Name: Harshita Mehra

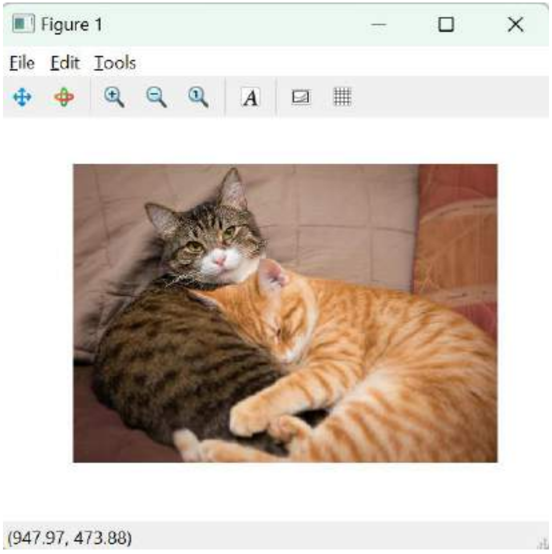
Course: BSc. Computer Science Hons

Roll No: 24921

DIP Practicals

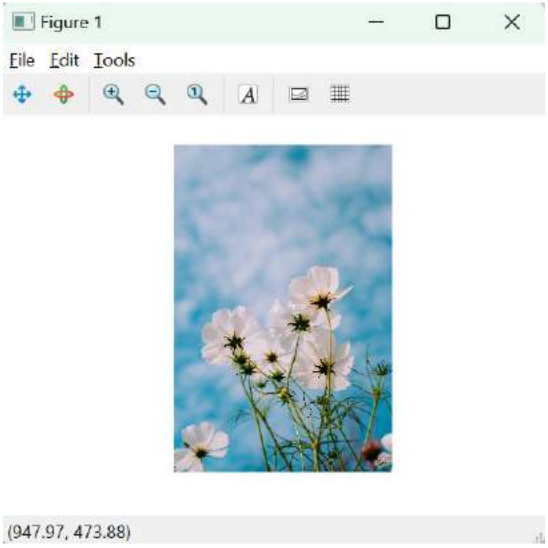
1. Write program to read and display digital image.
   a. Become familiar with basic commands

```
pkg load image;
I=imread('Cats.jpg');
Imshow(I);
```



   b. Read and display image.

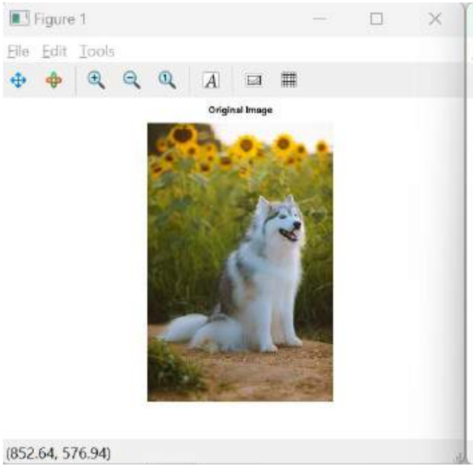
```
I=imread('Flower.jpg');
Imshow(I);
```

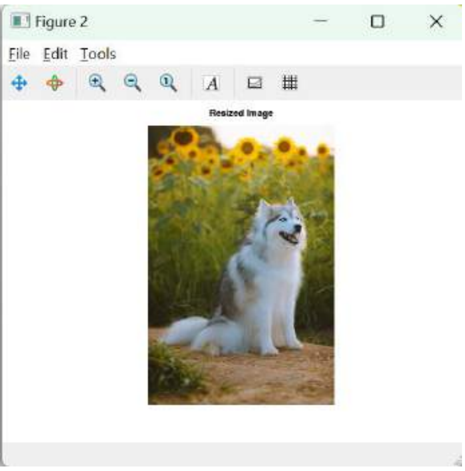


   c. Resize given image.

```
I=imread('Husky.jpg');
J=imresize(I,0.5);
```

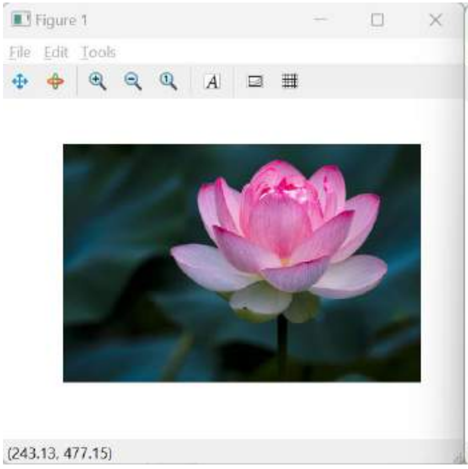
```
figure
imshow(I);
title(‘Original Image’);
figure
imshow(J);
title(‘Resized Image’);
```

