Image Processing

(01CE0507)

Department of Computer Engineering

5th Semester

Lab Manual

(July-Dec 2022)

Index

Lab	Programs	Date	Signature
1	Study of matlab image processing toolkit and various commands on matlab.		
2	Point processing in spatial domain a. Negation of an image b. Thresholding of an image c. Contrast Stretching of an image		
3	Write a program for histogram equalization.		
4	Write a program to apply various filtering techniques in matlab. a. Low pass filtering b. High pass filtering c. Median filtering		
5	Write a program for image segmentation a. Local thresholding b. Global thresholding		
6	Write a program for color image processing a. Color approximation b. Quantization		
7	Write a program, for Image restoration a. Facial Images b. Texture Images		
8	Write a program for edge detection.		
9	Write a program for smoothening and sharpening for 8-bit color image.		
10	Write a program to implement morphological operations.		

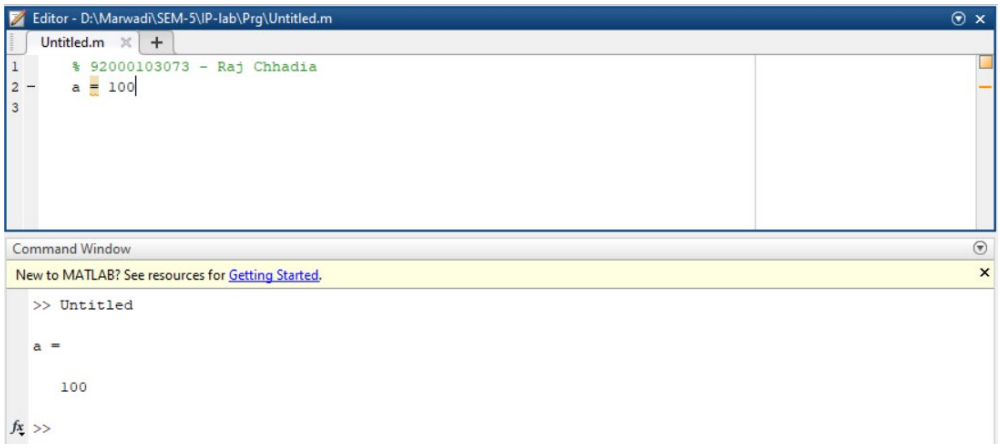
Practical 1

Aim: Study of matlab image processing toolkit and various commands on matlab.

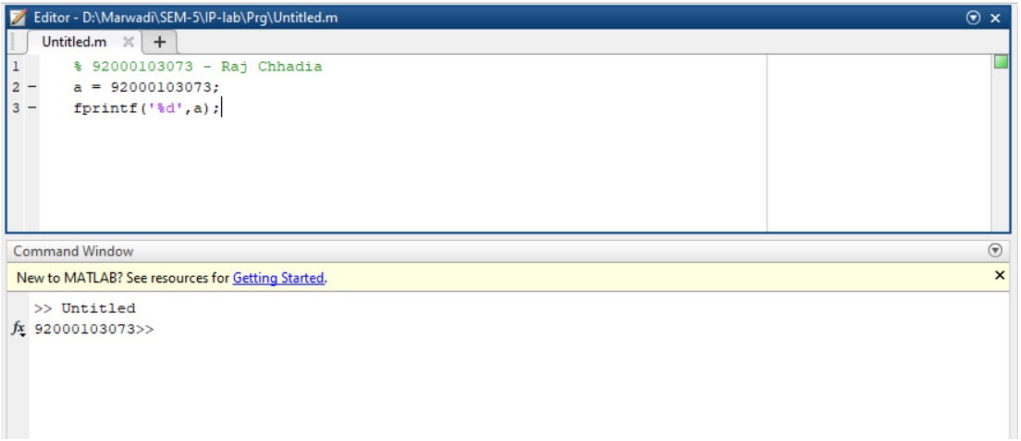
- **Introduction to Various component of MATLAB Tool like editor, command window, workspace.**
- **Basic Syntax, Variables, Commands**
  - **Use of Semicolon (;) in MATLAB**

1) Without Semicolon

→ Use semicolons to separate rows in an array creation command, or to suppress the output display of a line of code.

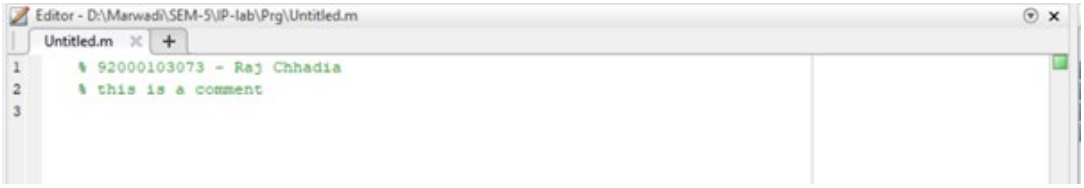


2) With Semicolon



○ Adding Comments

→ Comments are non-executable text within the body of a program. They can be declared by '%'



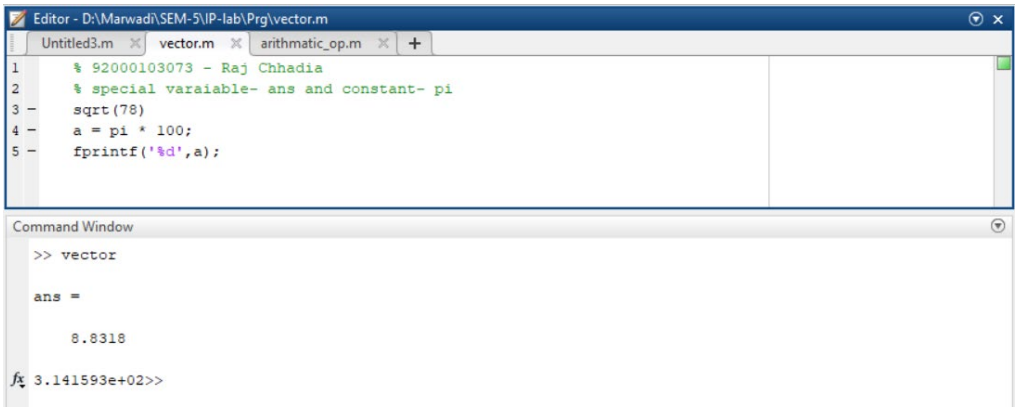
○ Commonly used Operators and Special Characters

Operator	Purpose
+	Plus; addition operator.
-	Minus; subtraction operator.
*	Scalar and matrix multiplication operator.
.*	Array multiplication operator.
^	Scalar and matrix exponentiation operator.
.^	Array exponentiation operator.
\	Left-division operator.
/	Right-division operator.
.\	Array left-division operator.
./	Array right-division operator.
:	Colon; generates regularly spaced elements and represents an entire row or column.
( )	Parentheses; encloses function arguments and array indices; overrides precedence.
[ ]	Brackets; enclosures array elements.
.	Decimal point.
…	Ellipsis; line-continuation operator
,	Comma; separates statements and elements in a row
;	Semicolon; separates columns and suppresses display.
%	Percent sign; designates a comment and specifies formatting.
'	Quote sign and transpose operator.
.'	Nonconjugated transpose operator.
=	Assignment operator.

Name	Meaning
ans	Most recent answer.
eps	Accuracy of floating-point precision.
i,j	The imaginary unit √-1.
Inf	Infinity.
NaN	Undefined numerical result (not a number).
pi	The number π

○ Special Variables and Constants

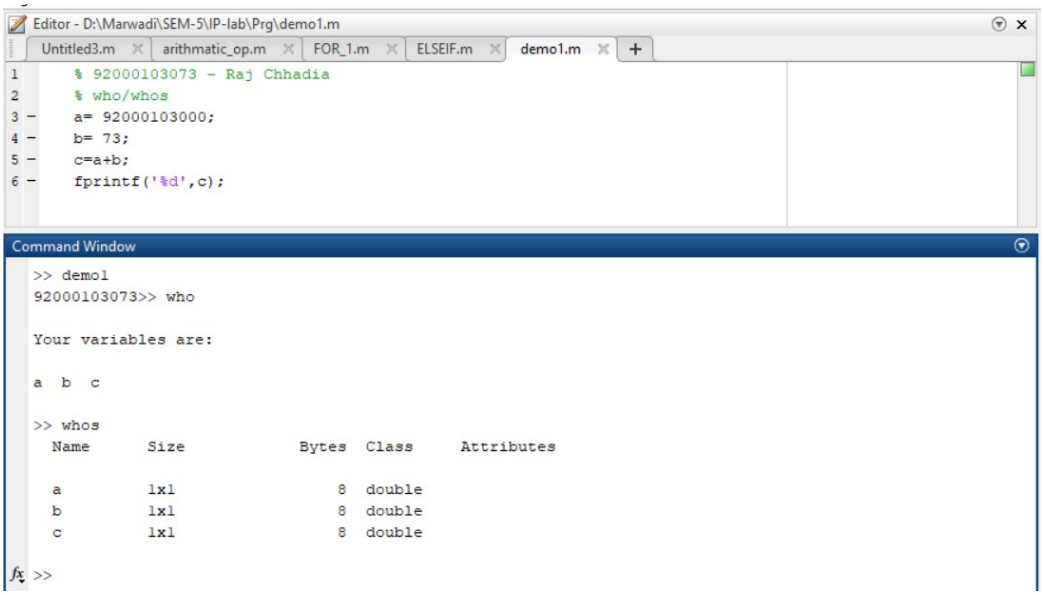
→ sqrt, pi are some of the special constants and ans is one of the special variables to store the answer.



○ save and load Command

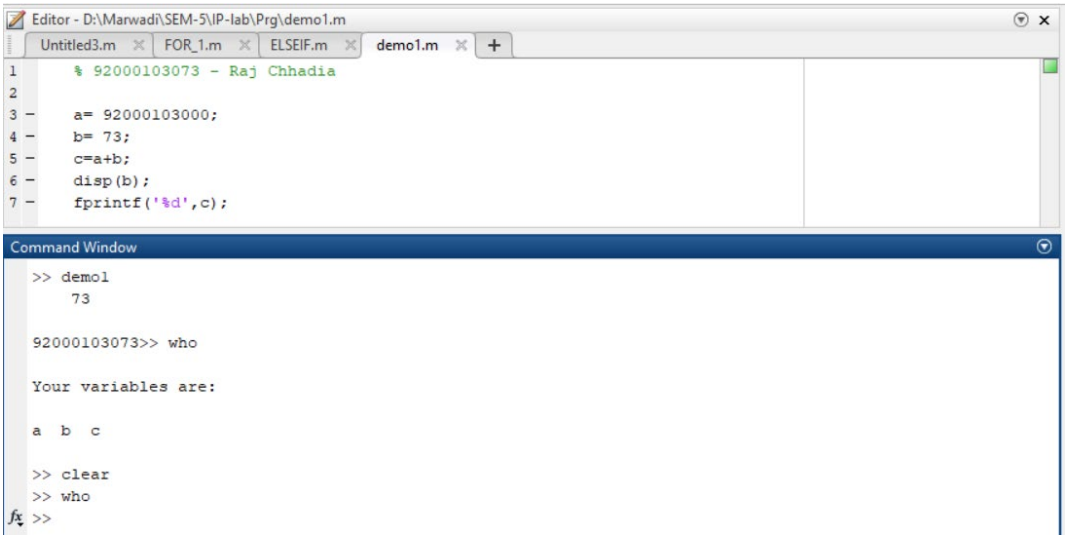
→ save command saves workspace variables in a file. load command loads workspace variables from a file.

- **who and whos command**



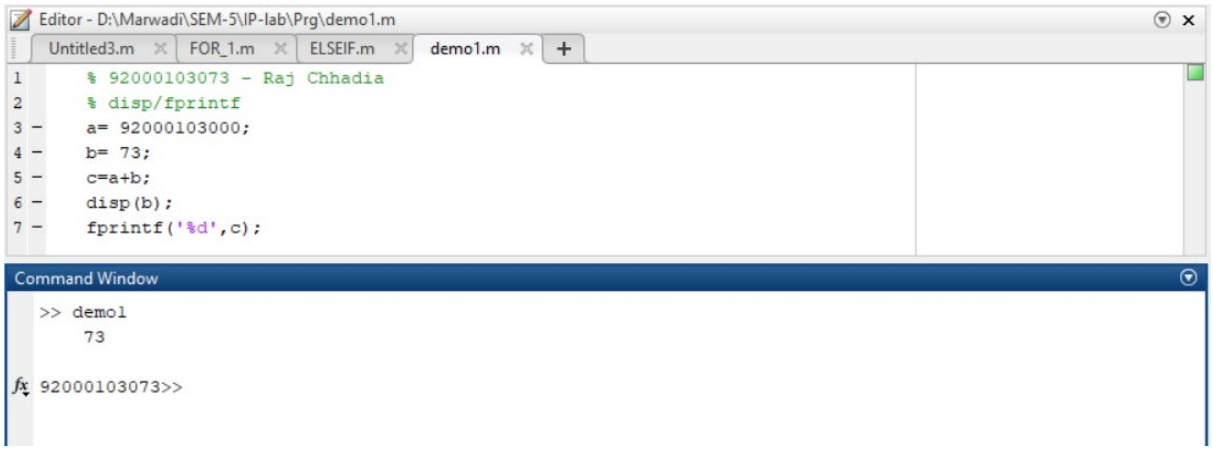
- **clear and clc Command**

→ clc clears command window. clear removes variables from memory.



- **disp and fprintf command**

→ disp displays contents of an array / string. fprintf performs formatted writes to screen.



- **input command**

- **M-Files**

→ Script files are program files with .m extension. In these files, you write series of commands, which you want to execute together.

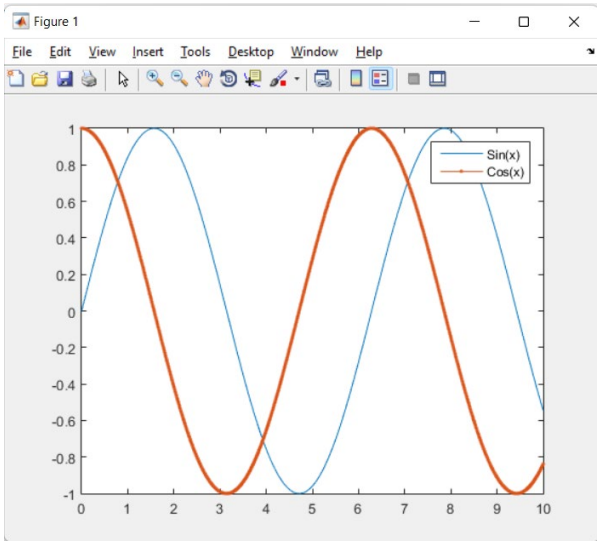
```
demo1.m | demo.txt
1   % 92000103073 - Raj Chhadia
2   a = 5; b = 7;
3   c = a + b
4   d = c + sin(b)
5   e = 5 * d
```

```
Command Window
>> demo1
c =
    12
d =
   12.6570
e =
   63.2849
fx >>
```

- **Plotting Commands**

→ plot command is used to plot a graph or curve.

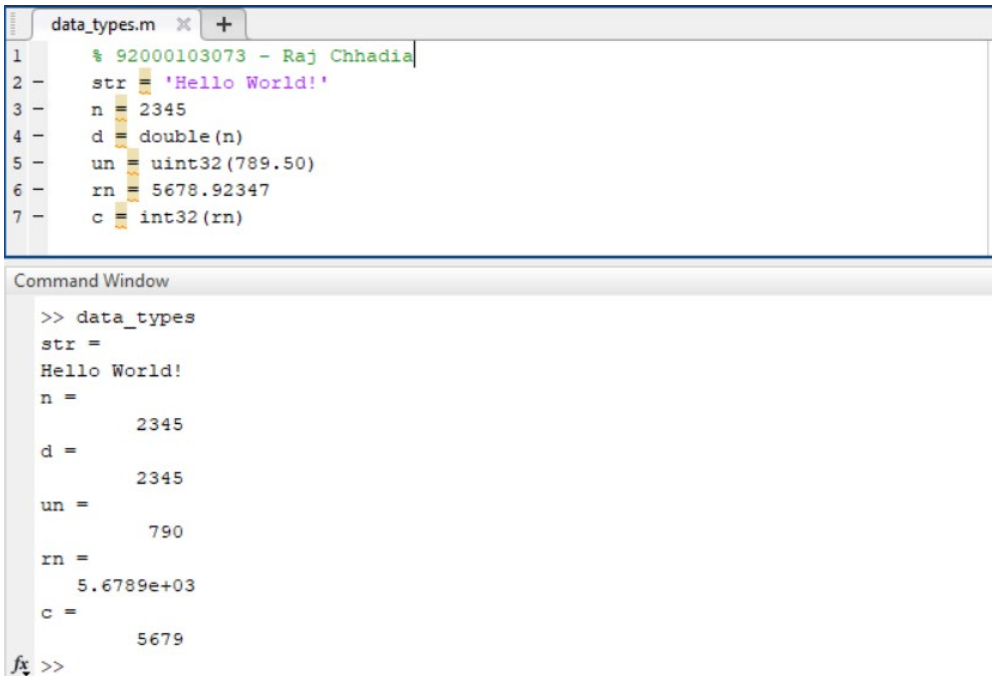
```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\demo1.m
demo1.m | demo.txt
1   % 92000103073 - Raj Chhadia
2   x = [0 : 0.01: 10];
3   y = sin(x);
4   g = cos(x);
5   plot(x, y, x, g, '.-'), legend('Sin(x)', 'Cos(x)')
```



