Window
New to MATLAB? See resources for Getting Started.

0      0      0

Cat function to add 3rd dimension

x(:,:,1) =

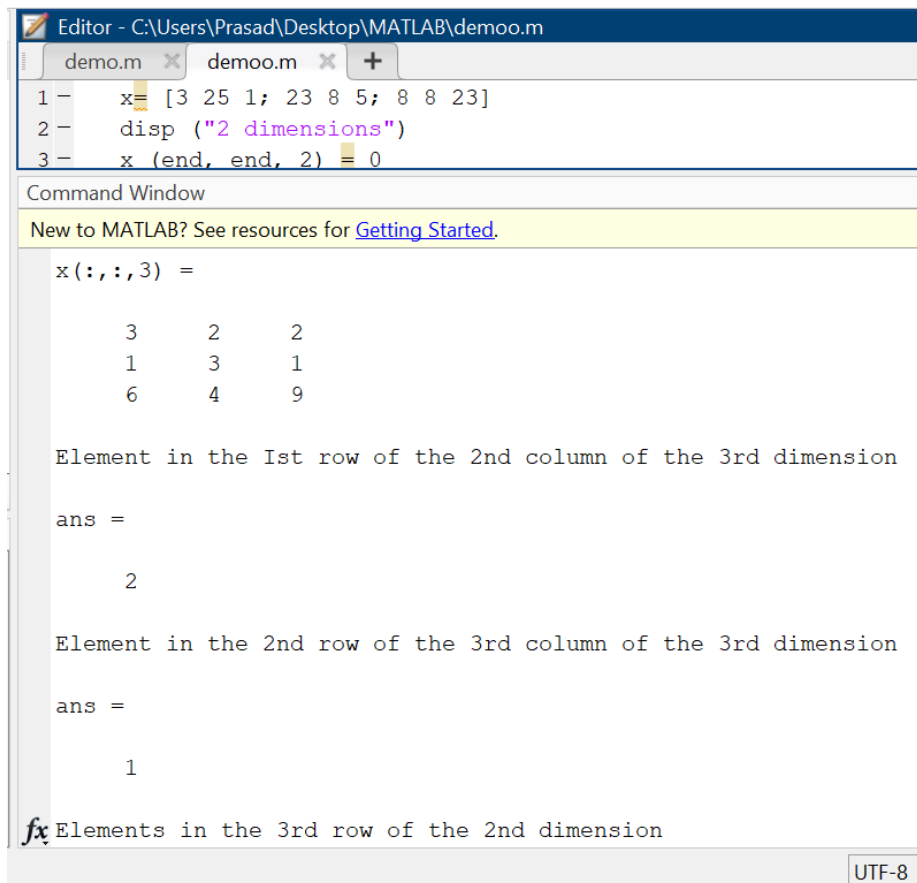
     3    25     1
    23     8     5
     8     8    23

x(:,:,2) =

     0     0     0
     0     0     0
     0     0     0

fx x(:,:,3) =

```



The image shows a MATLAB Editor window with a script named 'demo.m' and a Command Window. The script defines a 3D array 'x' and performs indexing operations. The Command Window shows the output of these operations, including a 3x3 matrix and scalar values.

```
Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demo.m x +
1 - x = [3 25 1; 23 8 5; 8 8 23]
2 - disp ("2 dimensions")
3 - x (end, end, 2) = 0

Command Window
New to MATLAB? See resources for Getting Started.

x(:, :, 3) =

     3     2     2
     1     3     1
     6     4     9

Element in the 1st row of the 2nd column of the 3rd dimension

ans =

     2

Element in the 2nd row of the 3rd column of the 3rd dimension

ans =

     1

fx Elements in the 3rd row of the 2nd dimension
UTF-8
```

3. Access the element in the 1st row of the 2nd column of the 3rd dimension.

Code:

MATLAB ▶

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1 - x = [3 25 1; 23 8 5; 8 8 23]
2 - disp ("2 dimensions")
3 - x (end, end, 2) = 0
4 - disp ("Cat function to add 3rd dimension")
5 - x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6 - disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7 - x (1, 2, 3)
8 - disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9 - x (2, 3, 3)
10 - disp ("Elements in the 3rd row of the 2nd dimension")
11 - x (3, :, 2)
12 - disp ("Elements in the 2nd row of the 3rd dimension")
13 - x (:, 2, 3)
14 - disp ("Delete 3rd dimension and change it")
15 - x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]
16