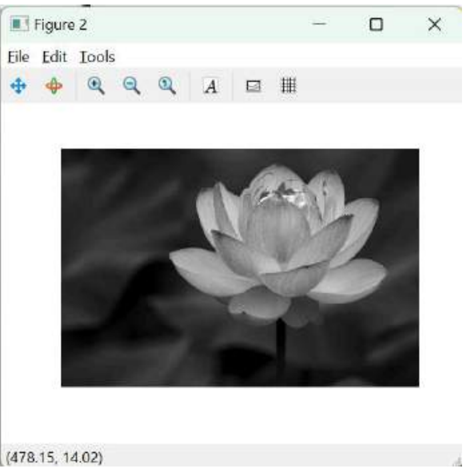




d. Convert given color image into gray-scale image.

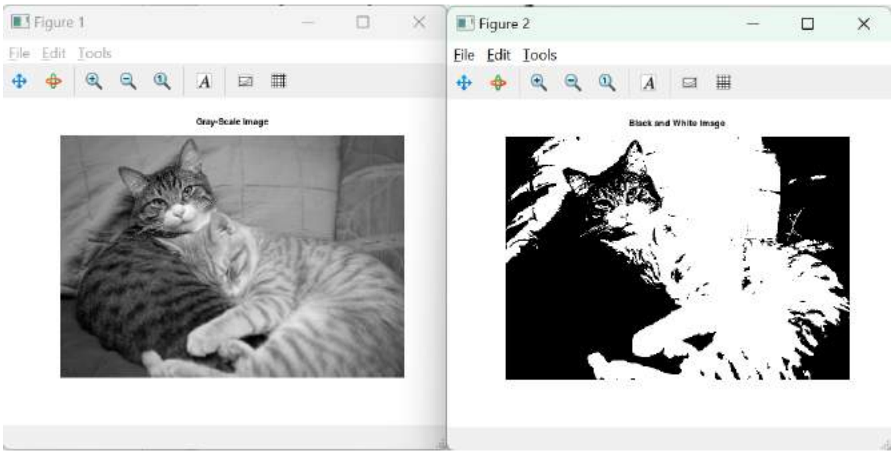
```
RGB=imread(‘Lotus.jpg’);
imshow(RGB);
I=rgb2gray(RGB);
figure
imshow(I);
```





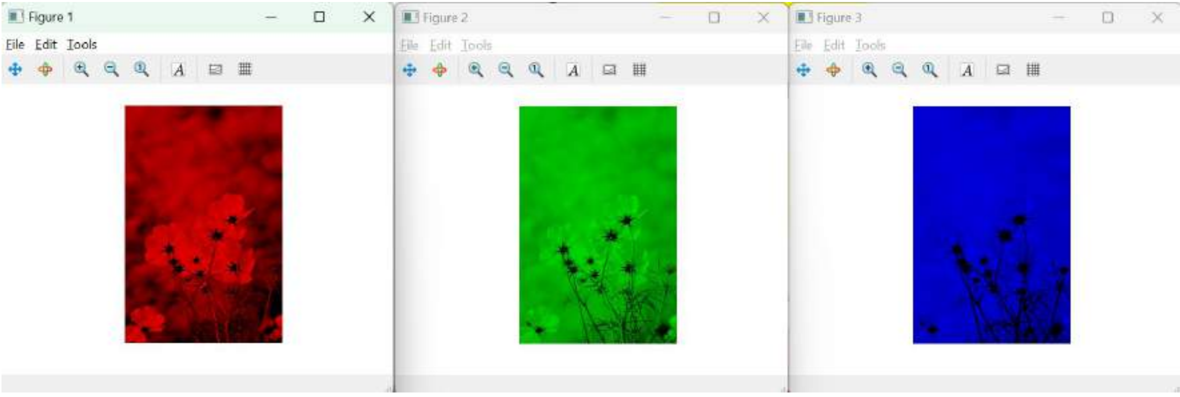
e. Convert given color image/gray-scale image into black and white image.

```
image=imread(‘Cats.jpg’);
I=rgb2gray(image);
figure
imshow(I);
title(‘Gray-Scale Image’);
J=im2bw(I);
figure
imshow(J);
title(‘Black and White Image’);
```



f. Separate color image into three R G & B planes.

