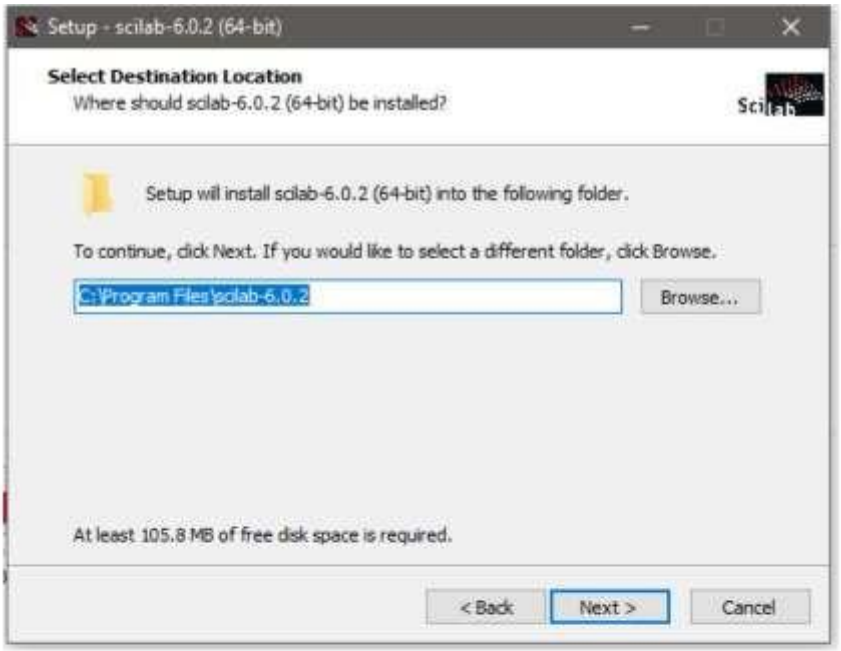
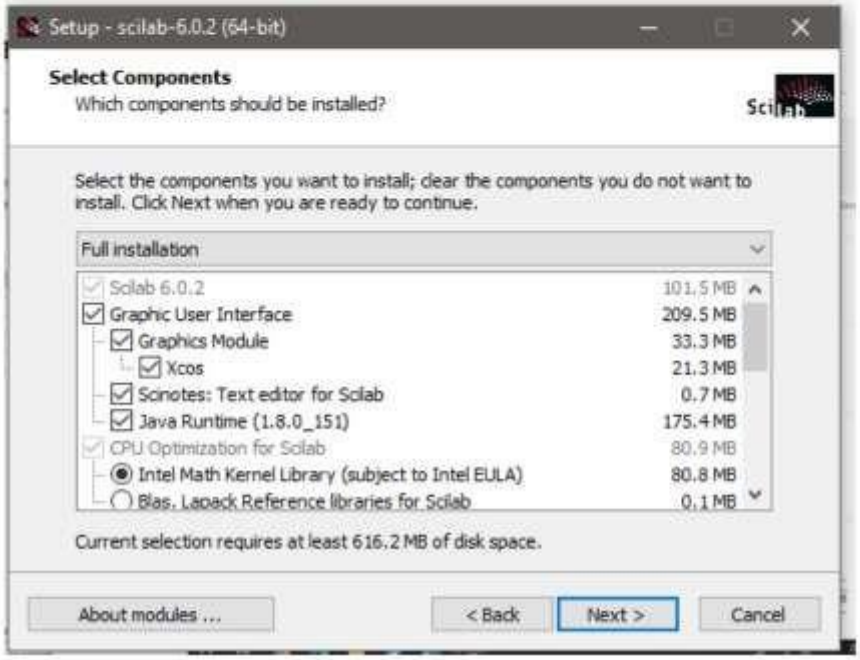
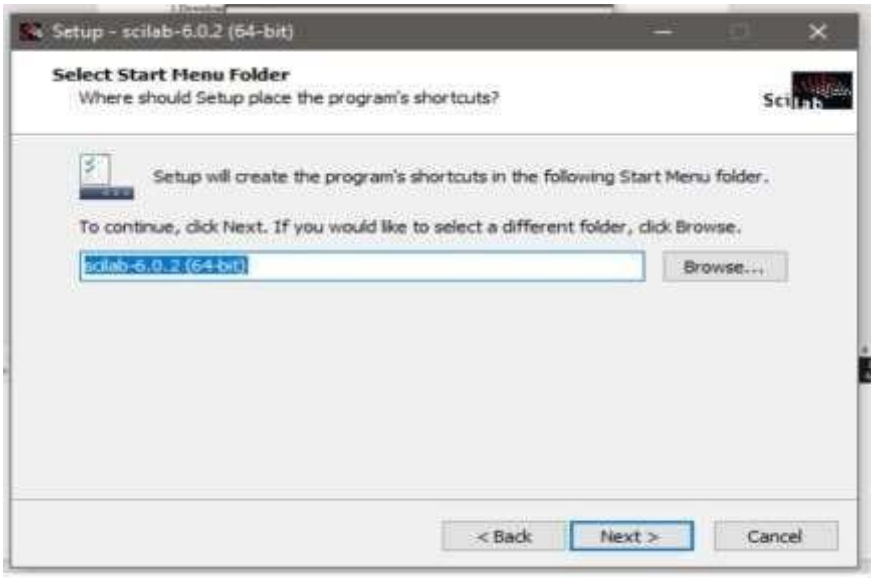
Practical No 1

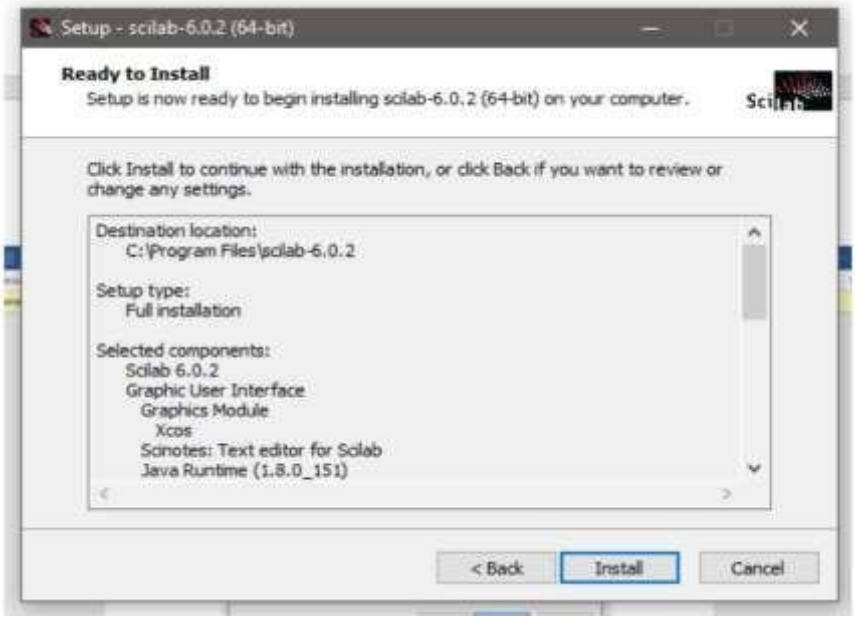
Required Software for Image Processing- Steps:

1. Download and Install Scilab 6.0.2



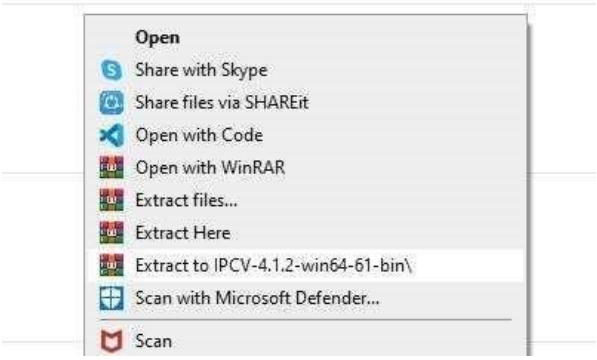




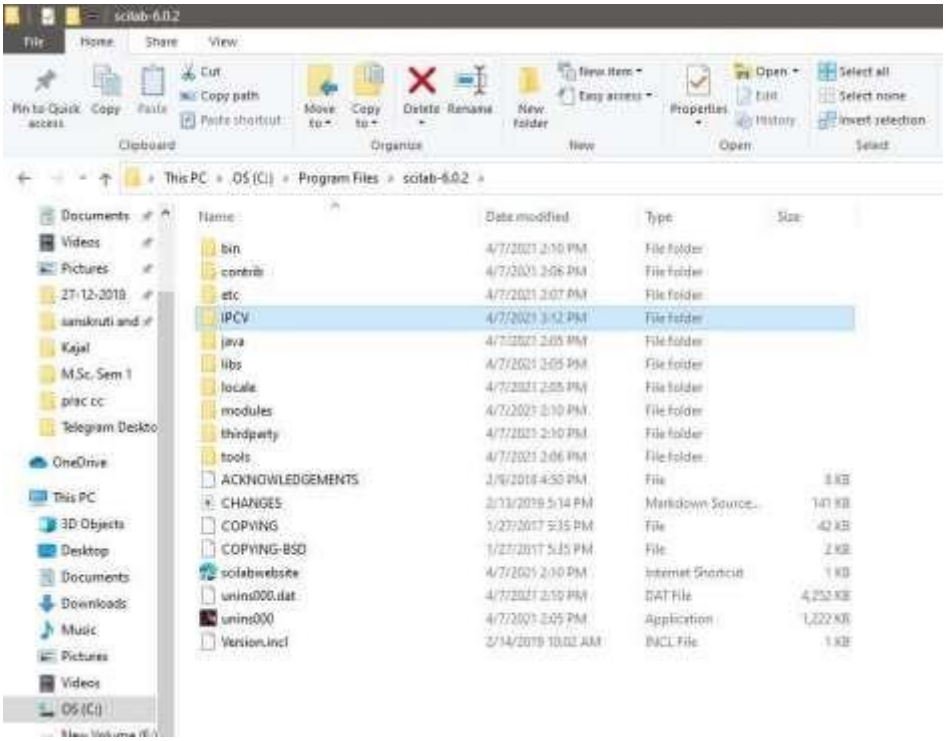


2.Download IPCV zip file

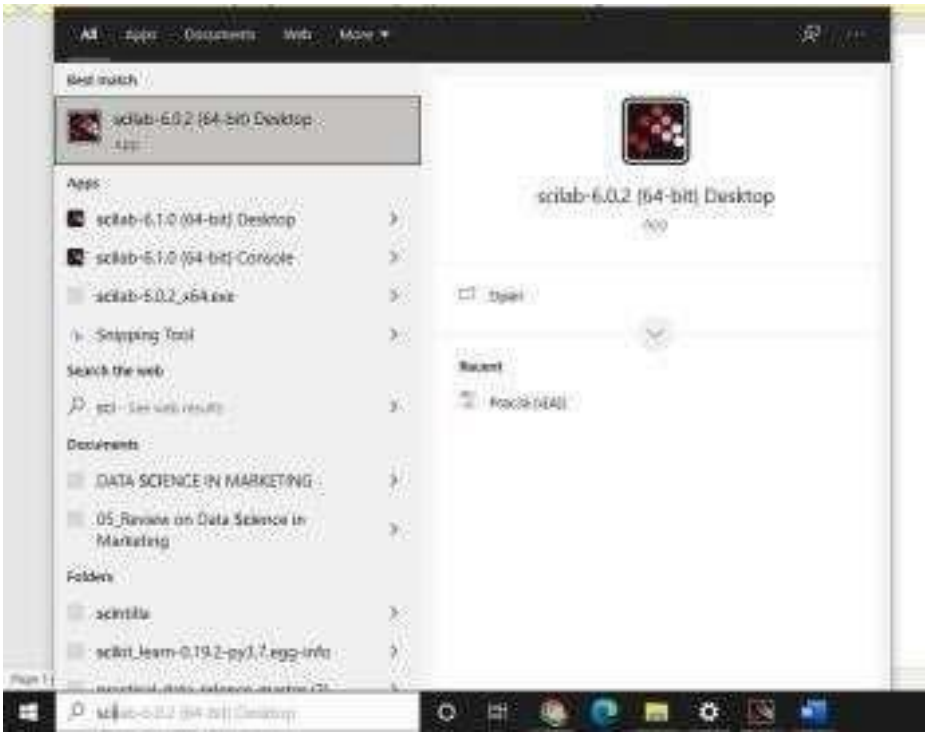
Extract IPVC Folder => Paste in Scilab Folder



3. Paste IPCV folder in C:\Program Files\scilab-6.0.2

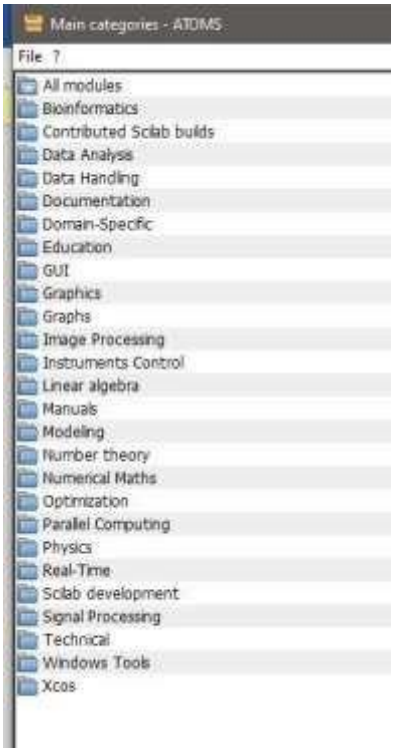


4. Open Scilab 6.0.2. Go to start → Search Scilab → Open



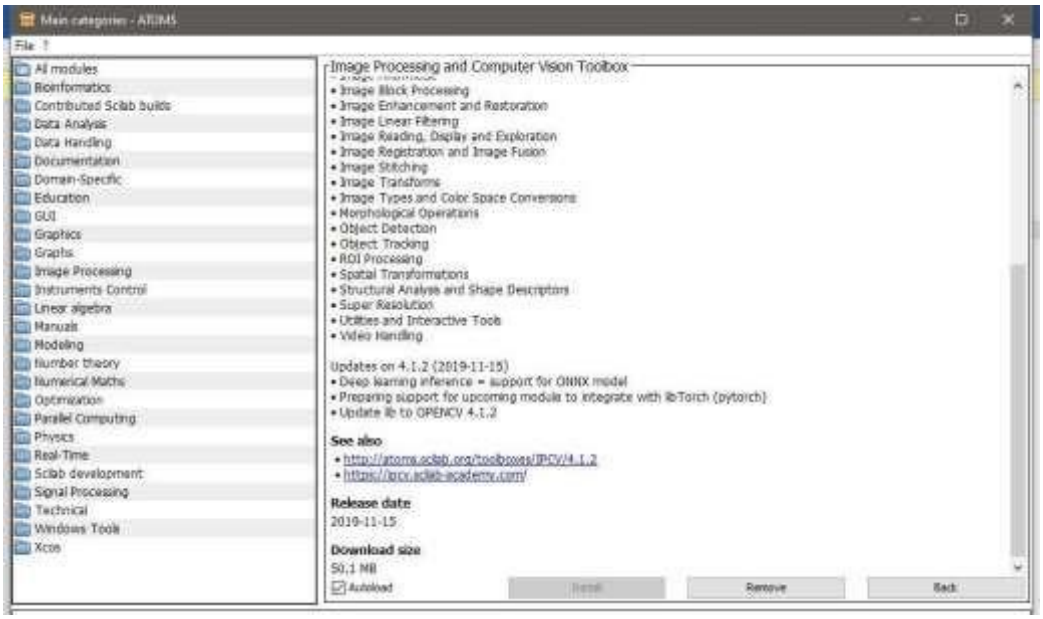
5. Click on Modul Manager.

6. Click on Image Processing.



7. Install Scilab computer vision modules

8. Install image processing and computer vision toolbox.



9. Open scilab 6.0.2 console

atomsInstall("C: Users\Downloads\IPCV-4.1.2-win64-61-bin")

Part A:

Part A: Aim- Program to calculate number of samples required for an image.

Code :
```
--> m=4;
--> n=6;
--> N=400;
--> N2=2+n;
--> Fs =m*N2*n*N2;
--> disp(Fs, 'Number of sample required to preserve the information in the image=')
```

Output: Number of sample required to preserve the information in the image= 1536.

Part B:

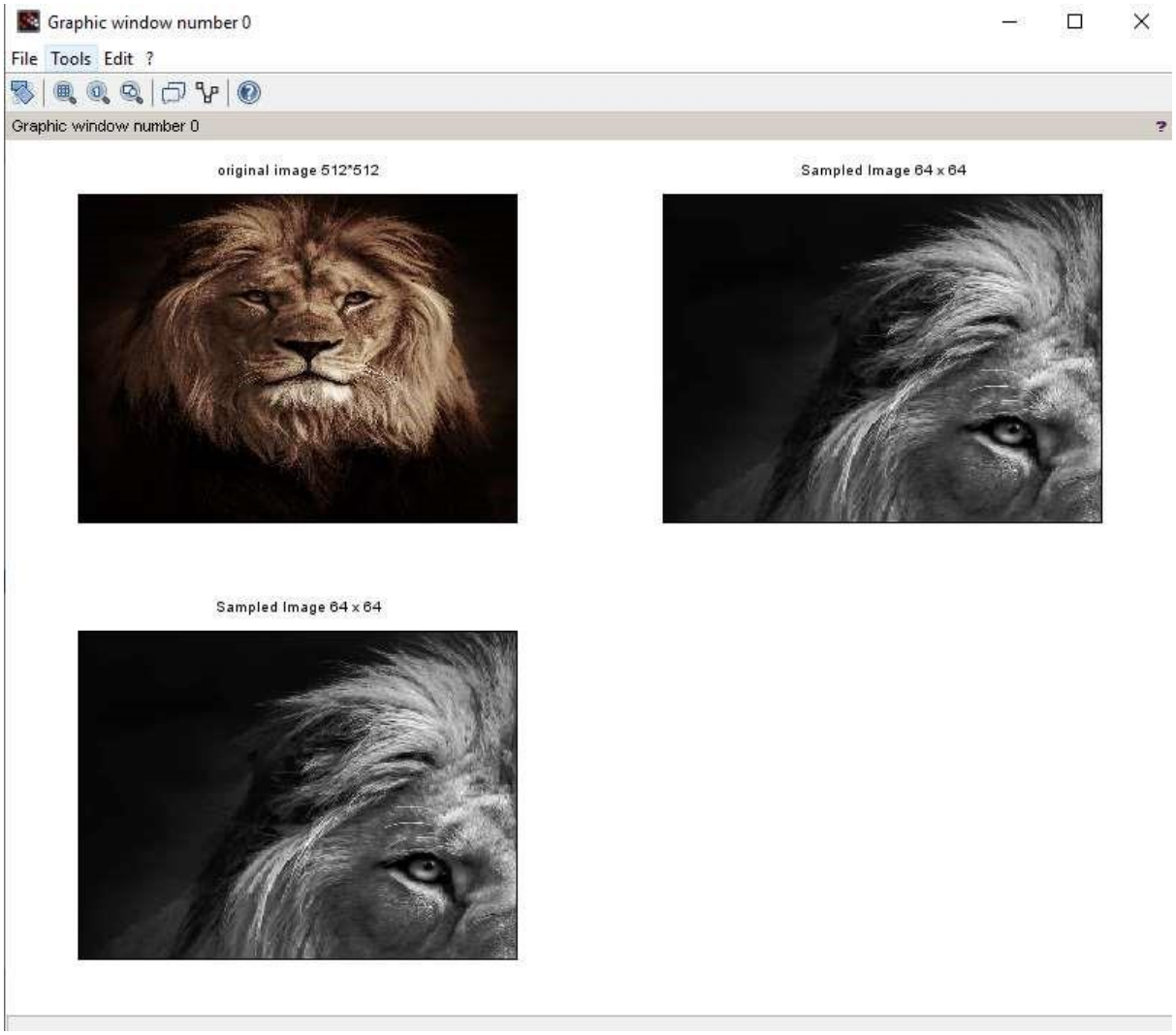
Aim- Program to study the effects of reducing the spatial resolution of a digital image.