➢ Data Types in MATLAB

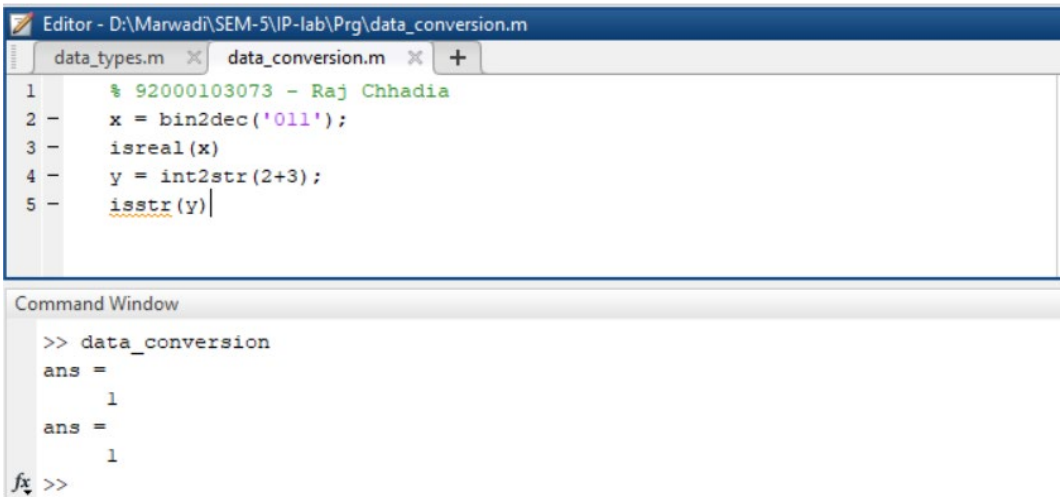
○ List of Data Types

No.	Data Types
1.	int8
2.	uint8
3.	int32
4.	uint32
5.	int64
6.	uint64
7.	double
8.	char
9.	cell array
10.	structure



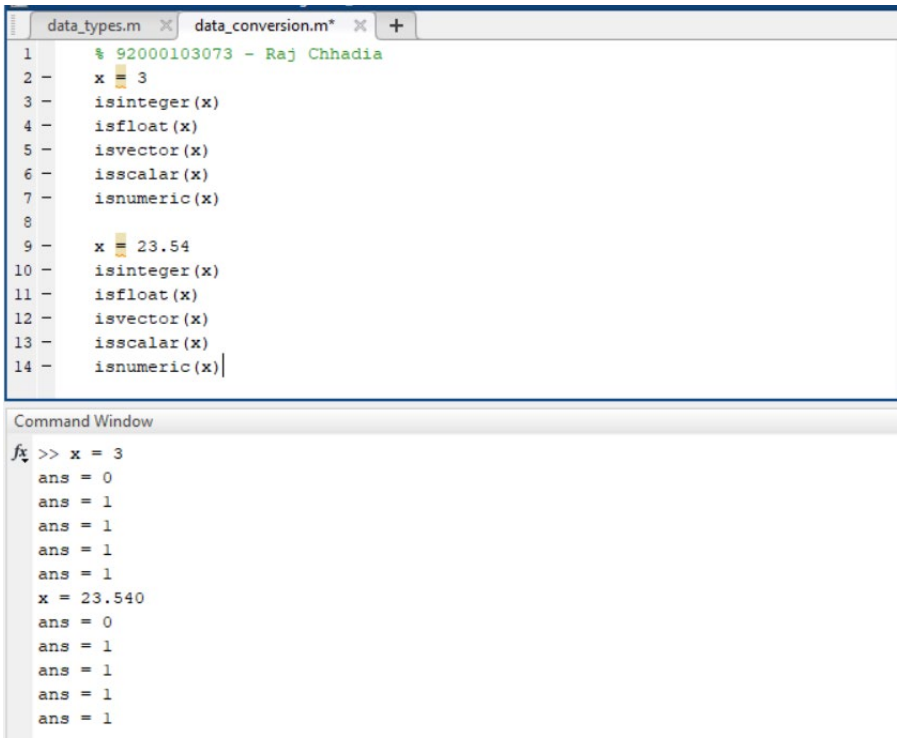
- **Data Type Conversion**

Function	Purpose
char	Convert to character array (string)
int2str	Convert integer data to string
num2str	Convert number to string
str2double	Convert string to double-precision value
str2num	Convert string to number
bin2dec	Convert binary number to decimal number
dec2base	Convert decimal to base N number in string
dec2bin	Convert decimal to binary number in string
hex2dec	Convert hexadecimal number string to decimal number
mat2cell	Convert array to cell array with potentially different sized cells
cellstr	Create cell array of strings from character array



- **Determination of Data Types**

Function	Purpose
isinteger	Determine if input is integer
isfloat	Determine if input is float
isvector	Determine if input is vector
isscalar	Determine if input is scalar
isnumeric	Determine if input is numeric array

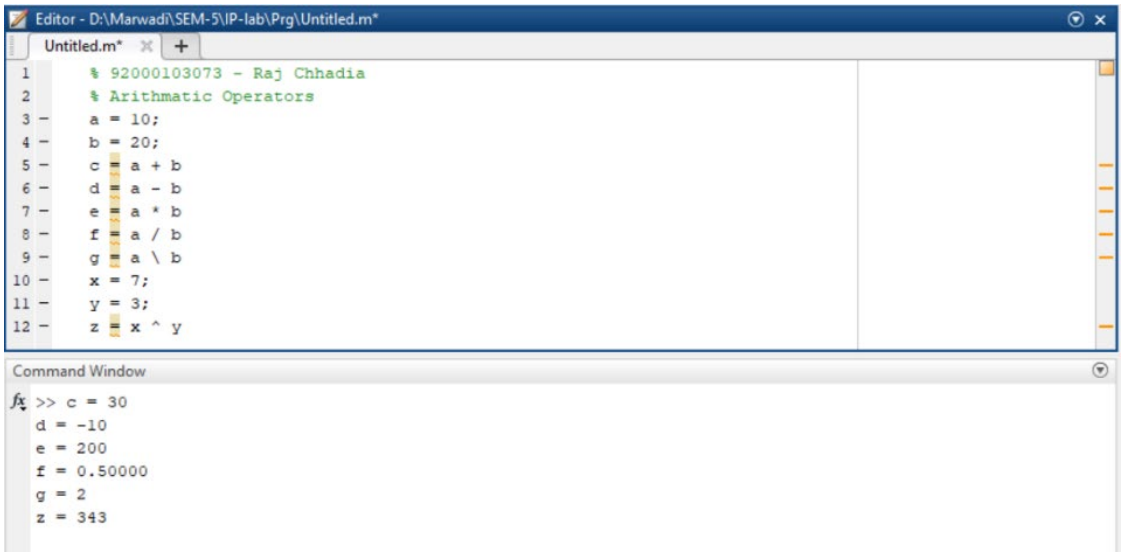


➢ Operators in MATLAB

○ Types of Operators

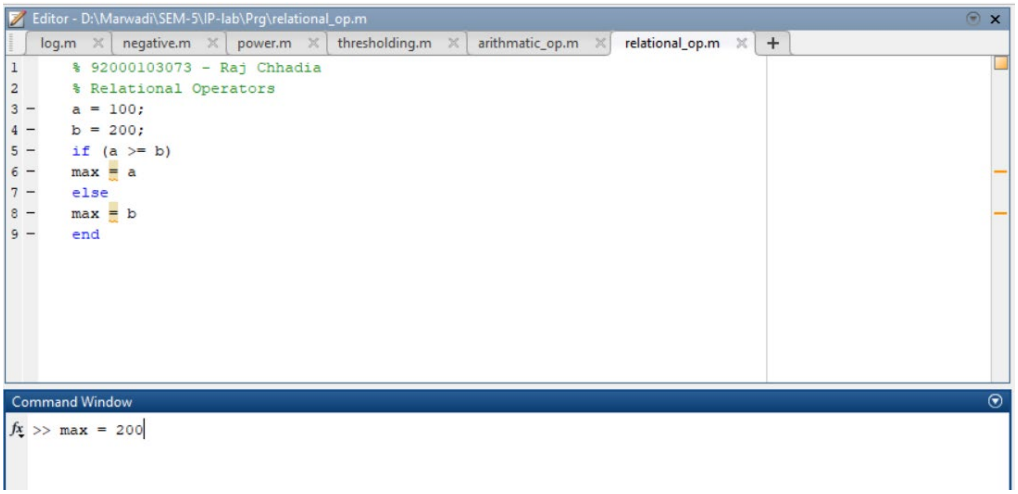
▪ Arithmetic Operators

No.	Symbol	Description
1.	+	Addition or unary plus. A+B adds the values stored in variables A and B.
2.	-	Subtraction or unary minus. A-B subtracts value of B from A.
3.	*	Matrix multiplication. C = A*B is the linear algebraic product of the matrices A and B.
4.	.*	Array multiplication. A.*B is the element-by-element product of the arrays A and B.
5.	/	Slash or matrix right division. B/A is roughly the same as B*inv(A).
6.	./	Array right division. A./B is the matrix with elements A(i,j)/B(i,j).
7.	\	Backslash or matrix left division. If A is a square matrix, A\B is roughly the same as inv(A)*B.
8.	.\	Array left division. A.\B is the matrix with elements B(i,j)/A(i,j).
9.	^	Matrix power. X^p is X to the power p, if p is a scalar. If p is an integer, the power is computed by repeated squaring.
10.	.^	Array power. A.^B is the matrix with elements A(i,j) to the B(i,j) power.
11.	‘	Matrix transpose. A' is the linear algebraic transpose of A.
12.	.’	Array transpose. A.' is the array transpose of A.



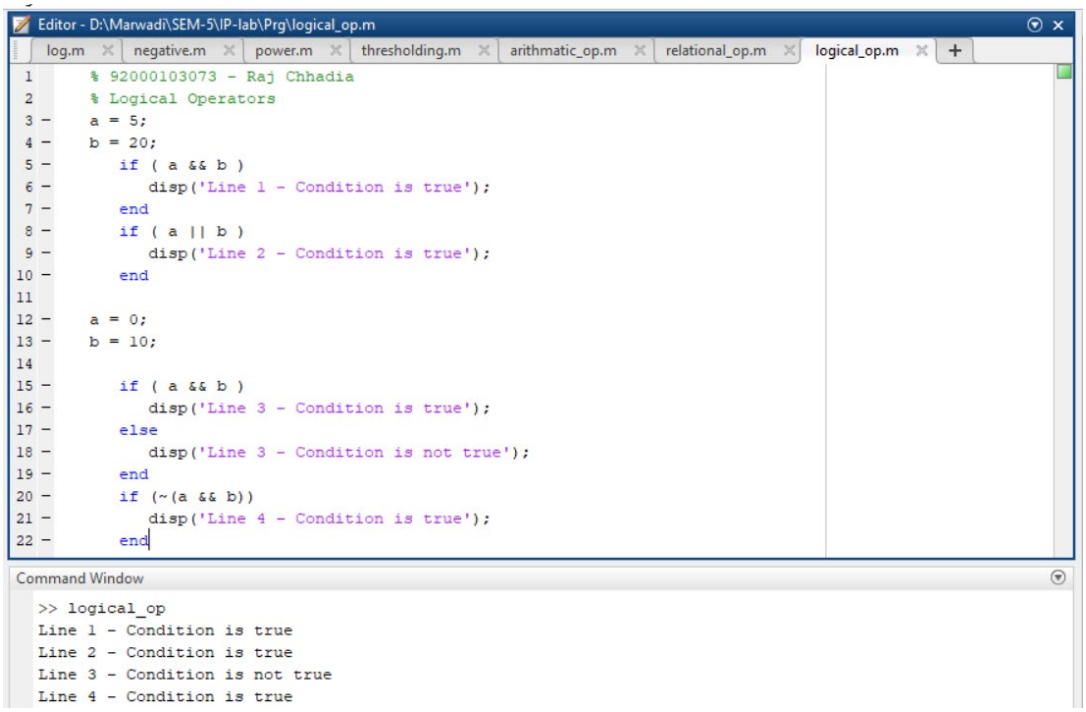
- **Relational Operators**

Symbol	Description
==	Determine Equality
>=	Determine greater than or equal to
>	Determine greater than
<=	Determine less than or equal to
<	Determine less than
~=	Determine inequality
isequal	Determine array equality.
isequaln	Determine array equality, treating NaN values as equal

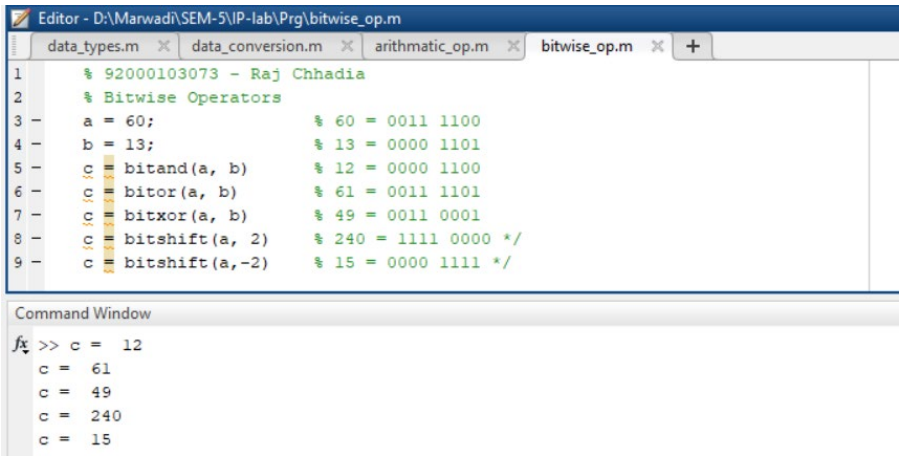


- **Logical Operators**

Symbol	Description
&& , ||	Short Circuit Operations
&	Logical AND
~	Logical NOT
|	Logical OR
xor	Logical XOR
all	Determine if all array elements are nonzero or not
any	Determine if any value is zero in array
false	Logical 0 (low)
true	Logical 1(high)