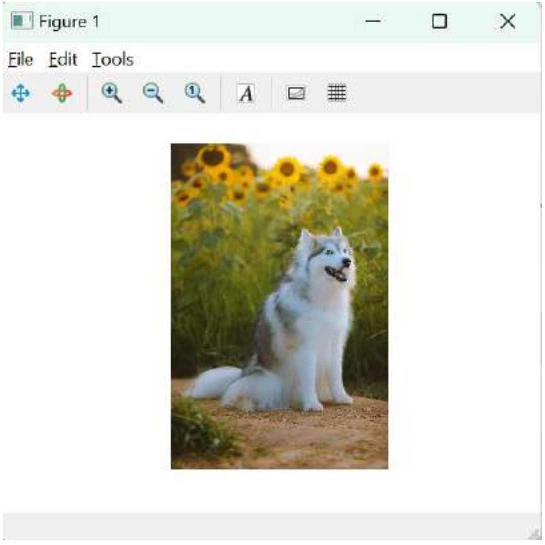
```
I=imread(‘Flower.jpg’);
%rows and columns in the image
r=size(I,1);
c=size(I,2);
%creating zero matrices
R=zeros(r,c,3);
G=zeros(r,c,3);
B=zeros(r,c,3);
%storing the corresponding color plane
%red plane
R(:,:,1)=I(:,:,1);
%green plane
G(:,:,2)=I(:,:,2);
%blue plane
B(:,:,3)=I(:,:,3);
%displaying the images
figure, imshow(uint8(R));
figure, imshow(uint8(G));
figure, imshow(uint8(B));
```



g. Create color image using R G & B three separate planes.

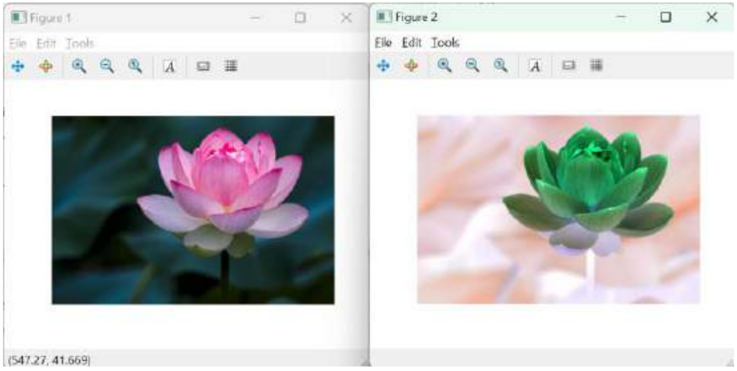
```
I=imread('Husky.jpg');
redChannel=I(:,:,1);
greenChannel=I(:,:,2);
```

```
blueChannel=I(:,:,3);
rgbImage=cat(3, redChannel, greenChannel, blueChannel);
figure, imshow(rgbImage);
```



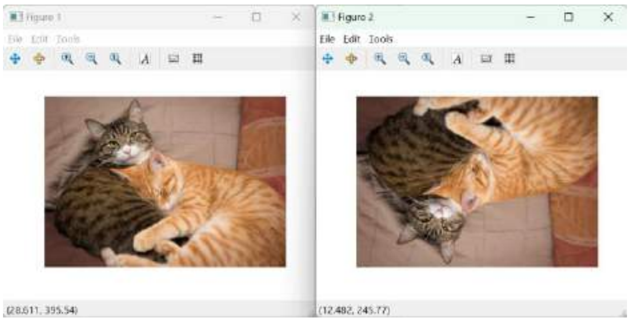
2. To write and execute image processing programs using point processing method.
   a. Obtain negative image.

```
a-imread(‘Lotus.jpg’);
imshow(a);
figure
d=255-a;
imshow(d);
```



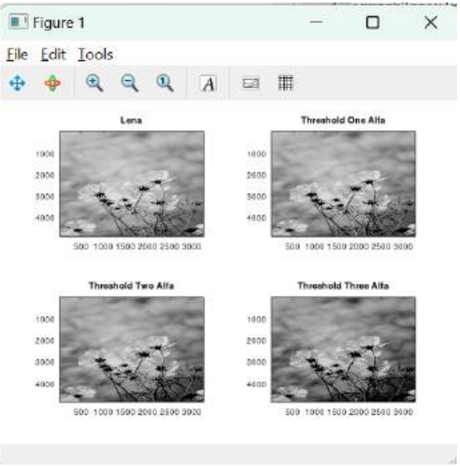
   b. Obtain flip image.

```
I=imread(‘Cats.jpg’);
imshow(I);
figure
I=flip(I);
imshow(I);
```