Code-

```
clc; clear
all;
Img=imread('C:\Users\5itlab\Documents\lion.jpg');
subplot(2,2,1), imshow(Img),
title('original image 512*512');
```

```
samp= zeros(256);
for i=1:1:512 for
j=1:1:512 if
modulo(i,2)==0
m=i/2; if
modulo(j,2)==0
n=j/2;
samp(i-m,j-n)=Img(i,j);
else
n=0;
end end
m=0
end end
sampImg128=mat2gray(samp); subplot(2,2,2),
imshow(sampImg128),
title('Sampled Image 64 x 64');
samp= zeros(32); for
i=1:1:512 for j=1:1:512
if modulo(i,16)==0
m=i/16*4; if
modulo(j,16)==0
n=j/16*4 samp(i-m,j-
n)= Img(i,j);
else
n=0;
end end
m=0
end end
samImg64=mat2gray(samp); subplot(2,2,3),
imshow(sampImg128),
title('Sampled Image 64 x 64');
samp = zeros(32); for
i=1:1:512
for j=1:1:512 if
modulo(i,16)==0
m=i/16*4; if
modulo(j,16)==0
n=j/16*4
samp(i-m,j-n)= Img(i,j);
else
n=0;
end end
m=0
end end
```

Output-



Part C:

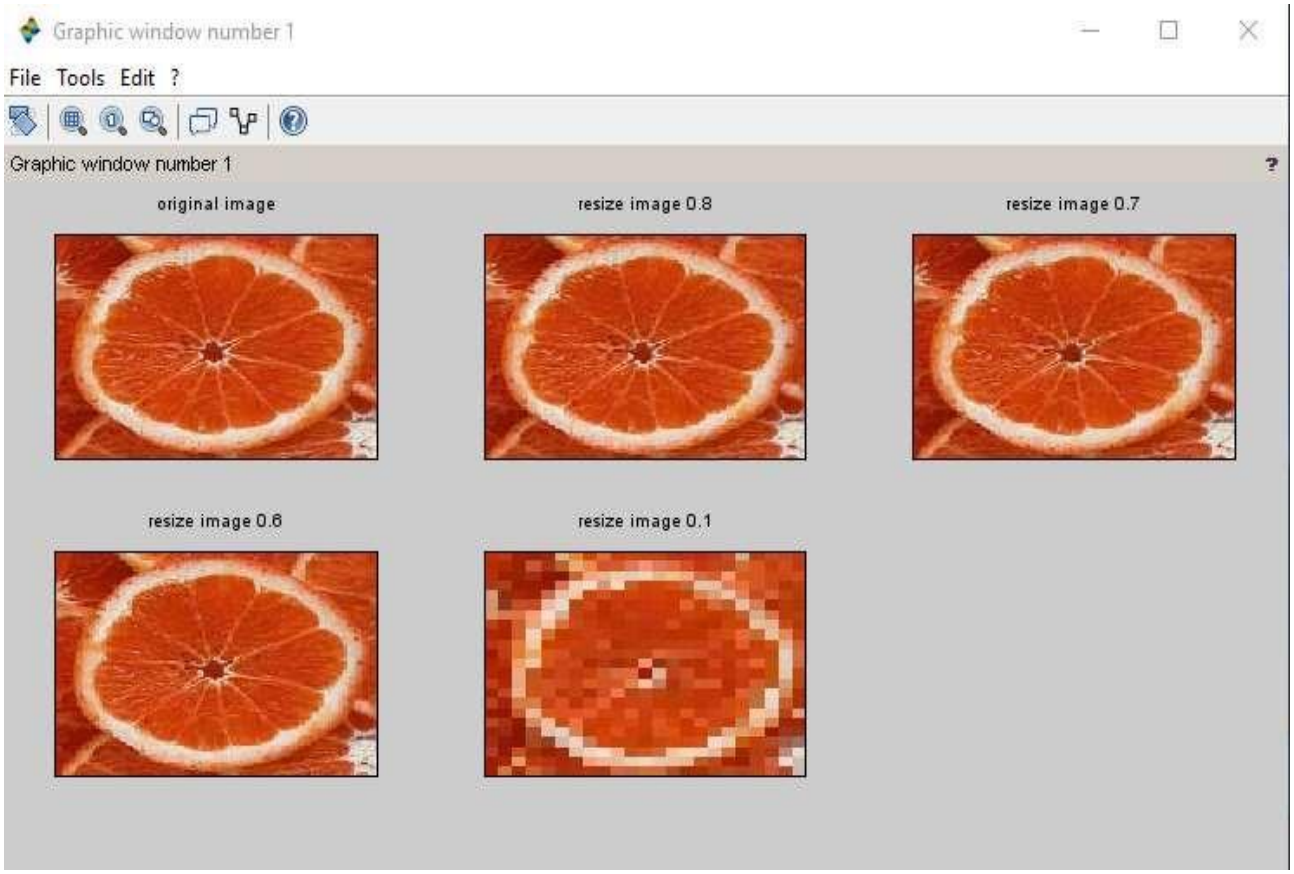
Aim- Program to study the effects of varying the number of intensity levels in a digital image.

Code-

```
clc;  clear all;
figure(1)
subplot(3,3,1);
i=imread('C:\Users\5itlab\Documents\orange.jpg');
imshow(i);
 title('original image');
subplot(3,3,2);
j=imresize(i,0.8);
imshow(j);
title('resize image 0.8');
subplot(3,3,3);
j=imresize(i,0.7);
imshow(j);
```

```
title('resize image 0.7');
subplot(3,3,4);
j=imresize(i,0.6);
imshow(j);
title('resize image 0.6');
subplot(3,3,5);
j=imresize(i,0.1);
 imshow(j);
title('resize image 0.1');
```

Output-



Practical No 2

Aim- Image Enhancement.

Part A:

Aim- Basic Intensity Transformation functions.

1. Program to perform Image negation.
2. Program to perform threshold on an image.
3. Program to perform Log transformation.
4. Power-law transformations.
5. Piecewise linear transformations.
    a) Contrast Stretching.
    b) Gray-level slicing with and without background.
    c) Bit-plane slicing.

Part B:

1. Program to plot the histogram of an image and categories.
2. Program to apply histogram equalization.

Part C:

Aim- Write a program to perform convolution and correlation.

Part D:

Aim- Write a program to apply smoothing and sharpening filters on grayscale and color images.

1. Low Pass.
2. High Pass.

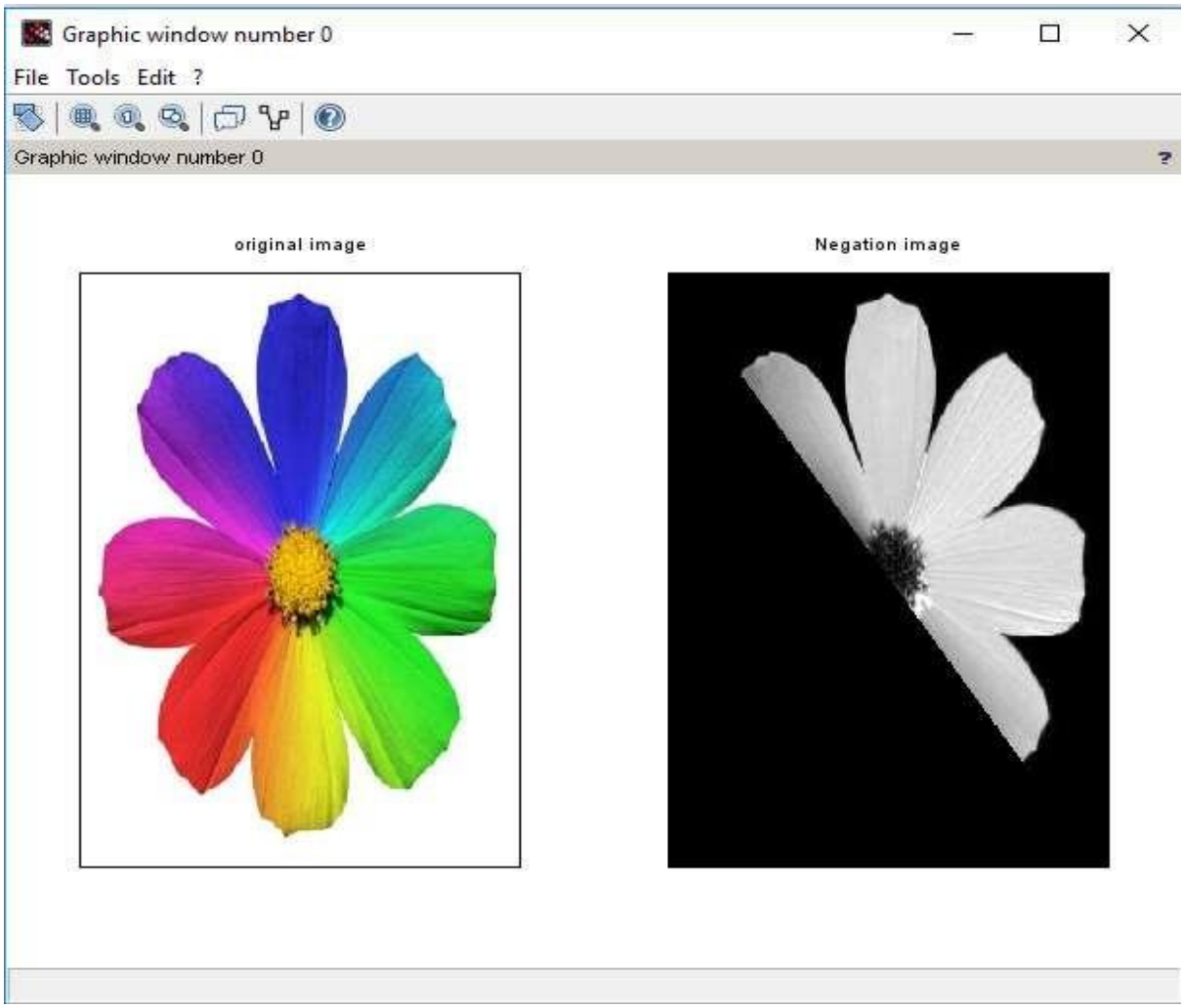
Part A:

1. Image Negation.

Code-

```
clc; clear
all;
a=imread('C:\Users\5itlab\Documents\flower.jpg');
subplot(1,2,1); imshow(a);
title('original image')
[m,n]=size(a); for
i=1:m
   for j=i:n
c(i,j)=255-a(i,j) end
end subplot(1,2,2);
imshow(c);
title('Negation image');
```

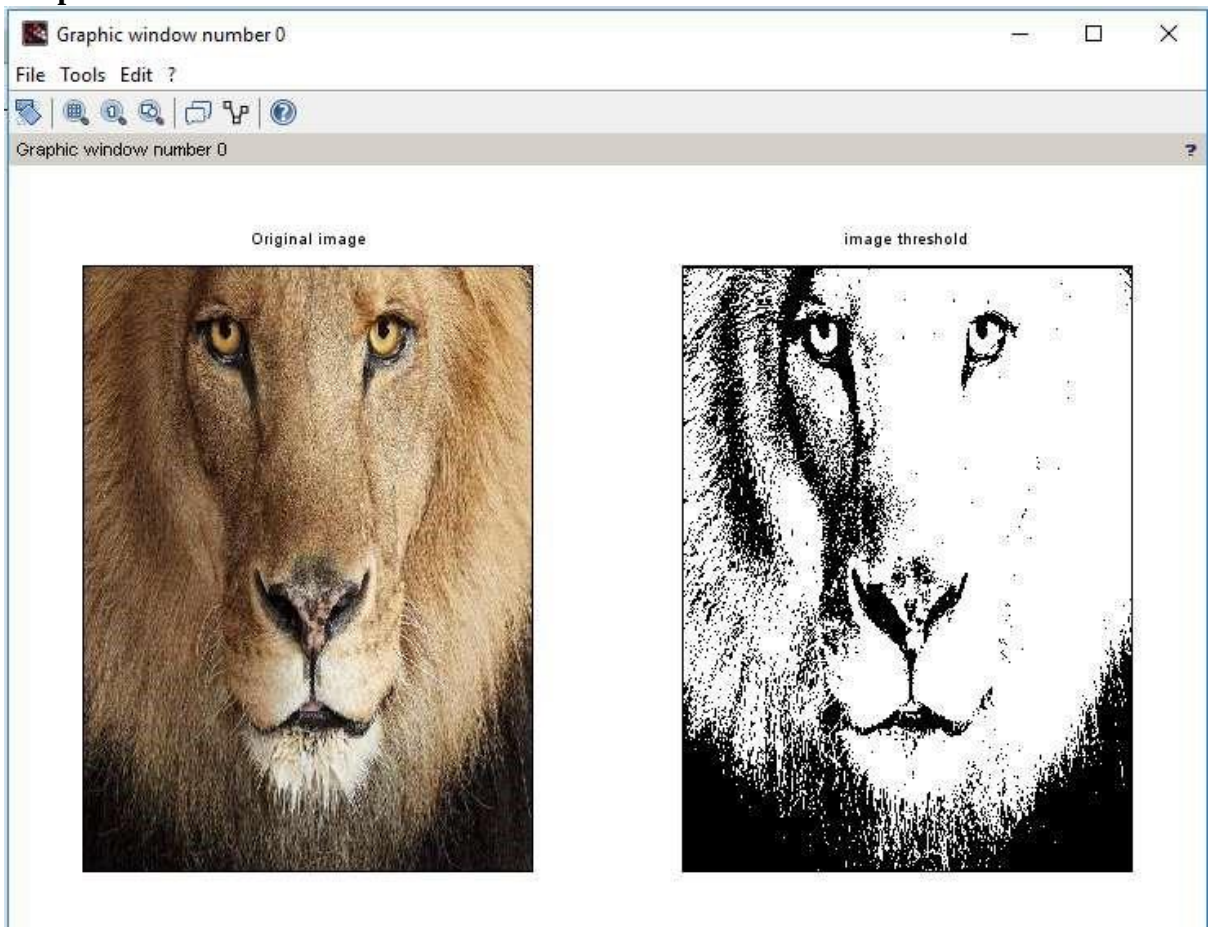
Output-



2. Threshold on an image Code-

```
clc;
clear all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
b=double(a); subplot(1,2,1); imshow(a);
title('Original image'); t=100; [m,n]=size(b); for
i=1:m     for j=1:n         if(b(i,j)<t)
c(i,j)=0;         else
          c(i,j)=255; end
end end subplot(1,2,2);
imshow(c);
title('image threshold');
```

Output-

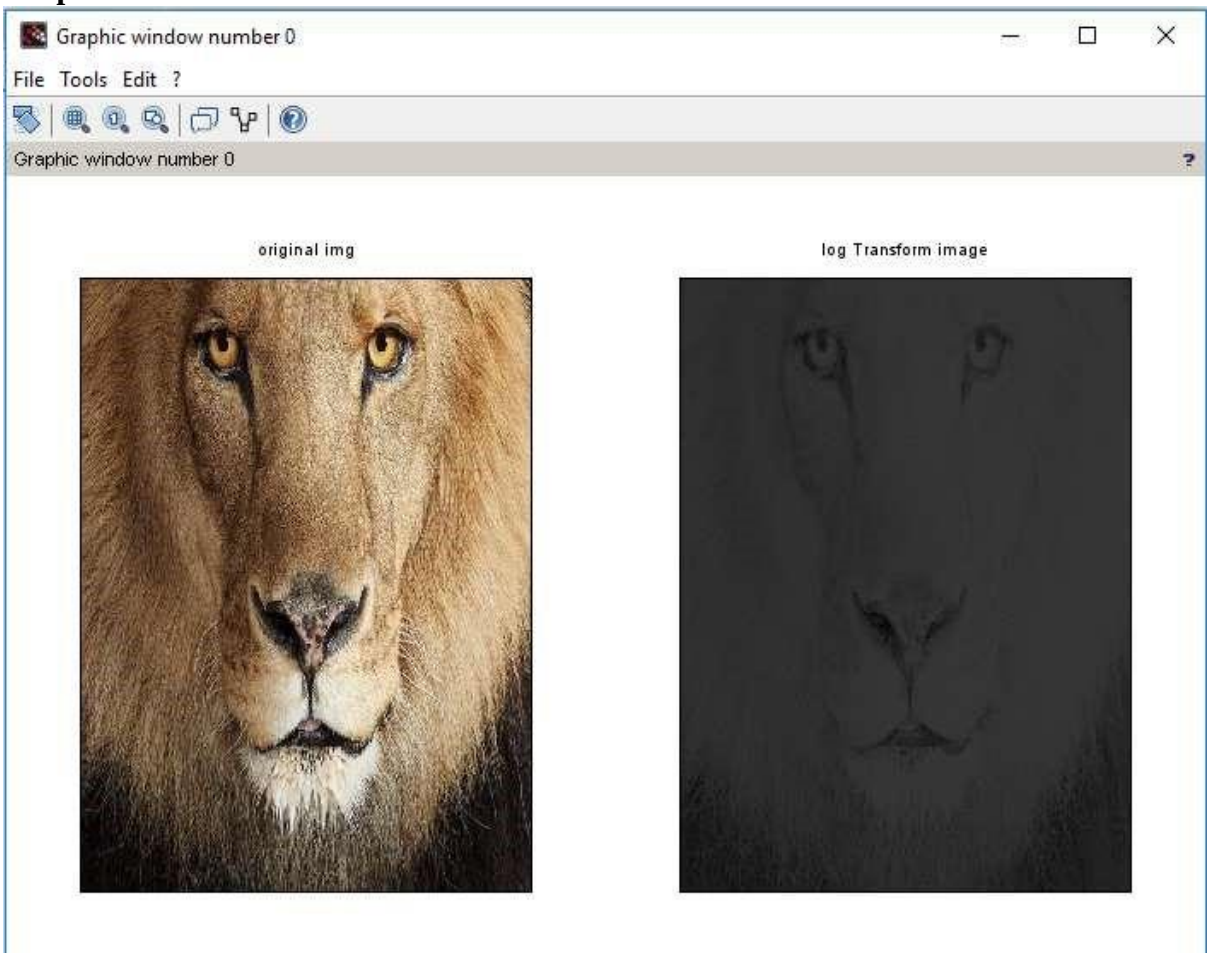


3. Program to Perform Log Transformation Code-

```
clc; clear all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
b=double(a); subplot(1,2,1);
imshow(a);
title('original img');
t=10;
[m,n]=size(b); for
i=1:m     for j=1:n
      c(i,j)=t*log(1+b(i,j));
      end end subplot(1,2,2);
imshow(uint8(c)); title('log
Transform image');
```