```

Output:

```

3 - x (end, end, 2) = 0
Command Window
New to MATLAB? See resources for Getting Started.
x (:, :, 3) =
    3    2    2
    1    3    1
    6    4    9
Element in the 1st row of the 2nd column of the 3rd dimension
ans =
    2
Element in the 2nd row of the 3rd column of the 3rd dimension
ans =
    1
fx Elements in the 3rd row of the 2nd dimension

```

4. Access the element in the 2nd row 3rd column of the 3rd dimension.

Code:

```

MATLAB ▶
Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1 - x = [3 25 1; 23 8 5; 8 8 23]
2 - disp ("2 dimensions")
3 - x (end, end, 2) = 0
4 - disp ("Cat function to add 3rd dimension")
5 - x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6 - disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7 - x (1, 2, 3)
8 - disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9 - x (2, 3, 3)
10 - disp ("Elements in the 3rd row of the 2nd dimension")
11 - x (3, :, 2)
12 - disp ("Elements in the 2nd row of the 3rd dimension")
13 - x (:, 2, 3)
14 - disp ("Delete 3rd dimension and change it")
15 - x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]

```

Output:

```

Command Window
New to MATLAB? See resources for Getting Started.

x (:, :, 3) =

     3     2     2
     1     3     1
     6     4     9

Element in the 1st row of the 2nd column of the 3rd dimension

ans =

     2

Element in the 2nd row of the 3rd column of the 3rd dimension

ans =

     1

fx Elements in the 3rd row of the 2nd dimension

```


5. Access all the elements in the 3rd row of 2nd dimension.

Code:

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m +
6 1 x = [3 25 1; 23 8 5; 8 8 23]
2 disp ("2 dimensions")
3 x (end, end, 2) = 0
4 disp ("Cat function to add 3rd dimension")
5 x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6 disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7 x (1, 2, 3)
8 disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9 x (2, 3, 3)
10 disp ("Elements in the 3rd row of the 2nd dimension")
11 x (3, :, 2)
12 disp ("Elements in the 2nd row of the 3rd dimension")
13 x (:, 2, 3)
14 disp ("Delete 3rd dimension and change it")
15 x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]
16

```

Output:

```

Command Window
New to MATLAB? See resources for Getting Started.

Element in the 2nd row of the 3rd column of the 3r

ans =

    1

Elements in the 3rd row of the 2nd dimension

ans =

    0    0    0

Elements in the 2nd row of the 3rd dimension

ans =

    2
    3
    4
fx

```

6. Access all the elements of the 2nd column 3rd dimension.

Code:

```
MATLAB
Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1 - x = [3 25 1; 23 8 5; 8 8 23]
2 - disp ("2 dimensions")
3 - x (end, end, 2) = 0
4 - disp ("Cat function to add 3rd dimension")
5 - x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6 - disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7 - x (1, 2, 3)
8 - disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9 - x (2, 3, 3)
10 - disp ("Elements in the 3rd row of the 2nd dimension")
11 - x (3, :, 2)
12 - disp ("Elements in the 2nd row of the 3rd dimension")
13 - x (:, 2, 3)
14 - disp ("Delete 3rd dimension and change it")
15 - x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]
16
```

Output:

```
3 - x (end, end, 2) = 0
Command Window
New to MATLAB? See resources for Getting Started.
Elements in the 2nd row of the 3rd dimension
ans =
    2
    3
    4
Delete 3rd dimension and change it
x(:, :, 1) =
    3    25     1
   23     8     5
    8     8    23
x(:, :, 2) =
fx
```

7. Delete dimension 3 and change it to [2 5 3; 8 9 0; 6 5 4]

Code