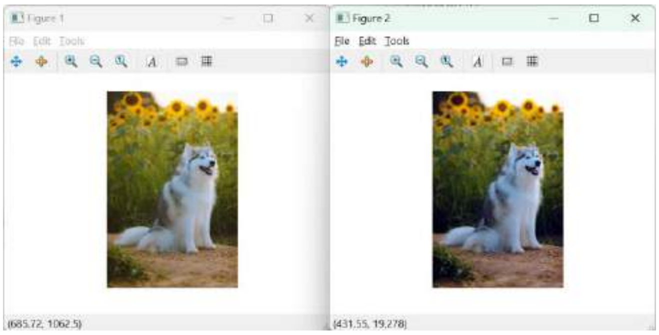
c. Thresholding

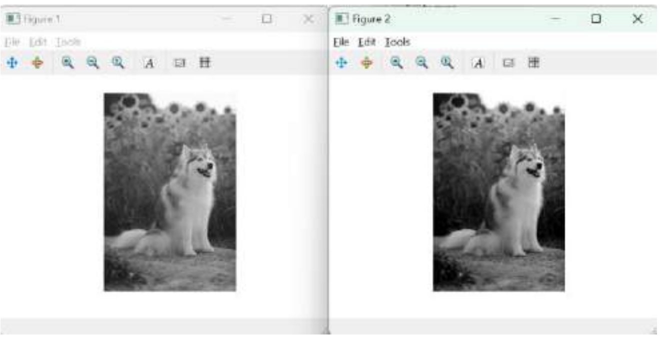
```
alfa=0.1;
x=imread('Flower.jpg');
ix=rgb2gray(x);
I_max=max(max(ix));
I_min=min(min(ix));
level1=alfa*(I_max-I_min)+I_min;
level2=2*level1;
level3=3*level1;
thix1=max(ix,level1.*ones(size(ix)));
thix2=max(ix,level2.*ones(size(ix)));
thix3=max(ix,level3.*ones(size(ix)));
figure(1);colormap(gray);
subplot(2,2,1);imagesc(ix);title("Lena");
subplot(2,2,2);imagesc(thix1);title('Threshold One Alfa');
subplot(2,2,3);imagesc(thix2);title('Threshold Two Alfa');
subplot(2,2,4);imagesc(thix3);title('Threshold Three Alfa');
```



d. Contrast stretching

```
I=imread(‘Husky.jpg’);
%I=rgb2gray(I);
figure
imshow(I);
J=imadjust(I, stretchlim(I), []);
imshow(J);
```