Output-

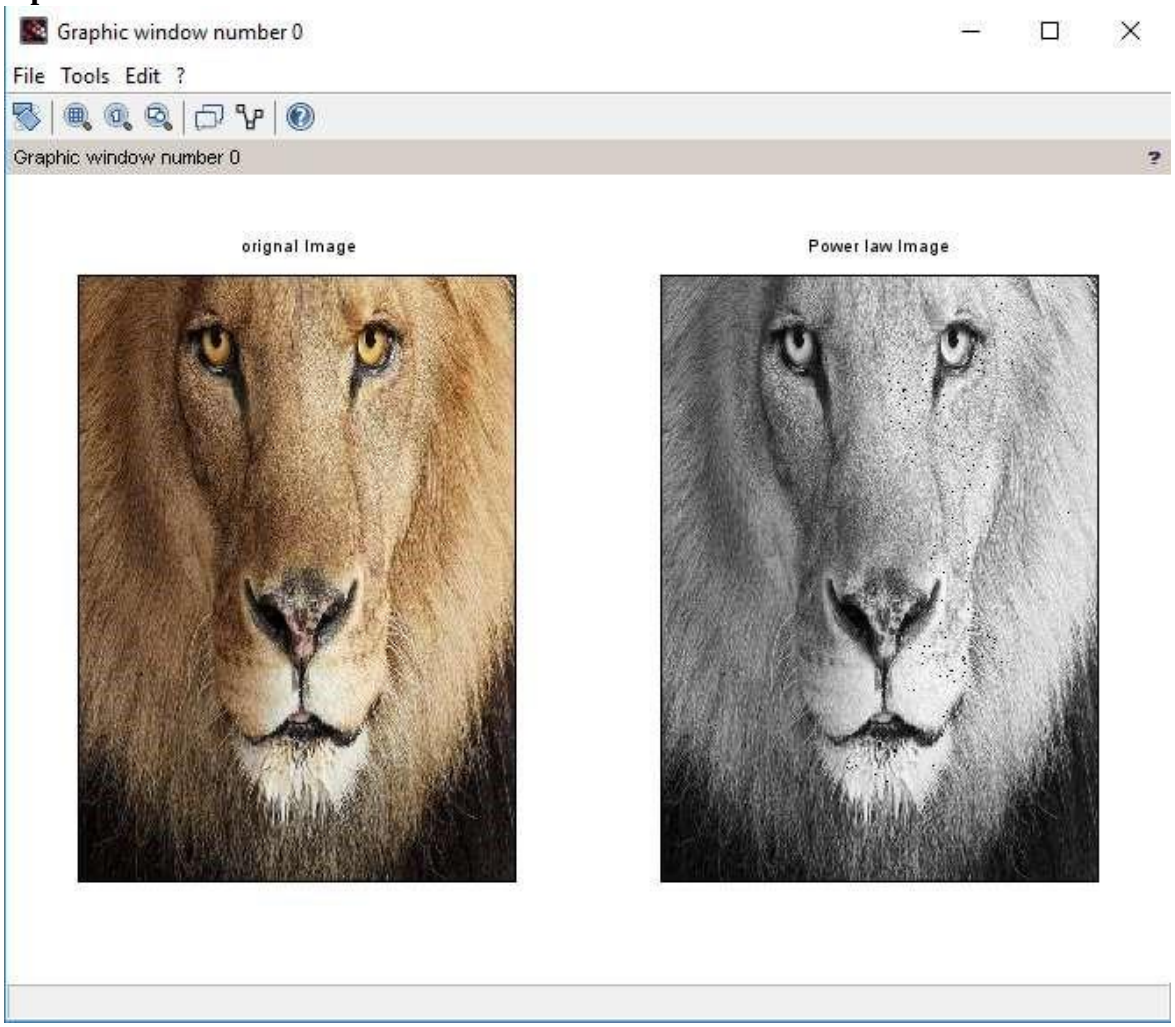
Output-



4. Power-law Transformation Code-

```
clc; clear
all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
b=double(a) subplot(1,2,1);
imshow(a);
title('orignal Image');
k=1;
gamma=1;
[m,n]=size(b);
for i=1:m
for j=1:n
      c(i,j)=k+(b(i,j)^gamma);
      end
end subplot(1,2,2);
imshow(uint8(c));
title('Power law Image');
```

Output-



5. Piecewise Linear Transformation

a) Contrast Stretching.

Code-

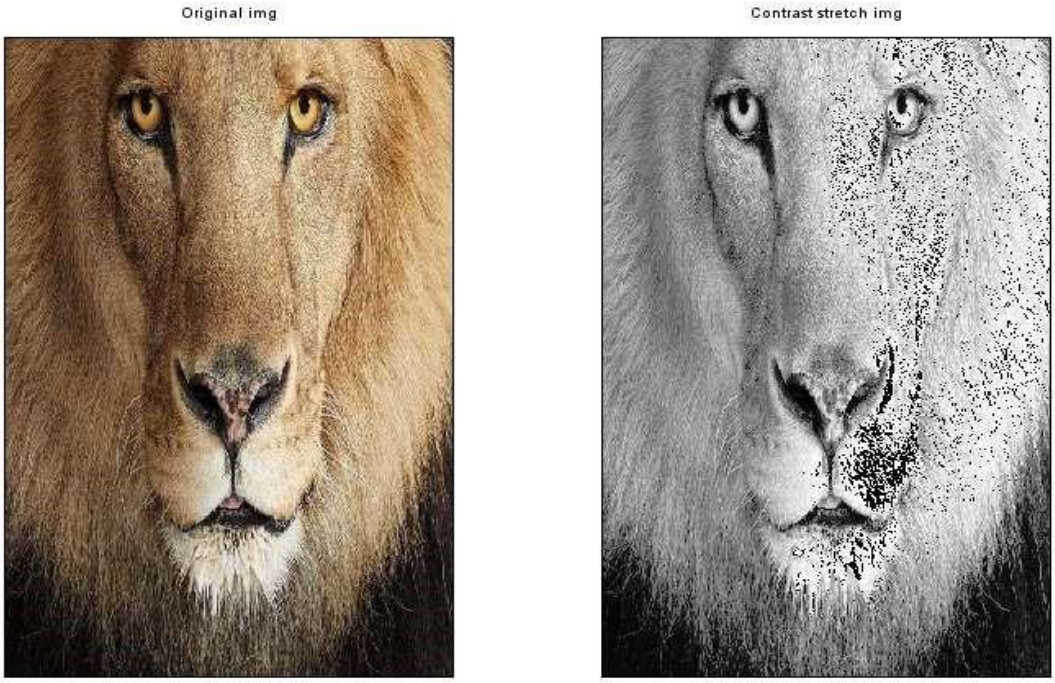
```
clc; clear
all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
b=double(a);  [m,n]=size(b); x1=input('Enter
x1'); x2=input('Enter x2');  y1=input('Enter y1');
y2=input('Enter y2');

slope1=y1/x1;  slope2=(y2-y1)/(x2-
x1); slope3=(255-y2)/(255-x2);
inter1=y1-slope2*x1;  inter2=y2-
slope3*x1; ics=zeros(m,n); for i=1:m
for j=1:n if(0<b(i,j)&&b(i,j)<x1)
ics(i,j)=slope1*b(i,j); else if(x1<b(i,j)&&b(i,j)<x2)
ics(i,j)=slope2*b(i,j)+inter1; else if(x2<b(i,j)&&b(i,j)<255)
ics(i,j)=slope3*b(i,j)+inter2;          end end  end end  end
```

Output-

subplot(1,2,1); imshow(a); title('Original img');
subplot(1,2,2); imshow(uint8(ics)); title('Contrast stretch img');

Input-



Output-

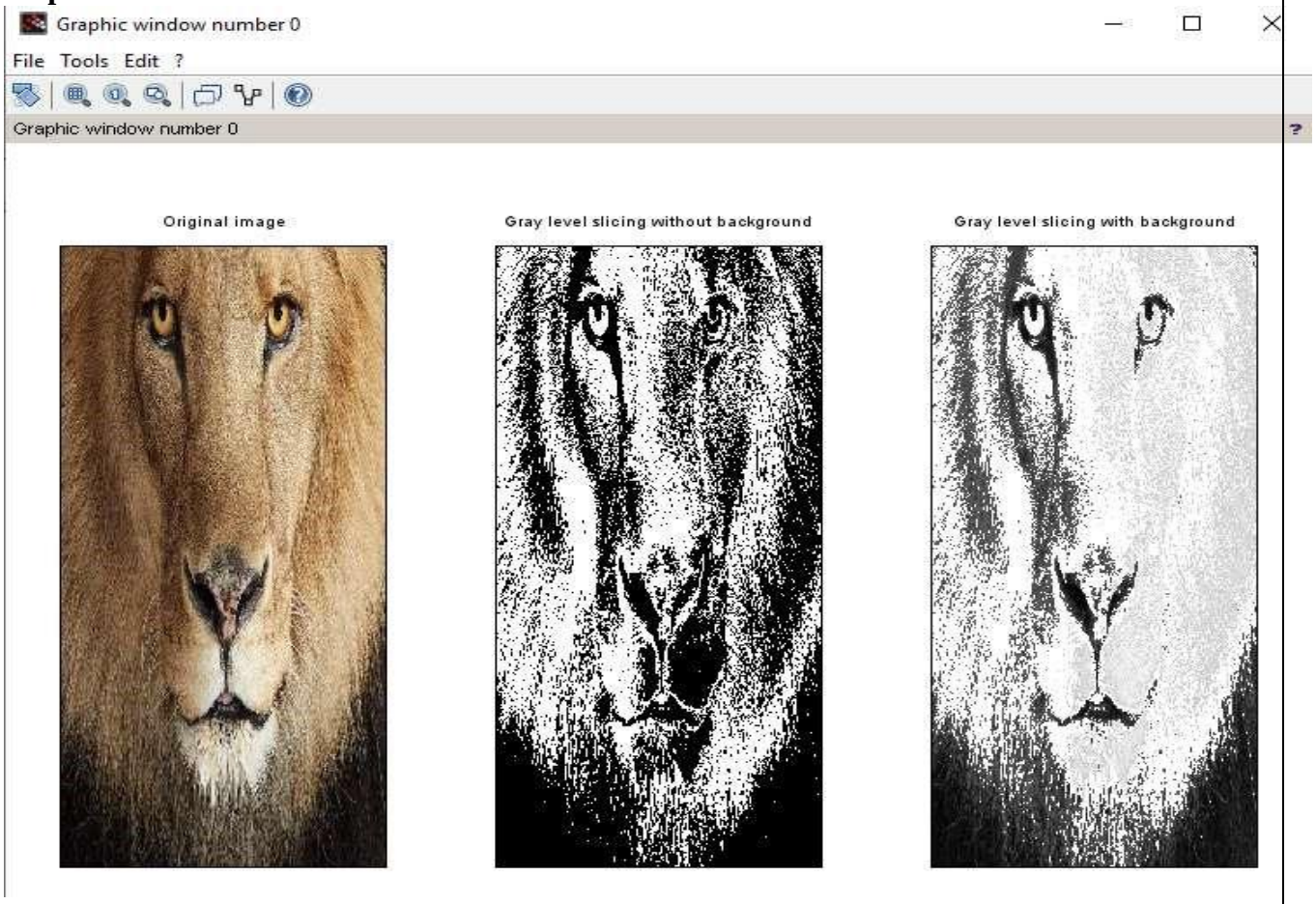
b) Grey-level Slicing with and without background.

Code-

```
clc;  clear
all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg");
b=double(a);  [m,n]=size(b); x1=input("Enter
x1");  x2=input("Enter x2"); c=zeros(m,n);
d=zeros(m,n); for i=1:m
for j=1:n
if(b(i,j)>=x1 && b(i,j)<=x2);
   c(i,j)=255;
else c(i,j)=0;
end end  end
for i=1:m
   for j=1:n
if(b(i,j)>=x1 && b(i,j)<=x2) d(i,j)=255;
else d(i,j)=b(i,j);
end end  end
subplot(1,3,1);
imshow(a);
title("Original image");
subplot(1,3,2);
imshow(uint8(c));
title("Gray level slicing without background");
subplot(1,3,3);  imshow(uint8(d));
title("Gray level slicing with background");
```

Input-

Output-



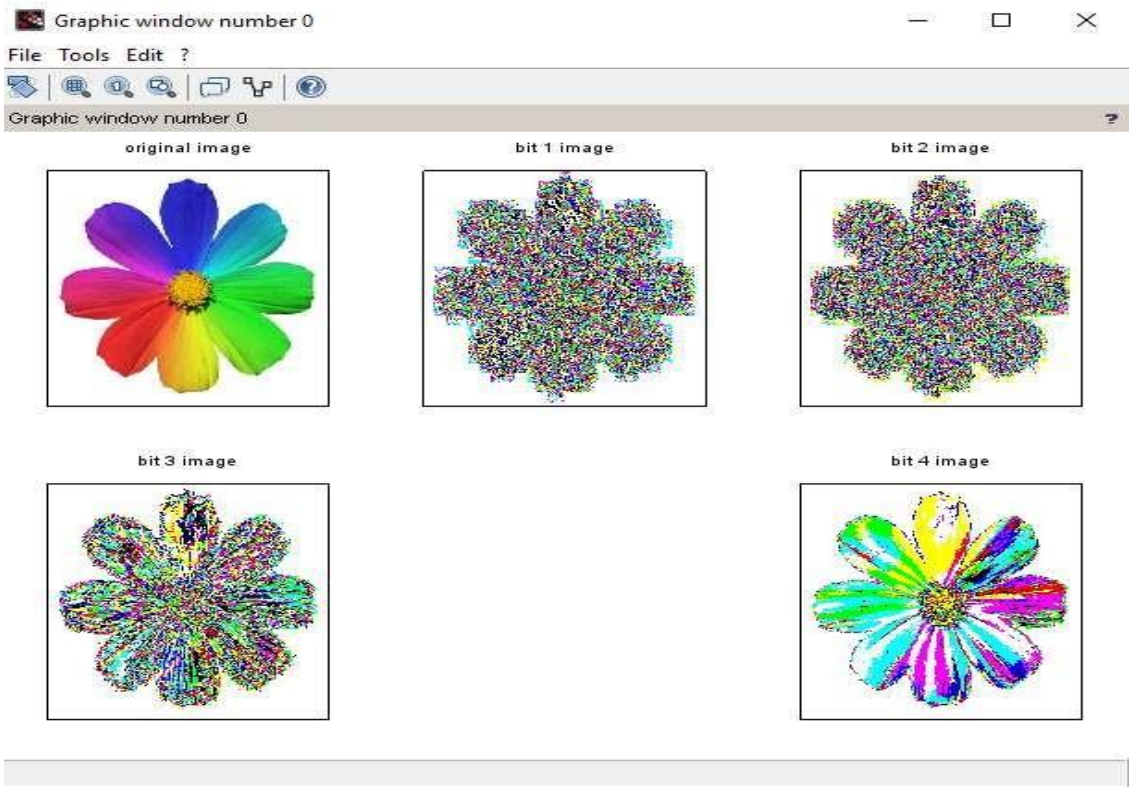
C) Bit-Plane Slicing. Code-

```
clc; clear
all;
a=imread('C:\Users\5itlab\Documents\flower.jpeg');
b=double(a); subplot(2,3,1); imshow(a);
title('original image');
f1=bitget(b,1);
subplot(2,3,2);
imshow(f1); title("bit
1 image");
f2=bitget(b,2);
subplot(2,3,3);
imshow(f2); title("bit
2 image");
f3=bitget(b,4);
subplot(2,3,4);
imshow(f3); title("bit
3 image");
f4=bitget(b,6);
subplot(2,3,6);
```

Output-

imshow(f4); title("bit
4 image");

Output-



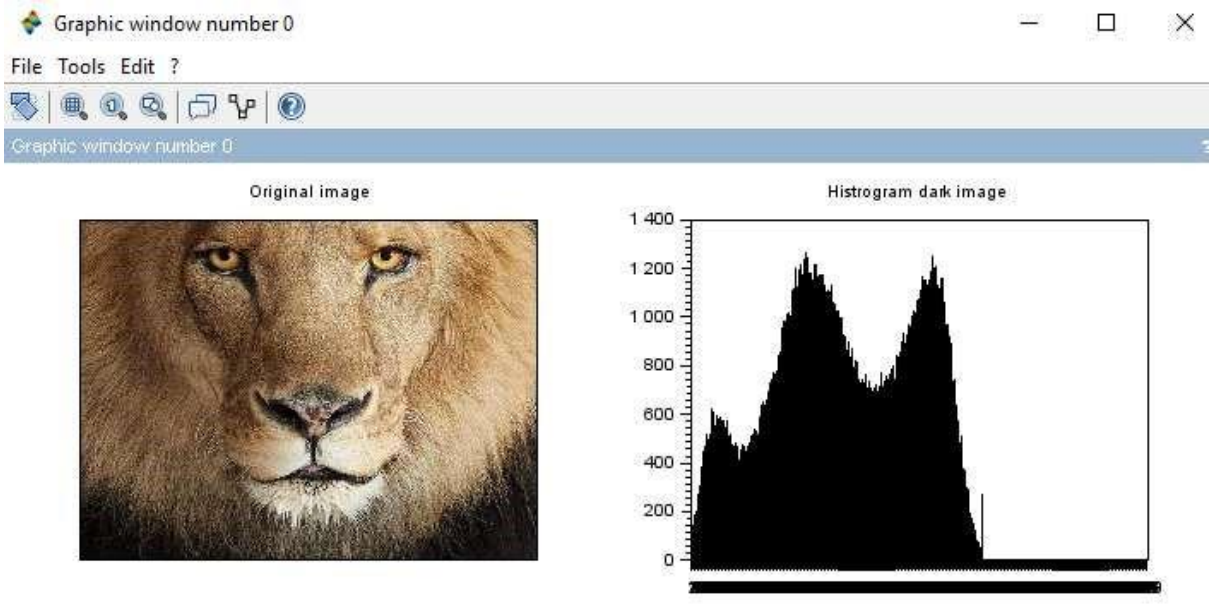
Part B:

1. Program to plot the histogram of an image and categories.

Code-

```
clc; clear all;
Img=imread('C:\Users\5itlab\Documents\lion.jpeg');
Img1=double(Img);
[row col]=size(Img1);
h=zeros(row,col);  for
n=1:1:row for
m=1:1:col; if
Img1(n,m)==0;
Img1(n,m)=1; end
end end for
n=1:1:row      for
m=1:1:col
t=Img1(n,m);
h(t)=h(t)+1; end  end
subplot(2,2,1),
imshow(Img),title('Original image')
subplot(2,2,2), bar(h),
title('Histrogram dark image');
```

Output-



Write a program to perform convolution and correlation.

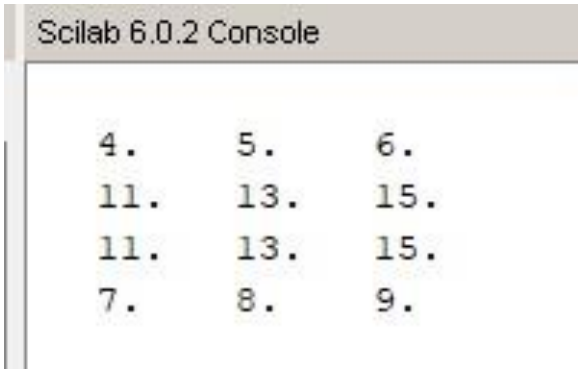
Part C:

1. **Convolution-**

Code-

```
clc; x=[4,5,6;7,8,9];
h=[1;1;1];
y=conv2(x,h);
disp(y);
```

Output-

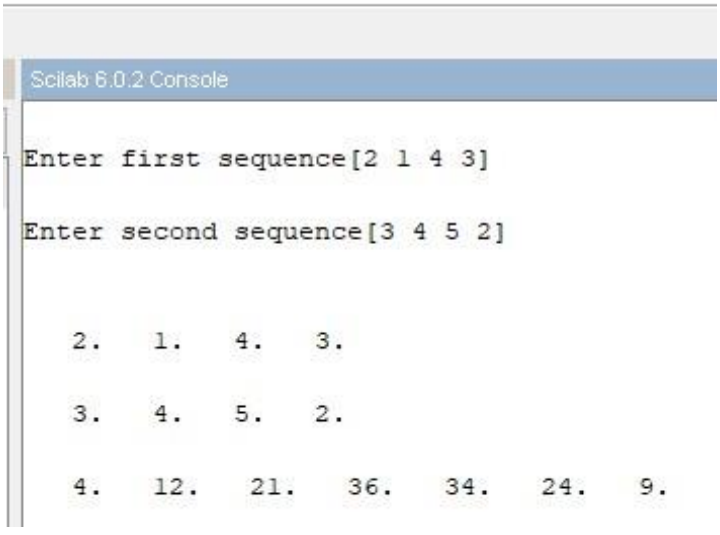


2. Correlation-

Code-

```
clc; x=input('Enter first
sequence'); h=input('Enter second
sequence'); y=xcorr(x,h); disp(x);
disp(h); disp(y);
```

Output-



Write a program to apply smoothing and sharpening filters on grayscale and color images.