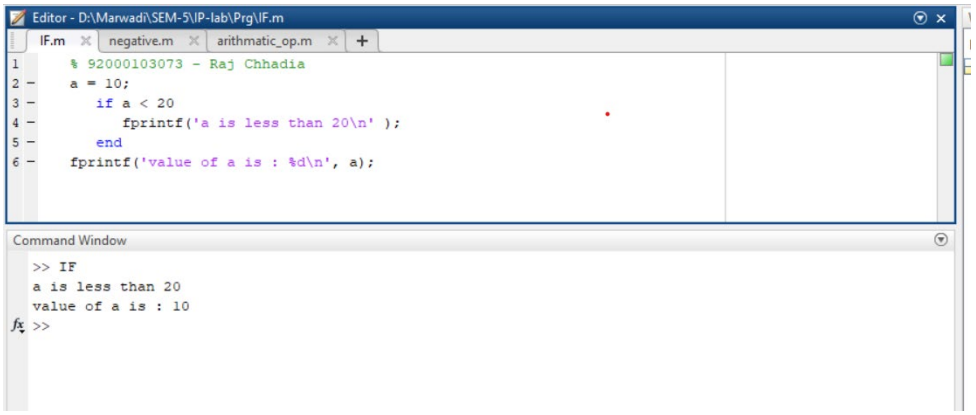


- **Bitwise Operations**

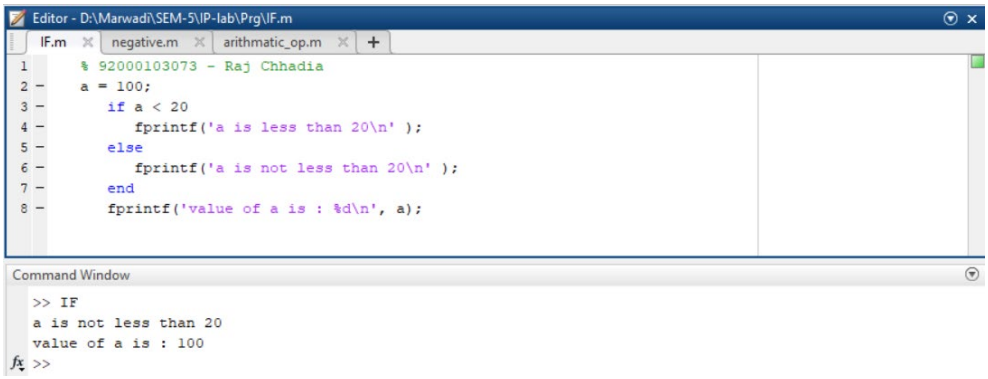


➢ Decision Making in MATLAB

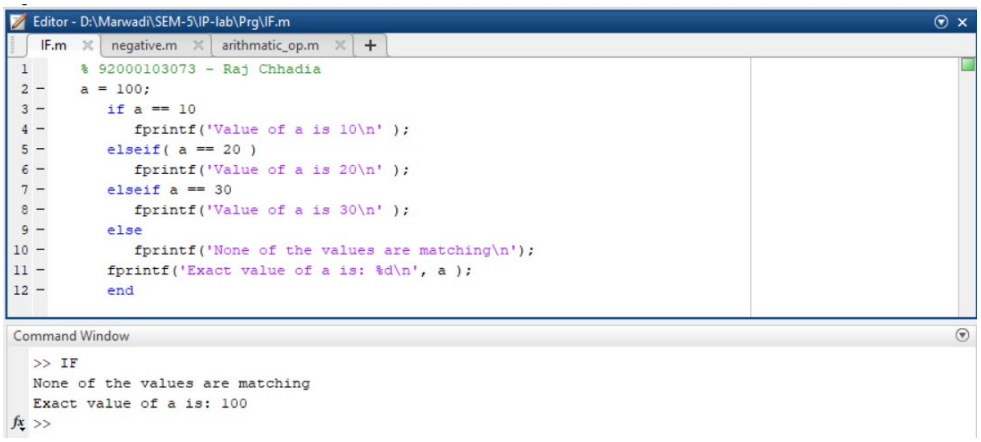
○ if ... end statement



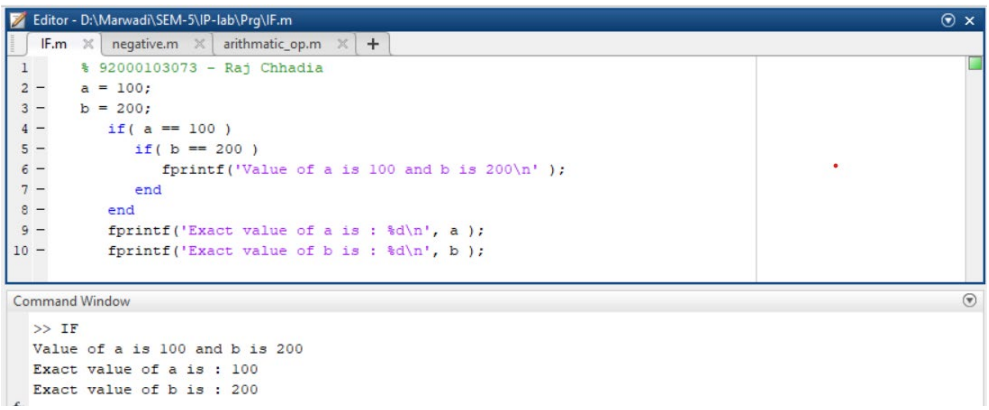
○ if...else...end statement



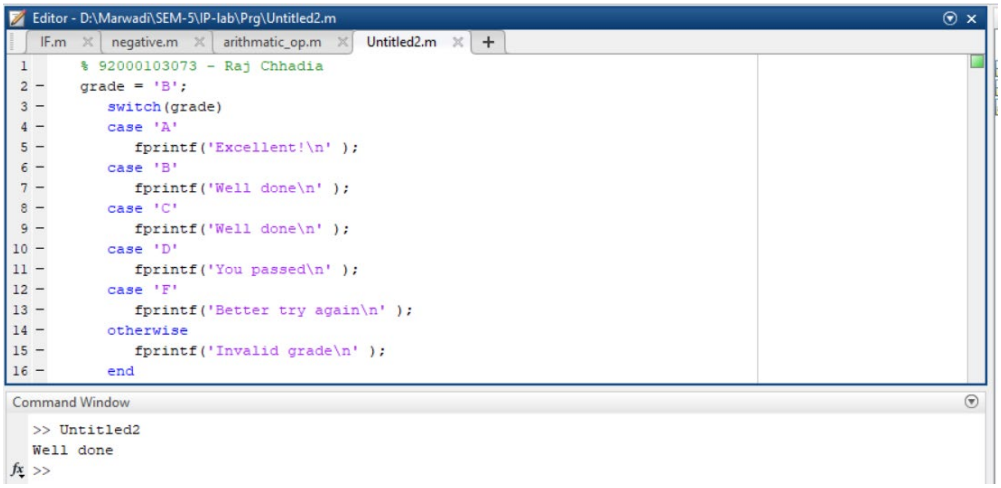
- **If... elseif...elseif...else...end statements**



- **nested if statements**

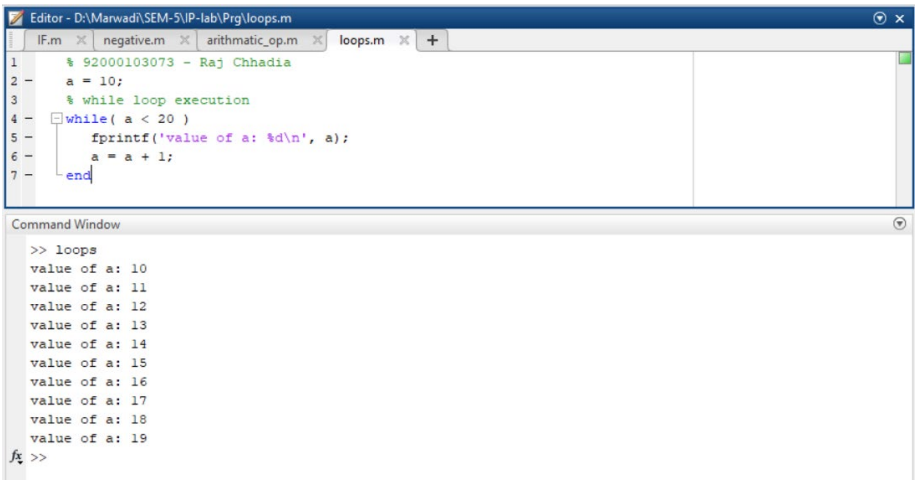


- **switch statement**

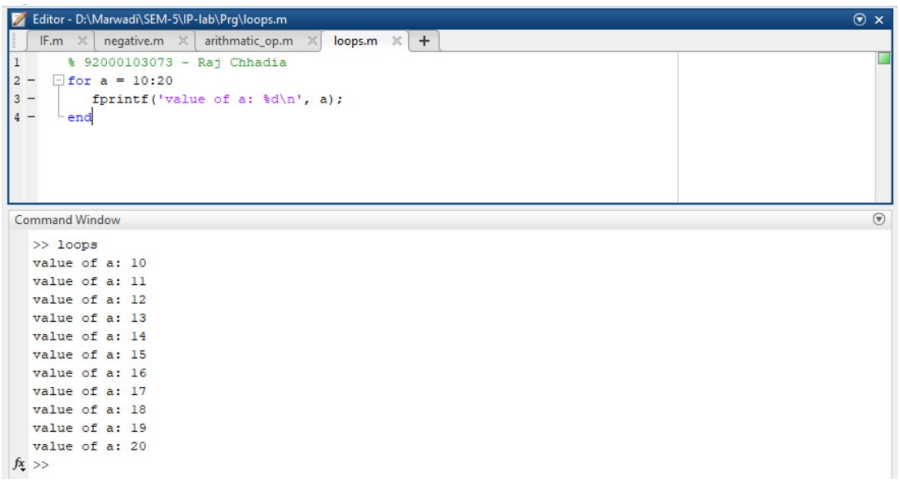


➢ Loop Types in MATLAB

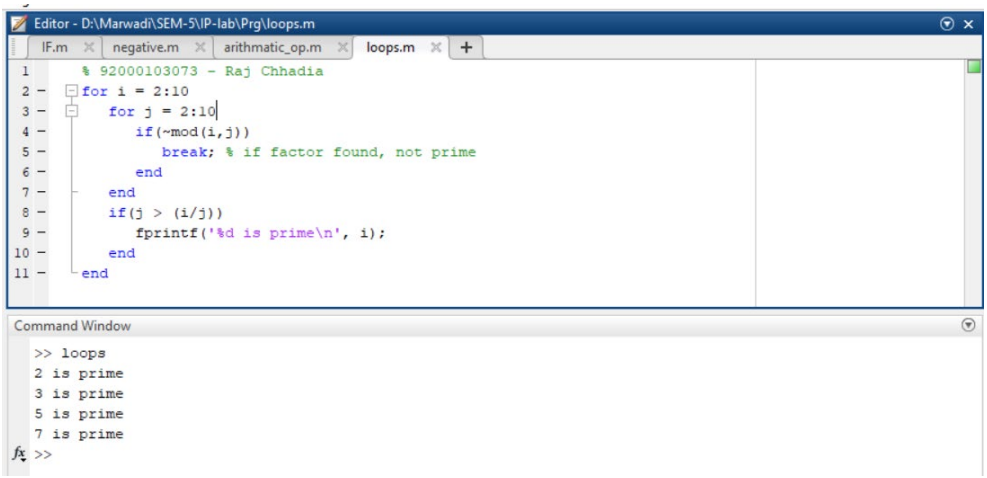
○ while loop



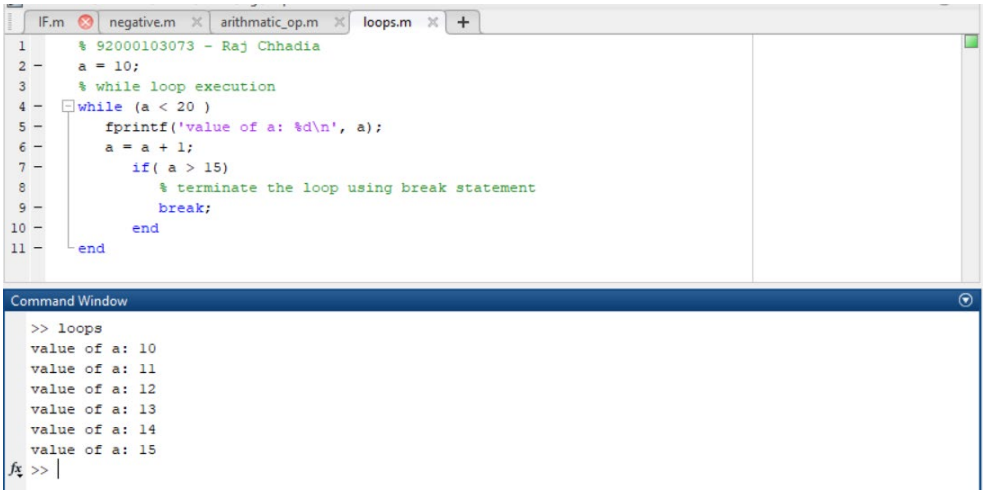
○ for loop



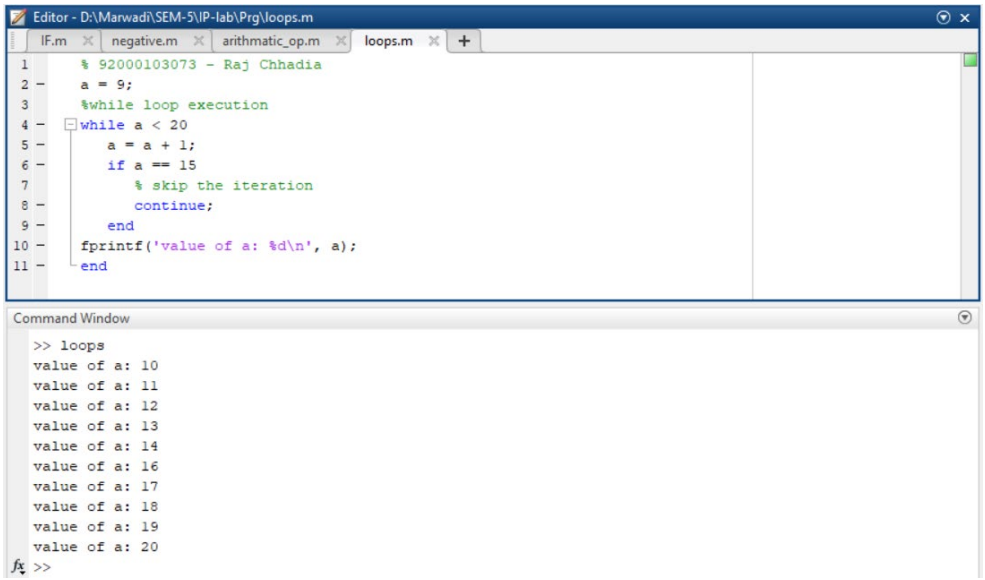
○ nested loops



- **break statement**

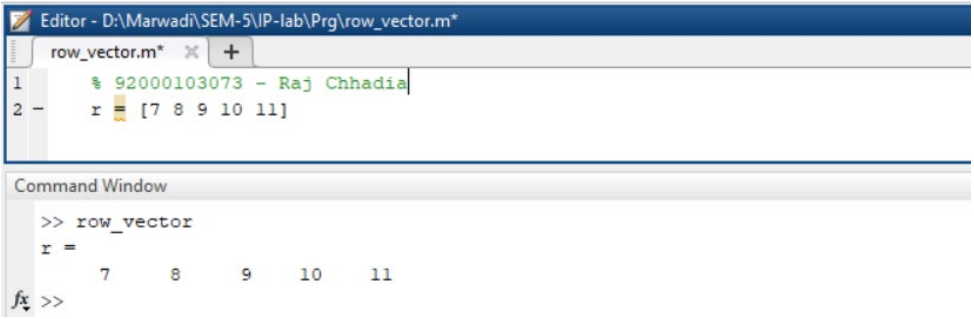


- **continue statement**

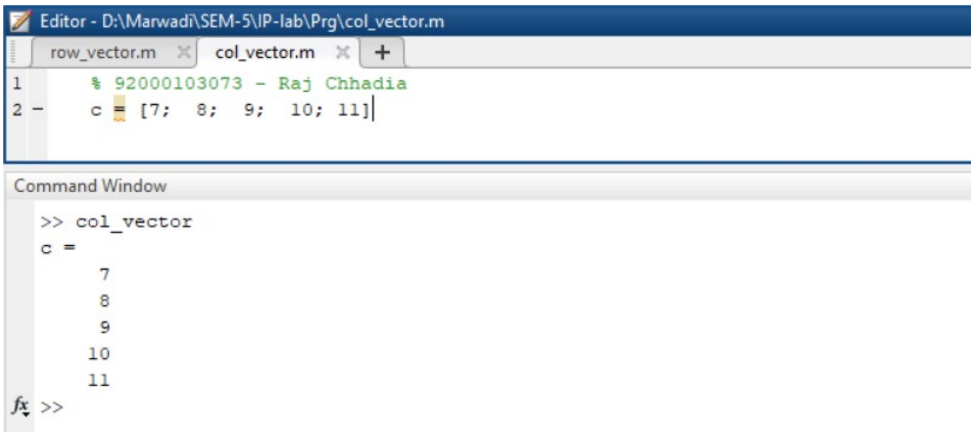


➢ Vectors in MATLAB

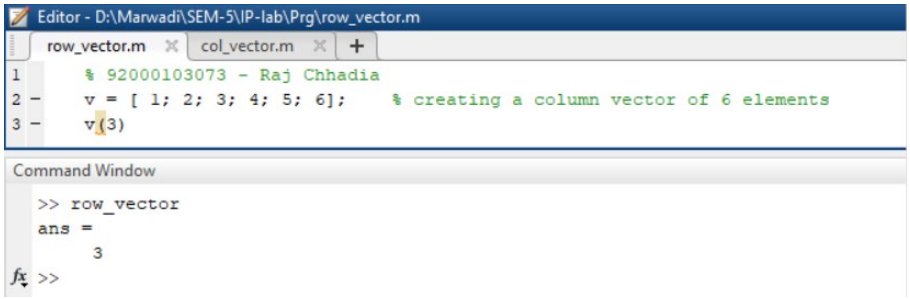
- **Row Vectors**



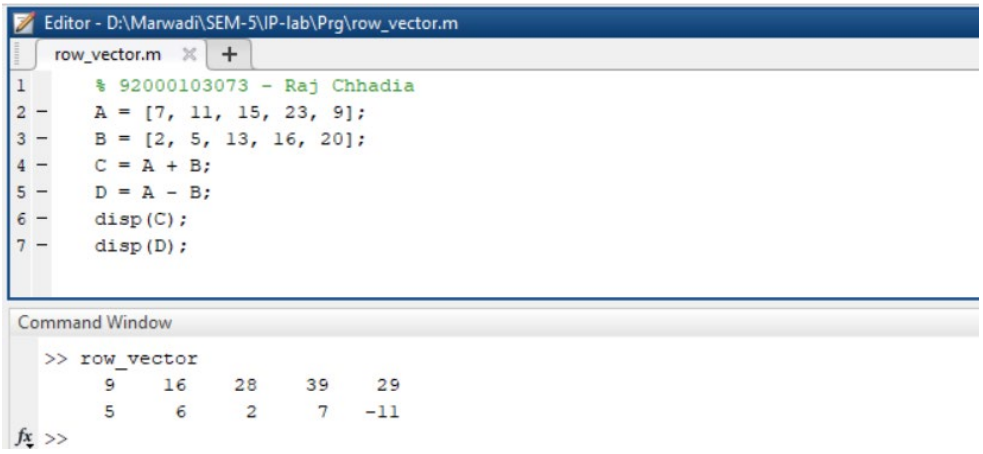
- **Column Vectors**



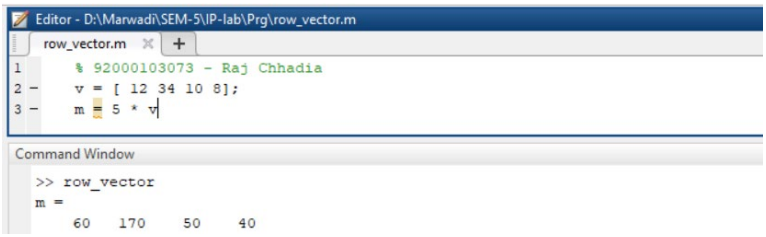
- **Referencing the Elements of a Vector**



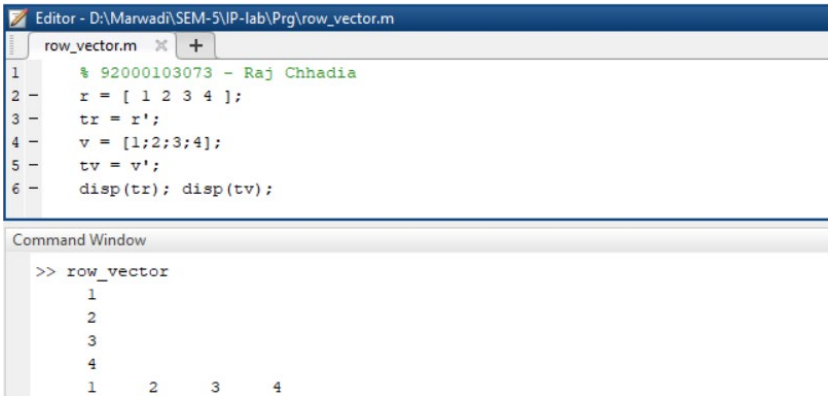
- **Vector Operations**
  - **Addition and Subtraction of Vectors**