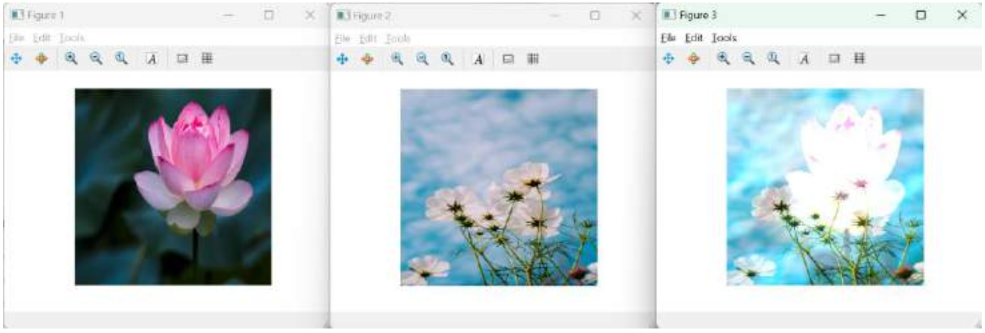




3. To write and execute programs for image arithmetic operations.

   a. Addition of two images

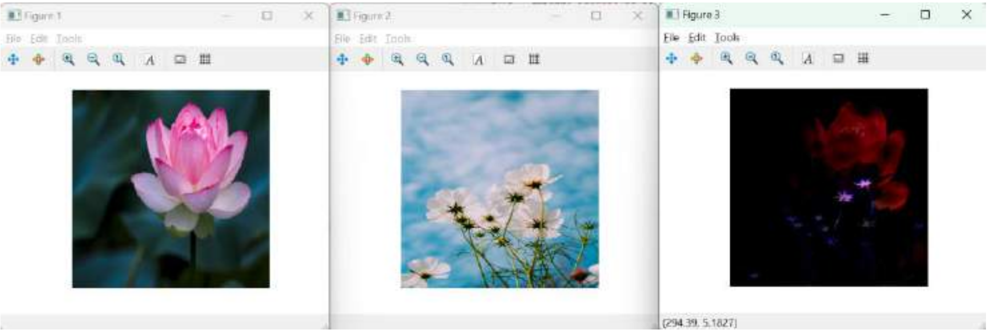
```
I = imread('Lotus.jpg');
a = imresize(I, [400,400]);
figure
imshow(a);
J = imread('Flower.jpg');
b =imresize(J, [400,400]);
figure
imshow(b);
K = imadd(a,b);
figure
imshow(K)
```



   b. Subtract one image from another image

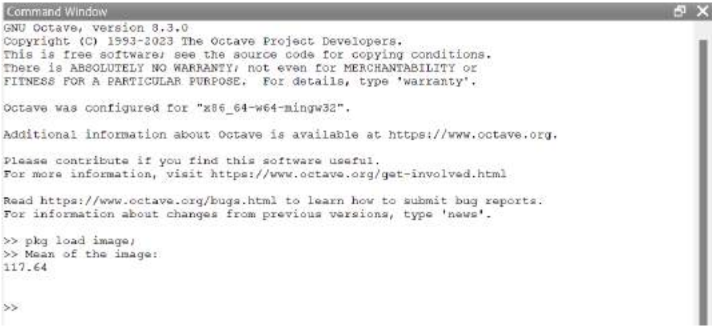
```
I = imread('Lotus.jpg');
a = imresize(I, [400,400]);
figure
imshow(a);
J = imread('Flower.jpg');
```

```
b =imresize(J, [400,400]);
figure
imshow(b);
K = imsubtract(a,b);
figure
imshow(K);
```



c. Calculate mean value of image

```
I = imread('Cats.jpg');
[m,n] = size(I);
pixel_sum = 0;
I = double(I);
for k = 1:m
 for j = 1:n
  pixel_sum+=I(k,j);
 endfor
endfor
px_mean = pixel_sum/(m*n);
disp("Mean of the image: "), disp(px_mean);
```

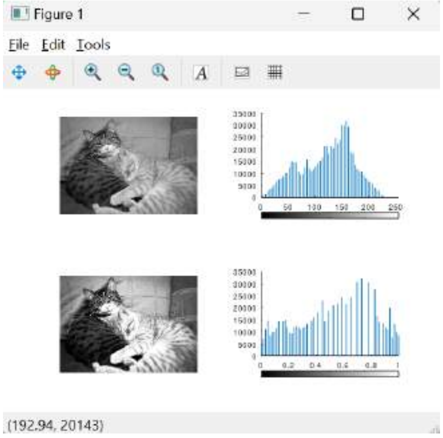


```
GNU Octave, version 8.3.0
Copyright (C) 1993-2023 The Octave Project Developers.
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE.  For details, type 'warranty'.

Octave was configured for "x86_64-w64-mingw32".

Additional information about Octave is available at https://www.octave.org.

Please contribute if you find this software useful.
For more information, visit https://www.octave.org/get-involved.html

Read https://www.octave.org/bugs.html to learn how to submit bug reports.
For information about changes from previous versions, type 'news'.

>> pkg load image;
>> Mean of the image:
117.64


>>
```

4. To write a program for histogram calculation and equalization.

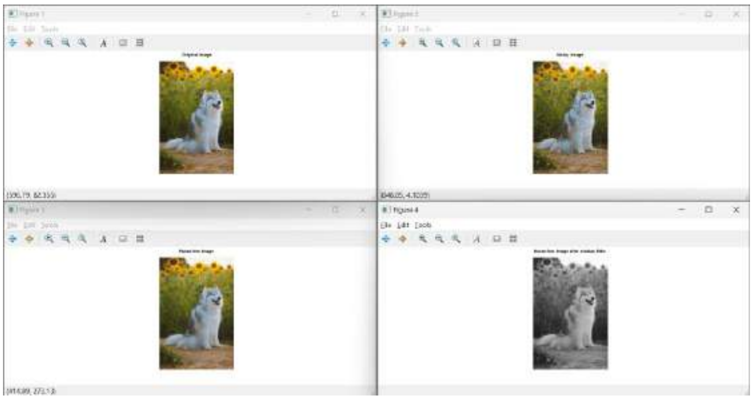
```
I=imread('Cats.jpg');
I=rgb2gray(I);
subplot(2,2,1);
imshow(I);
subplot(2,2,2);
imhist(I,64);
J=histeq(I);
subplot(2,2,3);
imshow(J);
subplot(2,2,4);
```