```

Editor - C:\Users\Prasad\Desktop\MATLAB\demoo.m
demo.m x demoo.m x +
1 x = [3 25 1; 23 8 5; 8 8 23]
2 disp ("2 dimensions")
3 x (end, end, 2) = 0
4 disp ("Cat function to add 3rd dimension")
5 x = cat (3, x, [3 2 2; 1 3 1; 6 4 9])
6 disp ("Element in the 1st row of the 2nd column of the 3rd dimension")
7 x (1, 2, 3)
8 disp ("Element in the 2nd row of the 3rd column of the 3rd dimension")
9 x (2, 3, 3)
10 disp ("Elements in the 3rd row of the 2nd dimension")
11 x (3, :, 2)
12 disp ("Elements in the 2nd row of the 3rd dimension")
13 x (:, 2, 3)
14 disp ("Delete 3rd dimension and change it")
15 x (:, :, 3) = [3 2 2; 3 4 5; 1 2 4]
16

```

Output:

```

Command Window
New to MATLAB? See resources for Getting Started.

4

Delete 3rd dimension and change it

x(:, :, 1) =

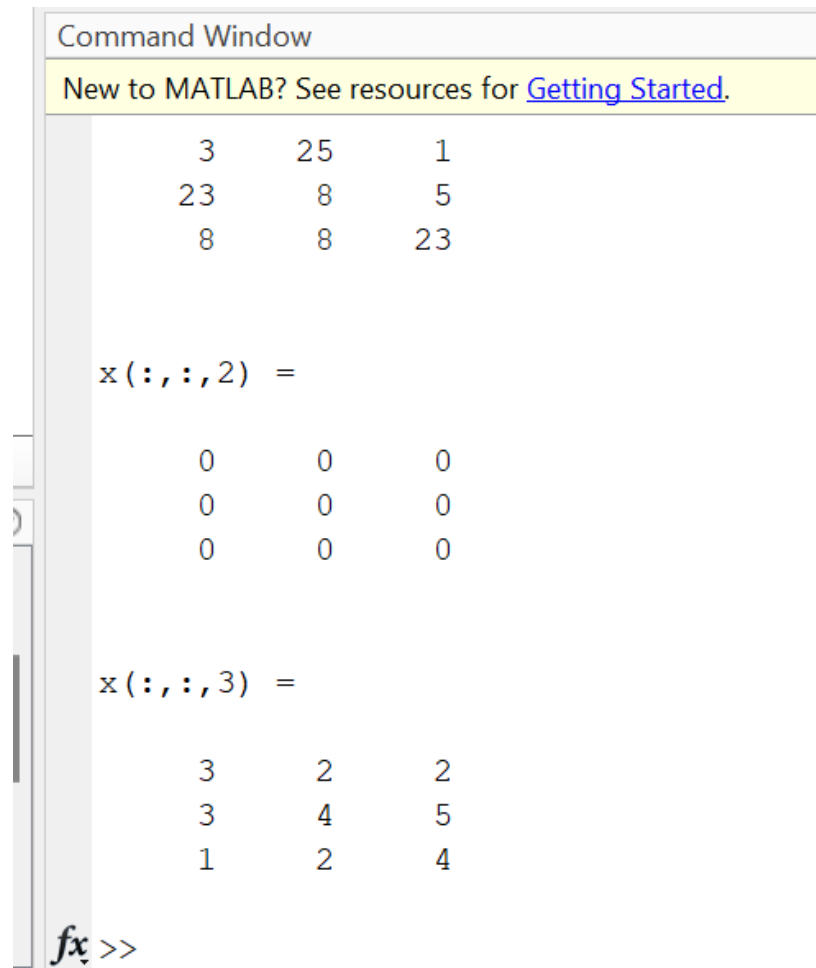
    3    25     1
   23     8     5
    8     8    23

x(:, :, 2) =

    0     0     0
    0     0     0
    0     0     0

fx x(:, :, 3) =

```



Command Window

New to MATLAB? See resources for [Getting Started](#).

```
3      25      1
23      8      5
8      8      23
```

`x(:, :, 2) =`

```
0      0      0
0      0      0
0      0      0
```

`x(:, :, 3) =`

```
3      2      2
3      4      5
1      2      4
```

fx >>