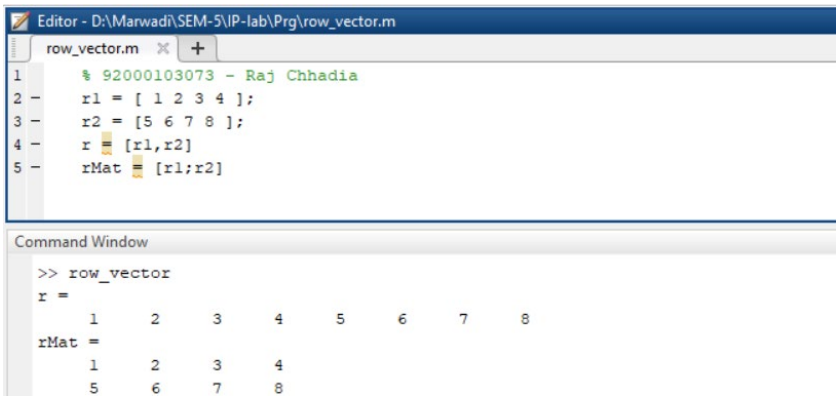
- **Scalar Multiplication of Vectors**



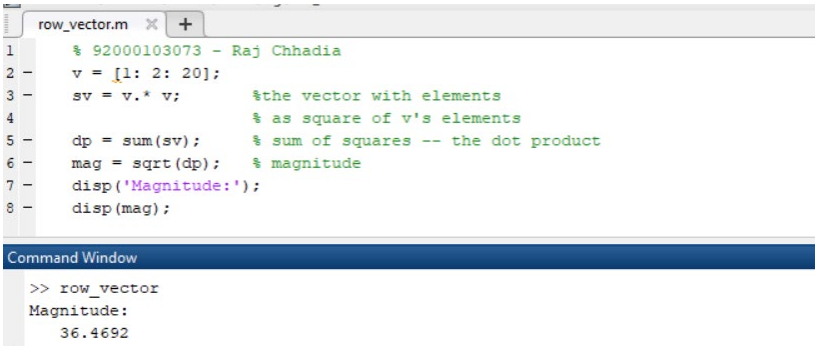
- **Transpose of a Vector**



- **Appending Vectors**



- **Magnitude of a Vector**



- **Vector Dot Product**

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\row_vector.m
row_vector.m
1      % 92000103073 - Raj Chhadia
2 -    v1 = [2 3 4];
3 -    v2 = [1 2 3];
4 -    dp = dot(v1, v2);
5 -    disp('Dot Product:');
6 -    disp(dp);
```

```
Command Window
>> row_vector
Dot Product:
    20
```

- **Vectors with Uniformly Spaced Elements**

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\row_vector.m
row_vector.m
1      % 92000103073 - Raj Chhadia
2 -    v = [1: 2: 20];
3 -    sqv = v.^2;
4 -    disp(v);
5 -    disp(sqv);
```

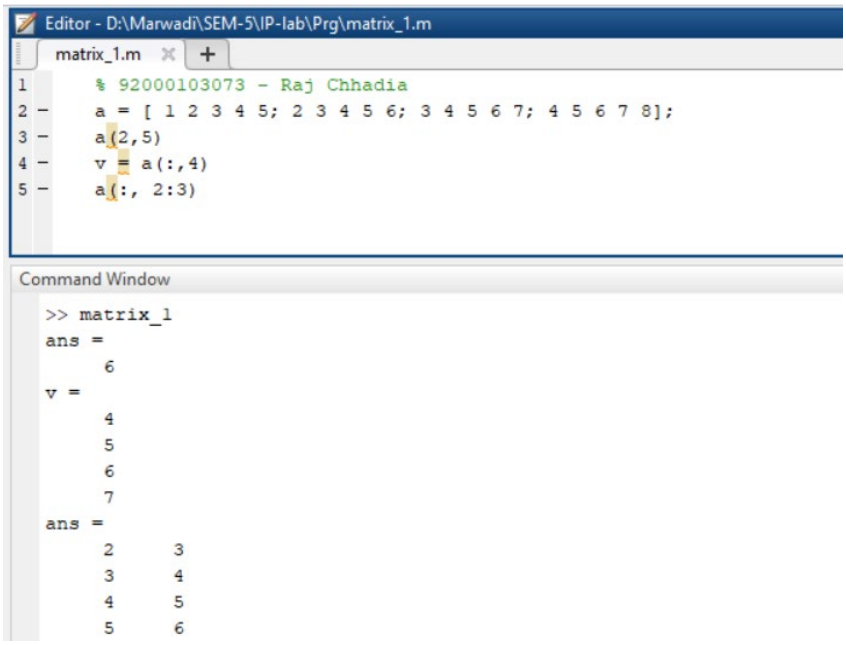
```
Command Window
>> row_vector
     1     3     5     7     9    11    13    15    17    19
     1     9    25    49    81   121   169   225   289   361
fx >>
```

➢ Matrix in MATLAB

○ Referencing the Elements of a Matrix

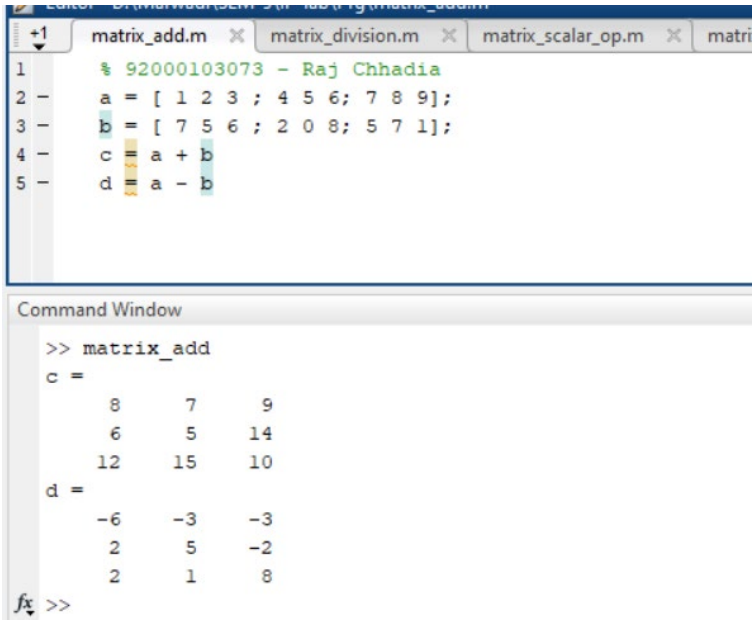
→ To reference an element in the mth row and nth column, of a matrix *mx*, we write − mx(m, n);

→ To reference all the elements in the mth column we type A(:,m).



○ Matrix Operations

▪ Addition and Subtraction of Matrices



- **Division of Matrices**

```
matrix_division.m | matrix_scalar_op.m | matrix_transpose.m | matrix_c
1     % 92000103073 - Raj Chhadia
2 -   a = [ 10 2 3 ; 4 5 6; 7 8 9];
3 -   b = [ 7 5 6 ; 5 6 7; 9 8 7];
4 -   c = a / b
5 -   d = a \ b
```

```
Command Window
>> matrix_division
c =
    3.0000   -2.0714   -0.0714
    0.0000    0.9286   -0.0714
   -0.0000    1.1429    0.1429
d =
    0.6667    0.1111   -0.1111
    1.6667   -2.2222   -6.7778
   -1.0000    2.7778    6.8889
fx >>
```

- **Scalar Operations of Matrices**

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\matrix_scalar_op.m
matrix_scalar_op.m | matrix_transpose.m | matrix_concat.m | matrix_n
1     % 92000103073 - Raj Chhadia
2 -   a = [ 10 12 23 ; 14 8 6; 27 8 9];
3 -   b = 2;
4 -   c = a + b
5 -   d = a - b
6 -   e = a * b
7 -   f = a / b
```

```
Command Window
>> matrix_scalar_op
c =
    12    14    25
    16    10     8
    29    10    11
d =
     8    10    21
    12     6     4
    25     6     7
e =
    20    24    46
    28    16    12
    54    16    18
f =
    5.0000    6.0000   11.5000
    7.0000    4.0000    3.0000
   13.5000    4.0000    4.5000
fx >>
```

- **Transpose of a Matrix**

Editor - D:\Marwadi\SEM-5\IP-lab\Prg\matrix_transpose.m

matrix_transpose.m | matrix_concat.m | matrix_multiplicat

```
% 92000103073 - Raj Chhadia
a = [ 10 12 23 ; 14 8 6; 27 8 9]
b = a'
```

Command Window

```
>> matrix_transpose
a =
    10    12    23
    14     8     6
    27     8     9
b =
    10    14    27
    12     8     8
    23     6     9
fx >>
```

- **Concatenating Matrices**

Editor - D:\Marwadi\SEM-5\IP-lab\Prg\matrix_concat.m

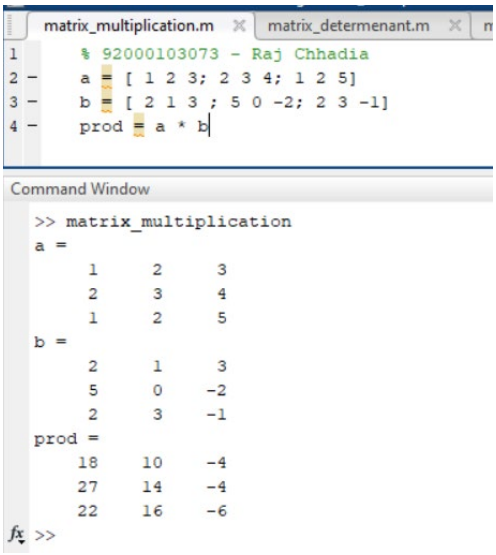
matrix_concat.m | matrix_multiplication.m | matrix_determenant.m

```
% 92000103073 - Raj Chhadia
a = [ 10 12 23 ; 14 8 6; 27 8 9]
b = [ 12 31 45 ; 8 0 -9; 45 2 11]
c = [a, b]
d = [a; b]
```

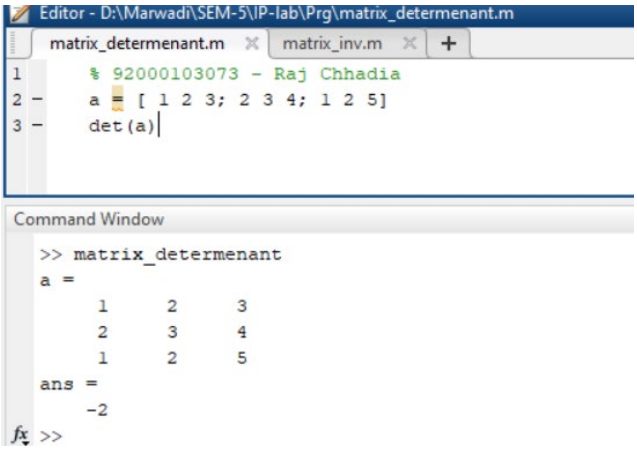
Command Window

```
>> matrix_concat
a =
    10    12    23
    14     8     6
    27     8     9
b =
    12    31    45
     8     0    -9
    45     2    11
c =
    10    12    23    12    31    45
    14     8     6     8     0    -9
    27     8     9    45     2    11
d =
    10    12    23
    14     8     6
    27     8     9
    12    31    45
     8     0    -9
    45     2    11
fx >>
```

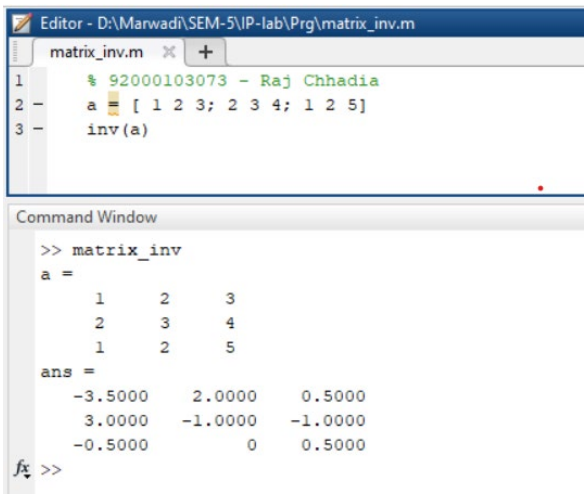
- **Matrix Multiplication**



- **Determinant of a Matrix**



- **Inverse of a Matrix**

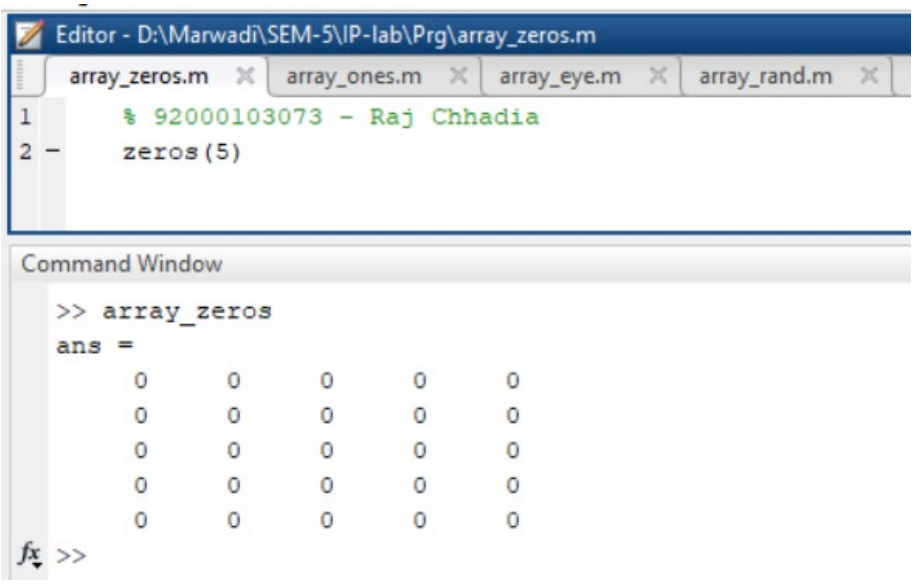


➢ Arrays in MATLAB

○ Special Arrays in MATLAB

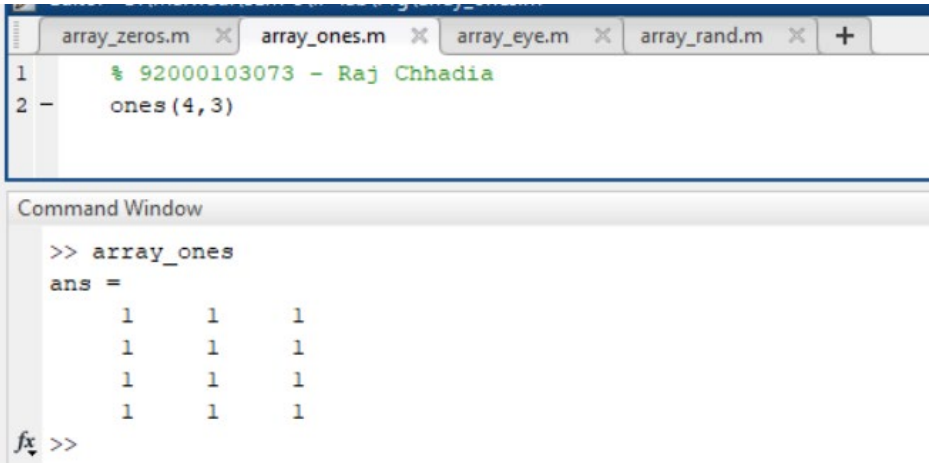
▪ zeros( ) function

→ The zeros() function creates an array of all zeros .



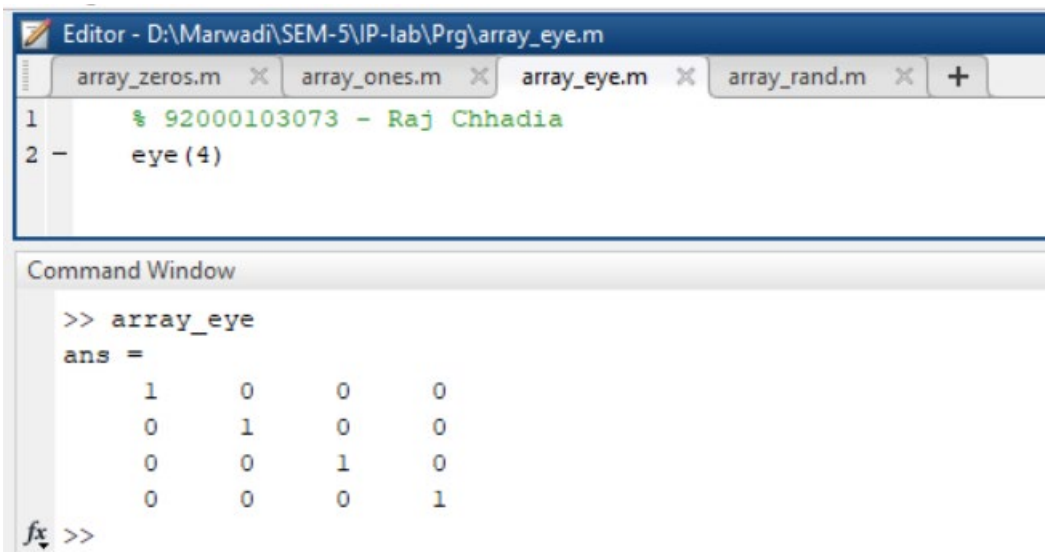
▪ ones( ) function

→ The ones() function creates an array of all ones.



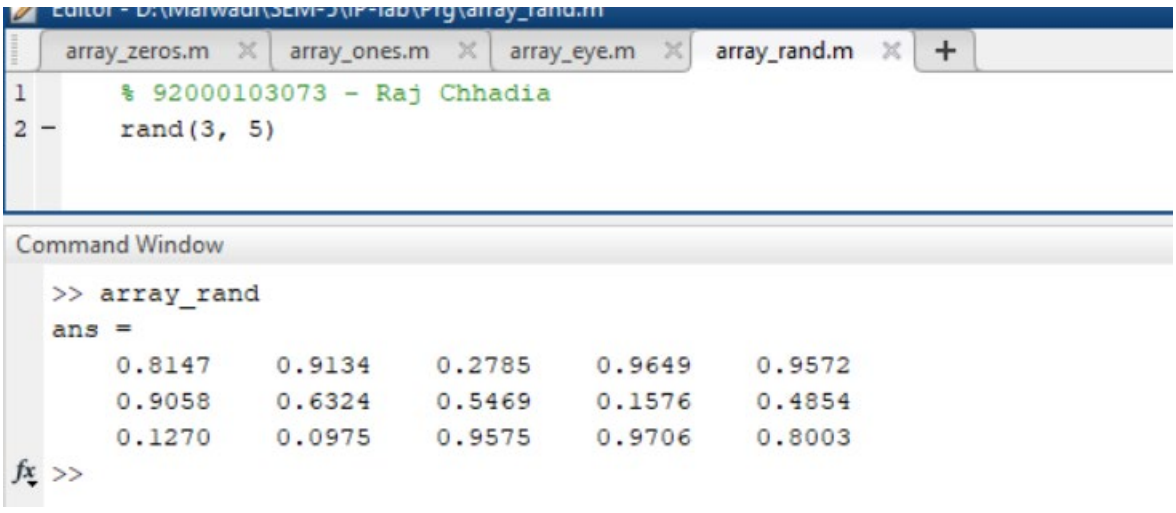
- **eye( ) function**

→ The eye() function creates an identity matrix.



- **rand( ) function**

→ The rand() function creates an array of uniformly distributed random numbers on (0,1) .



➢ Image Processing Toolkit

○ imread( ) and imshow( ) command

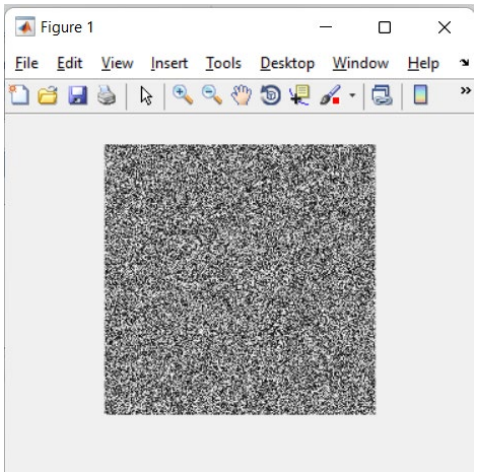
→ imread() reads the image from the file specified by filename, imshow() displays the grayscale image I in a figure