1.**Low Pass**

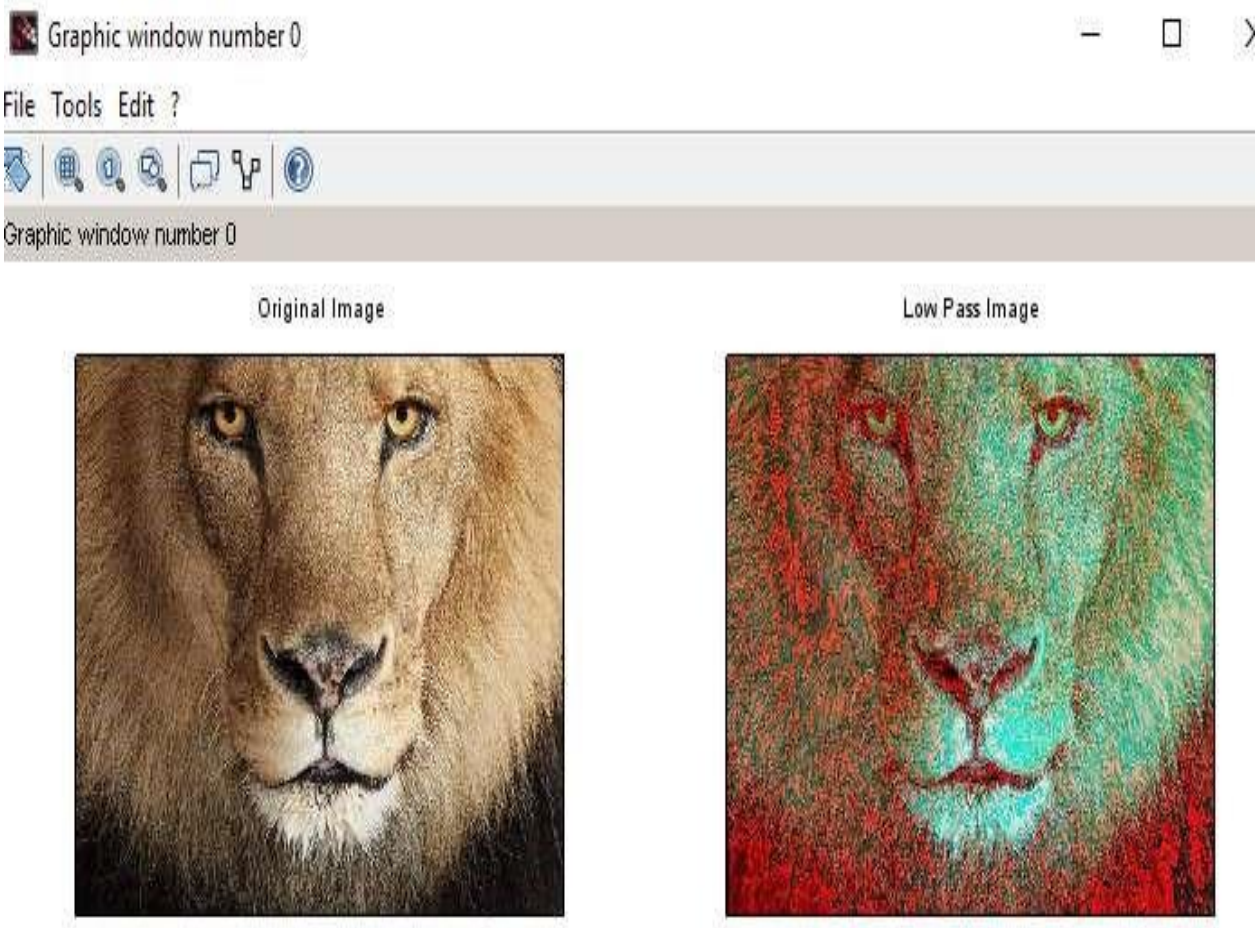
Code-

```
clc; clear all;
```

Part D:

```
a1=imread('C:\Users\5itlab\Documents\lion.jpeg');
a=double(a1); [m,n]=size(a); w=[1 1 1; 1 1 1; 1 1
1]; for i=2:m-1      for j=2:n-1
b(i,j)=(w(1)*a(i-1,j+1)+w(2)*a(i,j+1)+w(3)*a(i+1,j+1)+w(4)*a(i-
1,j)+w(5)*a(i,j)+w(6)*a(i+1,j)+w(7)*a(i-1,j-1)+w(8)*a(i,j-1)+w(9)*a(i+1,j-1))
end end subplot(2,2,1),  imshow(a1), title('Original Image'); subplot(2,2,2),
imshow(uint8(b)), title('Low Pass Image');
```

Output-

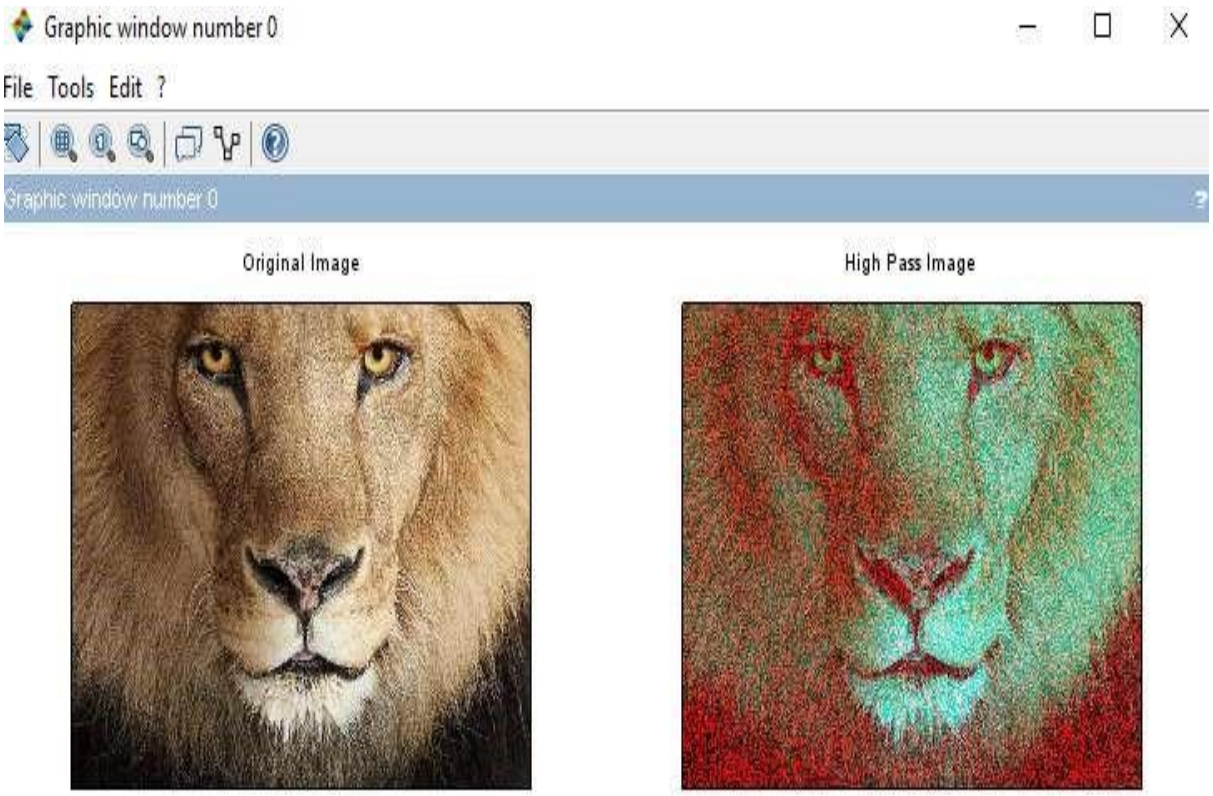


2.**High Pass**

Code-

```
clc; clear all;
a1=imread('C:\Users\5itlab\Documents\lion.jpeg');
a=double(a1); [m,n]=size(a);
w=[-1 -1 -1; -1 8 -1; -1 -1 -1];
for i=2:m-1
for j=2:n-1
b(i,j)=(w(1)*a(i-1,j+1)+w(2)*a(i,j+1)+w(3)*a(i+1,j+1)+w(4)*a(i-
1,j)+w(5)*a(i,j)+w(6)*a(i+1,j)+w(7)*a(i-1,j-1)+w(8)*a(i,j-1)+w(9)*a(i+1,j-1))
end end subplot(2,2,1),
imshow(a1),
title('Original Image');
subplot(2,2,2),
imshow(uint8(b)), title('High
Pass Image');
```

Output-



Practical No 3

Aim- Filtering in Frequency Domain.

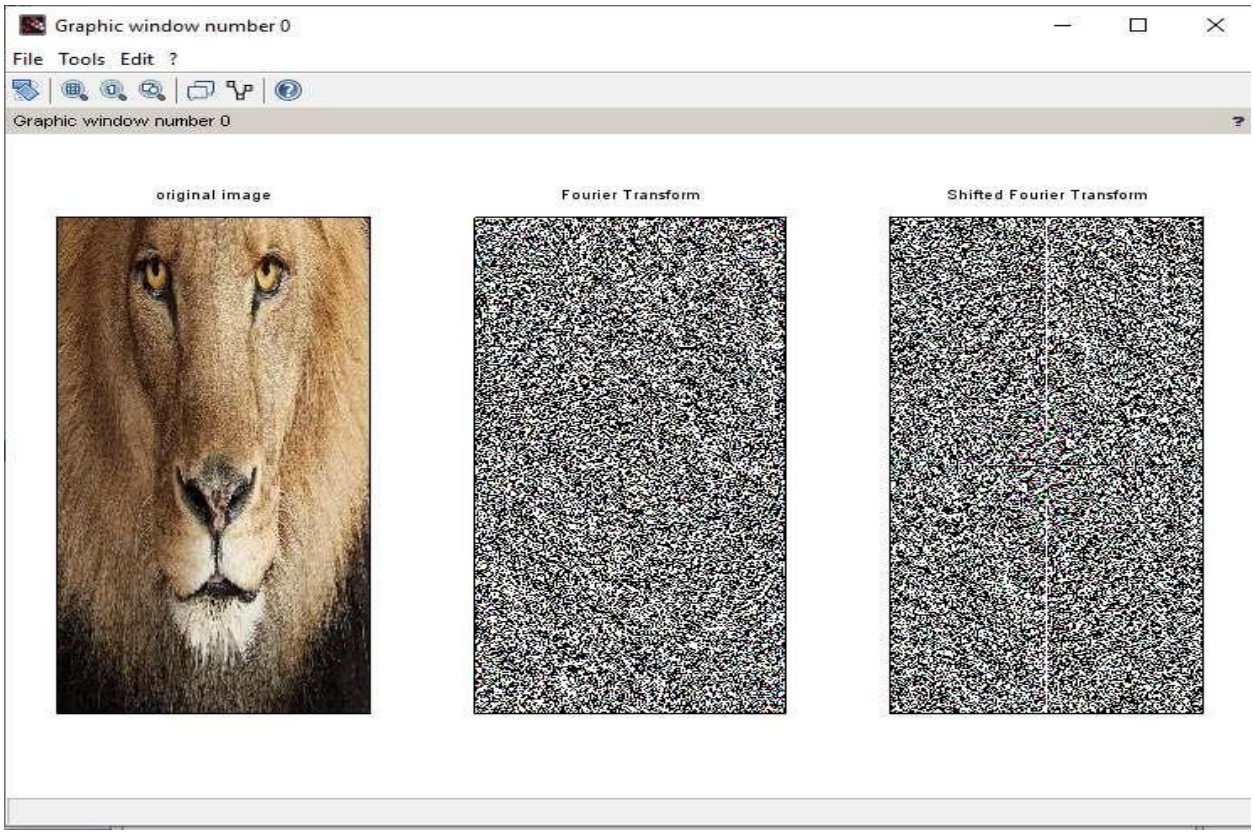
Part A:

Aim- Program to apply Discrete Fourier Transform on an image.

Code-

```
clc;
clear;
close;
i=imread('C:\Users\5itlab\Documents\lion.jpeg');
 subplot(1,3,1) imshow(i) title("original image")
i=double(i); j=fft2(i); subplot(1,3,2) imshow(j)
title("Fourier Transform") L=fftshift(real(j));
subplot(1,3,3) imshow(L)
title("Shifted Fourier Transform")
```

Output-



Part B:

Aim- Program to apply Low pass and High pass filters in frequency domain.

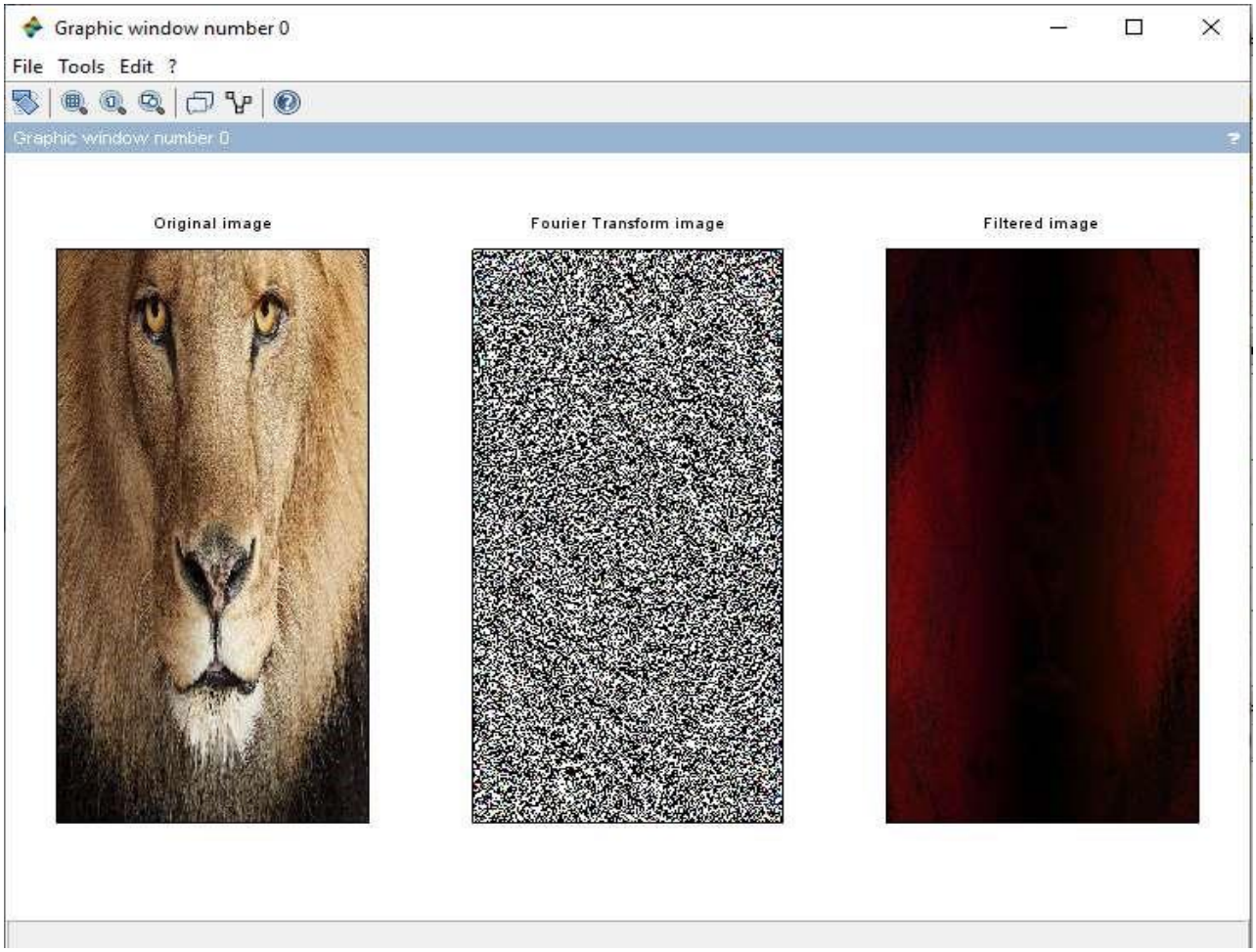
Low Pass Filter-

Code-

```
clc;
s=imread('C:\Users\5itlab\Documents\lion.jpeg');
s2=fft2(im2double(s)); h=ffilt('lp',3,0.2,0.6);
img=imfilter(s2,h); s4=real(ifft(img));
```

```
subplot(1,3,1); imshow(s); title("Original
image"); subplot(1,3,2);
imshow(s2);
title("Fourier Transform image");
subplot(1,3,3);
imshow(s4);
title("Filteredimage");
```

Output-



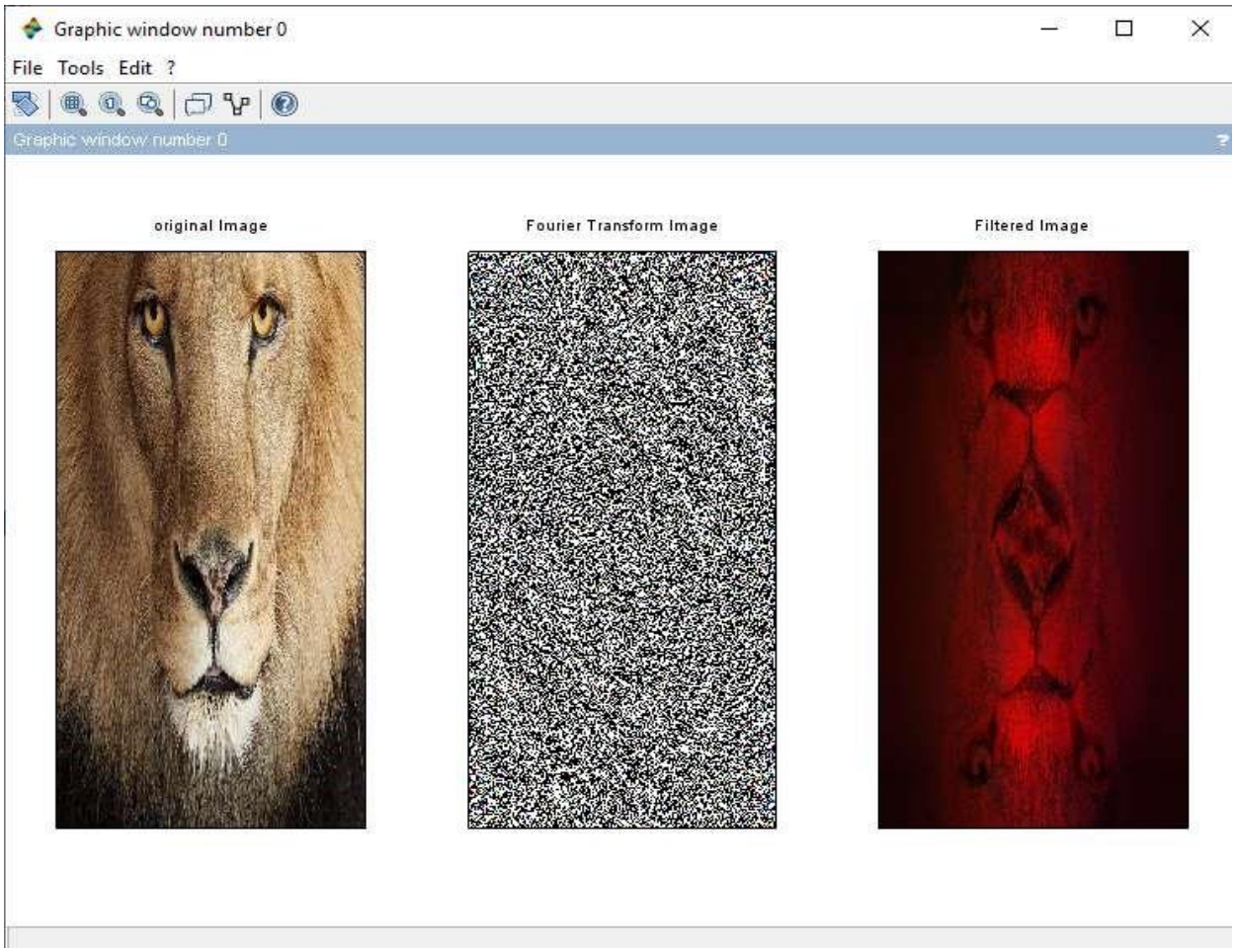
High Pass Filter-

Code-

```
clc;
s=imread('C:\Users\5itlab\Documents\lion.jpeg');
s2=fft2(im2double(s));
h=ffilt('hp',3,0.2,0.6);
img=imfilter(s2,h);
s4=real(ifft(img));
subplot(1,3,1);
imshow(s);
 title("original Image");
subplot(1,3,2);
```

```
imshow(s2);
title("Fourier Transform Image");
subplot(1,3,3); imshow(s4);
title("Filtered Image");
```