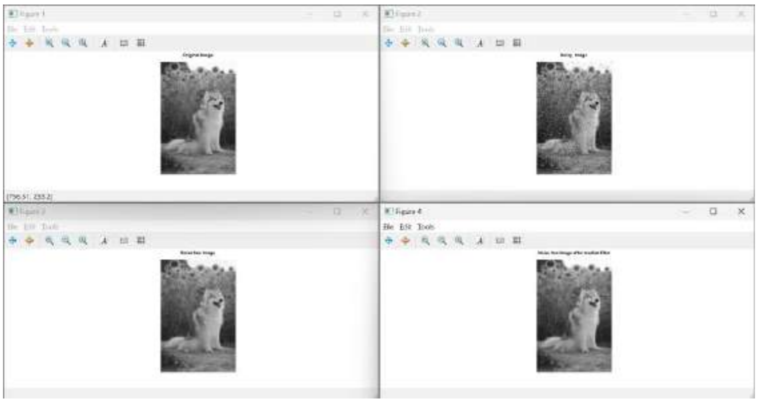
```
imhist(J,64);
```



5. To understand various image noise models and to write programs for
   a. Remove Salt and Pepper Noise

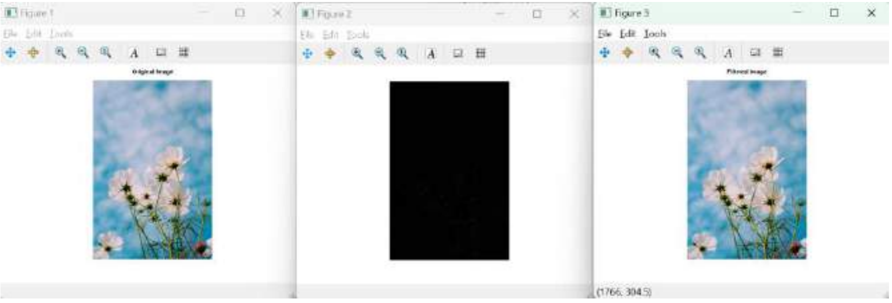
```
input_image=imread('Husky.jpg');
%input_image=rgb2gray(input_image);
figure
imshow(input_image);
title('Original Image');
J = imnoise(input_image,'Salt & Pepper',0.05);
figure
imshow(J);
title('Noisy  Image');

# remove salt and pepper noise using averaging filter
H = fspecial('average',[3,3]);
Kaverage = imfilter(J, H);
figure
imshow(Kaverage);
title('Noise free Image');

# remove salt and pepper noise using median filter
I = rgb2gray(J)
Kmedian = medfilt2(I);
%Kmedian=medfilt2(J);
figure
imshow(Kmedian);
title('Noise free Image after median filter');
```