```
% 92000103073 - Raj Chhadia
A = imread('cameraman.tif');
imshow(A)
title('Cameraman');
```

○ imwrite( ) command

→ imwrite() writes image data A to the file specified by filename.

```
% 92000103073 - Raj Chhadia
A = rand(250);
imwrite(A,'myGray.png')
imshow('myGray.png')
```



- **size( ) command**

→ size() is used to know number of pixels in a image.

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\toolkit.m
bit_plane.m | toolkit.m
1   % 92000103073 - Raj Chhadia
2 - x=imread('pepper.jpg');
3 - size(x,1)
4 - size(x,2)
```

```
Command Window
>> toolkit
ans =
   193
ans =
   261
fx >>
```

- **imfinfo( ) command**

→ imfinfo() returns a structure whose fields contain information about an image in a graphics file.

```
IP ToolKit.m | toolkit.m
1   % 92000103073 - Raj Chhadia
2 - info = imfinfo('myGray.png')
```

```
Command Window
>> toolkit
info =
                  Filename: 'D:\Marwadi\SEM-5\IP-lab\Prg\myGray.png'
               FileModDate: '25-Aug-2022 22:50:58'
                  FileSize: 62936
                    Format: 'png'
             FormatVersion: []
                     Width: 250
                    Height: 250
                  BitDepth: 8
                 ColorType: 'grayscale'
           FormatSignature: [137 80 78 71 13 10 26 10]
                  Colormap: []
                 Histogram: []
             InterlaceType: 'none'
              Transparency: 'none'
   SimpleTransparencyData: []
          BackgroundColor: []
          RenderingIntent: []
           Chromaticities: []
                    Gamma: []
               XResolution: []
               YResolution: []
           ResolutionUnit: []
                  XOffset: []
                  YOffset: []
```

- **impixel( ) command**

→ impixel() lets you select pixels interactively from the image in the current axes.

```
% 92000103073 - Raj Chhadia
RGB = imread('peppers.png');
c = [12 146 410];
r = [104 156 129];
pixels = impixel(RGB,c,r)
```

```
Command Window
>> toolkit
pixels =
    62    34    63
   166    54    60
    59    28    47
fx >>
```

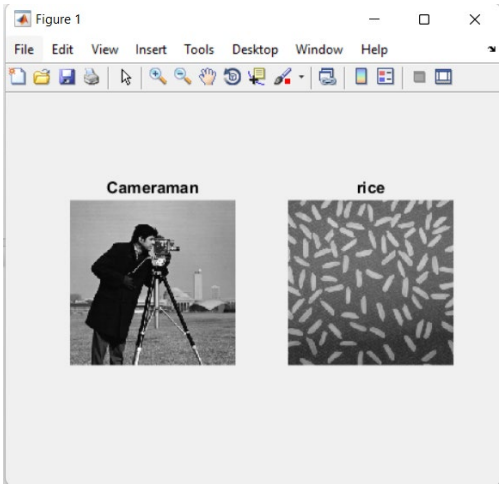
- **subplot( ) command**

→ subplot(m,n,p) divides the current figure into an m-by-n grid and creates axes in the position specified by p.

Editor - D:\Marwadi\SEM-5\IP-lab\Prg\toolkit.m

IP ToolKit.m | toolkit.m

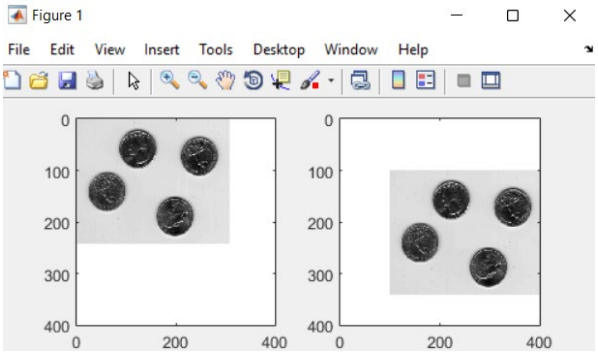
```
% 92000103073 - Raj Chhadia
subplot(1,2,1);
A = imread('cameraman.tif');
imshow(A)
title('Cameraman');


subplot(1,2,2);
B = imread('rice.png');
imshow(B)
title('rice');
```



○ imagesc( ) command

→ imagesc(C) displays the data in array C as an image that uses the full range of colors in the colormap.

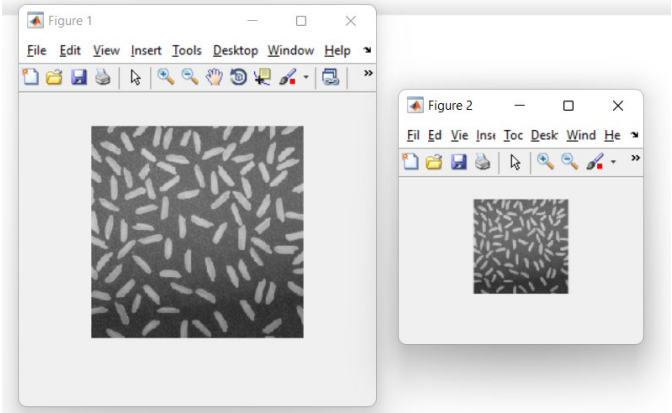
```
% 92000103073 - Raj Chhadia
i = imread('eight.tif');
figure;
subplot(2,2,1);
    imagesc(i);
    axis([0 400 0 400]);
    colormap(gray);
subplot(2,2,2);
    imagesc(100,100,i);
    axis([0 400 0 400]);
    colormap(gray);
```



○ imresize( ) command

→ B = imresize(A,scale) returns image B that is scale times the size of image A.

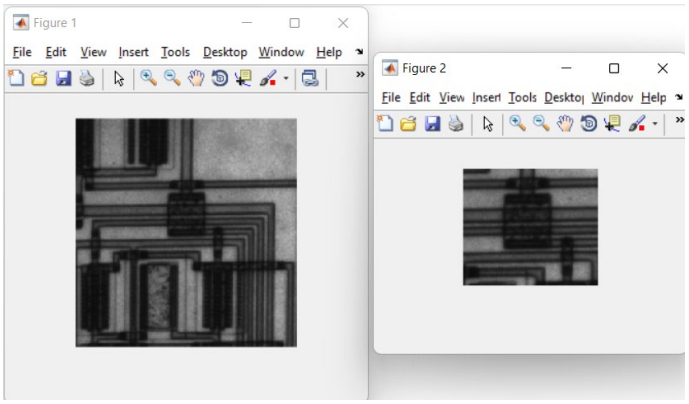
```
% 92000103073 - Raj Chhadia
I = imread('rice.png');
J = imresize(I, 0.5);
figure, imshow(I), figure, imshow(J)
```



- **imcrop( ) command**

→ imcrop() creates an interactive Crop Image tool associated with the grayscale, truecolor, or binary image displayed in the current figure.

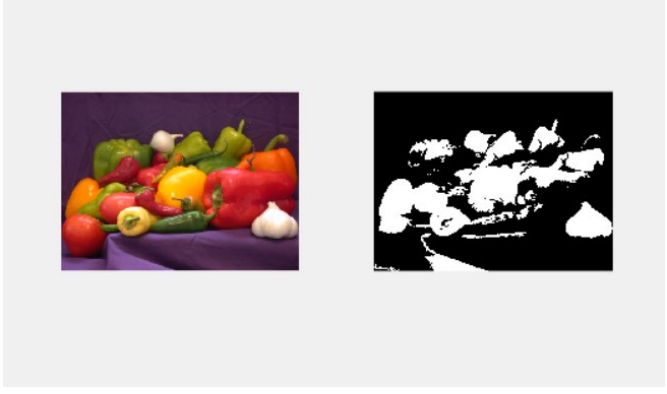
```
% 92000103073 - Raj Chhadia
I = imread('circuit.tif');
I2 = imcrop(I,[75 68 130 112]);
imshow(I), figure, imshow(I2)
```



- **im2bw( ) command**

→ im2bw() converts the grayscale image I to binary image.

```
% 92000103073 - Raj Chhadia
subplot(1,2,1);
    X = imread('peppers.png');
    imshow(X)
 subplot(1,2,2);
     BW = im2bw(X,0.4);
     imshow(BW)
```



- **rgb2gray( ) command**

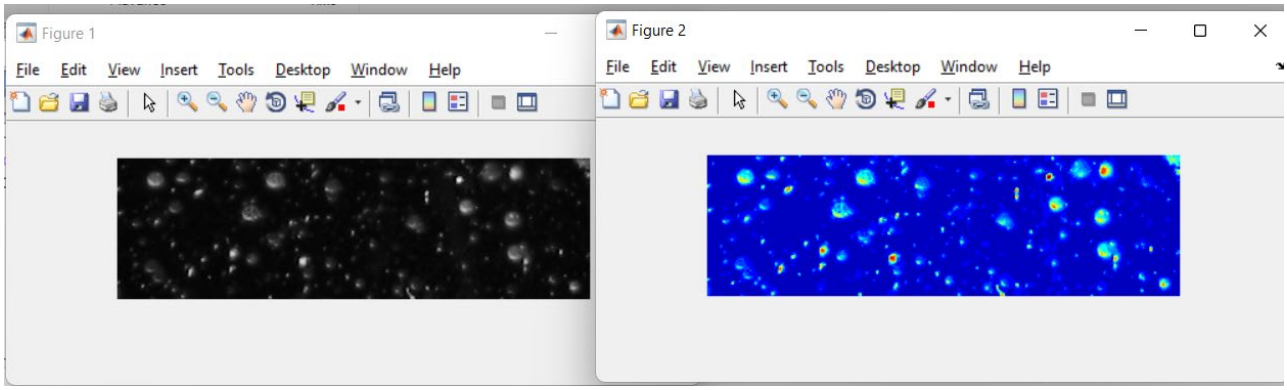
→ The rgb2gray() function converts RGB images to grayscale.

```
toolkit.m | IP_ToolKit.m
1   % 92000103073 - Raj Chhadia
2 - subplot(1,2,1);
3 -     RGB = imread('peppers.png');
4 -     imshow(RGB)
5 -  subplot(1,2,2);
6 -     I = rgb2gray(RGB);
7 -     imshow(I)
```


- **grayslice( ) command**

→ grayslice() converts a grayscale image to an indexed image by using multilevel thresholding approach.

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\toolkit.m
toolkit.m | IP_ToolKit.m
1   % 92000103073 - Raj Chhadia
2 - I = imread('snowflakes.png');
3 - X = grayslice(I,16);
4 - imshow(I)
5 - figure, imshow(X,jet(16))
```



- **imadd( ) command**

→ imadd() adds each element in array X with the corresponding element in array Y and returns the sum.

```
toolkit.m | IP_ToolKit.m
1      % 92000103073 - Raj Chhadia
2 -    X = uint8([ 255 0 75; 44 225 100]);
3 -    Y = uint8([ 50 50 50; 50 50 50 ]);
4 -    Z = imadd(X,Y)
```

```
Command Window
>> toolkit
Z =
  255   50  125
   94  255  150
fx >>
```

- **imsubtract( ) command**

→ imsubtract() subtracts each element in array Y from the corresponding element in array X and returns the difference.

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\toolkit.m
toolkit.m | IP_ToolKit.m
1      % 92000103073 - Raj Chhadia
2 -    X = uint8([ 255 10 75; 44 225 100]);
3 -    Y = uint8([ 50 50 50; 50 50 50 ]);
4 -    Z = imsubtract(X,Y)
```

```
Command Window
>> toolkit
Z =
  205    0   25
    0  175   50
fx >>
```

- **imdivide( ) command**

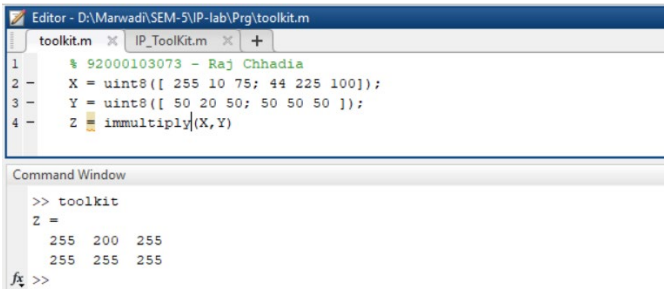
→ imdivide() divides each element in the array X by corresponding element in array Y.