Output-



Part C:

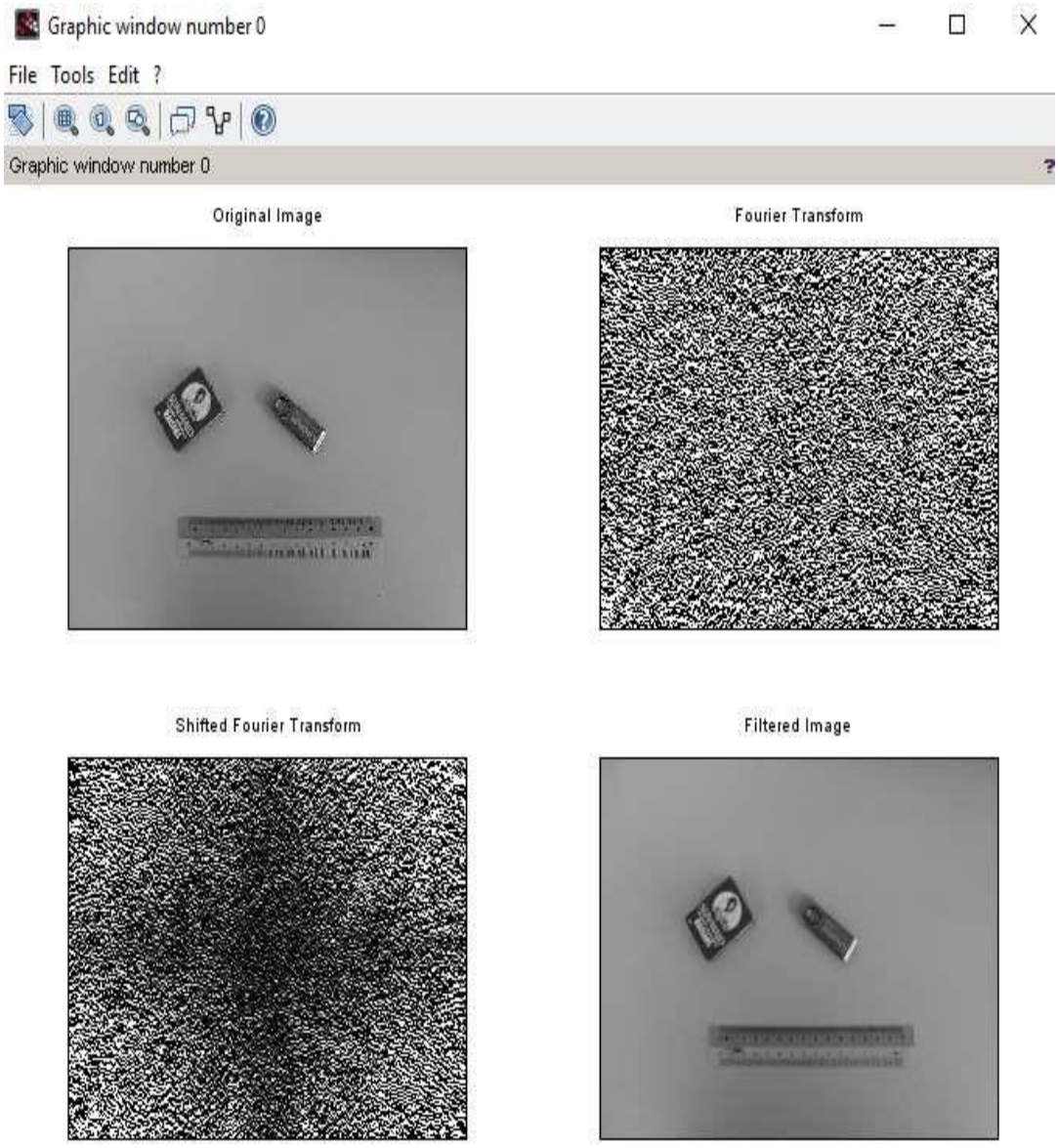
Aim- Program for Butterworth and gaussian filter in frequency domain.

Butterworth-

```
clc;
S=imread(fullpath(getIPCVpath()+"/images/measure_gray.jpg"));
h=mkfftfilter(S,'butterworth1',0.4);
S2= fft2(im2double(S));
S3 = S2.*fftshift(h); S4 =
real(ifft(S3));
subplot(2,2,1);
imshow(S); title("Original
Image"); subplot(2,2,2);
imshow(S2); title("Fourier
Transform");
```

```
subplot(2,2,3);
imshow(S3);
title("Shifted Fourier Transform");
subplot(2,2,4); imshow(S4);
title("Filtered Image");
```

Output-



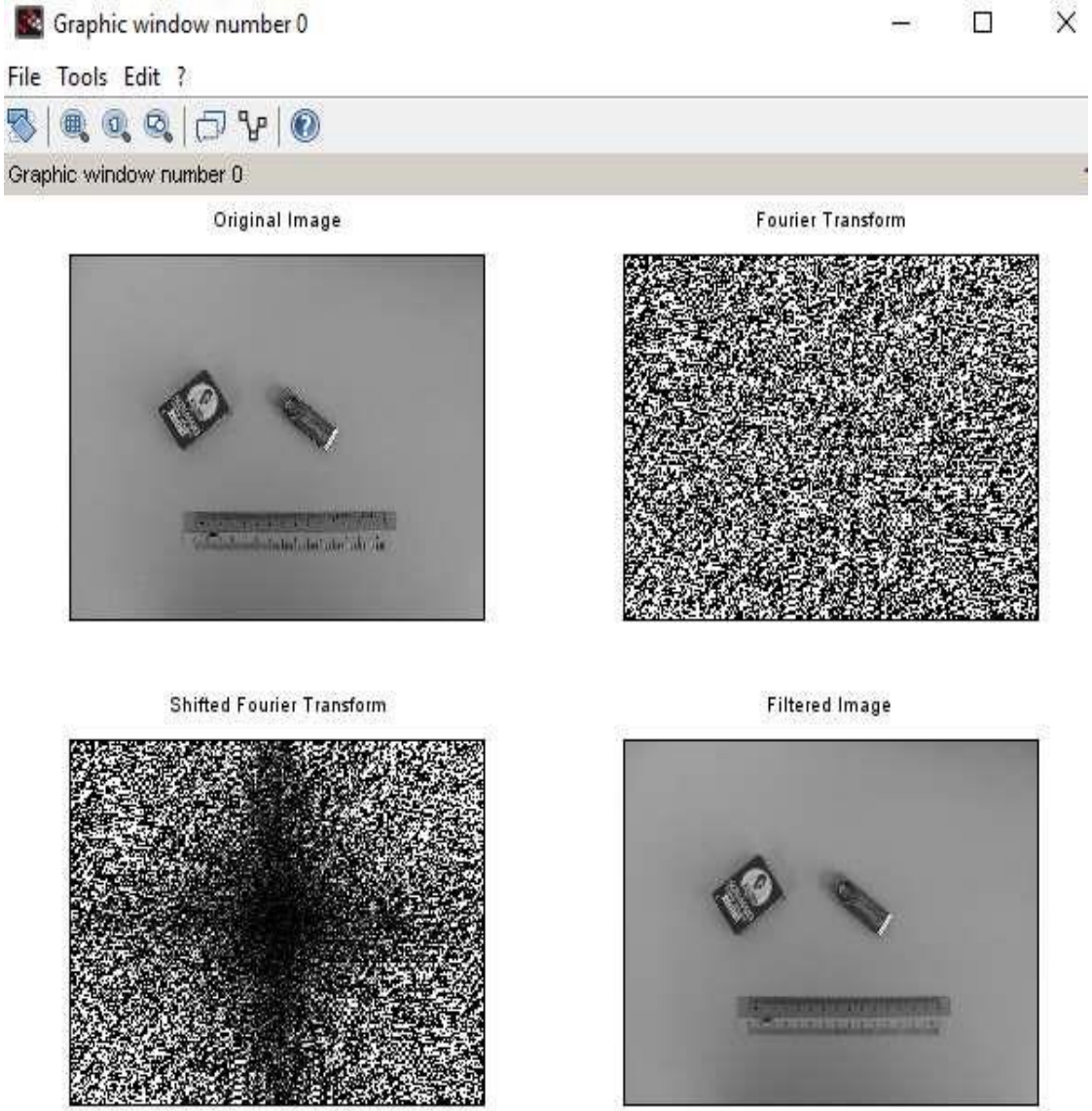
Gaussian-

Code-

```
clc;
S=imread(fullpath(getIPCVpath()+"/images/measure_gray.jpg"));
h=mkfftfilter(S,'gauss',0.8);
S2= fft2(im2double(S));
S3 = S2.*fftshift(h); S4 =
real(ifft(S3));
```

```
subplot(2,2,1);
imshow(S);
title("Original Image");
subplot(2,2,2);
imshow(S2);
 title("FourierTransform");
subplot(2,2,3);
imshow(S3);
title("Shifted Fourier Transform");
subplot(2,2,4); imshow(S4);
title("Filtered Image");
```

Output-



Practical No 4

Aim- Image Denoising.

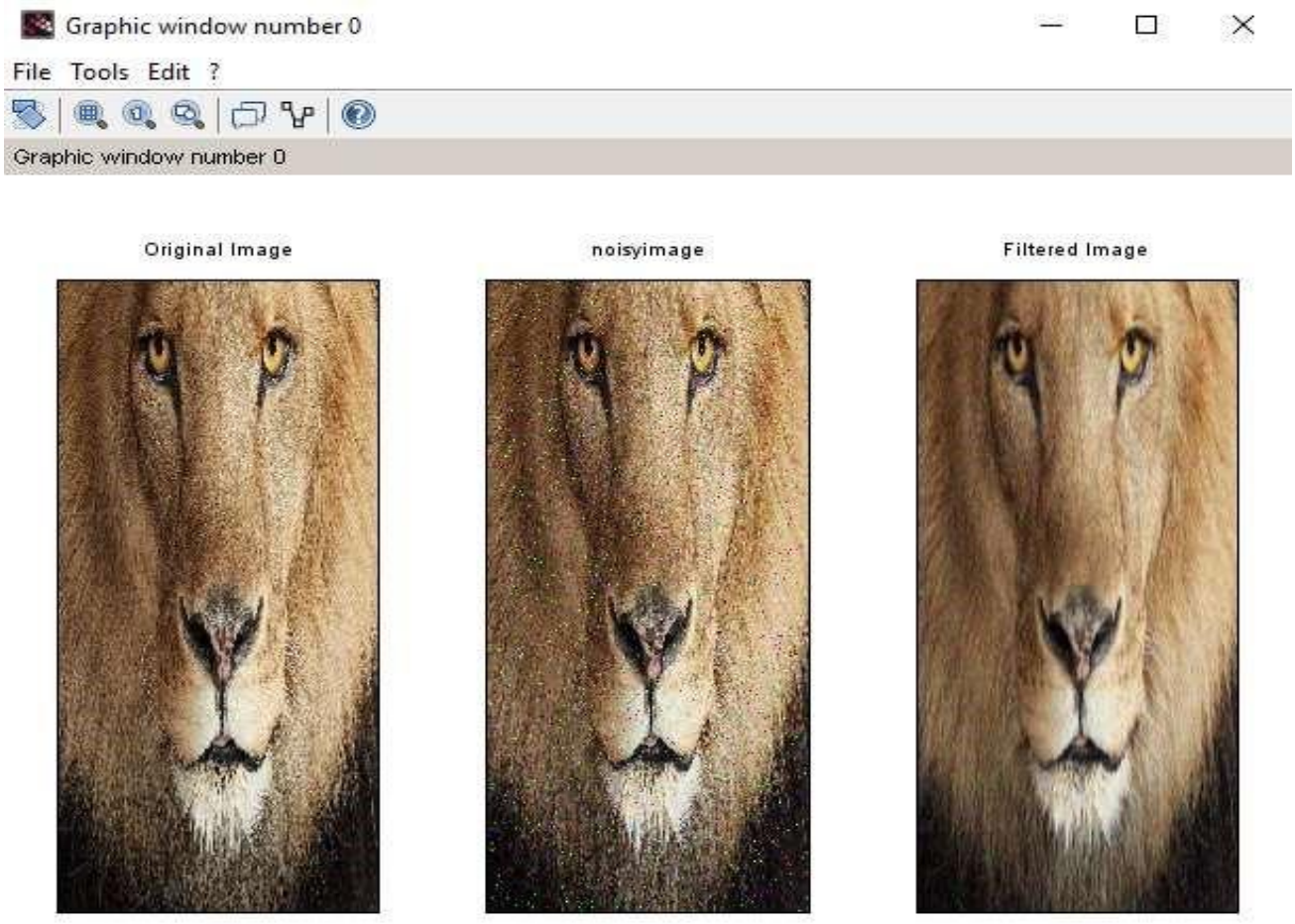
Part A:

Aim- Program to denoise using spatial mean, median.

Mean- Code-

```
clc;
i=imread('C:\Users\5itlab\Documents\lion.jpeg'); noisyimage=imnoise(i,'salt
& pepper',0.02);
f=fspecial('average',3);
filterimage=imfilter(noisyimage,f);
subplot(1,3,1); imshow(i);
title('Original Image');
subplot(1,3,2);
imshow(noisyimage);
title('noisyimage'); subplot(1,3,3);
imshow(filterimage);
title('Filtered Image');
```

Output-

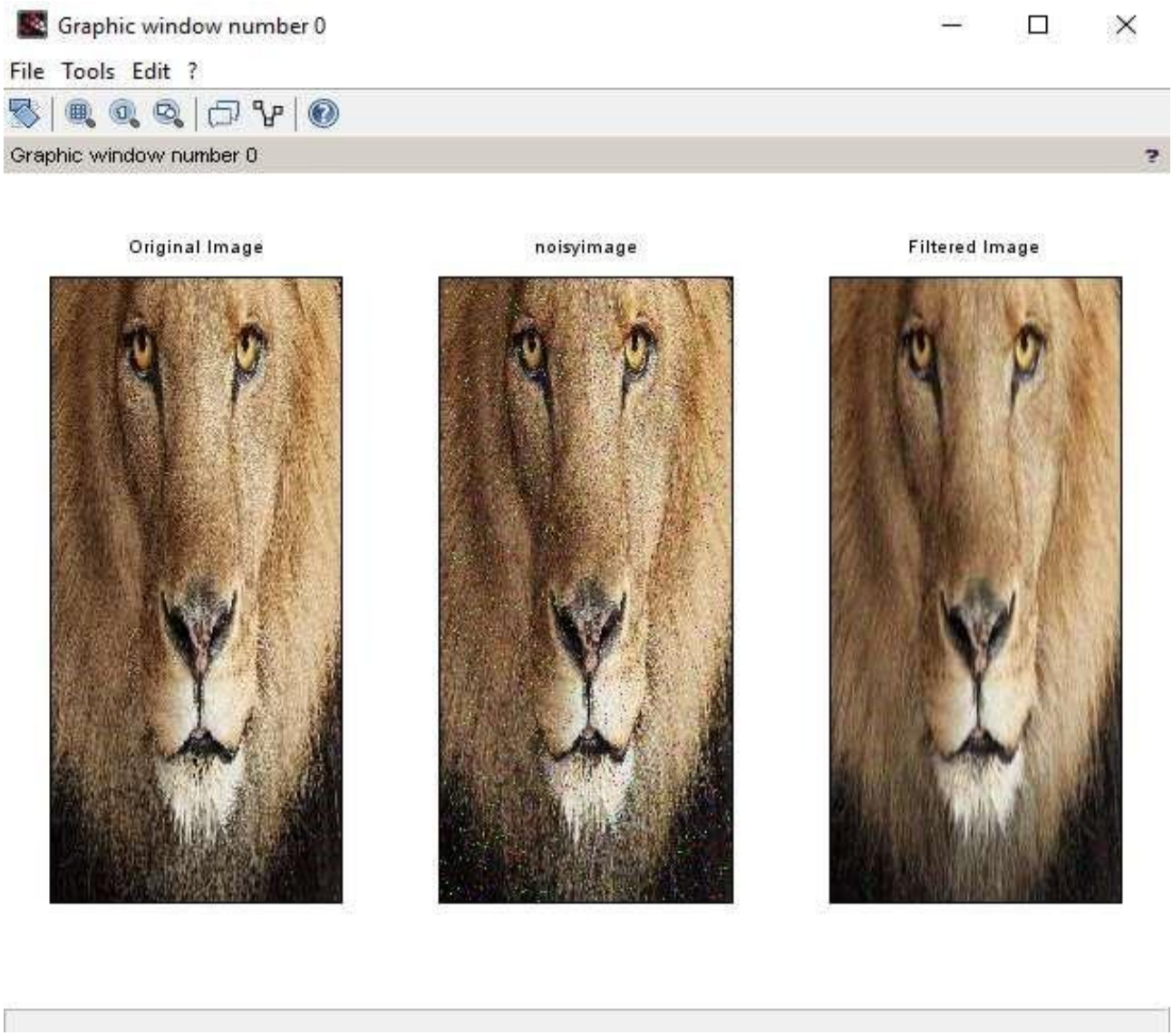


Median-

Code-

```
clc;
i=imread('C:\Users\5itlab\Documents\lion.jpeg');
noisyimage=imnoise(i,'salt & pepper',0.02);
imagefilter=immedian(noisyimage,[5,5,3]);
subplot(1,3,1); imshow(i); title('Original
Image'); subplot(1,3,2); imshow(noisyimage);
title('noisyimage'); subplot(1,3,3);
imshow(filterimage); title('Filtered Image');
```