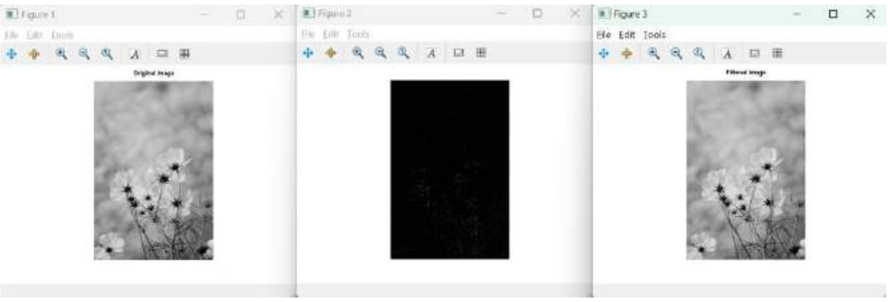


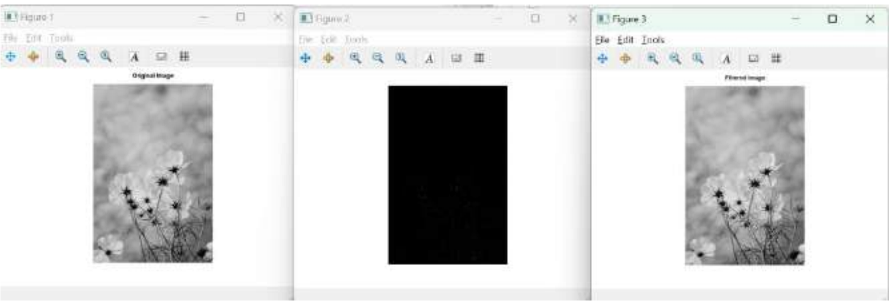


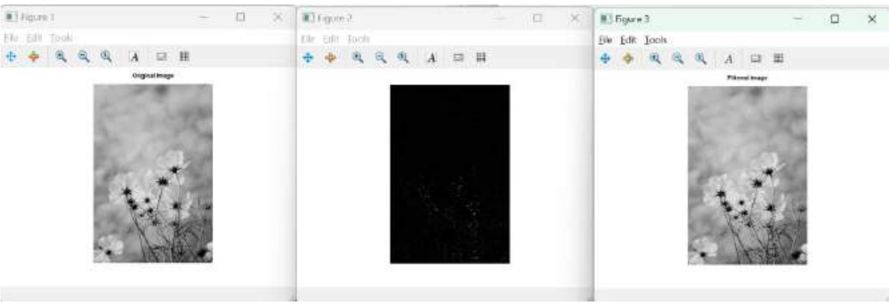
b. Laplacian Filter

```
I=imread('Flower.jpg');
I=rgb2gray(I);
figure
imshow(I);
title('Original Image');
lap= [0,1,0;1,-4,1;0,1,0];
%l= [1,1,1;1,-8,1;1,1,1];
%lap2= [0,-1,0;-1,4,-1;0,-1,0];
%l2= [-1,-1,-1;-1,8,-1;-1,-1,-1];
output= imfilter(I,lap);
figure
imshow(output);
filteredImage= imadd(I,output);
figure
imshow(filteredImage);
title('Filtered Image');
```





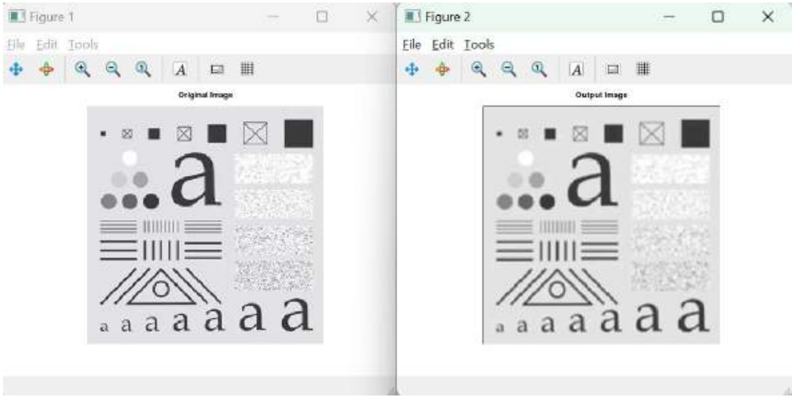




c. Mean Filter

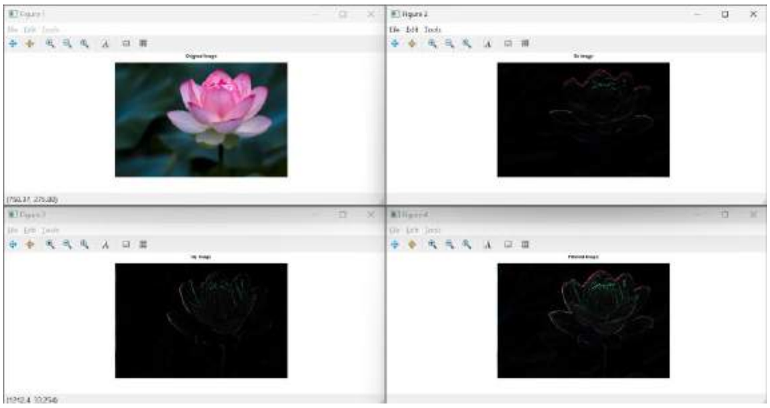
```
input_image =imread("mean_input_image.jpg");
#subplot(2,1,1);
figure
imshow(input_image);
title('Original Image');
input_image = double(input_image);
windowSize = 3; % Whatever you want.
kernel = ones(windowSize, windowSize) ;
Mx = [-1 0 1; -1 0 1; -1 0 1];
filtered_image = zeros(size(input_image));
for i = 1:size(input_image, 1) - 2
      for j = 1:size(input_image, 2) - 2
   filtered_image(i+1, j+1) = sum(sum(kernel .* input_image(i:i+2, j:j+2)));
   output_image(i+1, j+1) = filtered_image(i+1, j+1)/9;
  endfor
endfor
#filtered_image = (filtered_image / 9);
output_image = uint8(output_image);
#subplot(2,1,2);
figure
imshow(output_image);
```