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\toolkit.m
toolkit.m | IP_ToolKit.m
1      % 92000103073 - Raj Chhadia
2 -    X = uint8([ 255 10 75; 44 225 100]);
3 -    Y = uint8([ 50 20 50; 50 50 50 ]);
4 -    Z = imdivide(X,Y)
```

```
Command Window
>> toolkit
Z =
     5     1     2
     1     5     2
fx >>
```

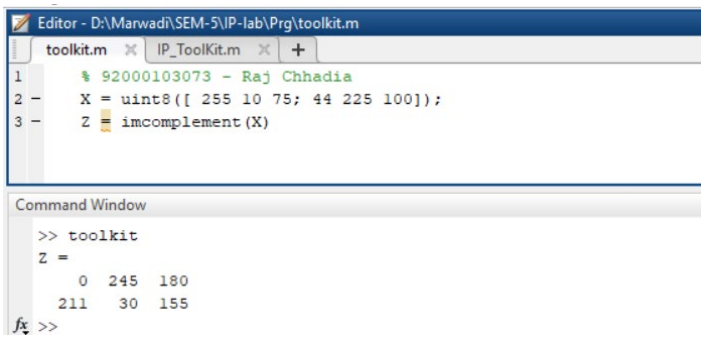
- **immultiply( ) command**

→ immultiply() multiplies each element in array X by the corresponding element in array Y and returns product.



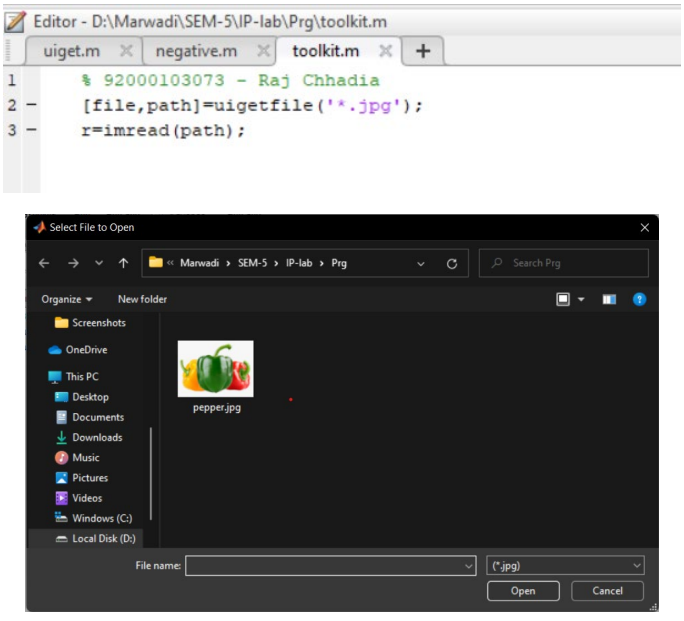
- **imcomplement( ) command**

→ imcomplement() computes the complement of the image I.



- **uigetfile( ) command**

→ uigetfile() opens file selection dialog box.



Practical 2

Aim: Point processing in spatial domain

a. Negation of an image
b. Thresholding of an image
c. Contrast Stretching of an image

a. Negation of an image

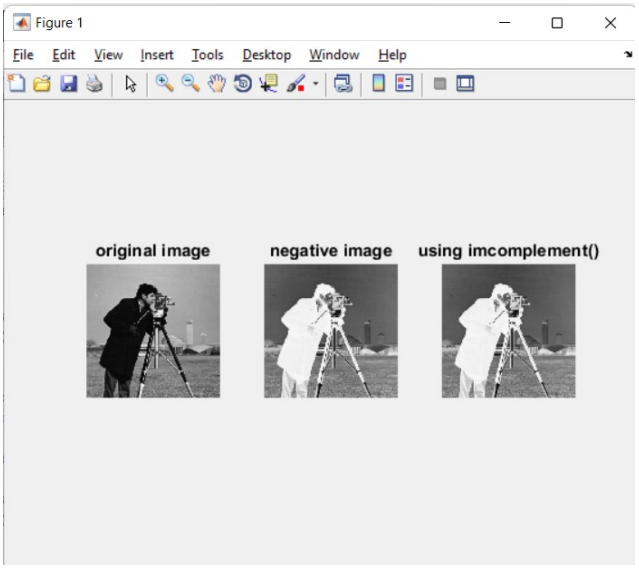
Code:

Editor - D:\Marwadi\SEM-5\IP-lab\Prg\negative.m

negative.m

```
r=imread('cameraman.tif');
s=r;

subplot(1,3,1);
    imshow (r);
    title('original image');

subplot(1,3,2);
    for row = 1 : size(r,1)
        for col = 1 : size(r,2)
            s(row,col) = 255 - r(row,col);
        end
    end
    imshow (s);
    title('negative image');

subplot(1,3,3);
    q=imcomplement(r);
    imshow(q);
    title('using imcomplement()');
fprintf('92000103073 Raj Chhadia');
```

Output:



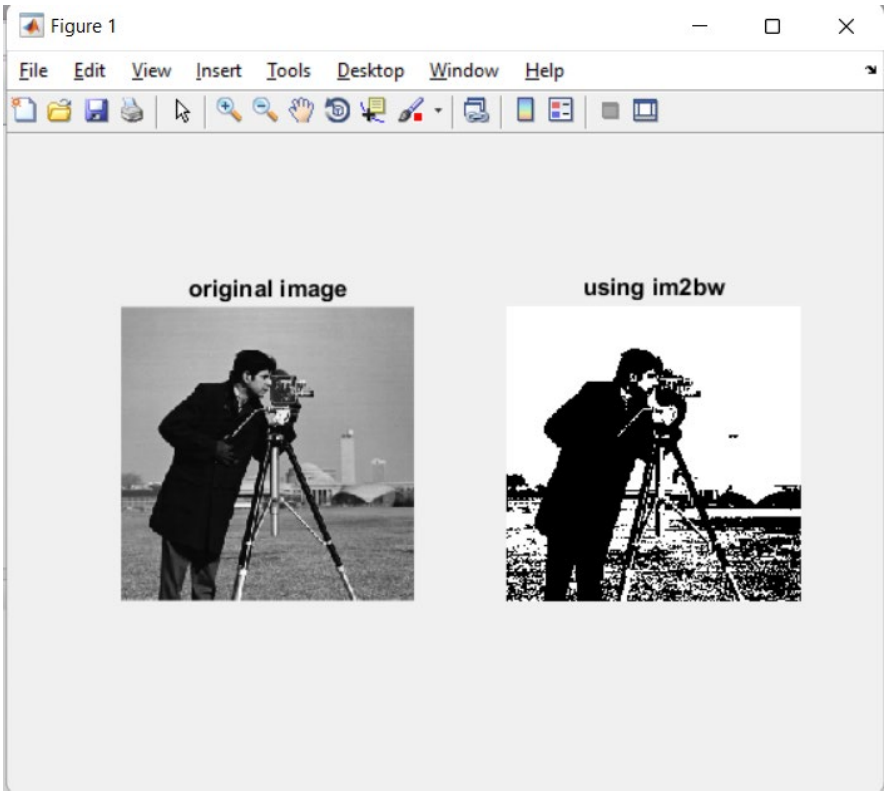
b. Thresholding of an image

Code:

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\thresholding.m
negative.m | thresholding.m
1 -    x=imread('cameraman.tif');
2
3 -    subplot(1,2,1);
4 -        imshow (x);
5 -        title('original image');
6
7 -    subplot(1,2,2);
8 -        g=im2bw(x);
9 -        imshow (g);
10 -       title('using im2bw');
11 -       fprintf('92000103073 Raj Chhadia');
12
```

```
Command Window
>> thresholding
92000103073 Raj Chhadia>>
```

Output:



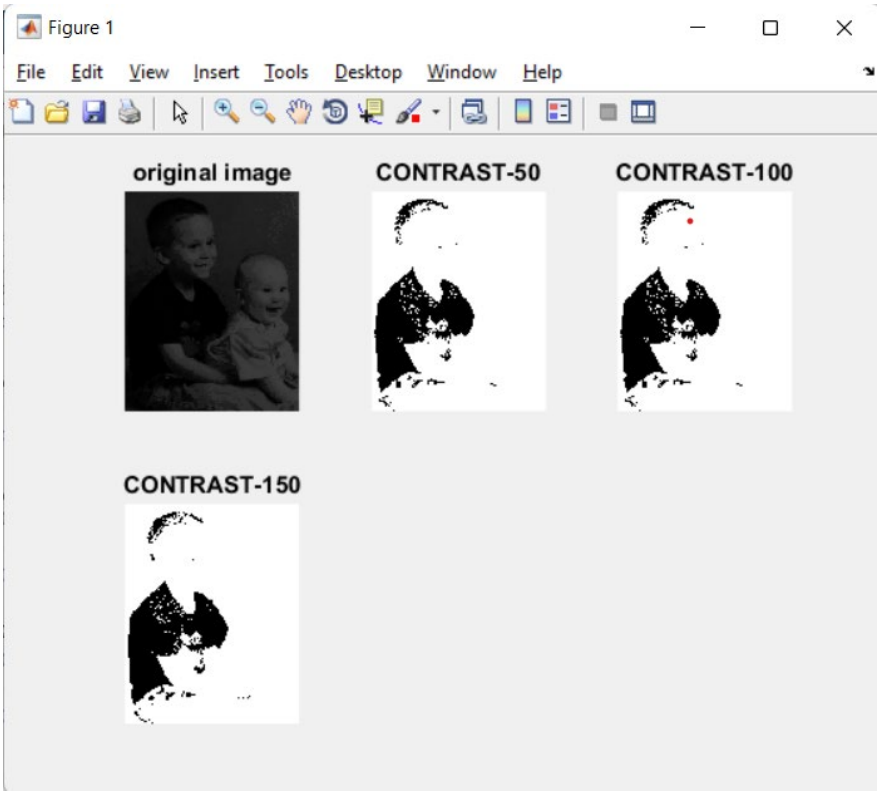
c. Contrast Stretching of an image

Code:

```
x=imread('kids.tif');

subplot(2,3,1);
    imshow (x);
    title('original image');

subplot(2,3,2);
    s=1./(50./(double(x)+eps ).^20);
    imshow (s);
    title('CONTRAST-50');

   subplot(2,3,3);
    s=1./(100./(double(x)+eps ).^20);
    imshow (s);
    title('CONTRAST-100');

    subplot(2,3,4);
    s=1./(150./(double(x)+eps ).^20);
    imshow (s);
    title('CONTRAST-150');
    fprintf('92000103073 Raj Chhadia');
```

```
>> constrast
92000103073 Raj Chhadia>>
```

Output:



Extra:

1. Log Transformation
2. Power-Law Functions
3. Gray-level slicing
4. Bit-plane slicing

1. Log Transformation

Code:

Editor - D:\Marwadi\SEM-5\IP-lab\Prg\log.m

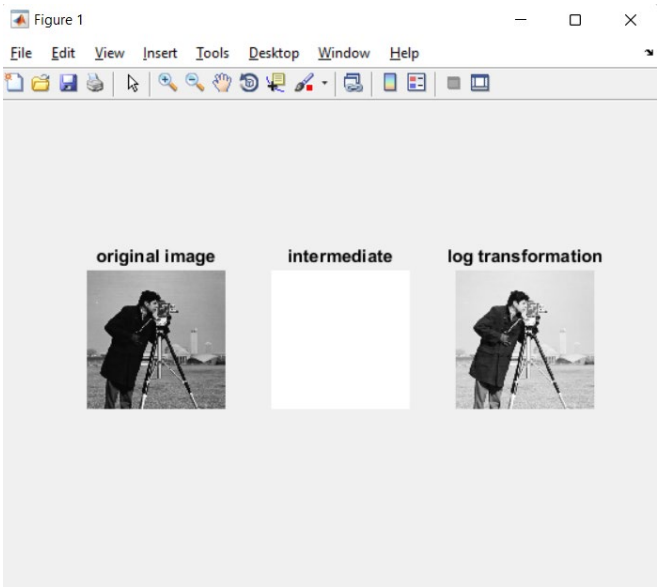
log.m

```
1 -    x=imread('cameraman.tif');
2
3 -    subplot(1,3,1);
4 -        imshow (x);
5 -        title('original image');
6
7 -    subplot(1,3,2);
8 -        g=log(1 + double(x));
9 -        imshow (g);
10 -       title('intermediate');
11
12 -   subplot(1,3,3);
13 -       gs=im2uint8(mat2gray(g));
14 -       imshow (gs);
15 -       title('log transformation');
16 -       fprintf('92000103073 Raj Chhadia');
```

Command Window

```
>> log
92000103073 Raj Chhadia>>
```

Output:



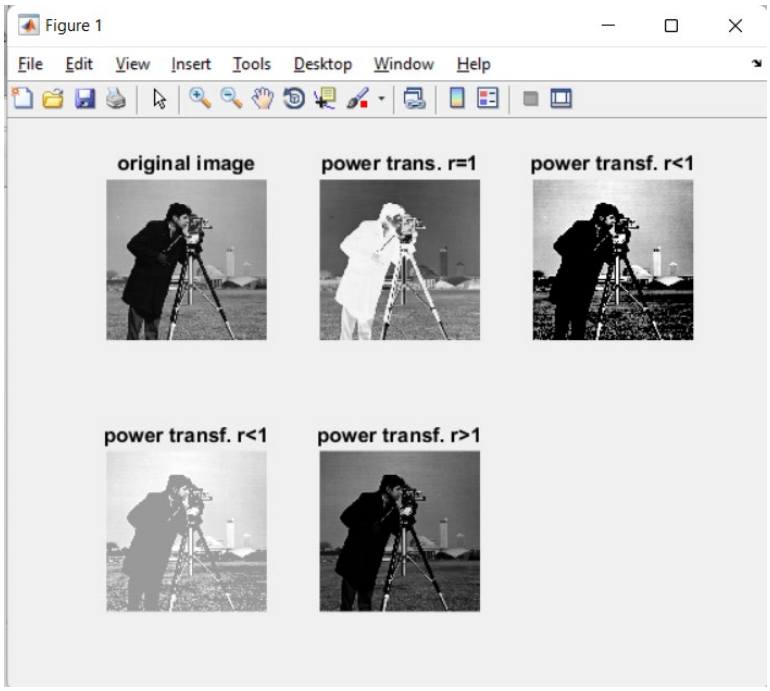
2. Power-Law Functions

Code:

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\power.m
power.m
1 -    x=imread('cameraman.tif');
2 -    subplot(2,3,1);
3 -        imshow (x);
4 -        title('original image');
5 -    subplot(2,3,2);
6 -        g=imadjust(x,[0 1],[1,0]);
7 -        imshow (g);
8 -        title('power trans. r=1');
9 -    subplot(2,3,3);
10 -       g=imadjust(x,[0.5 0.75],[0 1],0.5);
11 -       imshow (g);
12 -       title('power transf. r<1');
13 -   subplot(2,3,4);
14 -       g=imadjust(x,[0.5 0.75],[0.6 1],0.5);
15 -       imshow (g);
16 -       title('power transf. r<1');
17 -   subplot(2,3,5);
18 -       g=imadjust(x,[],[],2);
19 -       imshow (g);
20 -       title('power transf. r>1');
21 -       fprintf('92000103073 Raj Chhadia');
```

```
Command Window
>> power
92000103073 Raj Chhadia>>
```

Output:



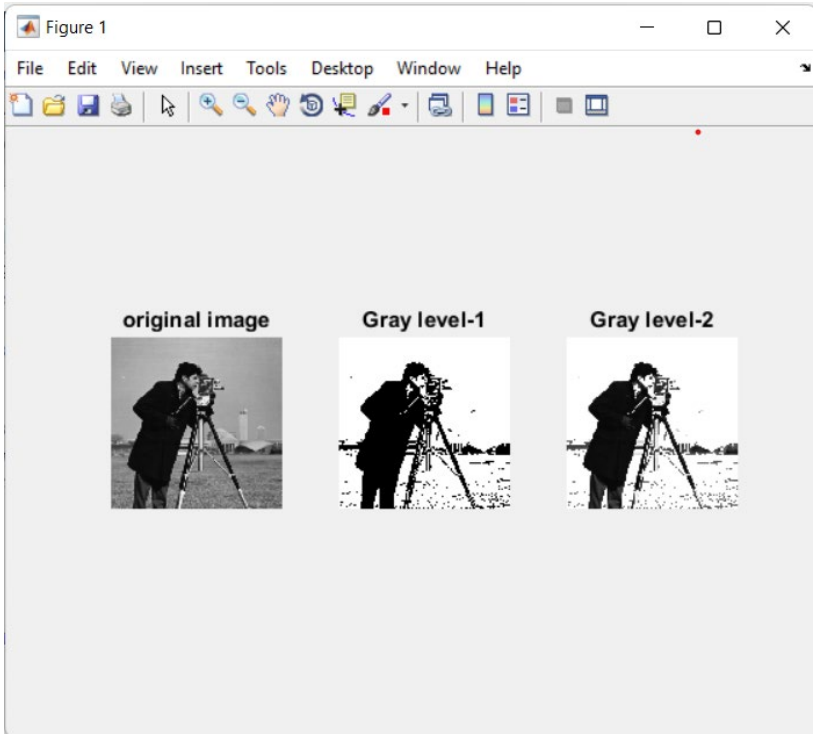
3. Gray-level slicing

Code:

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\gray_lvl.m*
thresholding.m | gray_lvl.m*

r=imread('cameraman.tif');
s=r;
subplot(1,3,1);
    imshow (r);
    title('original image');
subplot(1,3,2);
    for row = 1 : size(r,1)
        for col = 1 : size(r,2)
            if r(row,col) >100 && r(row,col) <200
                s(row,col)=255;
            else
                s(row,col) = 0;
            end
        end
    end
    imshow (s);
    title('Gray level-1');
subplot(1,3,3);
    for row = 1 : size(r,1)
        for col = 1 : size(r,2)
            if r(row,col) >100 && r(row,col) <200
                s(row,col)=255;
            else
                s(row,col) = r(row,col);
            end
        end
    end
    imshow (s);
```

```
Command Window
>> gray_lvl
92000103073 Raj Chhadia>>
```

Output:

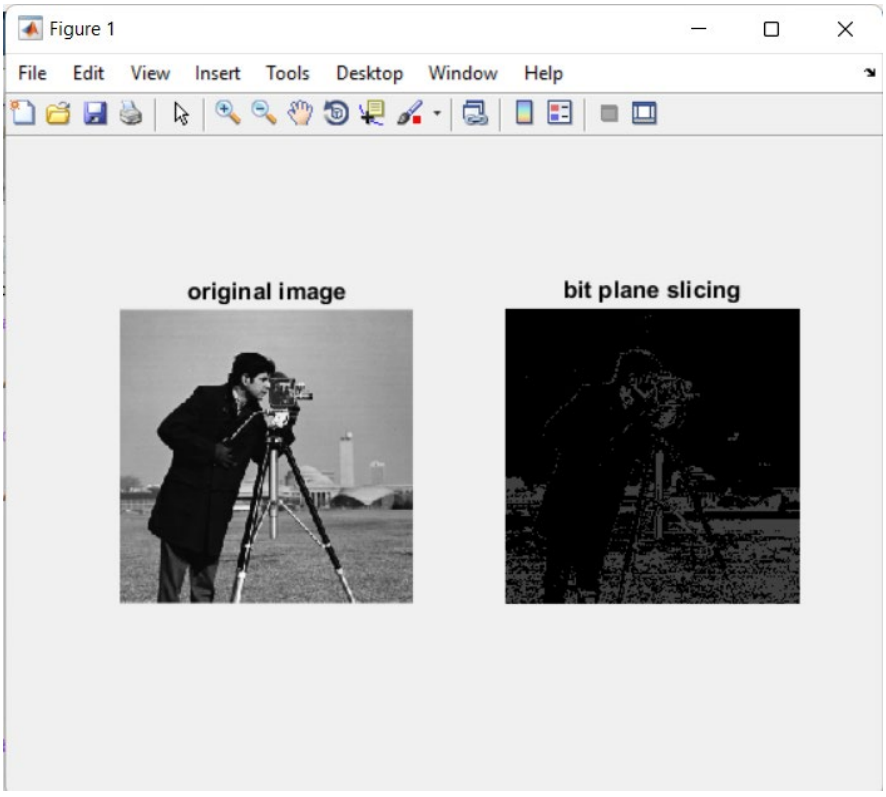


4. Bit-plane slicing

Code:

```
Editor - D:\Marwadi\SEM-5\IP-lab\Prg\bit_plane.m
thresholding.m | bit_plane.m

x=imread('cameraman.tif');

subplot(1,2,1);
    imshow (x);
    title('original image');

subplot(1,2,2);
    y =x*0;
    for row = 1:size(x,1)
        for col=1:size(x,2)
            b = bitget(x(row,col),7);
            y(row,col)=bitset(y(row,col),7,b);
        end
    end
    imshow (y);
    title('bit plane slicing');
    fprintf('92000103073 Raj Chhadia');

Command Window
>> bit_plane
92000103073 Raj Chhadia>>
```