Output-



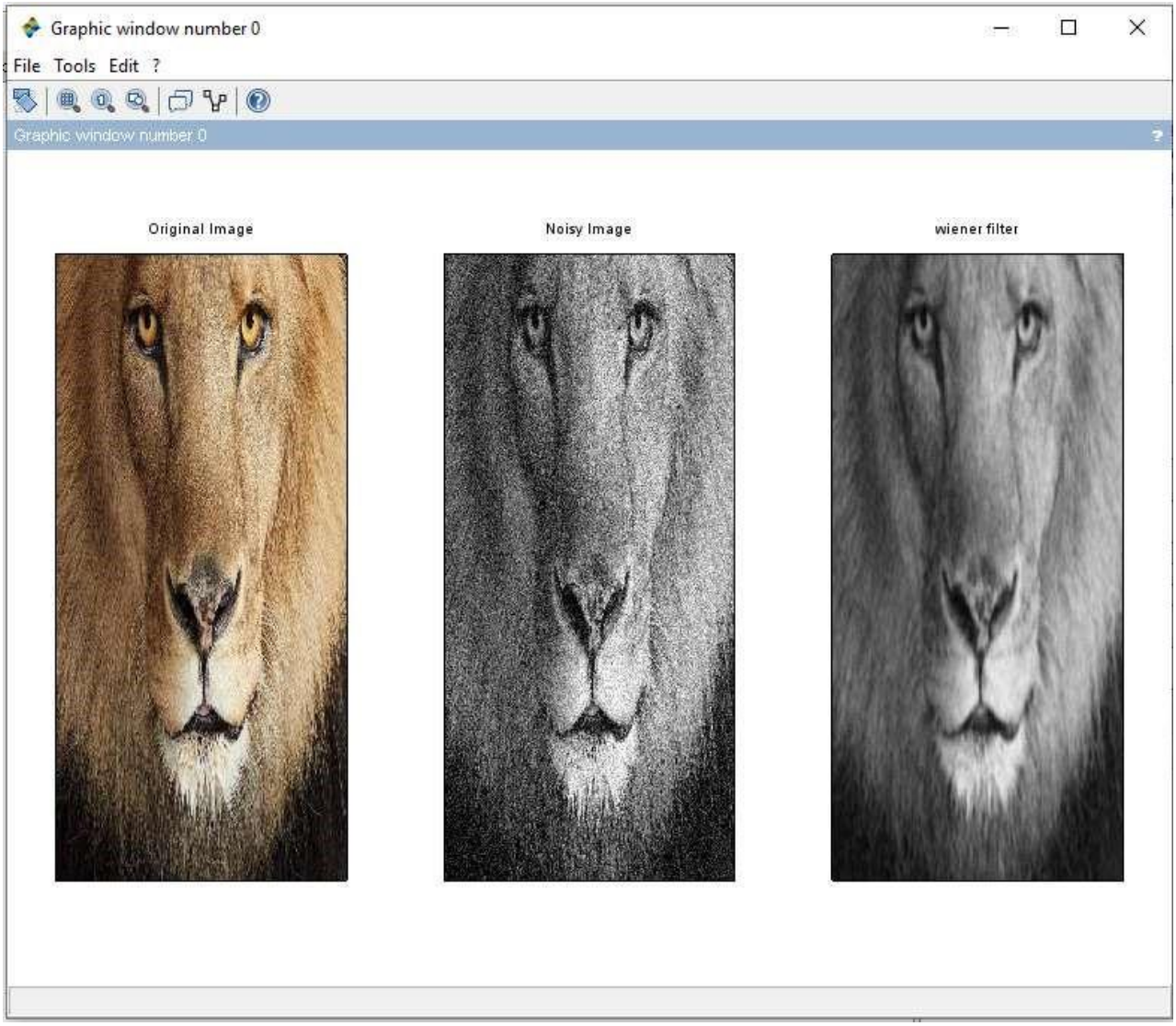
Part B:

Aim- Program for Image Weiner filters. **Code-**

```
clc;
```

```
i=imread('C:\Users\5itlab\Documents\lion.jpeg');
noisyimage=imnoise(i,'gaussian',0.04);
wienerfilter=imwiener2(noisyimage,[5,5,3],0.2);
subplot(1,3,1); imshow(i); title('Original
Image'); subplot(1,3,2); imshow(noisyimage);
title('noisyimage'); subplot(1,3,3);
imshow(wienerfilter); title(wiener filter ');
```

Output-



Practical No 5

Aim- Color Image Processing.

Part A:

Aim- Program to read a color image and segment into RGB planes, histogram of color image.

Code-

```
clc;
i=imread('C:\Users\5itlab\Documents\lion.jpeg');
r=size(i,1); c=size(i,2); R=zeros(r,c,3);
G=zeros(r,c,3);
B=zeros(r,c,3);
R(:,:,1)=i(:,:,1);
G(:,:,2)=i(:,:,2);
B(:,:,3)=i(:,:,3);
subplot(2,2,1);
imshow(i); title('Original
image'); subplot(2,2,2);
imshow(uint8(R));
title('Red Component');
subplot(2,2,3);
imshow(uint8(G));
title('Green component');
subplot(2,2,4);
imshow(uint8(B));
title('Blue component');
```

Output-

Part B:

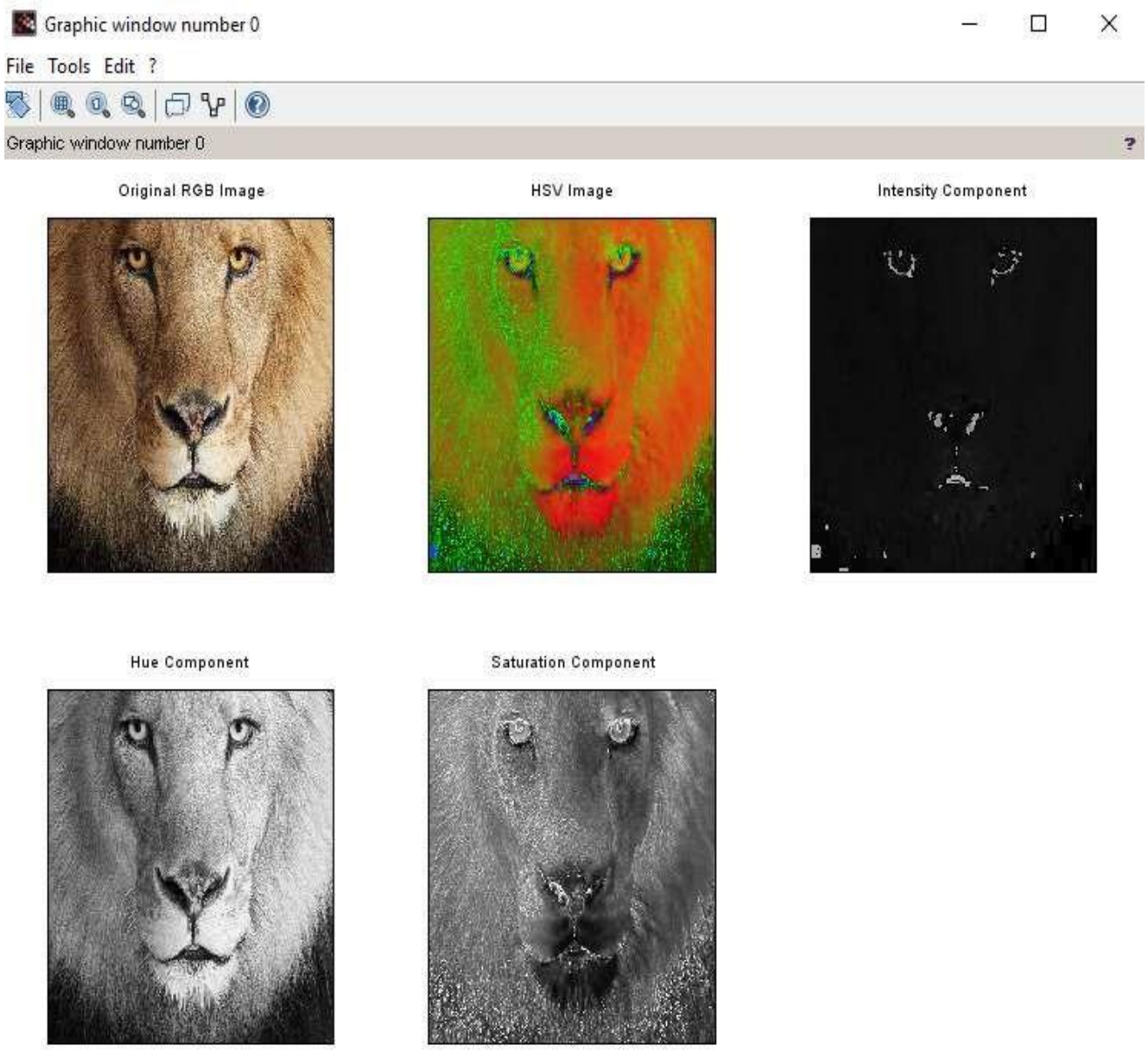
Aim- Program for converting from one color model to another model.

Code-

```
clc;
rgb=imread('C:\Users\5itlab\Documents\lion.jpeg');
hsv=rgb2hsv(rgb);
subplot(2,3,1);
imshow(rgb);
title('Original RGB Image');
subplot(2,3,2); imshow(hsv);
title('HSV Image');
subplot(2,3,3);
imshow(hsv(:,:,3));
title('Intensity Component');
```

```
subplot(2,3,4);
imshow(hsv(:,:,1));
title('Hue Component');
subplot(2,3,5);
imshow(hsv(:,:,2));
title('Saturation Component');
```

Output-



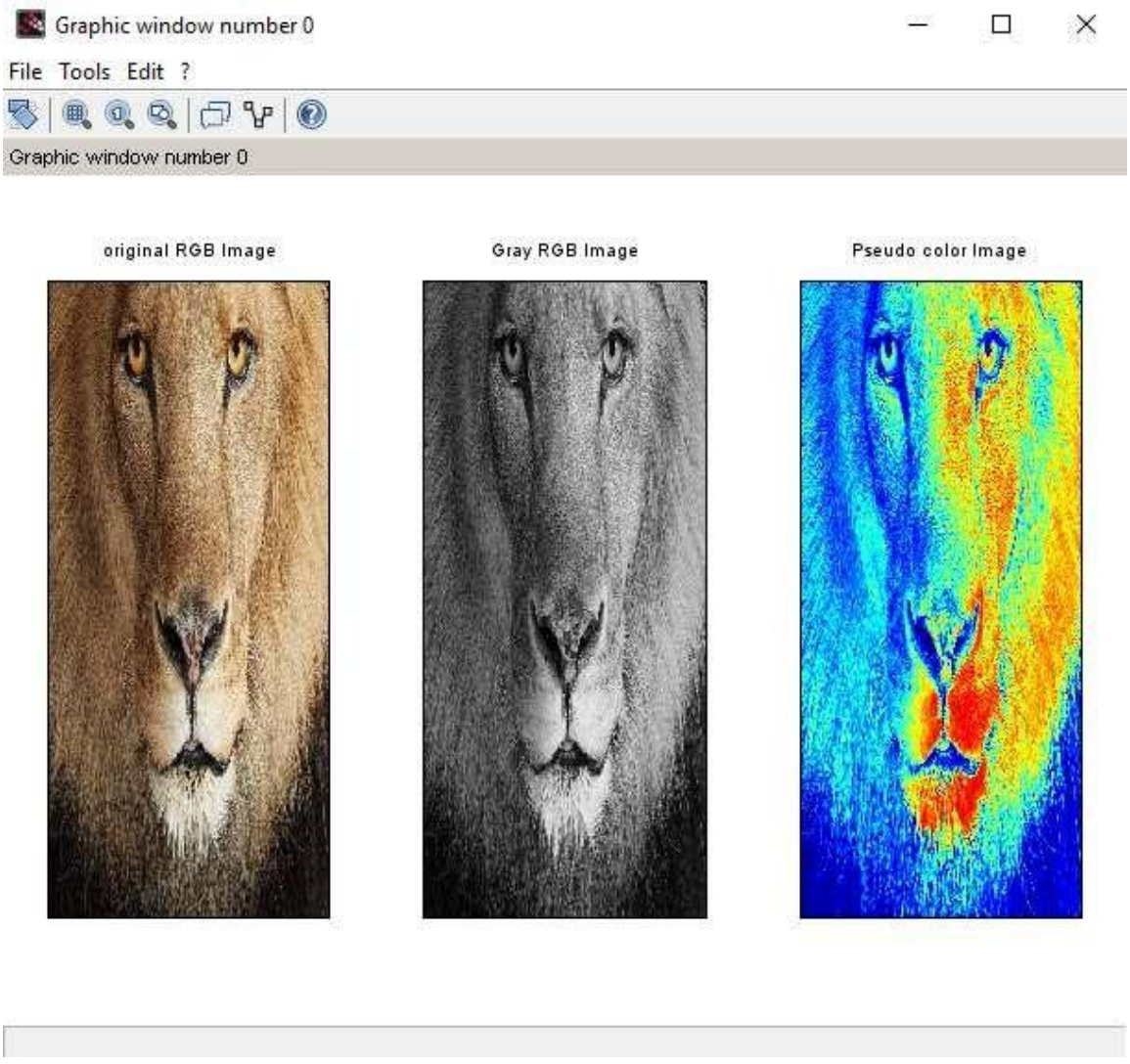
Part C:

Aim- Program to apply false colouring (pseudo) on a gray scale image.

Code-

```
clc;
rgb=imread('C:\Users\5itlab\Documents\lion.jpeg');
i=rgb2gray(rgb); subplot(1,3,1); imshow(rgb);
title('original RGB Image');
subplot(1,3,2); imshow(i);
title('Gray RGB Image');
subplot(1,3,3);
imshow(i,jetcolormap(256));
title('Pseudo color Image');
Histogram=imhist(i);
figure();
plot(0:255,'Histogram');
```

Output-



Practical No 6

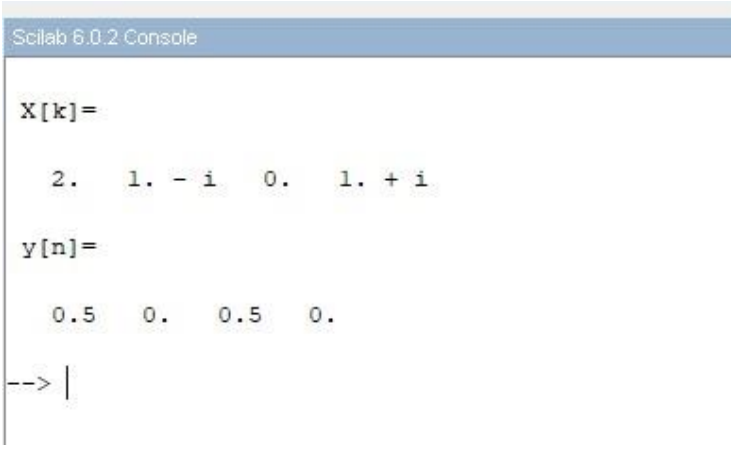
Aim- Fourier Related Transforms.

Program to compute Discrete Cosine Transforms.

Code-

```
clear; clc; close;
x=[1,1,0,0];
x=fft(x,-1);
y=[1,0,1,0];
y=fft(y,1);
disp(x,"X[k]=");
disp(y,"y[n]=");
```

Output-



Practical No 7

Aim- - Morphological Image Processing.

Part A:

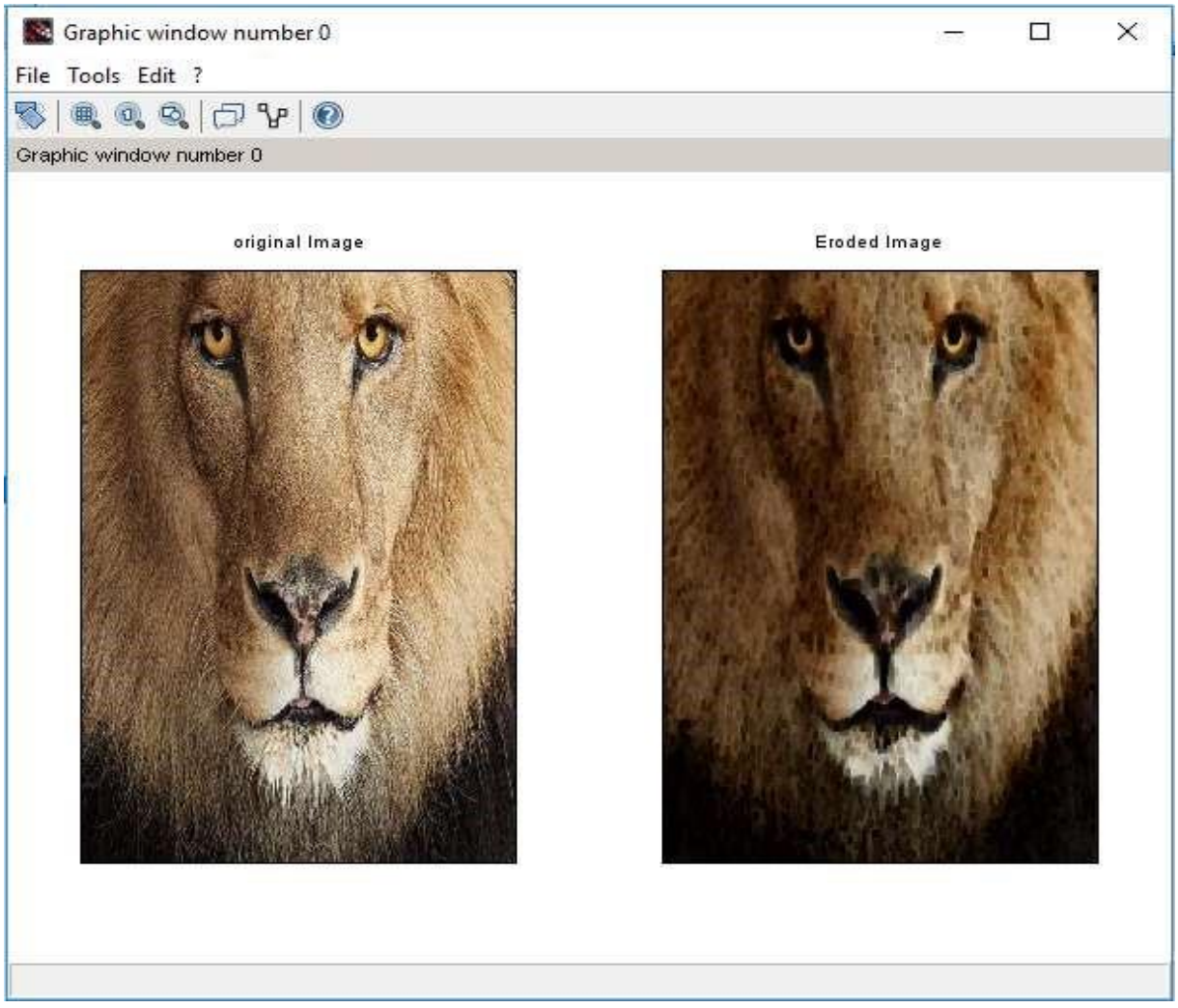
Aim- Program to apply erosion, dilation, opening, closing.