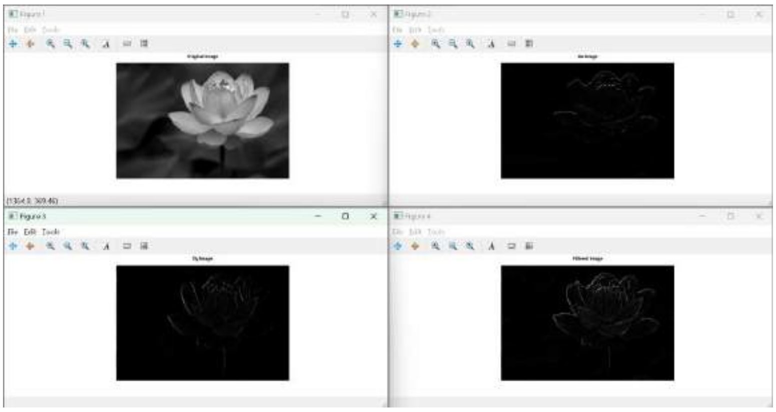
```
title('Output Image');
```



d. Prewitt Filter

```
I=imread('Lotus.jpg');
%I=rgb2gray(I);
%subplot(2,2,1);
figure
imshow(I);
title('Original Image');
%lap= [0,1,0;1,-4,1;0,1,0];
Gx= [-1,-1,-1;0,0,0;1,1,1];
outputGx= abs(imfilter(I,Gx));
figure
%subplot(2,2,2);
imshow(outputGx);
title('Gx Image');

Gy= [-1,0,1;-1,0,1;-1,0,1];
outputGy= abs( imfilter(I,Gy));
figure
%subplot(2,2,3);
imshow(outputGy);
title('Gy Image');
filteredImage= imadd(outputGx,outputGy);
figure
%subplot(2,2,4);
imshow(filteredImage);
title('Filtered Image');
```





6. Write and execute program for image morphological operations erosion and dilation.

```
% Read Input Image
input_image = imread("erosion.jpg");

% Displaying Input Image
input_image = uint8(input_image);
figure, imshow(input_image); title('Input Image');

% Convert the truecolor RGB image to bw image
input_image = im2bw(input_image);

% Convert the image to double
input_image = double(input_image);
figure;
imshow(input_image);

%se = [0,1,0;1,1,1;0,1,0];
se = strel("square", 3);
erodedI = imerode(input_image,se);
figure;
imshow(erodedI); title('Eroded Image');

dilateI = imdilate(erodedI, se);
figure;
imshow(dilateI); title('Dilated Image');
```

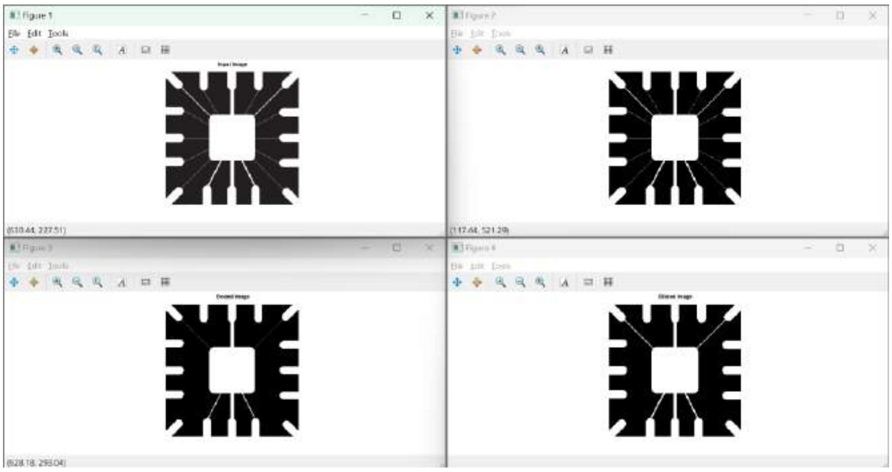


Figure 1
File Edit Tools
Input Image
(610.44, 227.51)
Figure 2
File Edit Tools
(117.64, 521.29)
Figure 3
File Edit Tools
Eroded Image
(628.18, 295.04)
Figure 4
File Edit Tools
Dilated Image