Output:



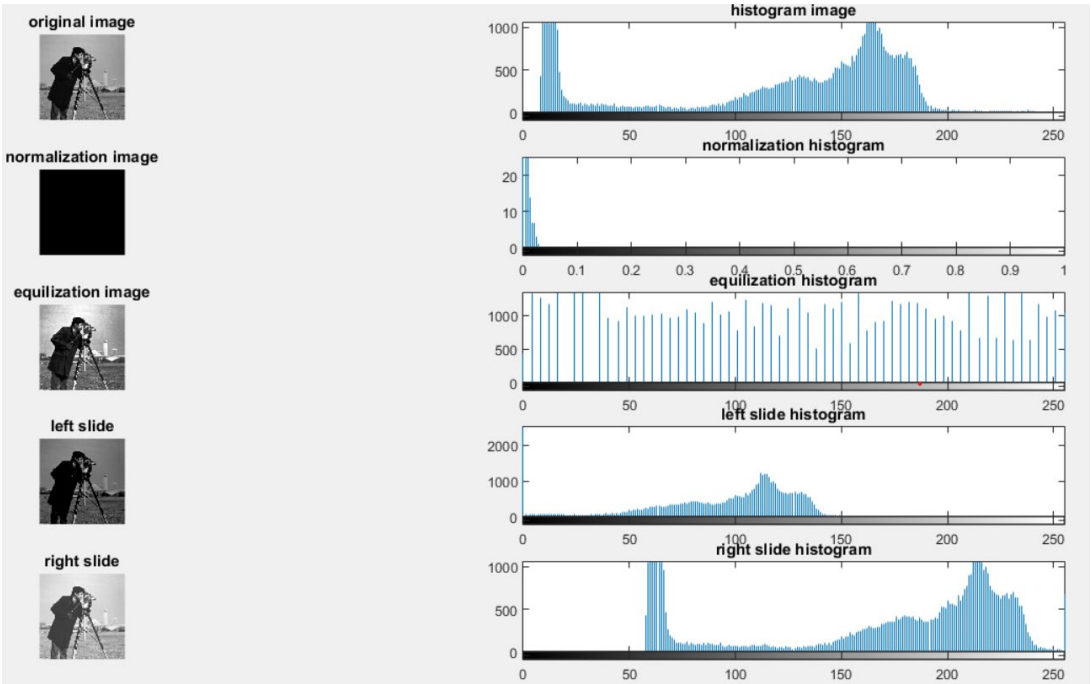
Practical 3

Aim: Write a program for histogram equalization.

Code:

```
r=imread('cameraman.tif');

subplot(5,2,1);
    imshow (r);
    title('original image');

subplot(5,2,2);
    imhist(r);
    title('histogram image');

subplot(5,2,3);
    p=r;
    total=numel(r);
    for i=1:size(r,1)
        for j=1:size(r,2)
            p(i,j)=r(i,j)/total;
        end
    end
    imshow(p);
    title('normalization image');

subplot(5,2,4);
    p=imhist(r)/numel(r);
    imhist (p);
    title('normalization histogram');

subplot(5,2,5);
    j=histeq(r);
    imshow (j);
    title('equilization image');

subplot(5,2,6);
    j=histeq(r);
    imhist (j);
    title('equilization histogram');

subplot(5,2,7);
    q=r;
    for i=1:size(r,1)
        for j=1:size(r,2)
            q(i,j)=r(i,j)-50;
        end
    end
    imshow(q);
    title('left slide');
subplot(5,2,8);
    imhist(q);
    title('left slide histogram');
subplot(5,2,9);
     s=r;
    for i=1:size(r,1)
        for j=1:size(r,2)
            s(i,j)=r(i,j)+50;
        end
    end
    imshow(s);
    title('right slide');
subplot(5,2,10);
    imhist(s);
    title('right slide histogram');
    fprintf('92000103073 Raj Chhadia');
```

Output:



Extra:

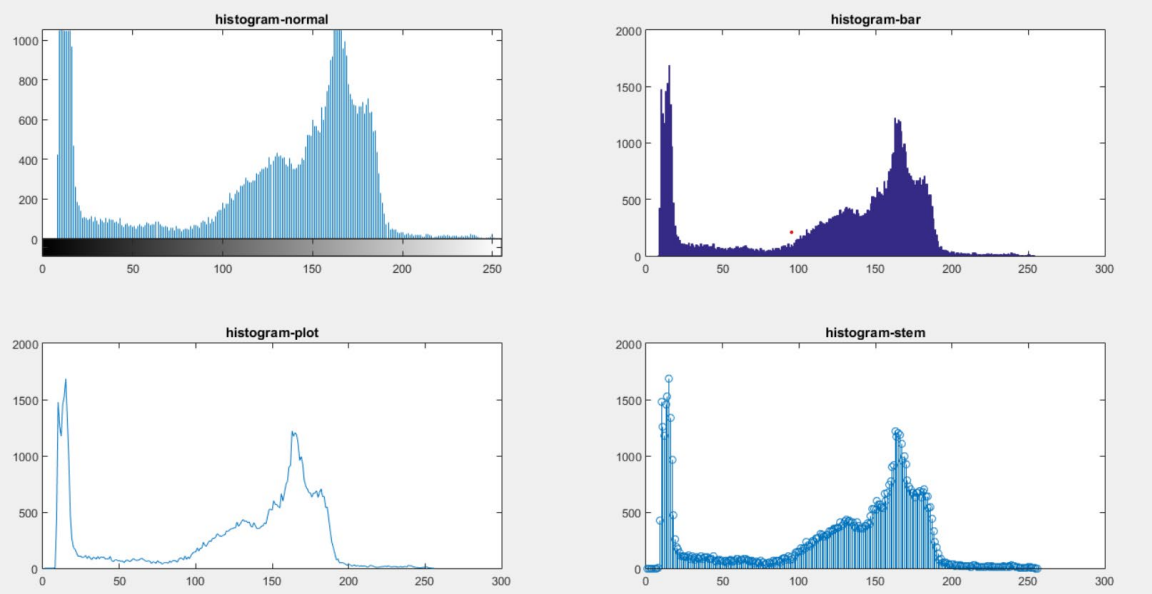
1. Histogram Types

Code:

histogram_types.m

```
1 -    r=imread('cameraman.tif');
2
3 -    subplot(2,2,1);
4 -        imhist(r);
5 -        title('histogram-normal');
6
7 -        subplot(2,2,2);
8 -        q = bar (p);
9 -        title('histogram-bar');
10
11 -       subplot(2,2,3);
12 -       s = plot(p);
13 -       title('histogram-plot');
14
15 -       subplot(2,2,4);
16 -       t = stem (p);
17 -       title('histogram-stem');
```

Output:



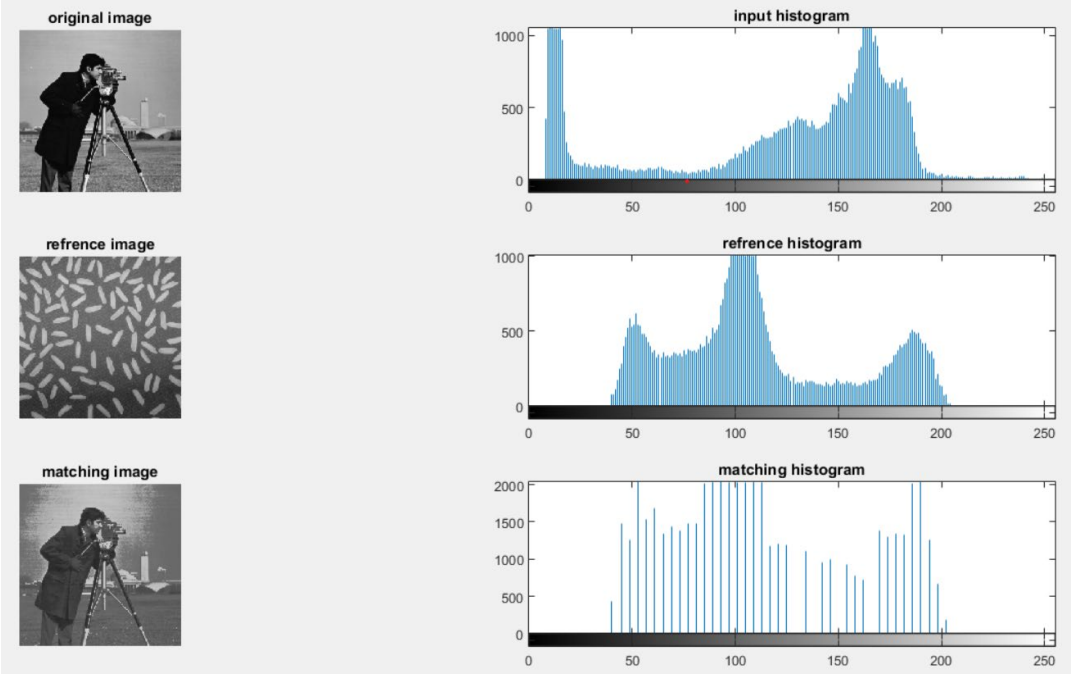
2. Histogram Matching

Code:

histogram_matching.m

```
r=imread('cameraman.tif');
s=imread('rice.png');

subplot(3,2,1);
    imshow (r);
    title('original image');

subplot(3,2,2);
    imhist(r);
    title('input histogram');

subplot(3,2,3);
    imshow (s);
    title('refrence image');

subplot(3,2,4);
    imhist(s);
    title('refrence histogram');

subplot(3,2,5);
    t = imhistmatch(r,s);
    imshow(t);
    title('matching image');

subplot(3,2,6);
    imhist(t);
    title('matching histogram');
    fprintf('92000103073 Raj Chhadia');
```

Output:



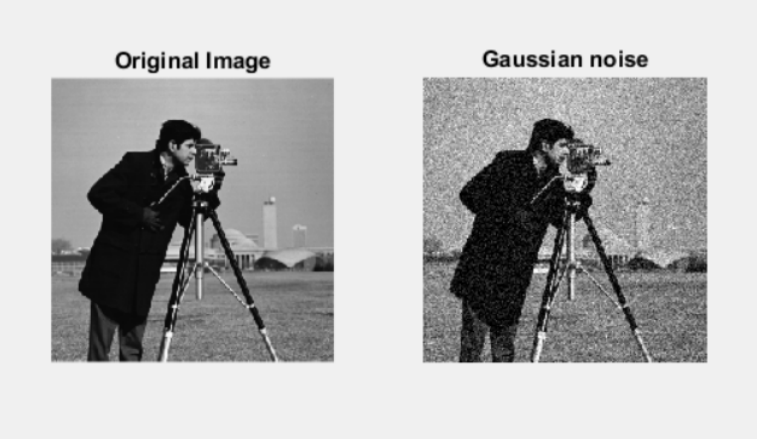
Practical 4

Aim: Write a program to apply various filtering techniques in Matlab.

- **Types of Noise (Gaussian Noise, Poisson noise, Salt & Pepper Noise, Speckle Noise)**
  - **Gaussian noise**

operator_robert.m | noise_gaussian.m

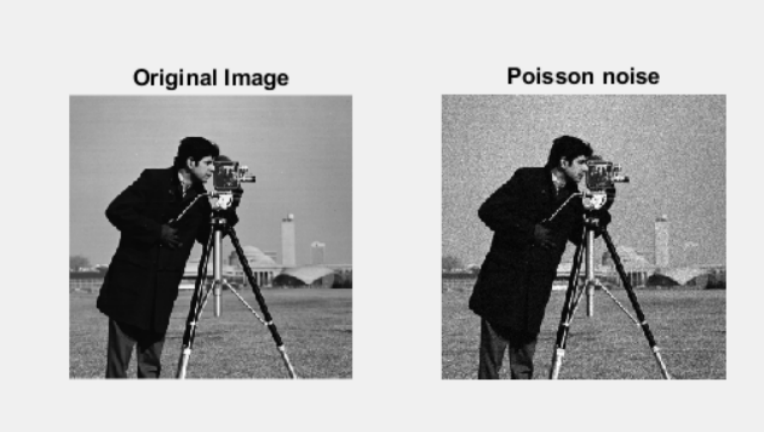
```
subplot(1,2,1);
    I = imread('cameraman.tif');
    imshow(I);
    title('Original Image');
subplot(1,2,2);
    J = imnoise(I,'gaussian');
    imshow(J);
    title('Gaussian noise');
    fprintf('92000103073 Raj Chhadia');
```



  - **Poisson noise**

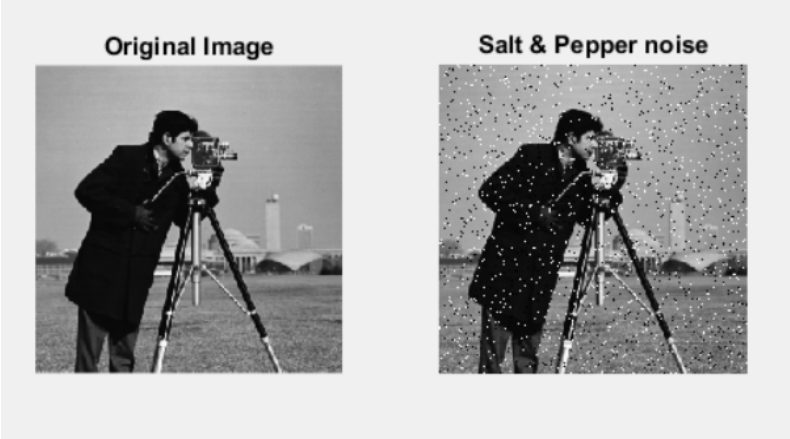
operator_robert.m | noise_poisson.m

```
subplot(1,2,1);
    I = imread('cameraman.tif');
    imshow(I);
    title('Original Image');
subplot(1,2,2);
    J = imnoise(I,'poisson');
    imshow(J);
    title('Poisson noise');
    fprintf('92000103073 Raj Chhadia');
```



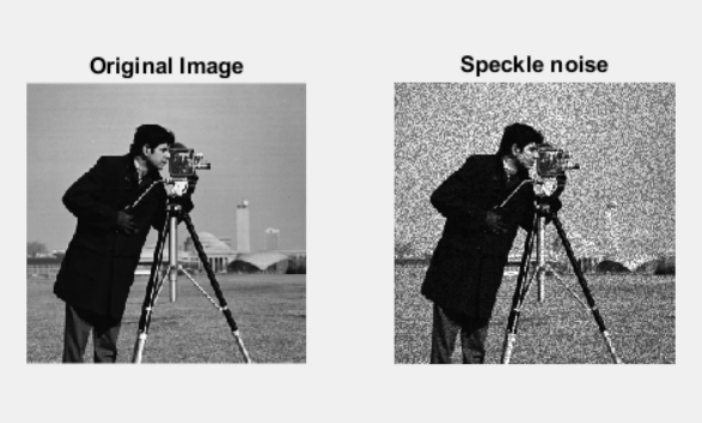
- **Salt & pepper noise**

```
subplot(1,2,1);
    I = imread('cameraman.tif');
    imshow(I);
    title('Original Image');
subplot(1,2,2);
    J = imnoise(I,'salt & pepper');
    imshow(J);
    title('Salt & Pepper noise');
    fprintf('92000103073 Raj Chhadia');
```



- **Speckle noise**

```
subplot(1,2,1);
    I = imread('cameraman.tif');
    imshow(I);
    title('Original Image');
subplot(1,2,2);
    J = imnoise(I,'speckle');
    imshow(J);
    title('Speckle noise');
    fprintf('92000103073 Raj Chhadia');
```



➢ Spatial Domain - Low Pass Filters / The Smoothing Spatial Filter

1. Linear Filters / Mean Filter

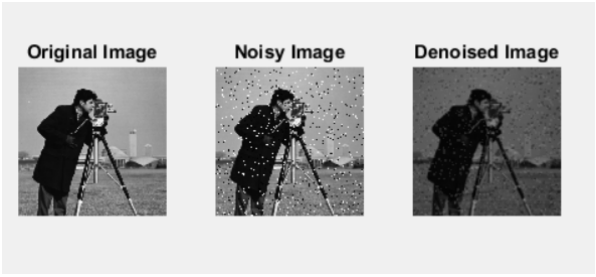
a. Averaging Filter / Standard Average Filter / Arithmetic Mean Filter

i. $N_8(P)$ Neighbor

```
%N8(P)
I =imread('cameraman.tif');
X = imnoise(I,'salt & pepper');
f=1/9*[1,1,1;1,1,1;1,1,1];

Z=filter2(f,X);
    subplot(1,3,1);
    imshow(I);
    title('Original Image');

subplot(1,3,2);
    imshow(X);
    title('Noisy Image');

subplot(1,3,3);
    imshow(uint8(Z));
    title('Denoised Image');
    fprintf('92000103073 Raj Chhadia');
```