Erosion-

Code-

```
clear all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('rect',5,5);
erosion=imerode(a,se);
subplot(1,2,1); imshow(a);
title('original Image');
subplot(1,2,2);
imshow(erosion); title('Eroded
Image');
```

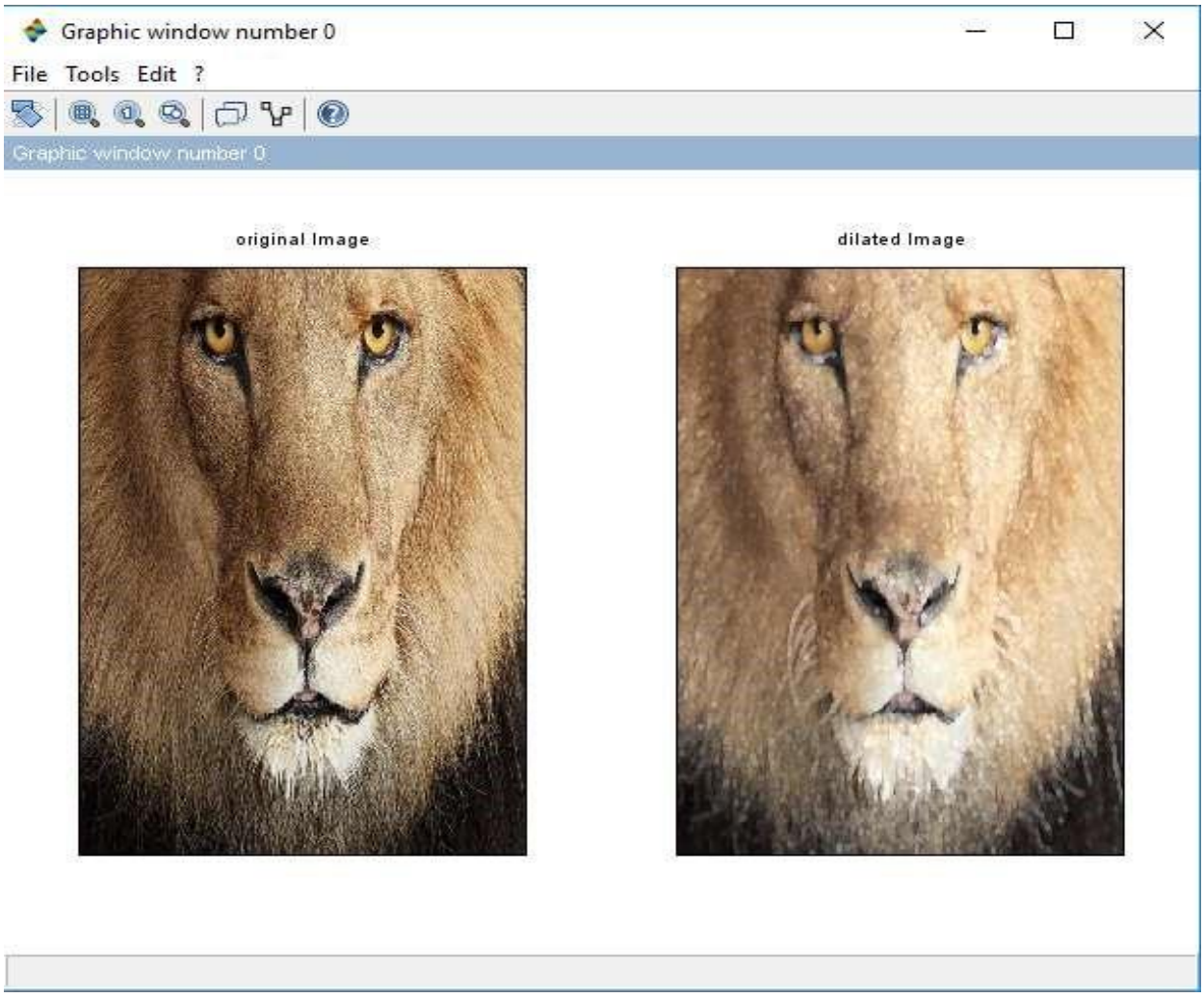
Output-



DilationCode-

```
clear all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('ellipse',5,5);
dilation=imdilate(a,se); subplot(1,2,1);
imshow(a);
title('original Image');
subplot(1,2,2);
imshow(dilation); title('dilated
Image');
```

Output-



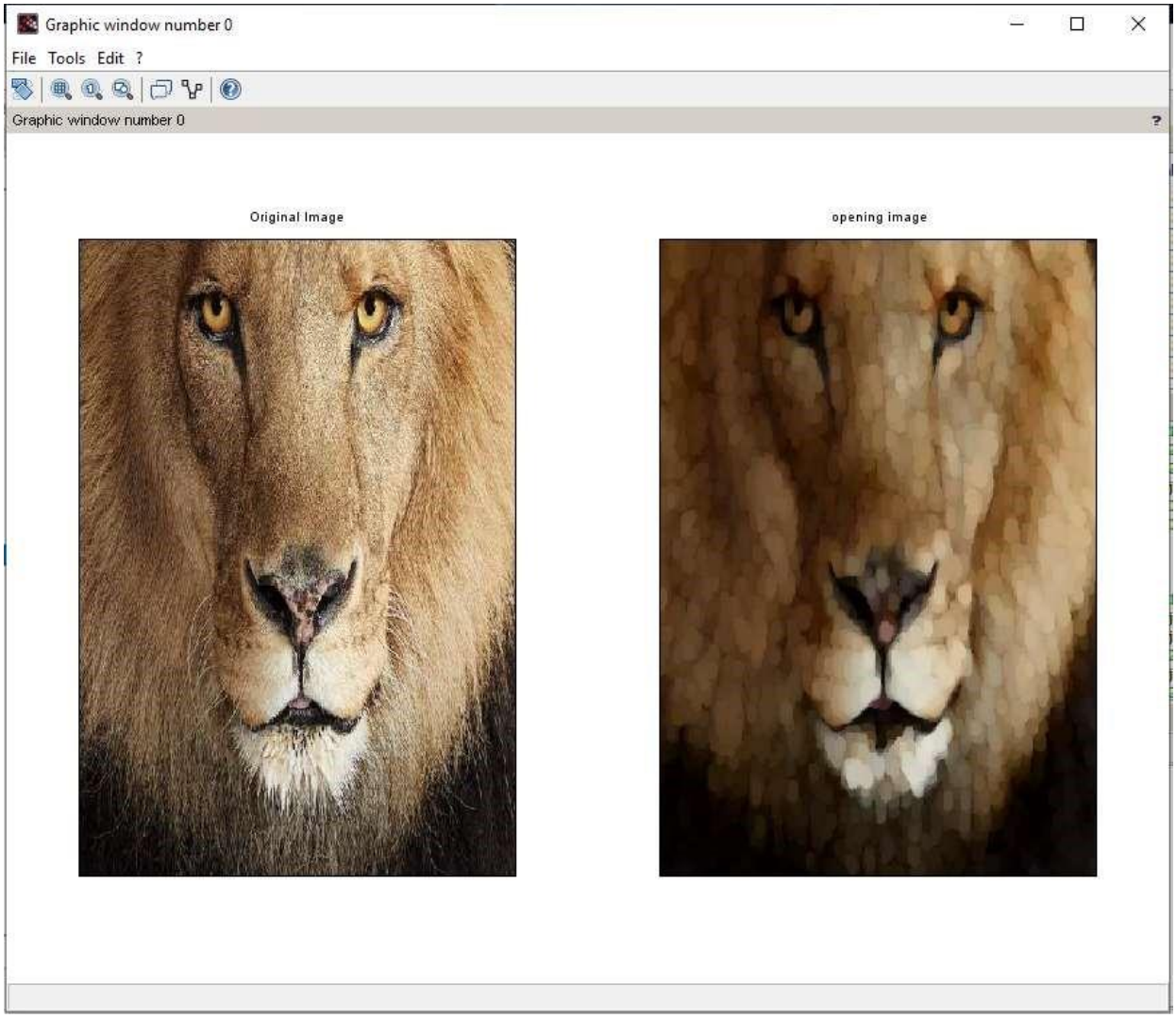
Opening

Code-

```
clc; clear
all;
```

```
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('ellipse',10,10);
erosion=imerode(a,se); opening=imdilate(erosion,se)
subplot(1,2,1);
imshow(a); title('Original
Image'); subplot(1,2,2);
imshow(opening);
title('opening image');
```

Output-



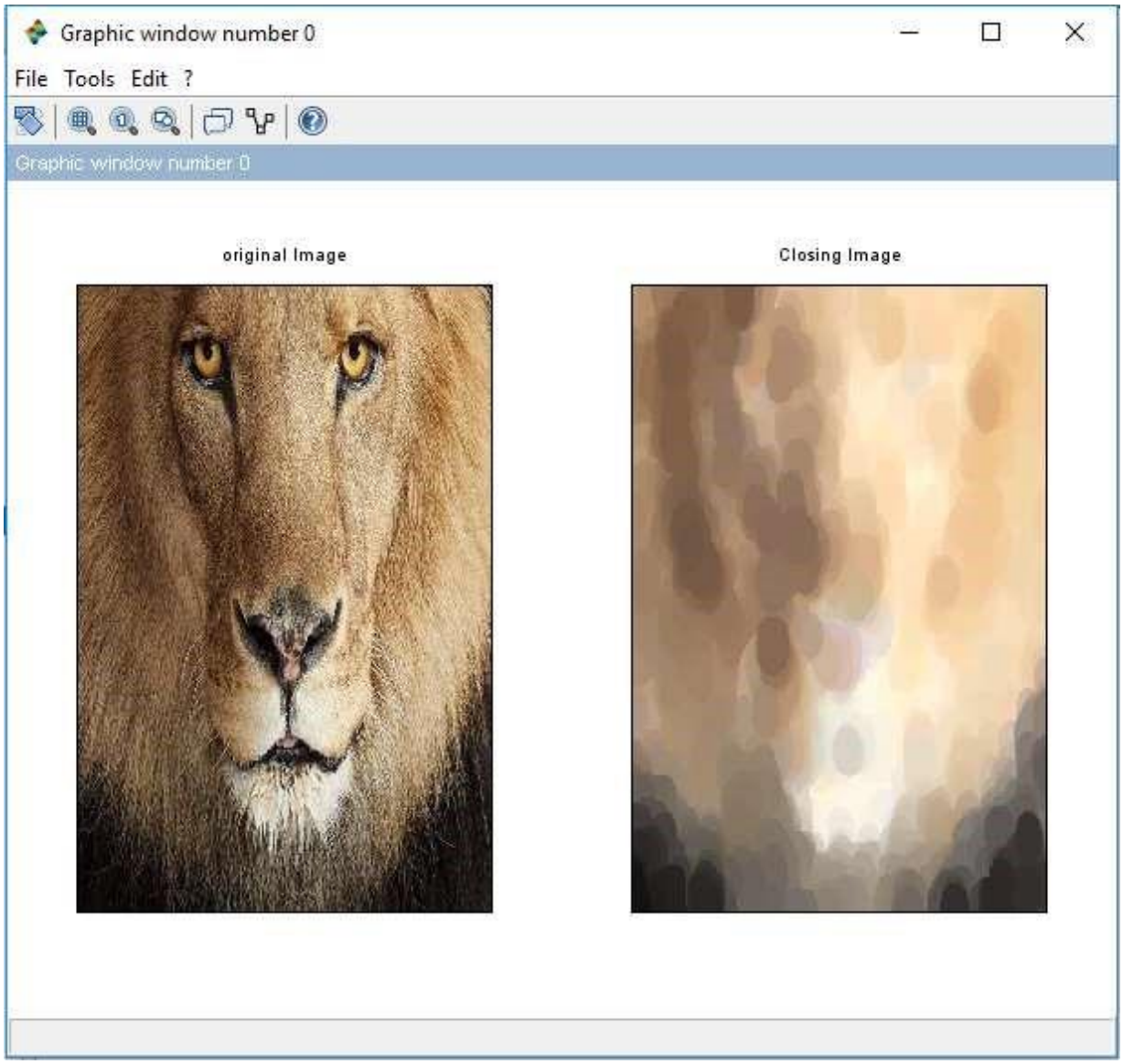
Closing

Code-

```
clear all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('ellipse',30,30);
dilate=imdilate(a,se); closing=imerode(dilate,se);
subplot(1,2,1); imshow(a);
title('original Image');
subplot(1,2,2);
```

```
imshow(closing); title('Closing
Image');
```

Output-



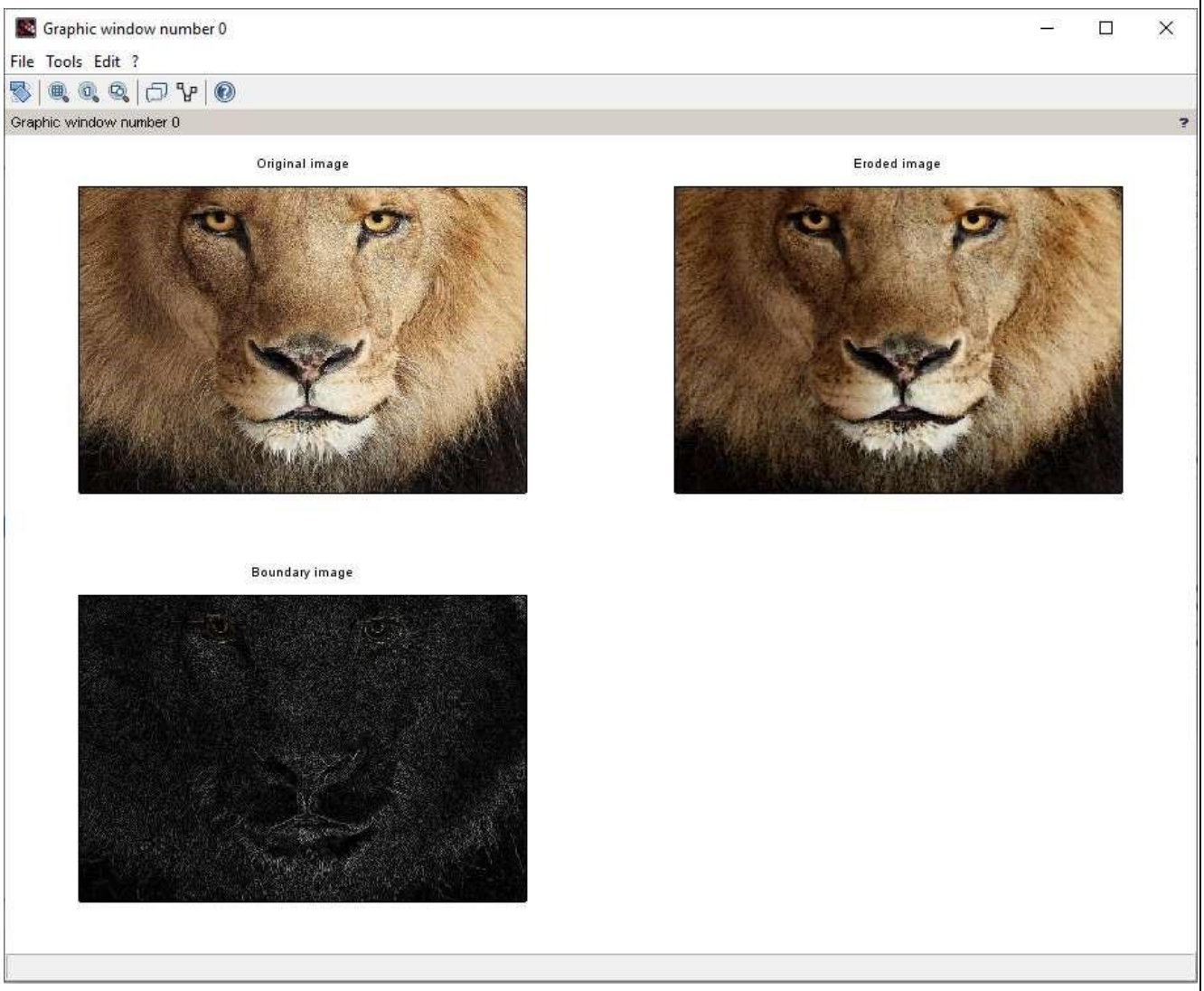
Part B**:**

Aim- Program for detecting boundary of an image.

Code-

```
clear all;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('ellipse',3,3);
erosion=imerode(a,se);
boundary=a-erosion
subplot(2,2,1); imshow(a);
title('Original image');
subplot(2,2,2);
imshow(erosion);
title('Eroded image');
subplot(2,2,3);
imshow(boundary);
title('Boundary image');
```

Output-



Part C**:**

Aim- Program to apply Hit-or-Miss transform.

Code-

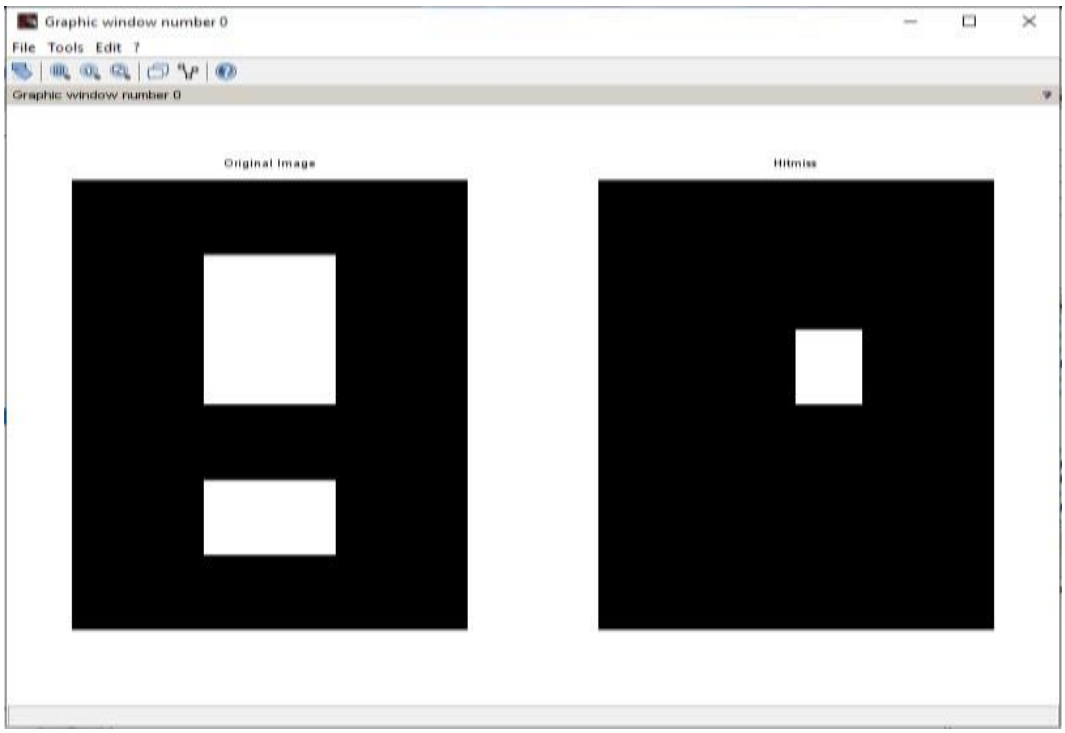
```
clc; clear all;
disp("Practical")
a=[0 0 0 0 0 0; 0
0 1 1 0 0;
0 0 1 1 0 0;
0 0 0 0 0 0;
0 0 1 1 0 0;
0 0 0 0 0 0];

a=im2bw(a,0.5);

se=[0 0 0 0; 0
1 1 0;
0 1 1 0;
0 0 0 0];

s2=imhitmiss(a,se);
subplot(1,2,1);
imshow(a); title('Original
Image') subplot(1,2,2);
imshow(s2);
title('Hitmiss')
```

Output-



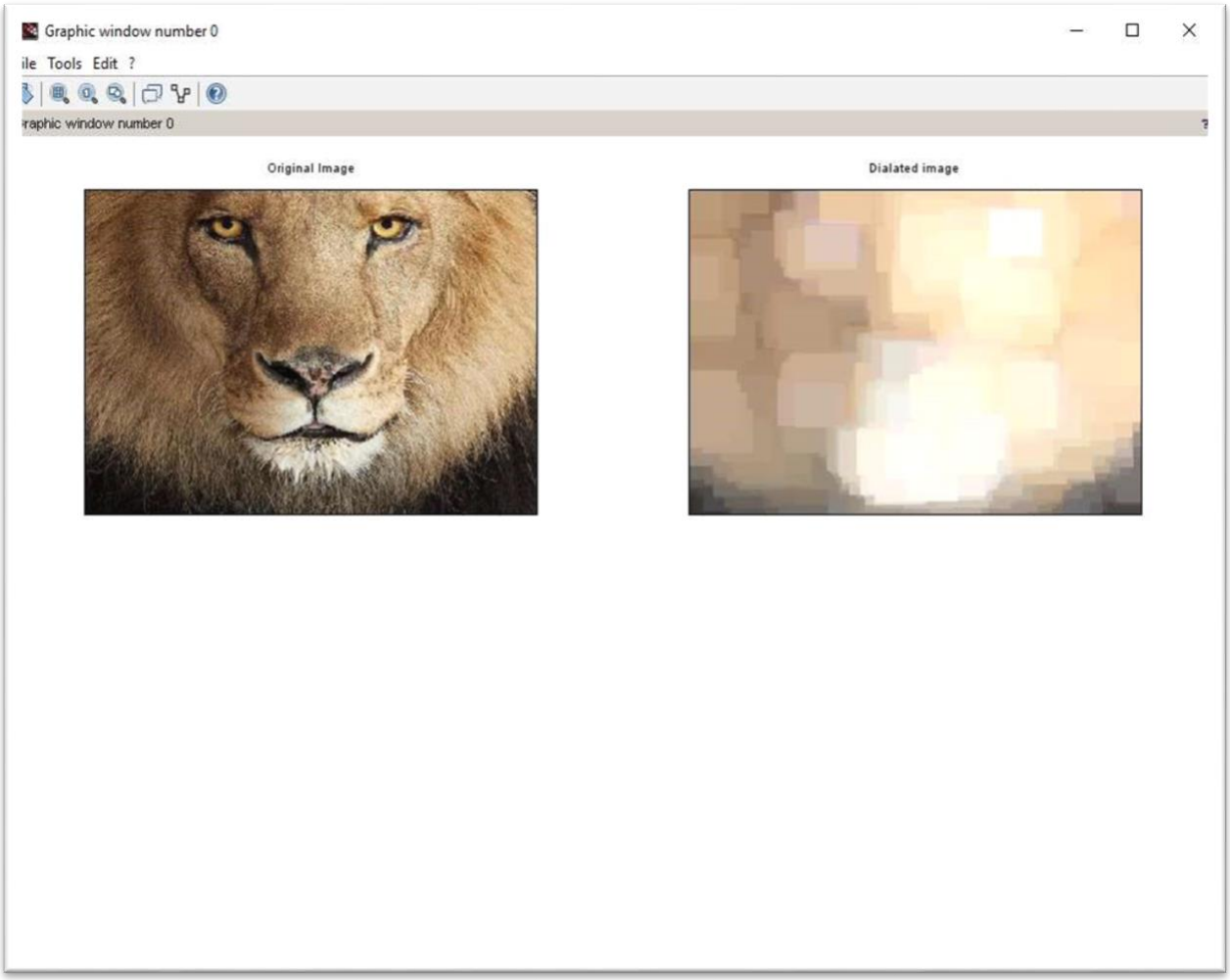
Part D**:**

Aim- Program to apply morphological gradient on an image.

Code-

```
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('rect',55,55);
dilation=imdilate(a,se);
erosion=imerode(a,se);
gradient=dilation-erosion subplot(2,2,1);
imshow(a);
title('Original Image');
subplot(2,2,2);
imshow(dilation);
title('Dialated image');
subplot(2,2,3);
```

Output-



Top-Hat/Bottom-hat Transformations.

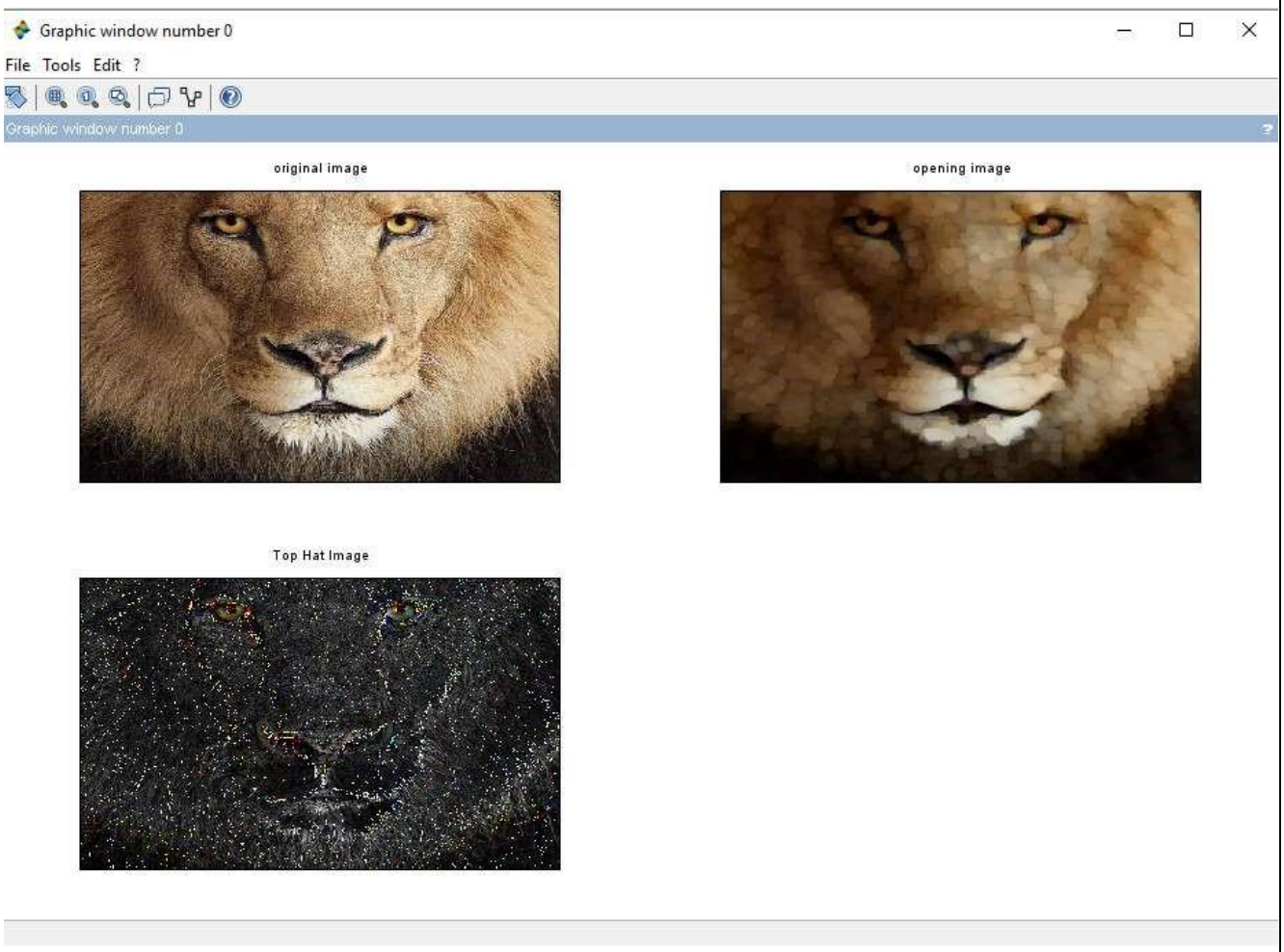
Top-Hat

Code-

```
clc;
a=imread('C:\Users\5itlab\Documents\lion.jpeg');
se=imcreatese('ellipse',10,10); erosion=imerode(a,se);
opening=imdilate(erosion,se)

top_hat=a-opening subplot(2,2,1);
imshow(a);
title('original image');
subplot(2,2,2);
imshow(opening);
title('opening image');
subplot(2,2,3);
imshow(top_hat); title('Top
Hat Image');
```

Output-

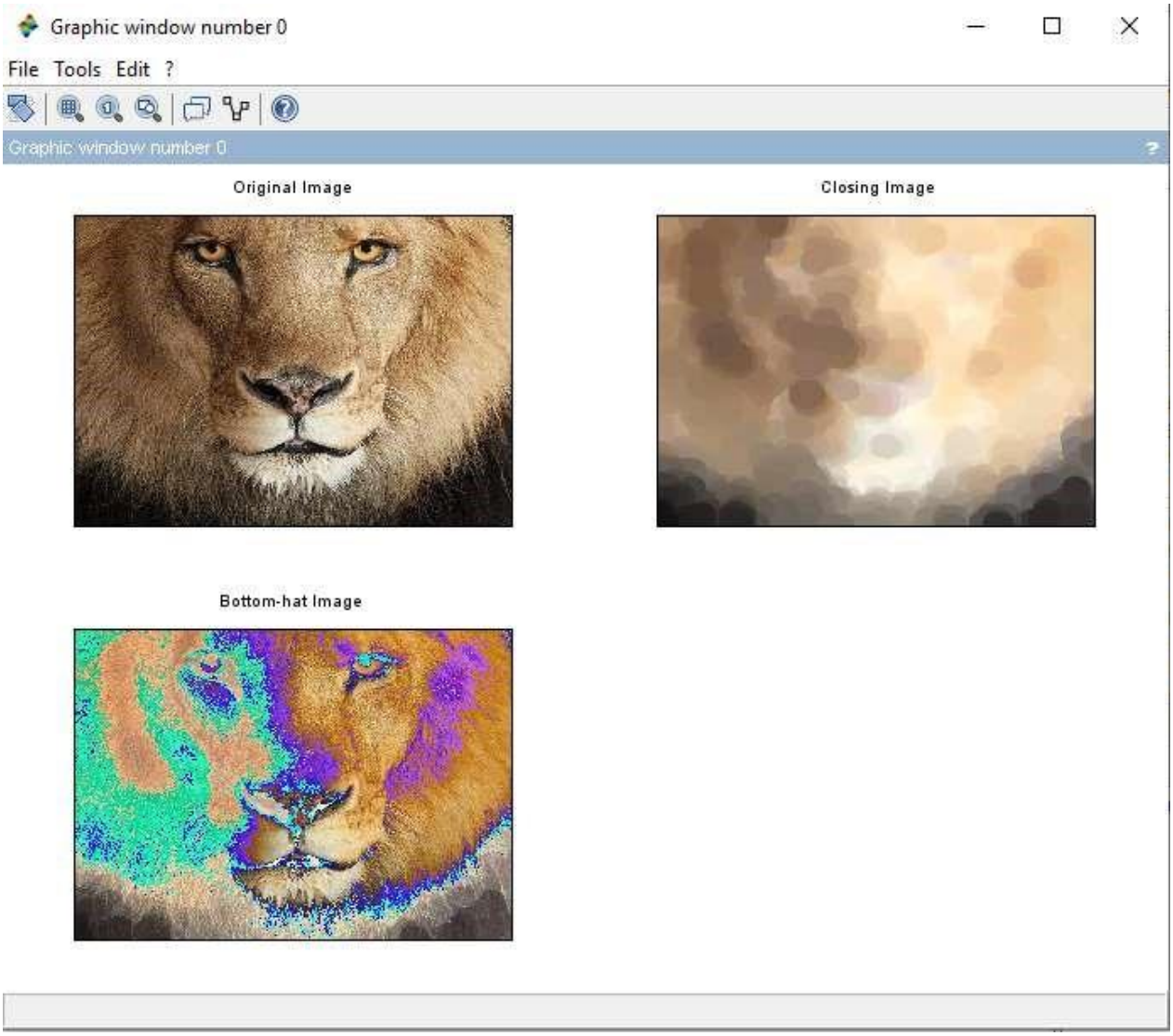


Bottom-Hat

Code-

```
clc;
a= imread('C:\Users\Documents\lion.jpeg');
se=imcreatese('ellipse',30,30);
dilate=imdilate(a,se);
closing=imerode(dilate,se)
bottom_hat=a+closing
subplot(2,2,1);  imshow(a);
title('Original Image');
subplot(2,2,2);
imshow(closing);
title('Closing Image');
subplot(2,2,3);
imshow(bottom_hat);
title('Bottom-hat Image');
```

Output-



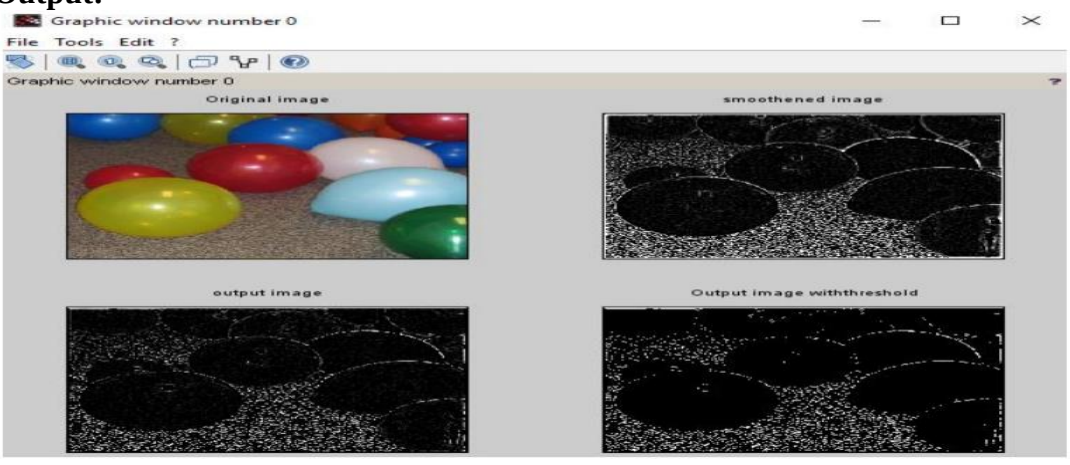
Practical No: 8

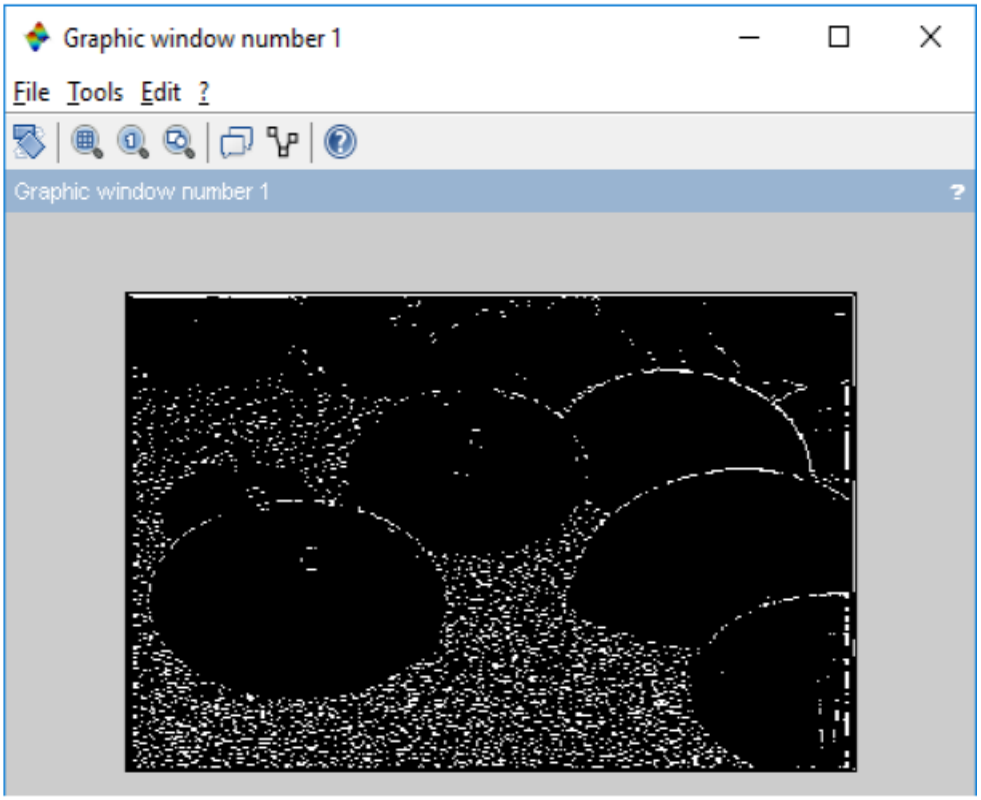
Aim: Part A. Image Segmentation. Program for Edge detection using Sobel.

Code:

```
clear all;
im= imread('C:\Users\5itlab\Documents\flags.jpg');
im=im2double(im);
gfilter= [ 0  0  1   0  0;
           0  1   2  1  0;
           1  2 -16 2  1;
           0  1   2  1  0;
           0  0   1  0  0  ];
          smim=conv2(im,gfilter)
          [rr,cc]=size(smim);
          zc=zeros([rr,cc]);
          for i= 2:rr-1
          for j= 2:cc-1
          if(smim(i,j)>0)
        if(smim(i,j+1)>=0 && smim(i,j-1)=0) || (smim(i,j+1)<0 && swim(I,j-1)>=0)
        zc(i,j)=smim(i,j+1);
       elseif(smim(i+1,j)>=0 && smim(i-1,j)<0) || (smim(i+1,j)<0 && swim(i-
1,j)>=0) zc(i,j)=smim(i,j+1);
     elseif(smim(i+1,j+1)>=0 && smim(i-1,j-1)<0) || (smim(i+1,j+1)<0 && smim(i-1,j-
1)>=0) zc(i,j)=smim(i,j+1);
   elseif(smim(i-1,j+1)>=0 && smim(i+1,j-1)<0) || (smim(i-1,j+1)<0 && smim(i+1,j-
  1)>=0) zc(i,j)=smim(i,j+1);
          end end
          end end
        otpt=im2uint8(zc);
        otptth = otpt>105;
        figure;
        subplot(2,2,1);imshow(im);title('Original image');
        subplot(2,2,2);imshow(smim);title('smoothened image');
        subplot(2,2,3);imshow(otpt);title('output image');
        subplot(2,2,4);imshow(otptth);
        title('Output image withthreshold');
        figure,imshow(otptth);
```

Output:





Part B. Image Segmentation. Program for Edge detection using Prewitt.

Code:

```
clc;
clear all;
// im= imread('C:\Users\5itlab\Documents\flags.jpg');
im = [ 0 0 0 0 0 0 0 0 0;
       0 0 0 0 1 0 0 0 0;
       0 0 0 1 1 1 0 0 0;
       0 0 1 1 1 1 1 0 0;
       0 1 1 1 1 1 1 1 0;
       0 0 0 0 0 0 0 0 0];
im=im2double(im);
gfilter= [ 0  0  1   0  0;
           0  1   2  1  0;
           1  2 -16 2  1;
           0  1   2  1  0;
           0  0   1  0  0  ];
smim=conv2(im,gfilter) disp(smim)
[rr,cc]=size(smim); zc=zeros([rr,cc]);
disp([rr,cc])
for i= 2:rr-1
for j=2:cc-1
if(smim(i,j)>0)
if(smim(i,j+1)>=0 && smim(i,j-1)<0) || (smim(i,j+1)<0 && smim(i,j-1)>=0)
zc(i,j)=smim(i,j+1);
elseif(smim(i+1,j)>=0 && smim(i-1,j)<0) || (smim(i+1,j)<0 && smim(i-1,j)>=0)
zc(i,j)=smim(i,j+1);
```

```
  elseif(smim(i+1,j+1)>=0 && smim(i-1,j-1)<0) || (smim(i+1,j+1)<0 && smim(i-1,j-1)>=0) zc(i,j)=smim(i,j+1);
   elseif(smim(i-1,j+1)>=0 && smim(i+1,j-1)<0) || (smim(i-1,j+1)<0 && smim(i+1,j-1)>=0) zc(i,j)=smim(i,j+1);
                    end end end
          otpt=im2uint8(zc);
          otptth = otpt>105;
          figure;
          subplot(2,2,1);    imshow(im);    title('Original image');
          subplot(2,2,2);    imshow(smim); title('smoothened image');
          subplot(2,2,3);    imshow(otpt);   title('output image');
          subplot(2,2,4);    imshow(otptth); title('Output image withthreshold');
          //% final result
          figure,imshow(otptth);
```

Output :