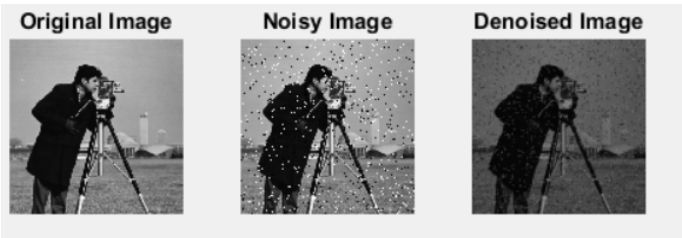
ii. $N_4(P)$ Neighbor

```
%N4(P)
I =imread('cameraman.tif');
X = imnoise(I,'salt & pepper');
f=1/9*[1,0,1;0,1,0;1,0,1];

Z=filter2(f,X);
    subplot(1,3,1);
    imshow(I);
    title('Original Image');

subplot(1,3,2);
    imshow(X);
    title('Noisy Image');

subplot(1,3,3);
    imshow(uint8(Z));
    title('Denoised Image');
    fprintf('92000103073 Raj Chhadia');
```



iii. $N_D(P)$ Neighbor

```
%Nd(P)
I =imread('cameraman.tif');
X = imnoise(I,'salt & pepper');
f=1/9*[0,1,0;1,1,1;0,1,0];

Z=filter2(f,X);
    subplot(1,3,1);
    imshow(I);
    title('Original Image');

subplot(1,3,2);
    imshow(X);
    title('Noisy Image');

subplot(1,3,3);
    imshow(uint8(Z));
    title('Denoised Image');
    fprintf('92000103073 Raj Chhadia');
```



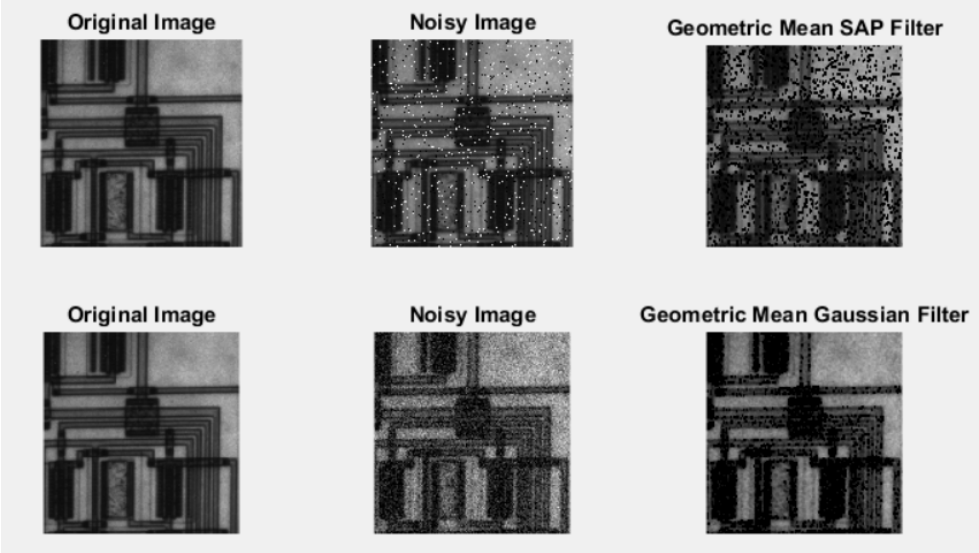
b. Weighted Averaging Filter / Gaussian Filter

```
fprintf('92000103073 Raj Chhadia');
I =imread('cameraman.tif');
X =imnoise(I,'gaussian');
f =1/16*[1,2,1;2,4,2;1,2,1];
Z =filter2(f,X);
figure;
subplot(1,3,1);
imshow(I);
title('Original image');

subplot(1,3,2);
imshow(X);
title('Noisy image');

subplot(1,3,3);
imshow(uint8(Z));
title('Denoised image');
```

c. Geometric Mean

filter_geometric.m

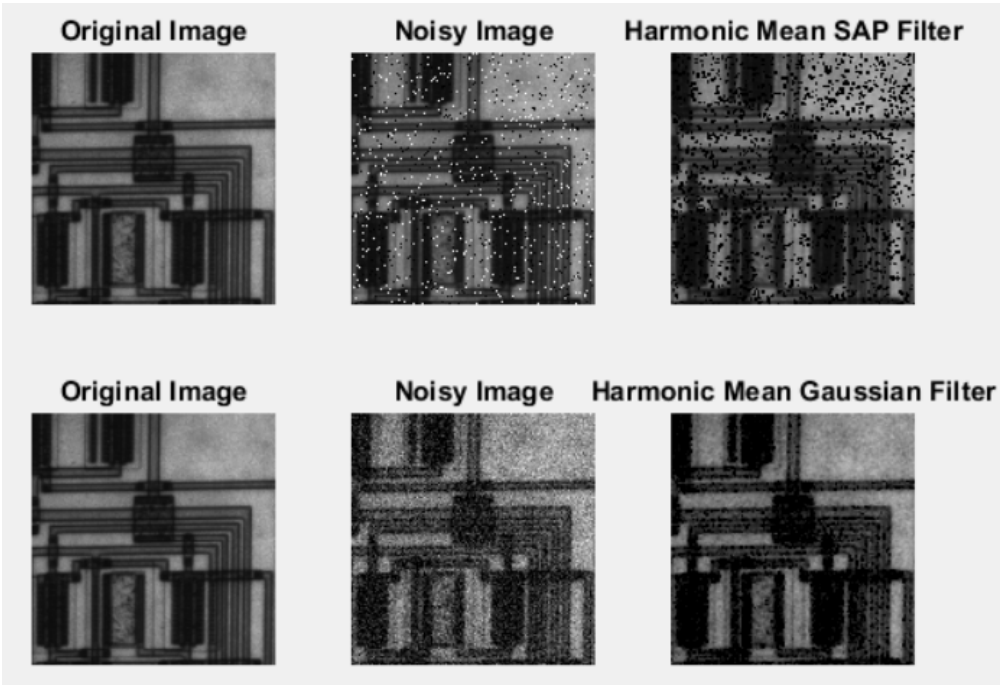
```
fprintf('92000103073 Raj Chhadia');
I =imread('circuit.tif');
NI = imnoise(I,'salt & pepper');
NI1 = im2double(NI);
NL = imnoise(I,'gaussian');
NL1 = im2double(NL);
f=[1,1,1;1,1,1;1,1,1];

subplot(2,3,1);
imshow(I);
title('Original Image');

subplot(2,3,2);
imshow(NI);
title('Noisy Image');

subplot(2,3,3);
f1= exp(imfilter(log(NI1),f,'replicate')).^(1/9);
imshow(f1);
title('Geometric Mean SAP Filter');


subplot(2,3,4);
imshow(I);
title('Original Image');

subplot(2,3,5);
imshow(NL);
title('Noisy Image');

subplot(2,3,6);
f2= exp(imfilter(log(NL1),f,'replicate')).^(1/9);
imshow(f2);
title('Geometric Mean Gaussian Filter');
```



d. Harmonic Mean

```
fprintf('92000103073 Raj Chhadia');
I =imread('circuit.tif');
NI = imnoise(I,'salt & pepper');
NI1 = im2double(NI);
NL = imnoise(I,'gaussian');
NL1 = im2double(NL);
f=[1,1,1;1,1,1;1,1,1];

subplot(2,3,1);
imshow(I);
title('Original Image');

subplot(2,3,2);
imshow(NI);
title('Noisy Image');

subplot(2,3,3);
```

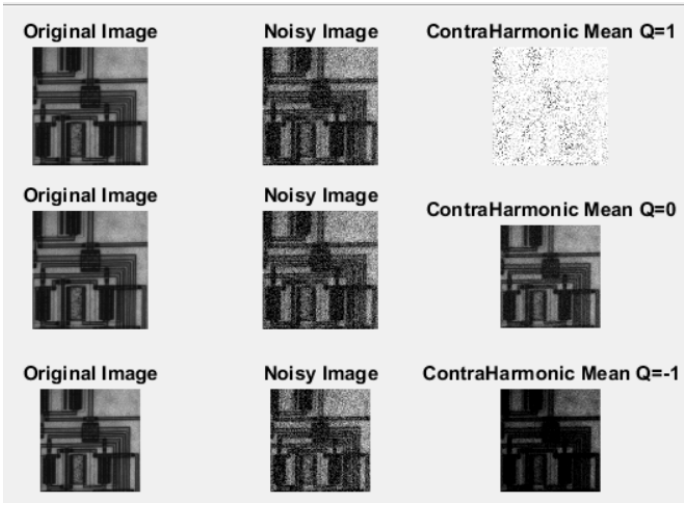
```
18 -    f1= 3*3./imfilter(1./(NI1+eps),f,'replicate');
19 -    imshow(f1);
20 -    title('Harmonic Mean SAP Filter');
21
22
23 -    subplot(2,3,4);
24 -    imshow(I);
25 -    title('Original Image');
26
27 -    subplot(2,3,5);
28 -    imshow(NL);
29 -    title('Noisy Image');
30
31 -    subplot(2,3,6);
32 -    f2=  3*3./imfilter(1./(NL1+eps),f,'replicate');
33 -    imshow(f2);
34 -    title('Harmonic Mean Gaussian Filter');
```



e. Contraharmonic Mean

```
fprintf('92000103073 Raj Chhadia');
I =imread('circuit.tif');
NI = imnoise(I,'gaussian');
NI1 = im2double(NI);
f=[1,1,1;1,1,1;1,1,1];

subplot(3,3,1);
imshow(I);
title('Original Image');

subplot(3,3,2);
imshow(NI);
title('Noisy Image');

subplot(3,3,3);
c2=imfilter(NI1.^(1+1),f,'replicate');
c3= f2./(imfilter(NI1.^1,f,'replicate')+eps);
imshow(c3);
title('ContraHarmonic Mean Q=1');

subplot(3,3,4);
imshow(I);
title('Original Image');
```

```
24
25 -    subplot(3,3,5);
26 -    imshow(NI);
27 -    title('Noisy Image');
28
29 -    subplot(3,3,6);
30 -    f2=imfilter(NI1.^(0+1),f,'replicate');
31 -    f3= f2./(imfilter(NI1.^0,f,'replicate')+eps);
32 -    imshow(f3);
33 -    title('ContraHarmonic Mean Q=0');
34
35 -    subplot(3,3,7);
36 -    imshow(I);
37 -    title('Original Image');
38
39 -    subplot(3,3,8);
40 -    imshow(NI);
41 -    title('Noisy Image');
42
43 -    subplot(3,3,9);
44 -    b2=imfilter(NI1.^(-1+1),f,'replicate');
45 -    b3= f2./(imfilter(NI1.^-1,f,'replicate')+eps);
46 -    imshow(b3);
```



2. Non-Linear Filters / Order Statistics Filter

a. Median Filtering

```
fprintf('92000103073 Raj Chhadia');
I =imread('cameraman.tif');
        NI = imnoise(I,'gaussian');
        %NI1 = im2double(NI);
      f=[1,1,1;1,1,1;1,1,1];

subplot(1,3,1);
imshow(I);
title('Original Image');

subplot(1,3,2);
imshow(NI);
title('Noisy Image');

subplot(1,3,3);
c2=ordfilt2(NI,5,f);
imshow(c2);
title('Median Filter');
```

b. Max Filtering

```
fprintf('92000103073 Raj Chhadia');
I =imread('cameraman.tif');
        NI = imnoise(I,'gaussian');
        %NI1 = im2double(NI);
      f=[1,1,1;1,1,1;1,1,1];

subplot(1,3,1);
imshow(I);
title('Original Image');

subplot(1,3,2);
imshow(NI);
title('Noisy Image');

subplot(1,3,3);
c2=ordfilt2(NI,9,f);
imshow(c2);
title('Max Filter');
```

c. Min Filtering

```
fprintf('92000103073 Raj Chhadia');
I =imread('cameraman.tif');
        NI = imnoise(I,'gaussian');
        %NI1 = im2double(NI);
      f=[1,1,1;1,1,1;1,1,1];

subplot(1,3,1);
imshow(I);
title('Original Image');

subplot(1,3,2);
imshow(NI);
title('Noisy Image');

subplot(1,3,3);
c2=ordfilt2(NI,1,f);
imshow(c2);
title('Min Filter');
```

d. Mid-point Filtering

```
fprintf('92000103073 Raj Chhadia');
I =imread('cameraman.tif');
        NI = imnoise(I,'gaussian');
        %NI1 = im2double(NI);
      f=[1,1,1;1,1,1;1,1,1];

subplot(1,3,1);
imshow(I);
title('Original Image');

subplot(1,3,2);
imshow(NI);
title('Noisy Image');

subplot(1,3,3);
c2=ordfilt2(NI,1,f);
c3=ordfilt2(NI,9,f);
G=imlincomb(0.5,c2,0.5,c3);
imshow(c2);
title('Midpoint Filter');
```

Practical 8

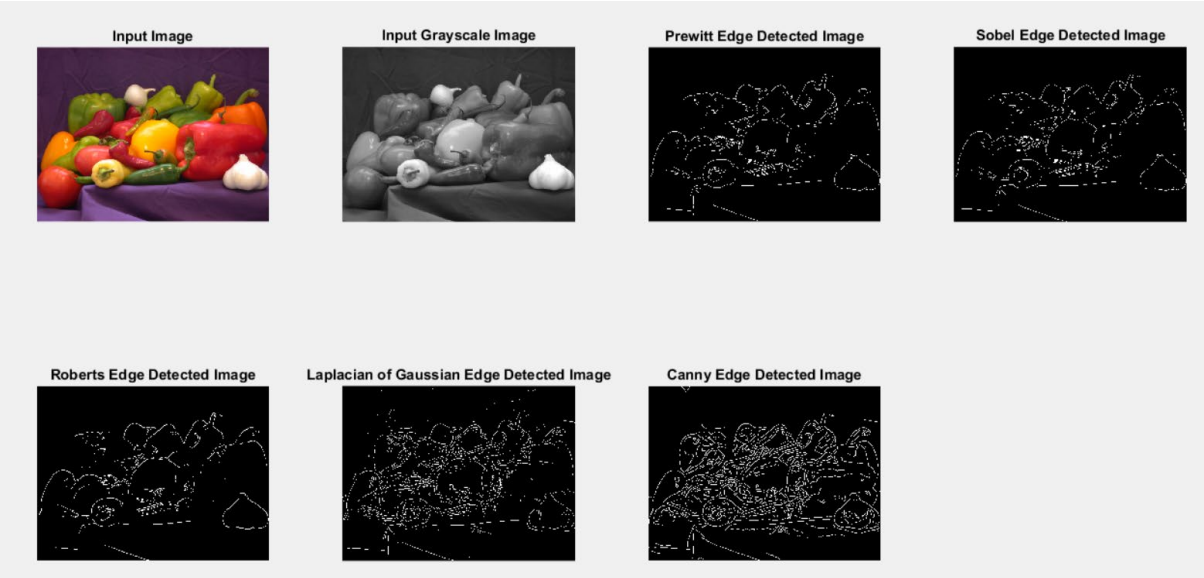
Aim: Write a program for edge detection.

(Apply Robert Operator, Prewitt Operator, Sobel Operator, Laplacian of Gaussion Operator and Canny Edge Detection methods to perform Edge Detection)

Code:

```
fprintf('92000103073 Raj Chhadia');
% Displaying Input Image
subplot(2,4,1);
    input_image = imread('peppers.png');
    imshow(input_image);
    title('Input Image');

% Convert the RGB image to the grayscale image
subplot(2,4,2);
    input_image = rgb2gray(input_image);
    imshow(input_image);
    title('Input Grayscale Image');

%Apply prewitt Operator
subplot(2,4,3);
    filtered_image1 = edge(input_image, 'prewitt');
    imshow(filtered_image1);
    title('Prewitt Edge Detected Image');

%Apply Sobel Operator
subplot(2,4,4);
    filtered_image2 = edge(input_image, 'sobel');
    imshow(filtered_image2);
    title('Sobel Edge Detected Image');

%Apply Roberts Operator
subplot(2,4,5);
    filtered_image3 = edge(input_image, 'roberts');
    imshow(filtered_image3);
    title('Roberts Edge Detected Image');

%Apply 'Laplacian of Gaussian Operator
subplot(2,4,6);
    filtered_image4 = edge(input_image, 'log');
    imshow(filtered_image4);
    title('Laplacian of Gaussian Edge Detected Image');

%Apply canny edge Detection algorithm
subplot(2,4,7);
    filtered_image4 = edge(input_image, 'canny');
    imshow(filtered_image4);
    title('Canny Edge Detected Image');
```

Output:



Practical 9

Aim: Write a program for smoothening and sharpening for 8-bit color image.

(Apply Laplacian of Gaussion Operator to Perform Edge Detection)

Code:

```
fprintf('92000103073 Raj Chhadia');
subplot(2,3,1);
    % Edge detection using Laplacian Filter.
    k=imread('peppers.png');
    imshow(input_image); title('Input Image');

subplot(2,3,2);
    % Convert rgb image to grayscale.
    k1=rgb2gray(k);
    imshow(k1); title('Input Grayscale Image');

subplot(2,3,3);
    % Display the noisy image.
    NI =imnoise(k1, 'gaussian');
    imshow(NI); title('Noisy Image');

subplot(2,3,4)
   % Create the gaussian Filter.
    GI=fspecial('gaussian', 5, 1);
    % Define the Laplacian Filter.
    Lap=[0 -1 0; -1 4 -1; 0 -1 0];
    % Convolve the noisy image
    % with Gaussian Filter first.
    Output1=conv2(NI, GI, 'same');
    % Display the Gaussian of noisy _ image.
    imshow(uint8(Output1)); title('Apply Gaussian Filter');

subplot(2,3,5)
    % Convolve the resultant
    % image with Laplacian filter.
    Output2=conv2(Output1, Lap, 'same');
    % Display the Laplacian of Gaussian resultant image.
    imshow(Output2); title('Apply Laplacian Operator');
```

Output: