PRACTICAL REPORT
ON
IMAGE PROCCESING

SUBMITTED BY
SHAHEEN MOHSIN ANSARI

ROLL NO: 01

SUBMITTED TO
Mrs.Sangeeta Jena

MSc. (INFORMATION TECHNOLOGY) SEM -II
2021 – 2022

CONDUCTED AT
CHIKITSAK SAMUHA'S
S. S. & L.S. PATKAR COLLEGE OF ARTS & SCIENCE
AND
V. P. VARDE COLLEGE OF COMMERCE & ECONOMICS
Affiliated to University of Mumbai
GOREGAON (W). MUMBAI -400062

INDEX

I		Aim
Practical 1	a	Program to calculate number of samples required for an image.
	b	Program to study the effects of reducing the spatial resolution of a digital image.
	c	Program to study the effects of varying the number of intensity levels in a digital image.
Practical 2	A	Basic Intensity Transformation functions
		i. Program to perform Image negation. ii. Program to perform threshold on an image. iii. Program to perform Log transformation. iv. Power-law transformations. v. Piecewise linear transformations. a. Contrast Stretching. b. Gray-level slicing with and without background. c. Bit-plane slicing.
	B	Program to plot the histogram of an image and categories.
	C	Write a program to perform convolution and correlation.
	D	Write a program to apply smoothing and sharpening filters on grayscale and color images. a) Low Pass. b) High Pass. Note: Use all kernels mentioned in the reference book.
Practical 3		Filtering in Frequency Domain.
		a) Program to apply Discrete Fourier Transform on an image.
		b) Program to apply Low pass and High pass filters in frequency domain.
		c) Program for butterworth and gaussion filter in frequency domain.
Practical 4		Image Denoising.
		i . Program to denoise using spatial mean, median.
		ii. Program for Image Weiner filters.
Practical 5		Color Image Processing.
		i. Program to read a color image and segment into RGB planes, histogram of color image.
		ii. Program for converting from one color model to another model.
		iii. Program to apply false colouring (pseudo) on a gray scale image.

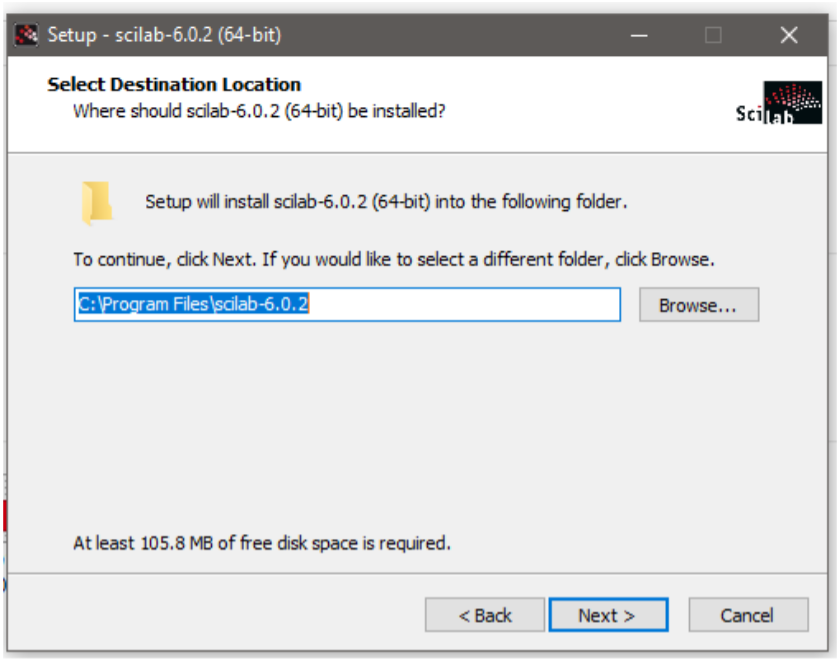
Practical 6	Fourier Related Transforms.
	Program to compute Discrete Cosine Transforms.
Practical 7	Morphological Image Processing.
	i. Program to apply erosion, dilation, opening, closing.
	ii. Program for detecting boundary of an image.
	iii. Program to apply Hit-or-Miss transform.
	iv. Program to apply morphological gradient on an image.
	v. Program to apply Top-Hat/Bottom-hat Transformations.
Practical 8.	Image Segmentation.
	Program for Edge detection using Sobel, Prewitt.

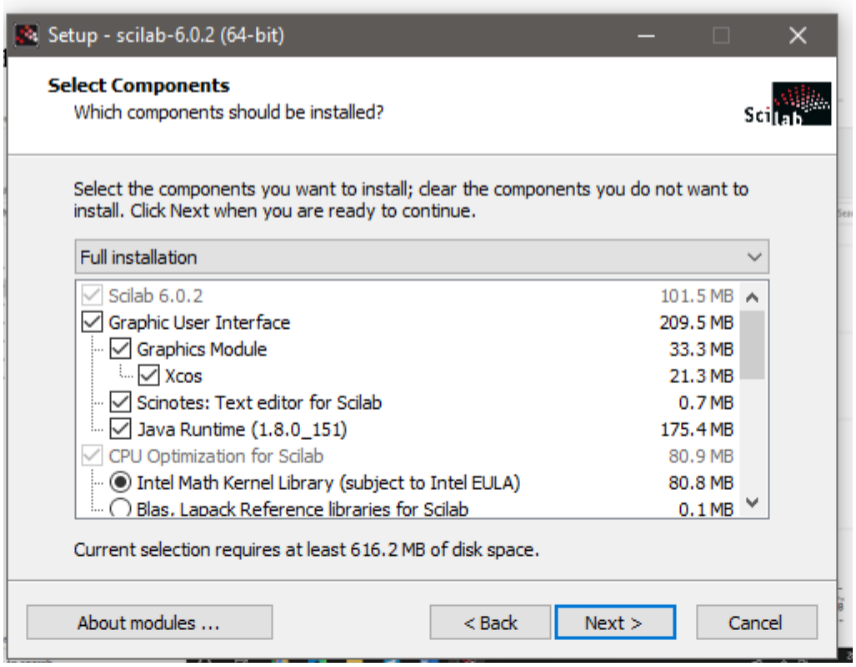
Practical No 1

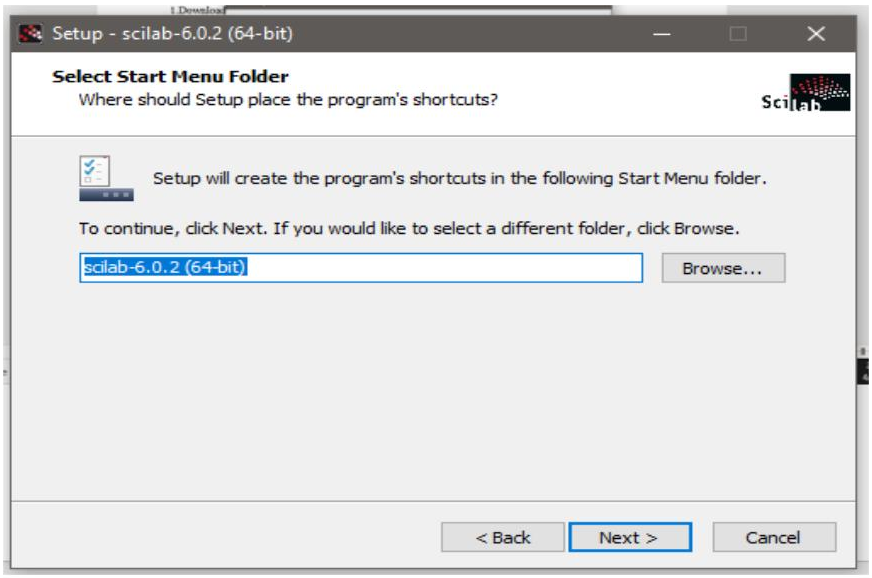
Required Software for Image Processing-

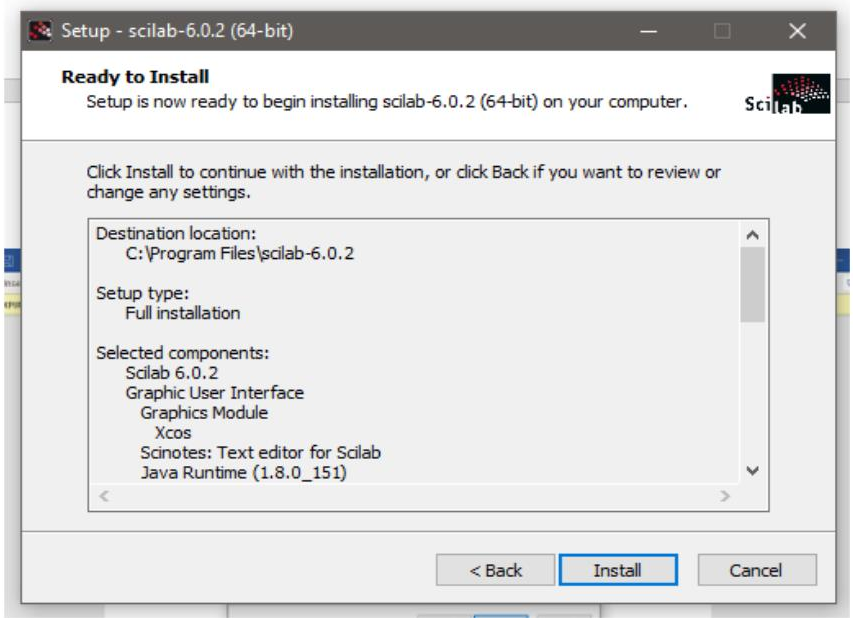
Steps:

1. Download and Install Scilab 6.0.2



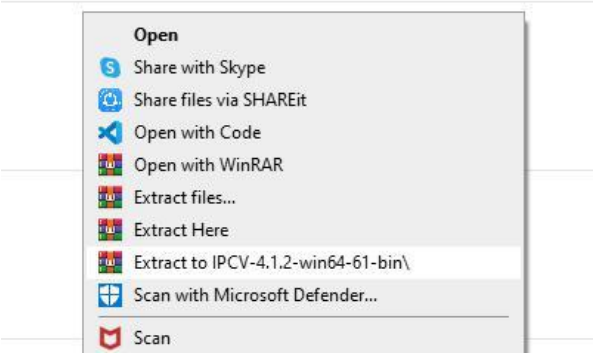




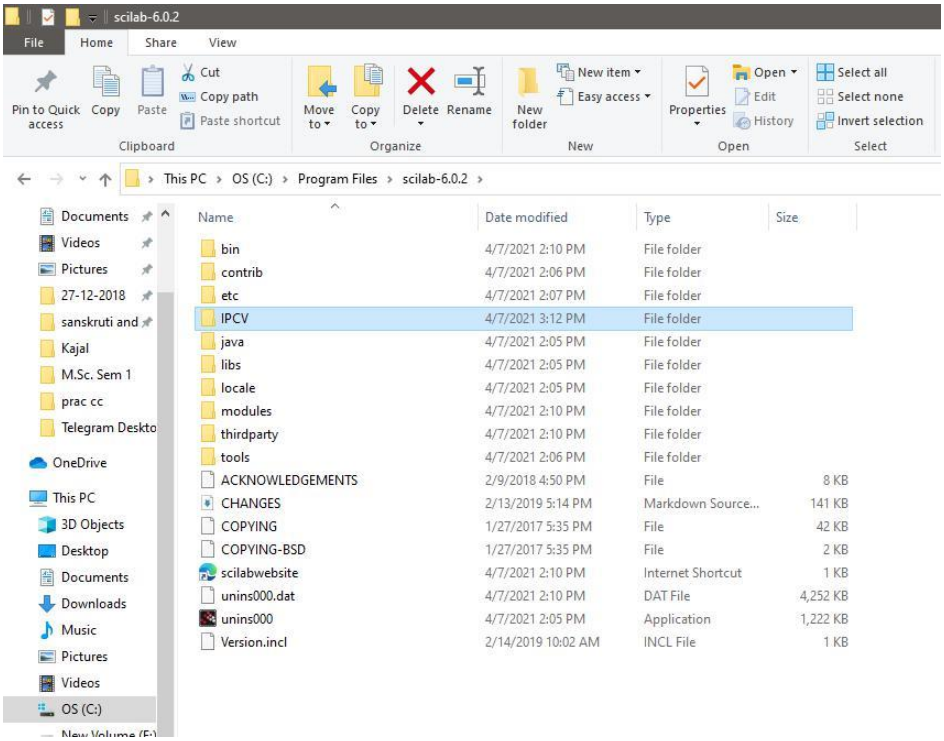


2. Download IPCV zip file

Extract IPCV folder => Paste in scilab folder

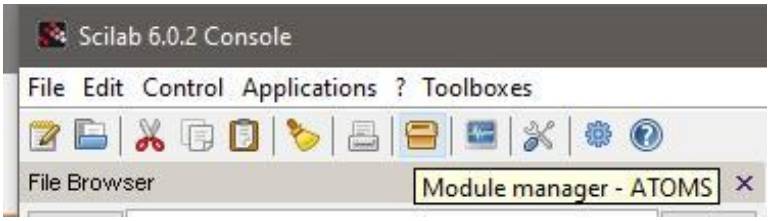


3. Paste IPCV folder in C:\Program Files\scilab-6.0.2

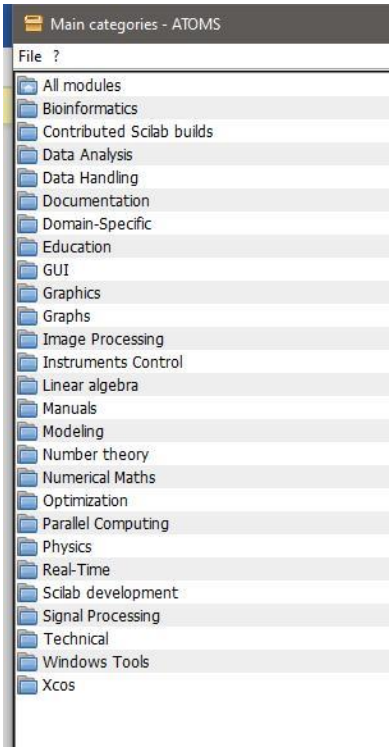


4. Open scilab 6.0.2. Go to start → Search scilab → Open.

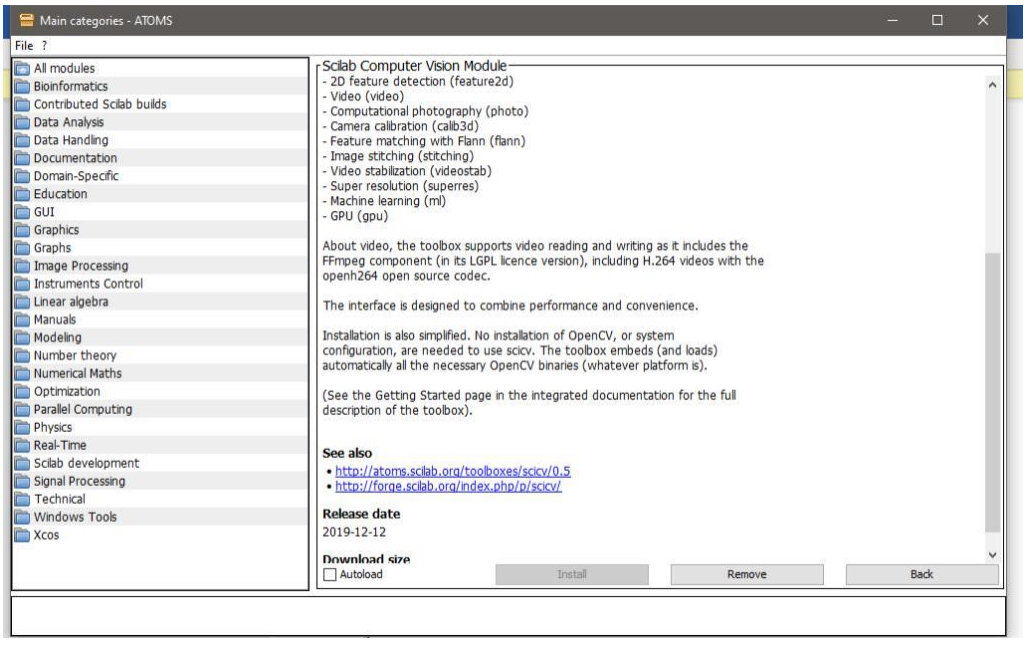
5. Click on Modul Manager.



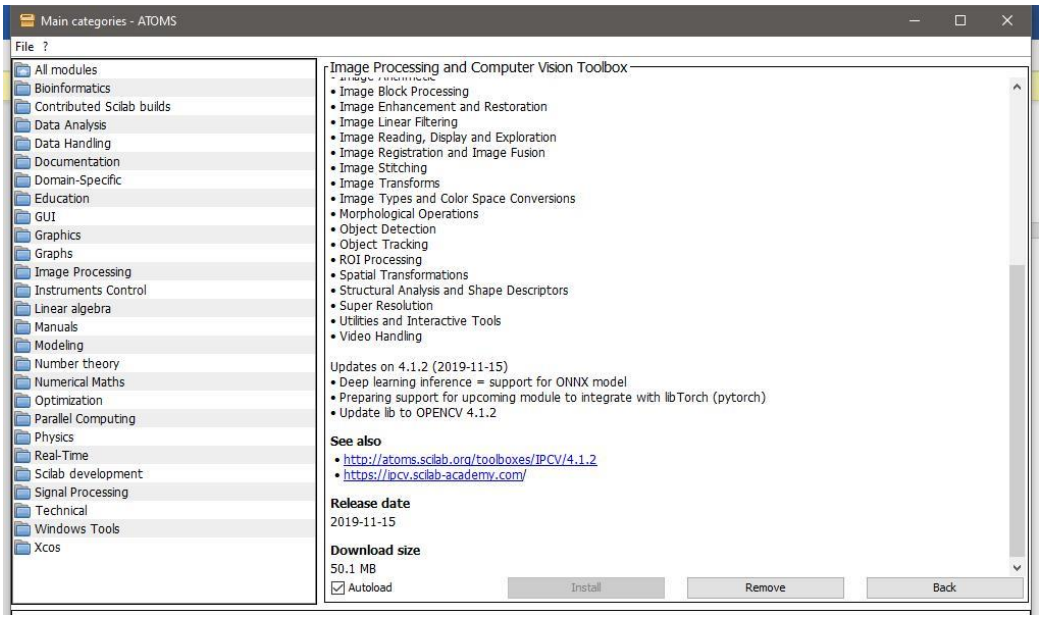
6. Click on Image Processing.



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:

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

Code-

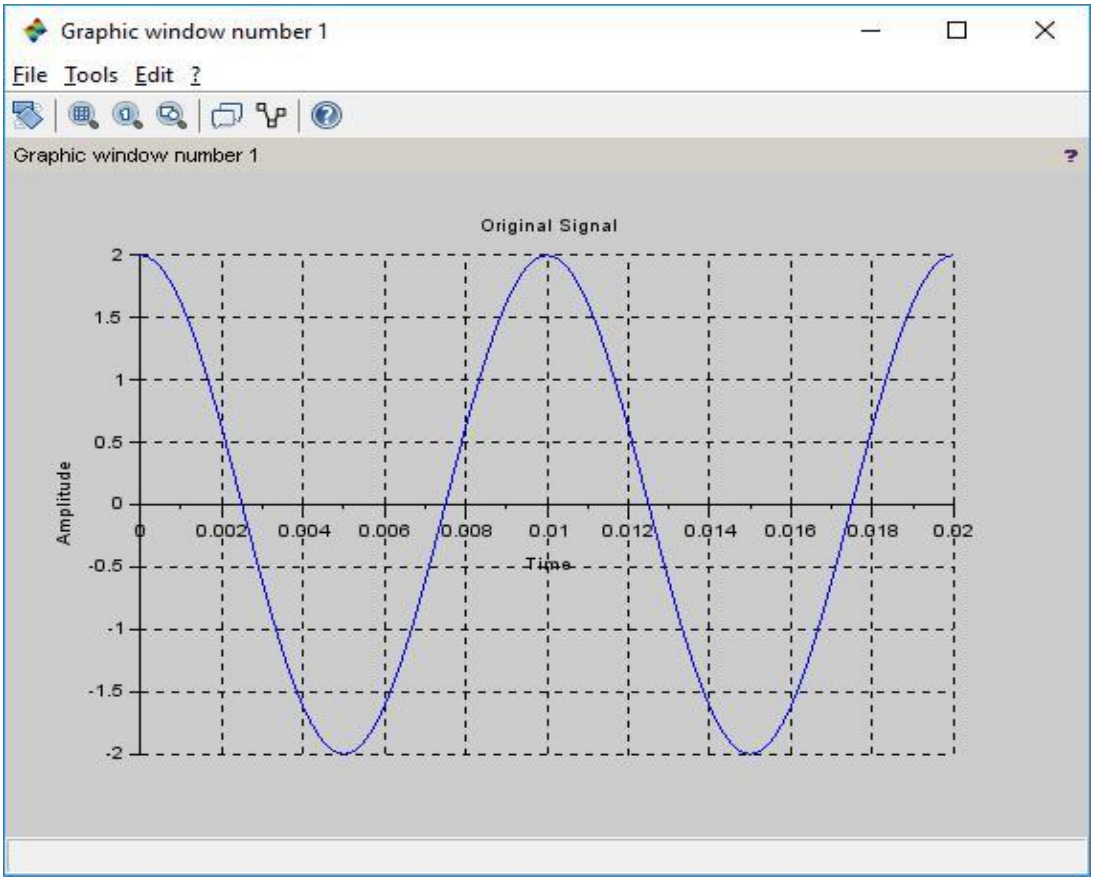
```
clc;
clear all;
fm=input('Input signal frequency');
k=input('no. of cycles');
A=input('Enter amplitude signal');

tm=0:1/(fm*fm):k/fm;
x=A*cos(2*%pi*fm*tm);
figure(1);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot(tm,x);
```

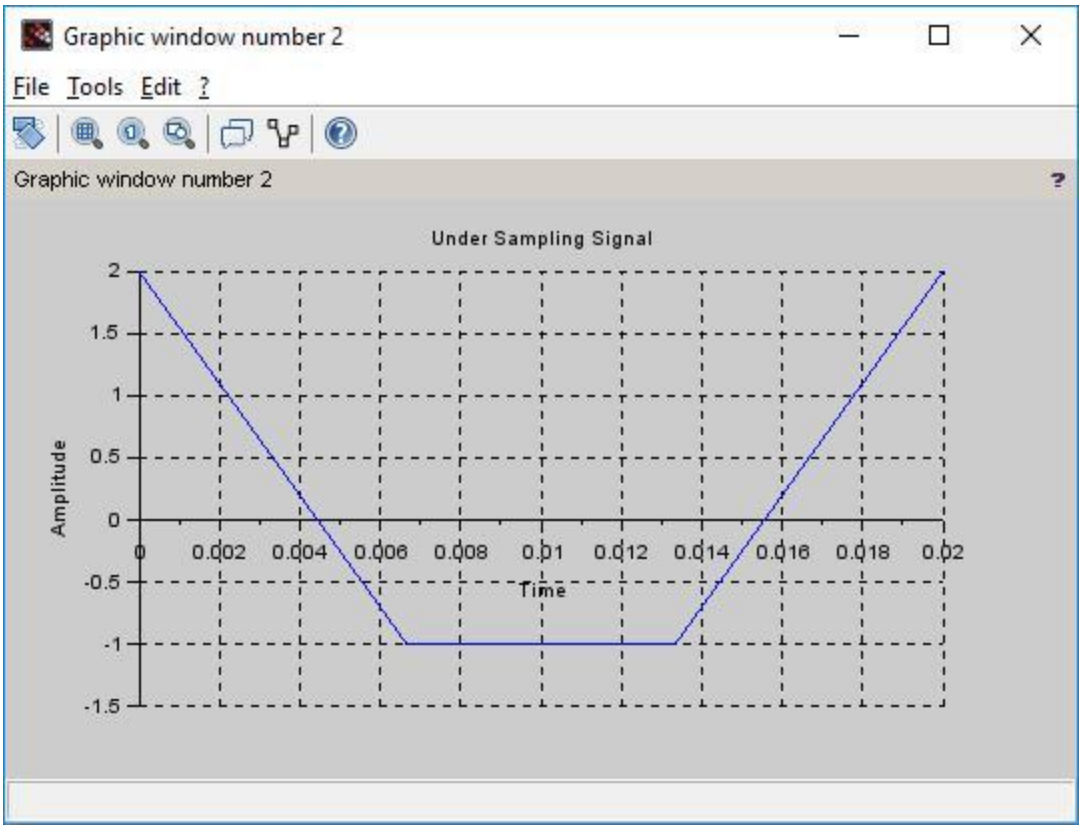
```
title('Original Signal');
xlabel('Time');
ylabel('Amplitude');
xgrid(1)

fnsq=2*fm;
fs=(3/4)*fnsq;
n=0:1/fs:k/fm;
x=A*cos(2*%pi*fm*n);
figure(2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot(n,x);
title('Under Sampling Signal');
xlabel('Time');
ylabel('Amplitude');
xgrid(1)

fnyq=2*fm;
fo=fnyq;
o=0:1/fo:k/fm;
x=A*cos(2*%pi*fm*o);
figure(3);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot(o,x);
title('nyquist Sampling');
xlabel('Time');
ylabel('Amplitude');
xgrid(1)

fo=10*fnyq;
o=0:1/fo:k/fm;
x=A*cos(2*%pi*fm*o);
figure(4);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d3('gnn',o,x);
plot(o,x,'r');
title('Over Sampling signal');
xlabel('Time');
ylabel('Amplitude');
xgrid(1)
```

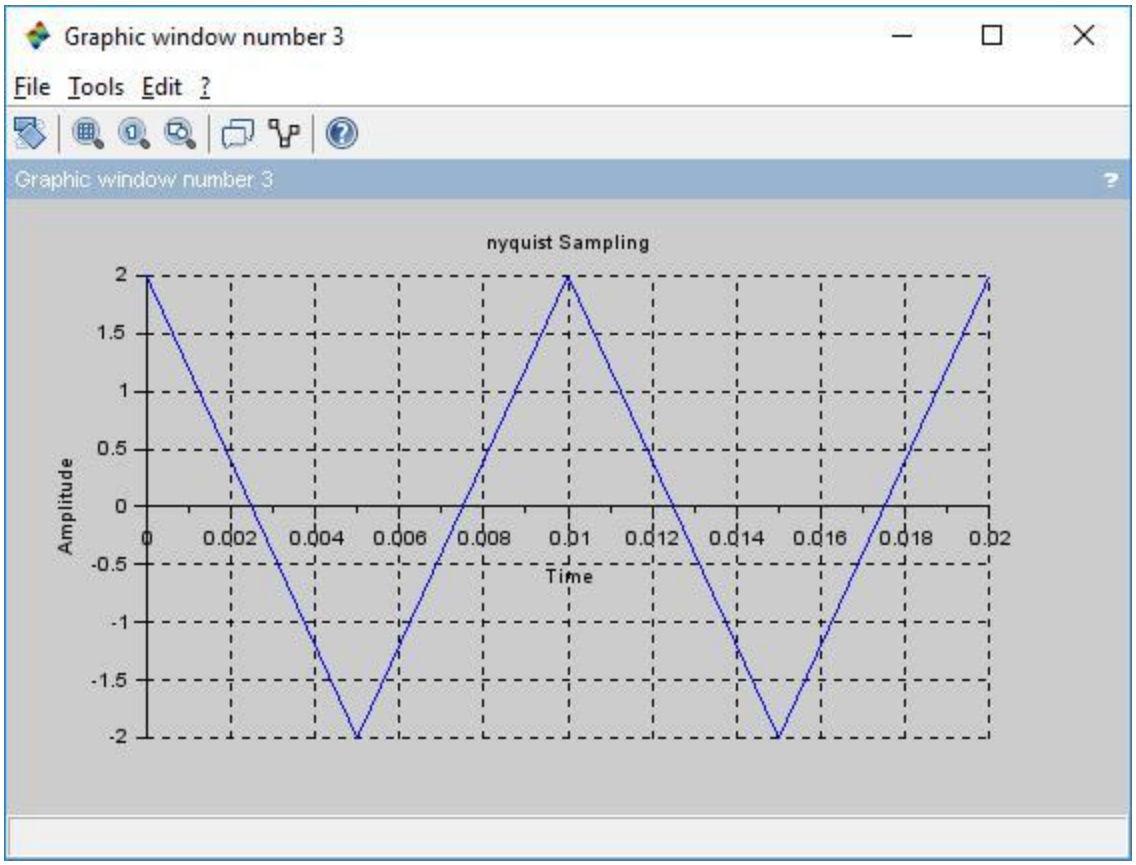
Input- (Example of Input-100,2,2)

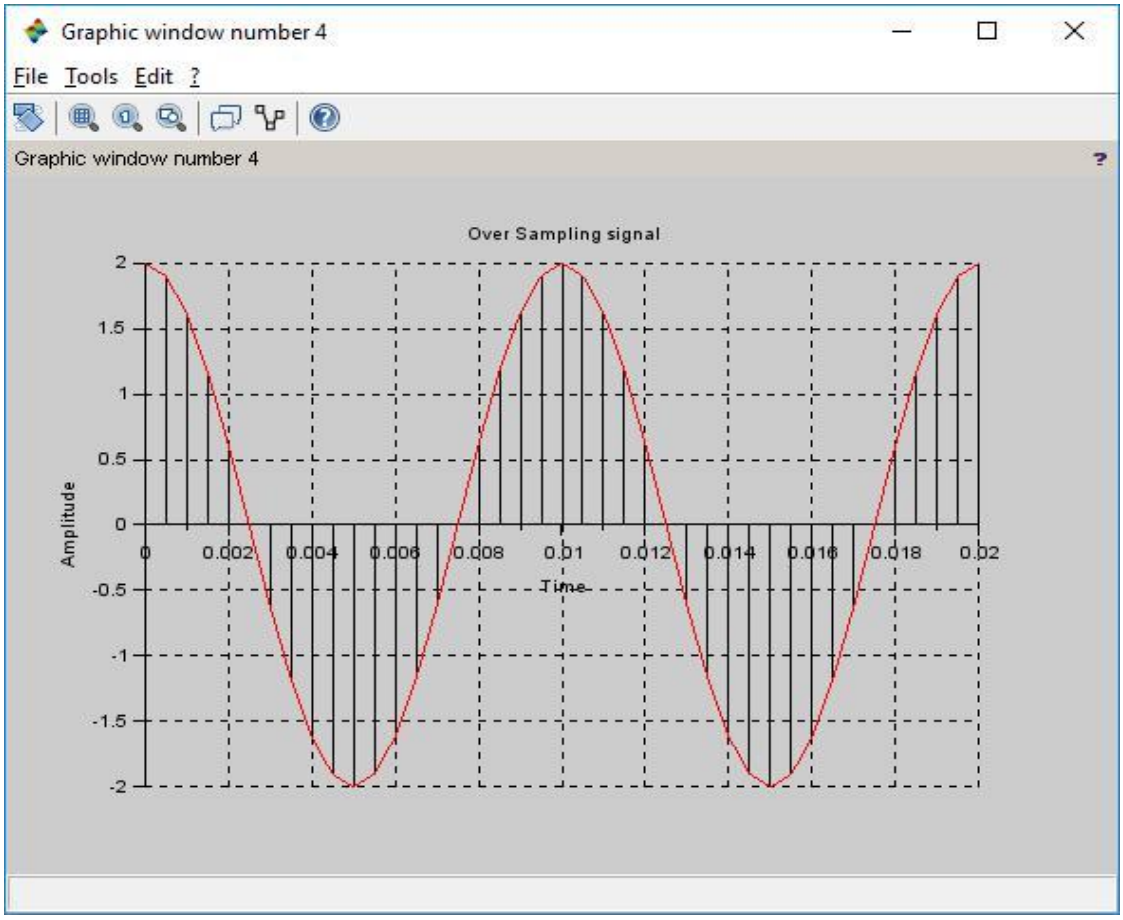
```
Input signal frequency100

no. of cycles2

Enter amplitude signal2


-->
```

Output-




Graphic window number 2
File Tools Edit ?
Graphic window number 2
Under Sampling Signal
Amplitude
Time
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02


Graphic window number 3
File Tools Edit ?
Graphic window number 3
nyquist Sampling
Amplitude
Time
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02



Part B:

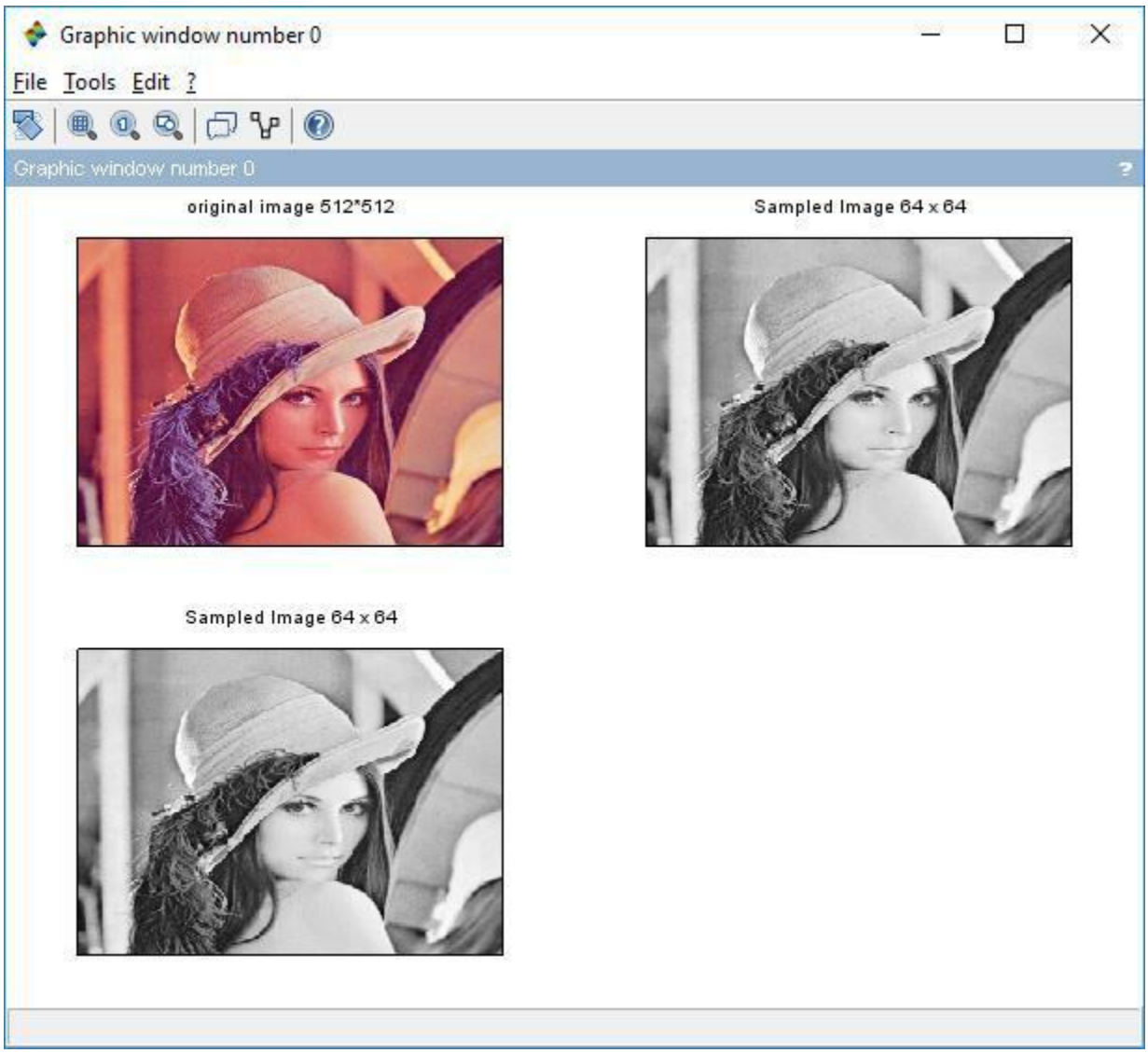
Aim- Program to study the effects of reducing the spatial resolution of a digital image.

Code-

```
clc;
clear all;

Img=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\lena.png');
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:

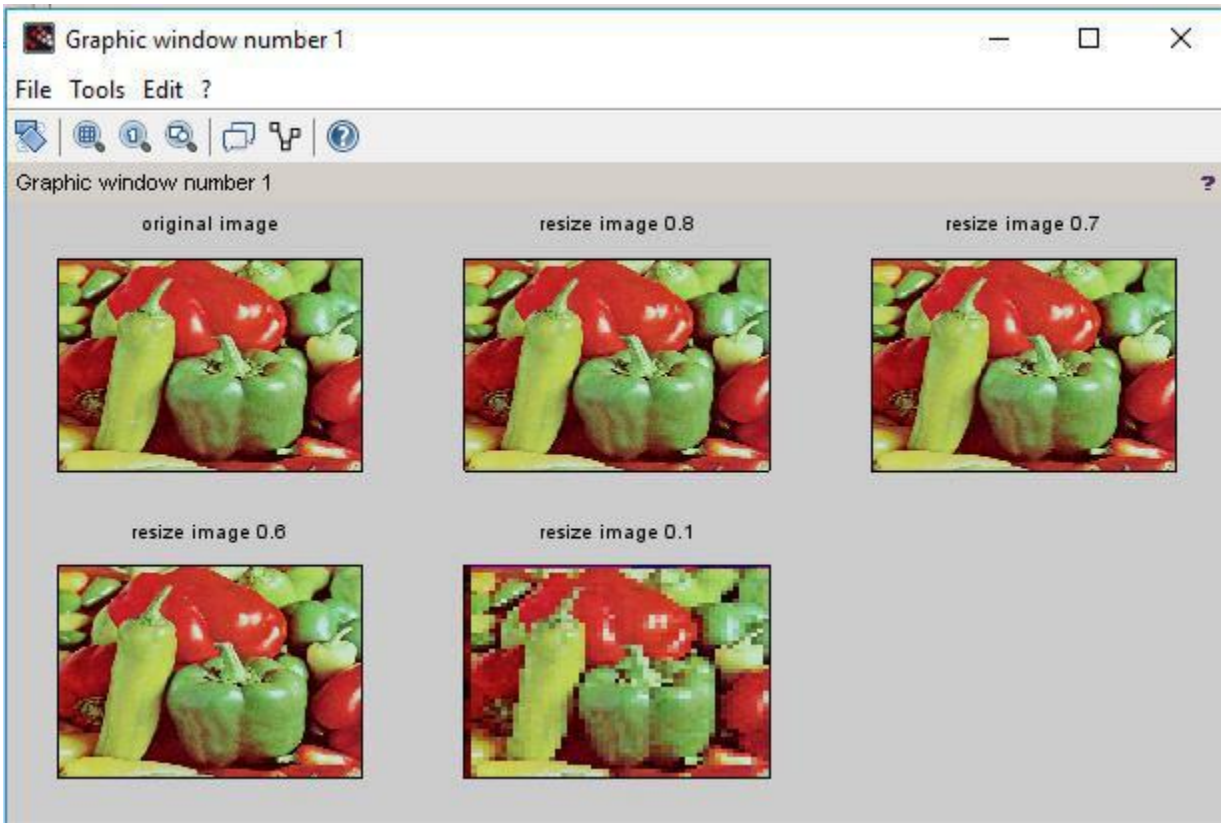
Aim- Program to study the effects of varying the number of intensity levels in a digital image.

Code-

```
clc;
clear all;
figure(1)
///cheak board
subplot(3,3,1);
i=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\peppers.png'); 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.

Code-

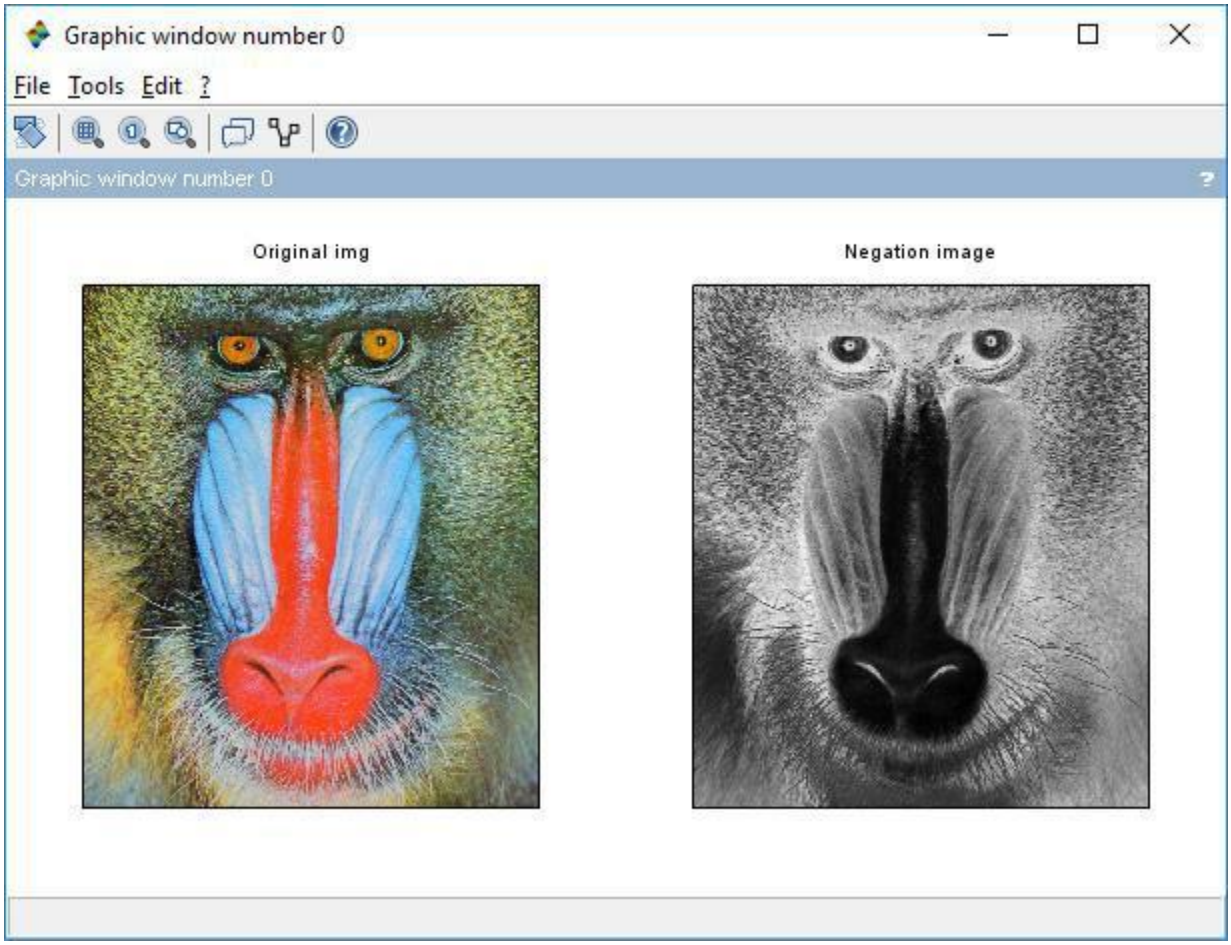
1. Image negation.

Code-

```
clc;
clear all;
a = imread('C:\Program Files\scilab-
6.0.2\IPCV\images\baboon.png'); subplot(1,2,1);
imshow(a)
title('Original img')

[m,n] = size(a);
for i = 1:m
   for j = 1:n
      c(i,j) = 255 - a(i,j)
   end
```

```
end

subplot(1,2,2);
imshow(c)
title('Negation image')
```

Output-

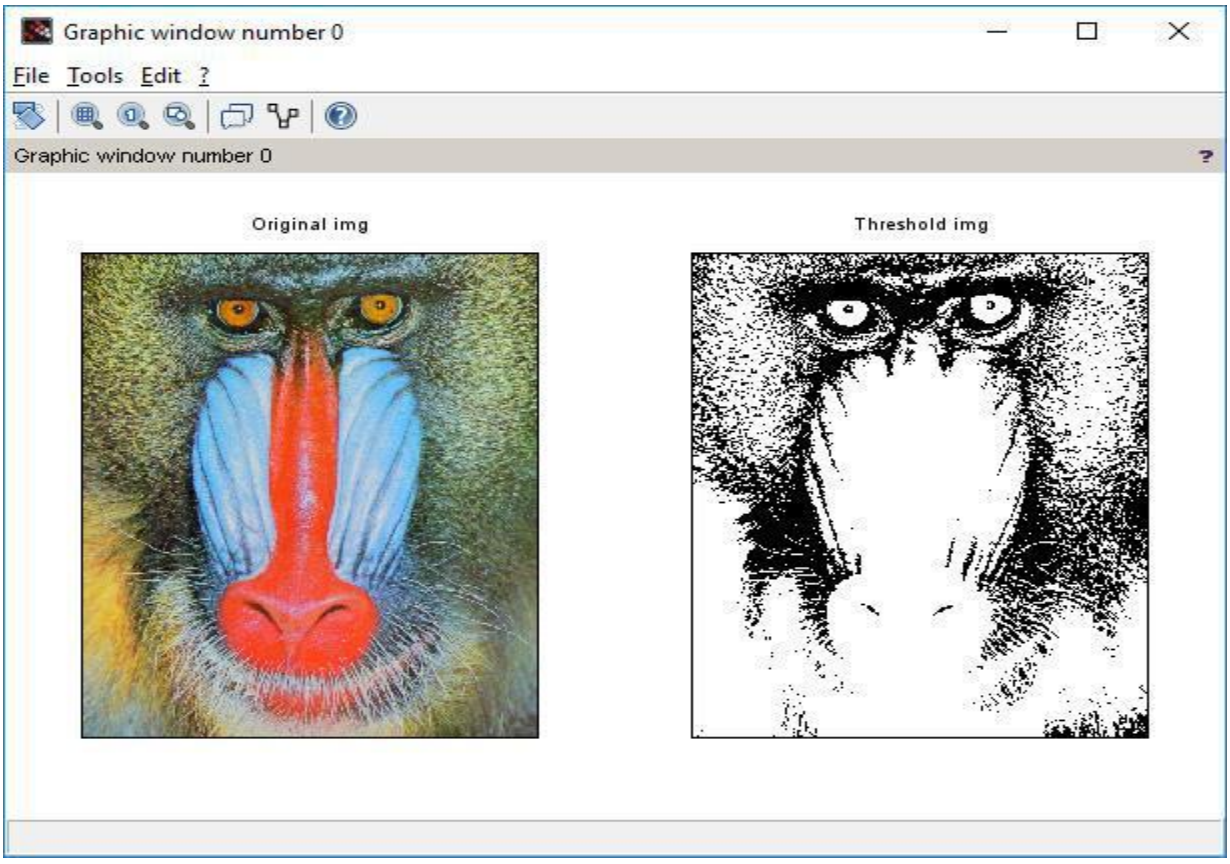


2. Threshold on an image. Code-

```
clc;
clear all;
clc;
clear all;
a = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\baboon.png'); b=double(a);
```

```
subplot(1,2,1);
imshow(a);
title('Original img');

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('Threshold img');
```

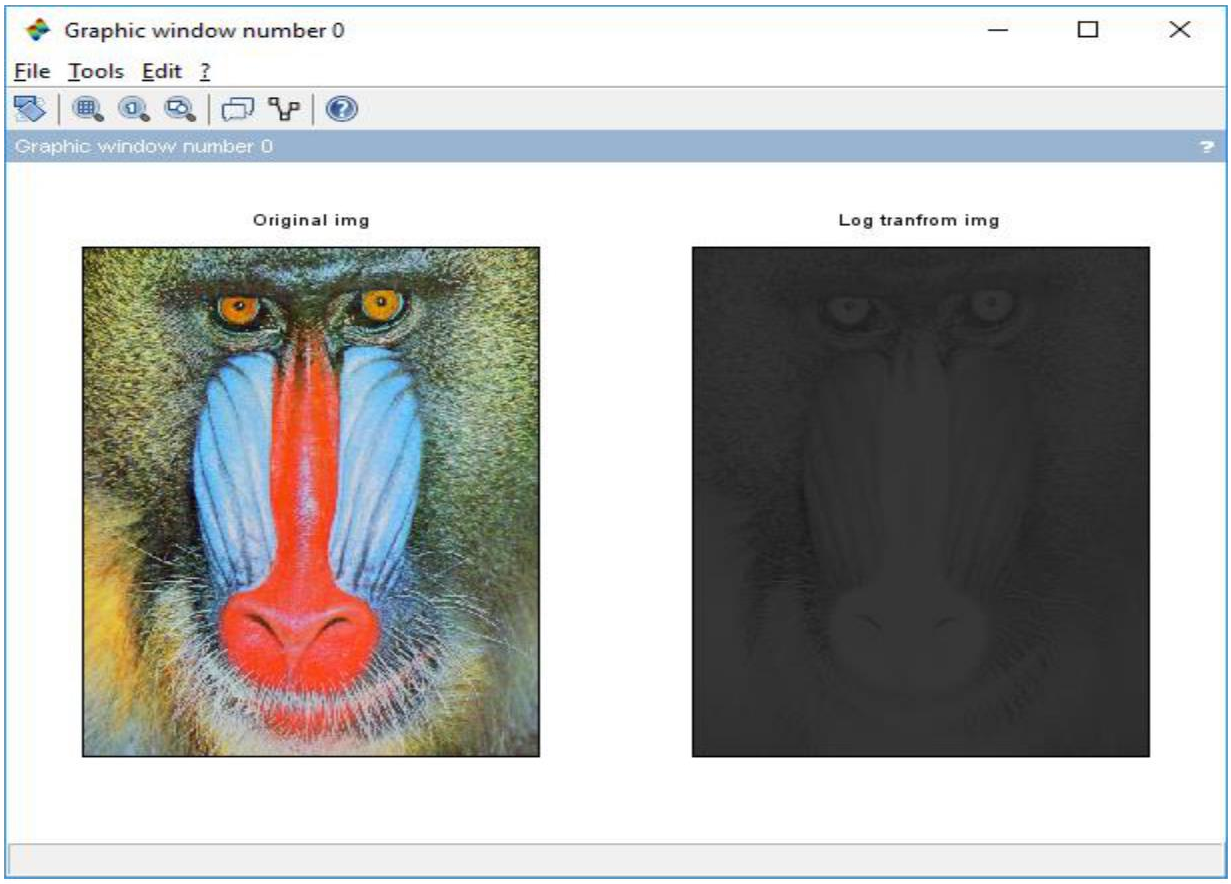
Output-



3. Program to perform Log transformation. Code-

```
clc;
clear all;
a = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\baboon.png'); b=double(a);
subplot(1,2,1);
imshow(a);
title('Original img');

t=10; //constant value
[m,n] = size(b);
for i = 1:m
for j = 1:n
c(i,j)=t*log(1+b(i,j)); //s=k*log(1+r)
end

end
subplot(1,2,2);
imshow(uint8(c));
title('Log tranfrom img');
```

Output-



Variable Browser

	Name	Value	Type	Visibility
int	a	512x512x3	Integer	local
	b	512x512x3	Double	local
int	c	512x512	Integer	local
	i	512	Double	local
	j	512	Double	local
	m	512	Double	local
	n	512	Double	local
	t	10	Double	local

4. Power-law

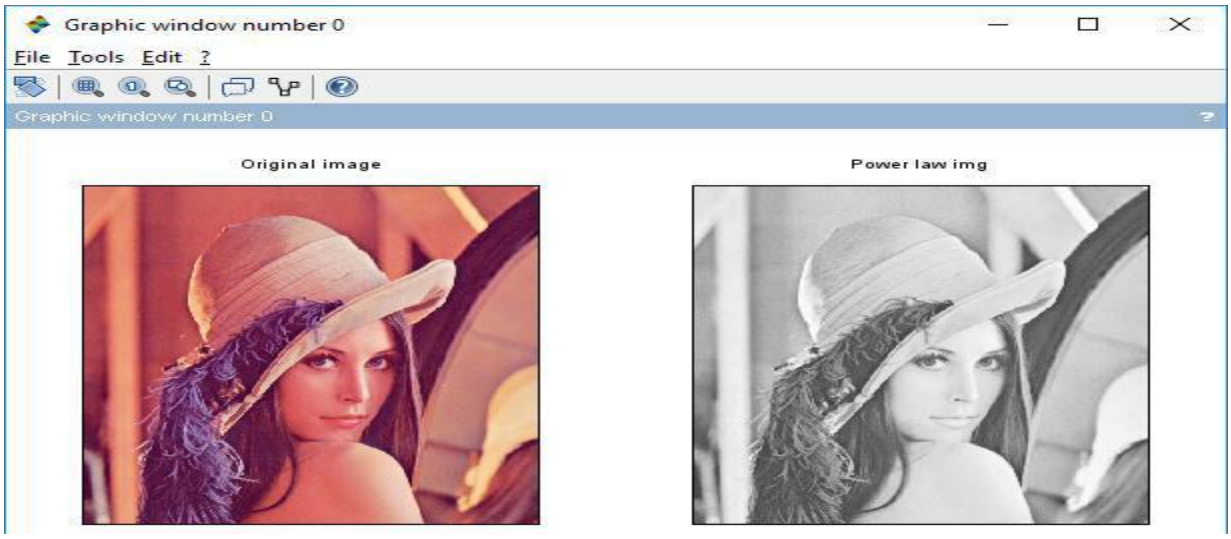
transformations. Code-

clear all;

a = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\lena.png'); b=double(a)

```
subplot(1,2,1);
imshow(a);
title("Original 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 img");
```

Output-



5. Piecewise linear transformations.

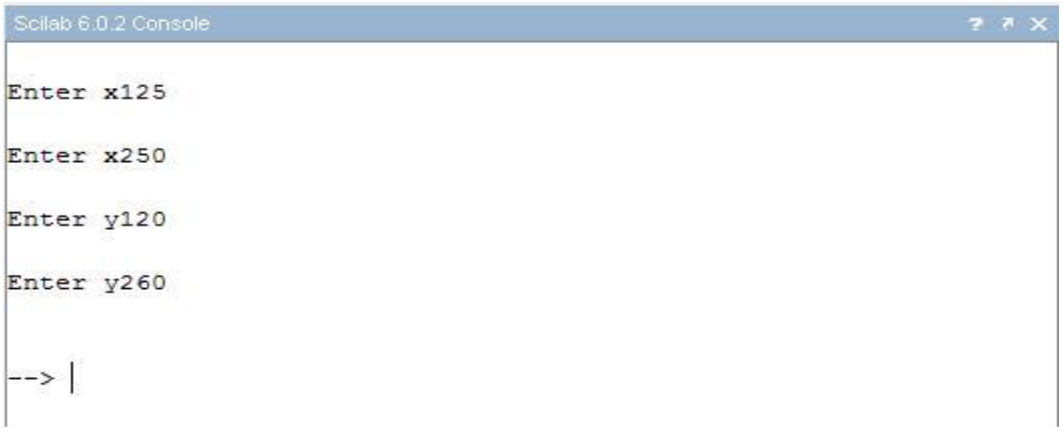
a) Contrast Stretching. Code-

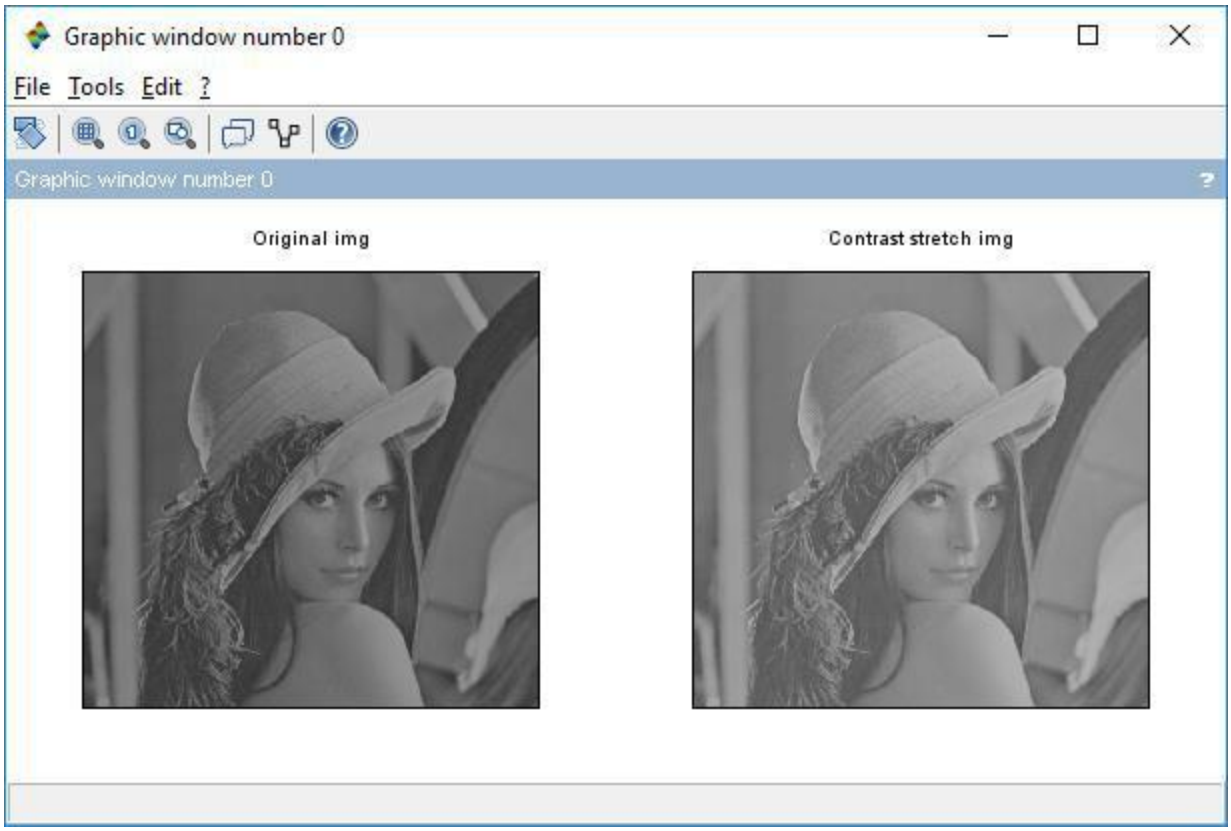
```
clc;
clear all;
a = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\Lena_dark.png'); b=double(a);
[m,n]=size(b);//to get digital image in the form of matrix m by n
x1=input('Enter x1');
x2=input('Enter x2');
```

```
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)=slop2*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
subplot(1,2,1);
imshow(a);
title('Original img');
subplot(1,2,2);
imshow(uint8(ics));
title('Contrast stretch img');
```

Output-

Input-





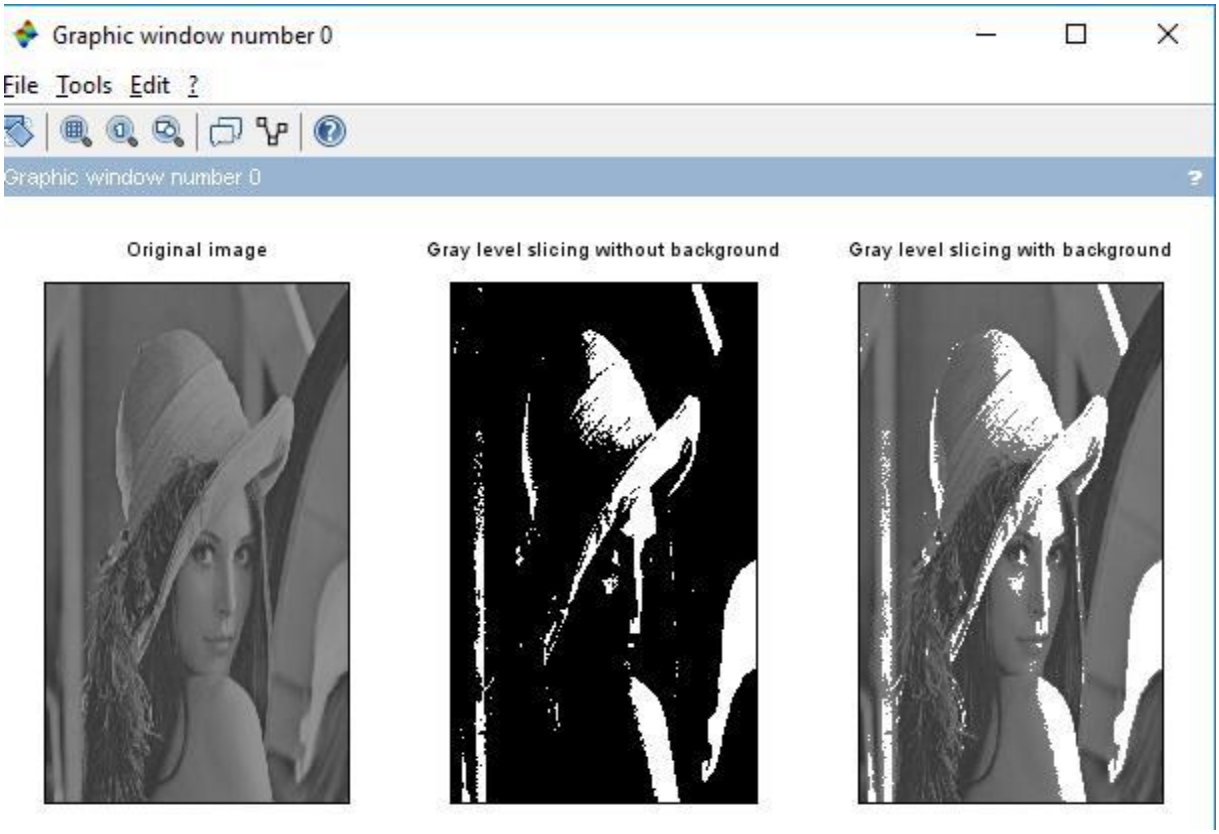
b) Gray-level slicing with and without background. Code-

```
clc;
clear all;
a = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\Lena_dark.png'); 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");
```

Output-

Input-

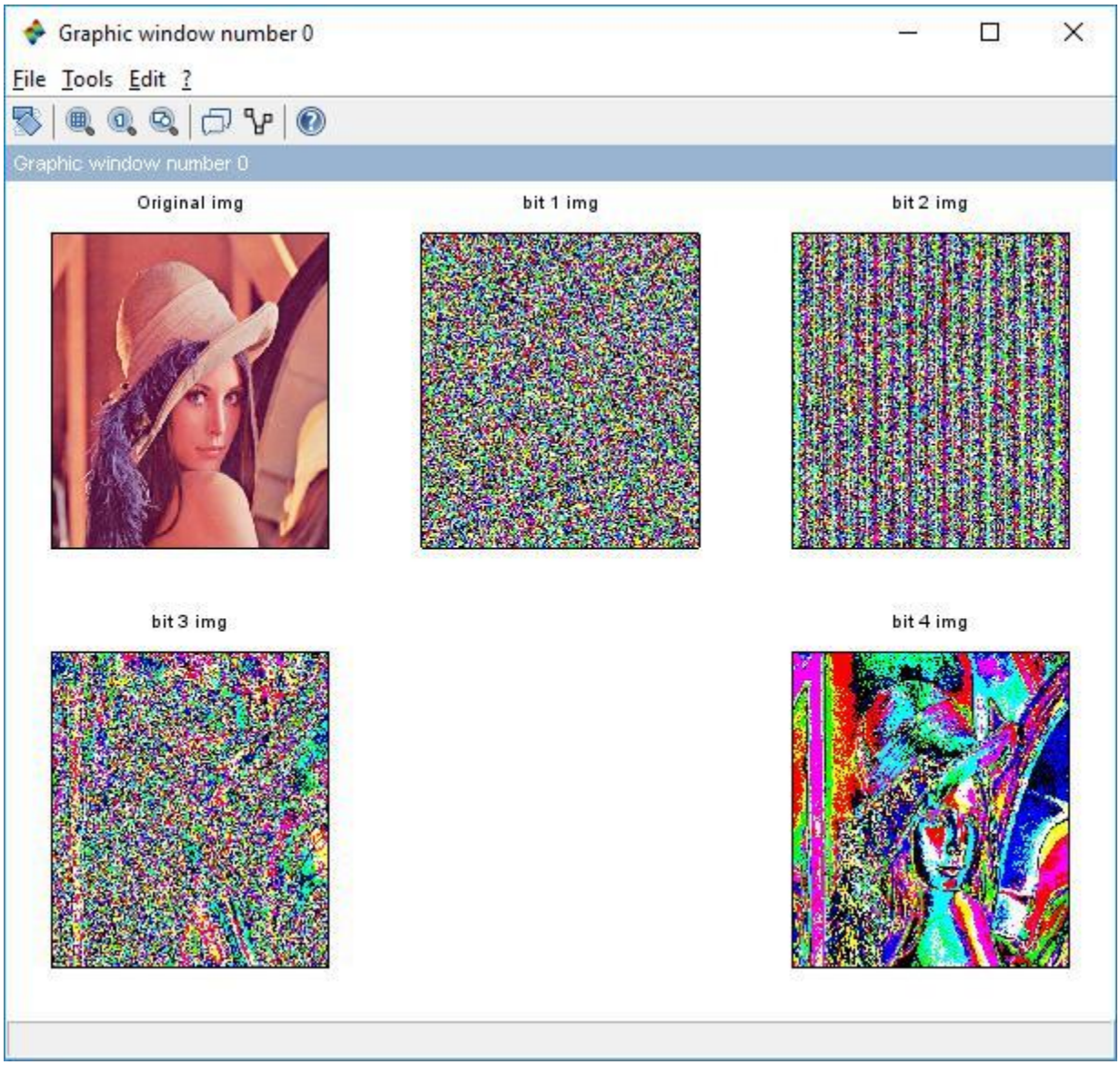


c) Bit-plane slicing. Code-

```
clc;
clear all;
a = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\lena.png'); b=double(a);
subplot(2,3,1);
imshow(a);
title("Original img");

f1=bitget(b,1),
subplot(2,3,2);
imshow(f1);
title("bit 1 img");

f2=bitget(b,2),
subplot(2,3,3);
imshow(f2);
title("bit 2 img");
```

```
f3=bitget(b,4),
subplot(2,3,4);
imshow(f3);
title("bit 3 img");

f4=bitget(b,6),
subplot(2,3,6);
imshow(f4);
title("bit 4 img");
```

Output-



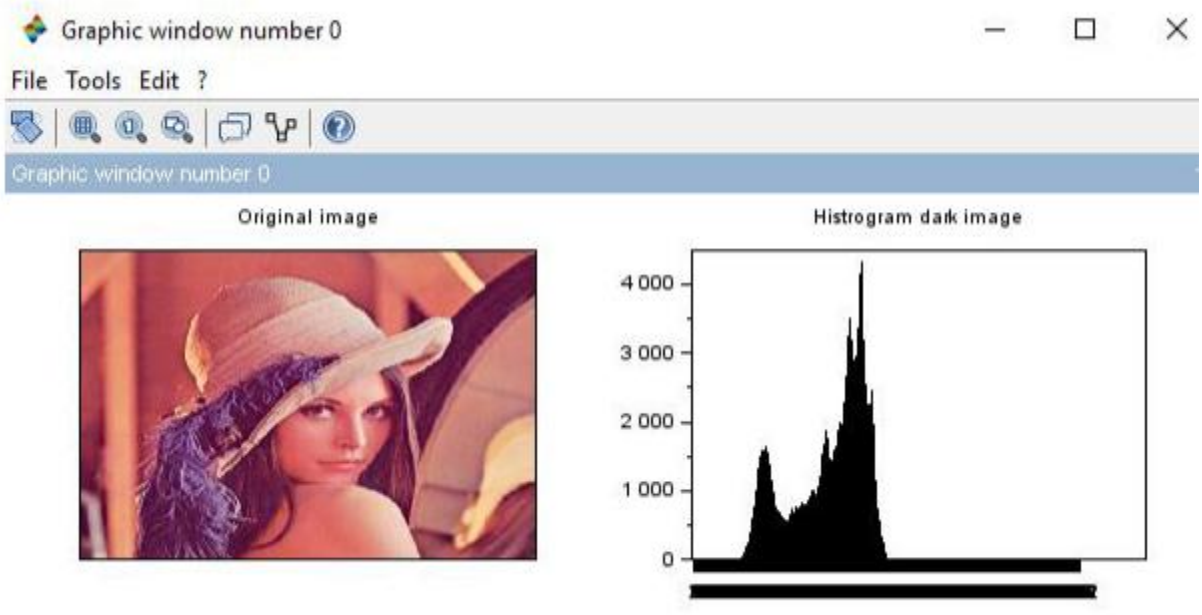
Part B:

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

Code-

```
clc;
clear all;
Img= imread('C:\Program Files (x86)\scilab-
6.0.2\IPCV\images\lena.png'); 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-



2. Program to apply histogram

equalization. Code-

Part C:

Write a program to perform convolution and correlation.

1. convolution

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

Output-

```
    4.    5.    6.
    11.   13.   15.
    11.   13.   15.
    7.    8.    9.

-->
```

2. Correlation.

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

disp(x);
disp(h);
disp(y);
```

Output-

```
Enter first sequence[1 2 3 4]

Enter second sequence[4 3 2 1]


   1.   2.   3.   4.

   4.   3.   2.   1.

   1.   4.   10.   20.   25.   24.   16.

-->
```

Part D:

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

1. Low Pass.

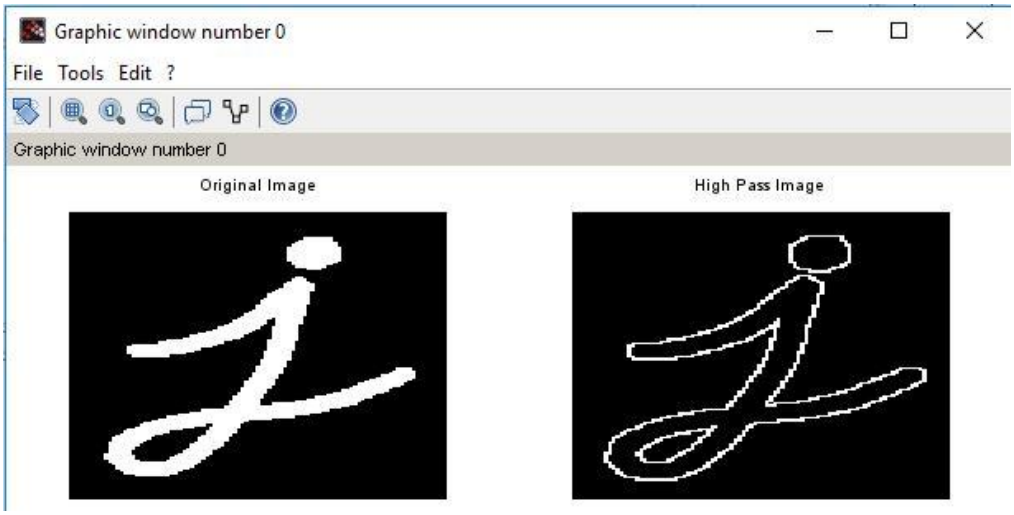
```
//low pass filter for smooting
clc;
clear all;
a1=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\morpex.png');
a=double(a1);
[m,n]=size(a);
w=[1 1 1; 1 1 1; 1 1 1];
for i=2:m-1
   for j=2:n-1
      b(i,j)=(w(1)*a(i-1,j+1)+w(2)*a(i,j+1)+w(3)*a(i+1,j+1)+w(4)*a(i-
1,j)+w(5)*a(i,j)+w(6)*a(i+1,j)+w(7)*a(i-1,j-1)+w(8)*a(i,j-1)+w(9)*a(i+1,j-1))
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.

```
//high pass filter for smooting
clc;
clear all;
a1=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\morpex.png');
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.

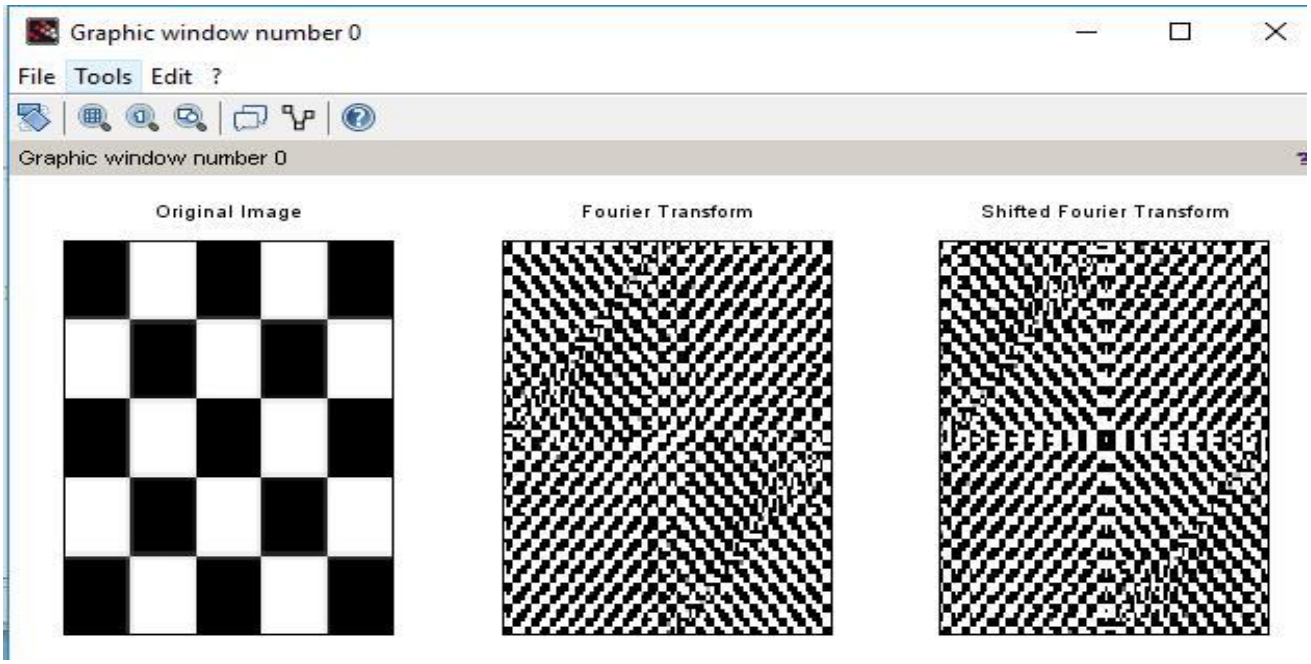
Part A:

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

Code-

```
clc;
clear;
close;
I=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\checkerbox.png');
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:

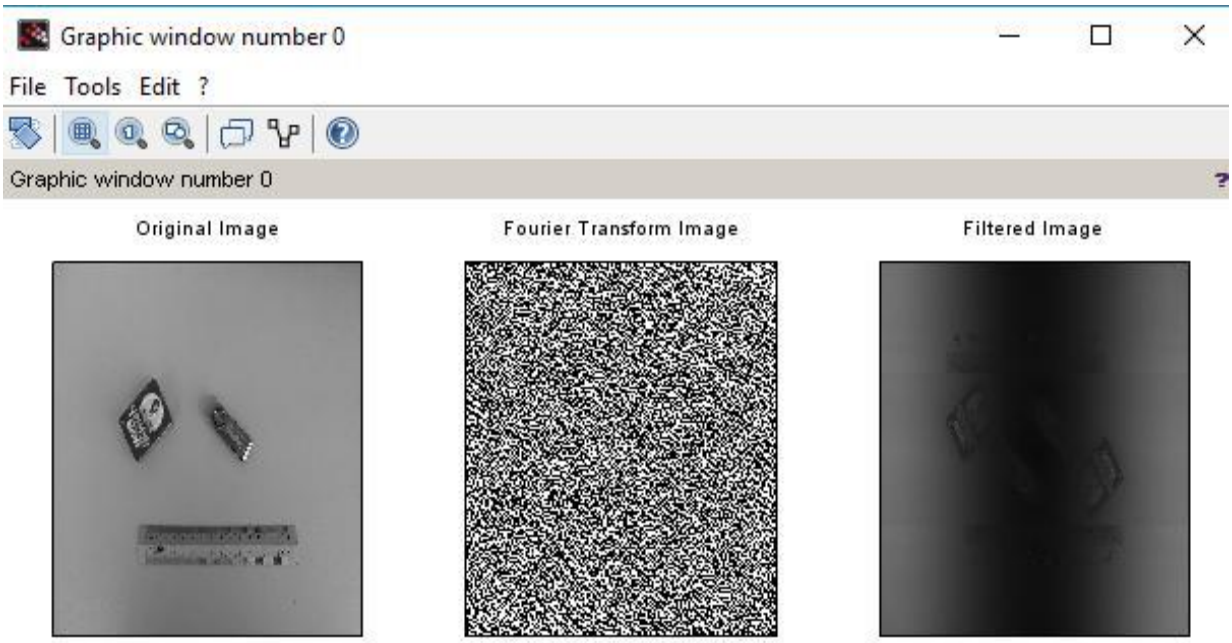
Aim- Program to apply Low pass and High pass filters in frequency domain.

Low pass filter-

Code-

```
S=imread("C:\Program Files (x86)\scilab-6.0.2\IPCV\images\measure_gray.jpg");
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("Filtered Image");
```

Output-

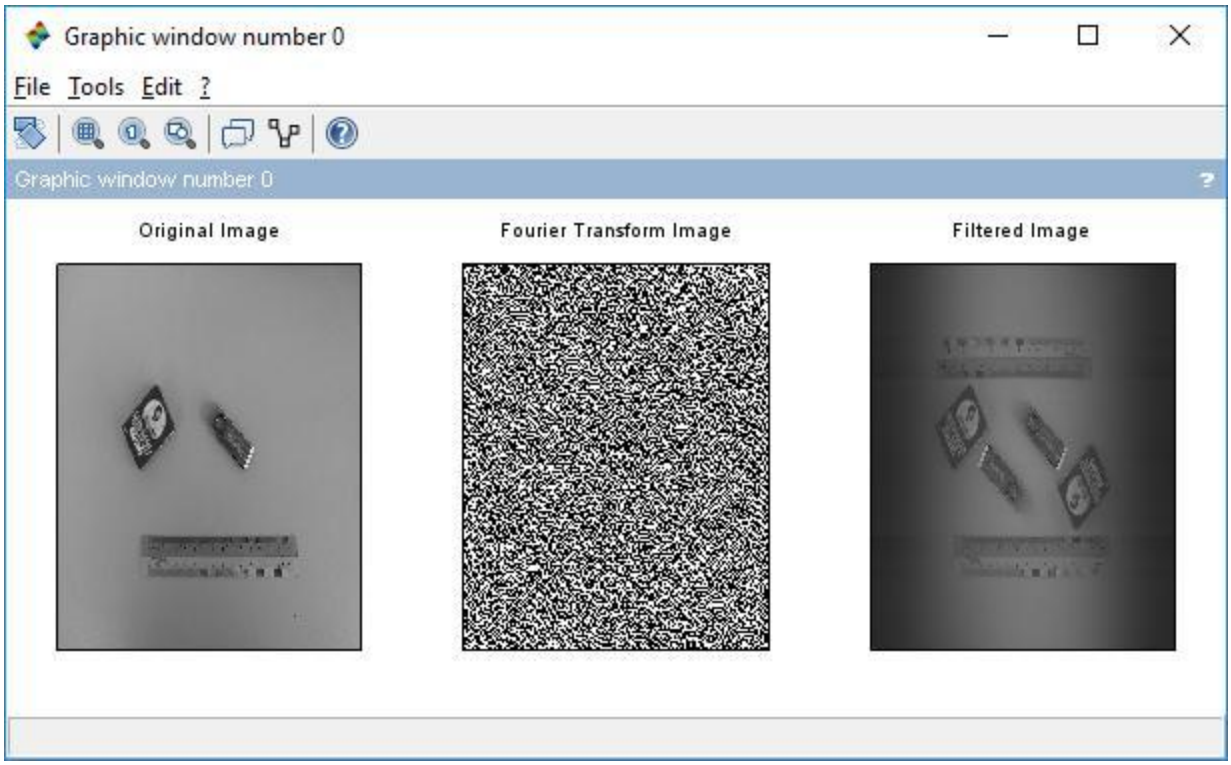


High pass filter-

Code-

```
S=imread("C:\Program Files (x86)\scilab-6.0.2\IPCV\images\measure_gray.jpg");
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 gaussion filter in frequency domain.

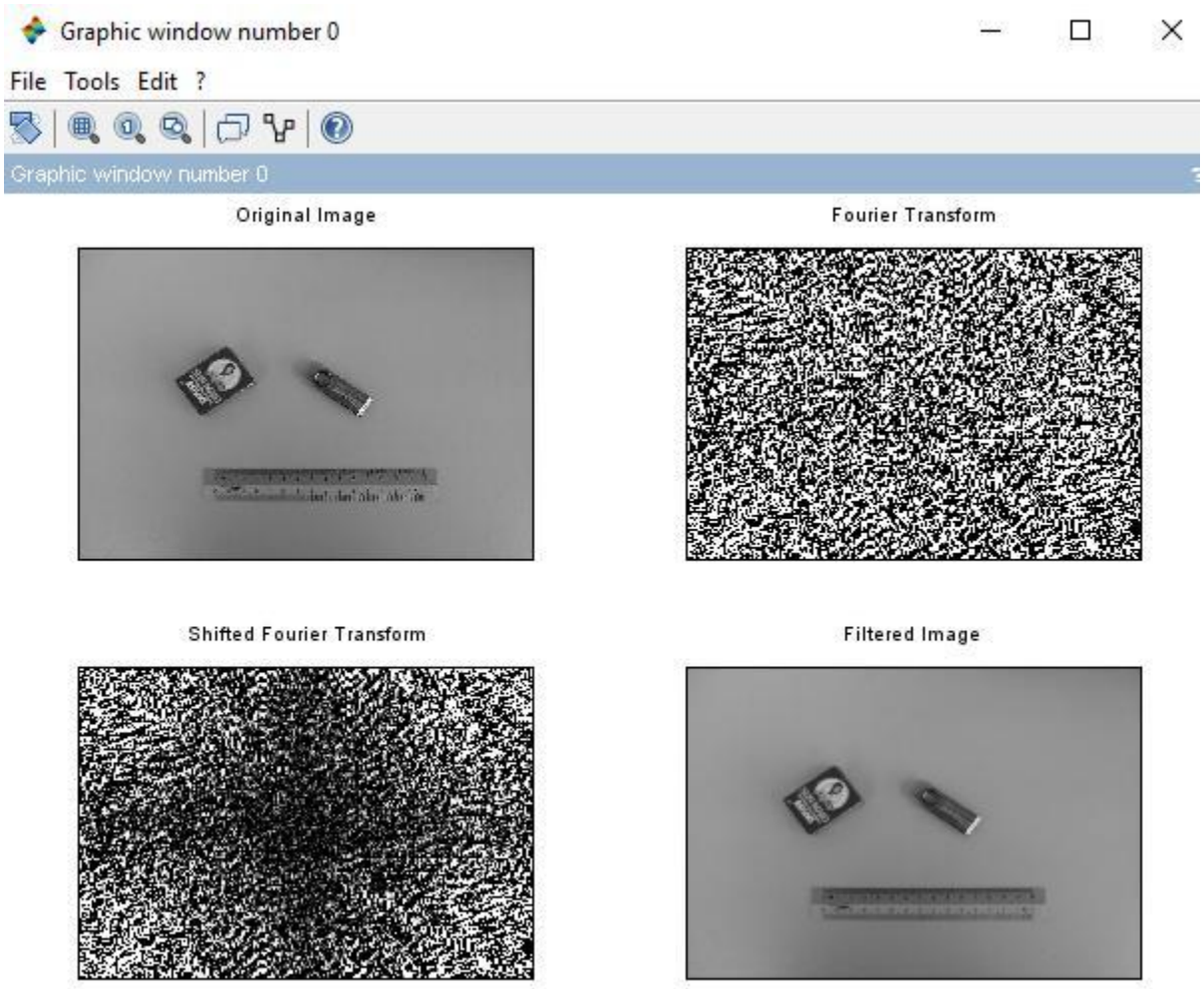
Butterworth-

Code-

```
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-



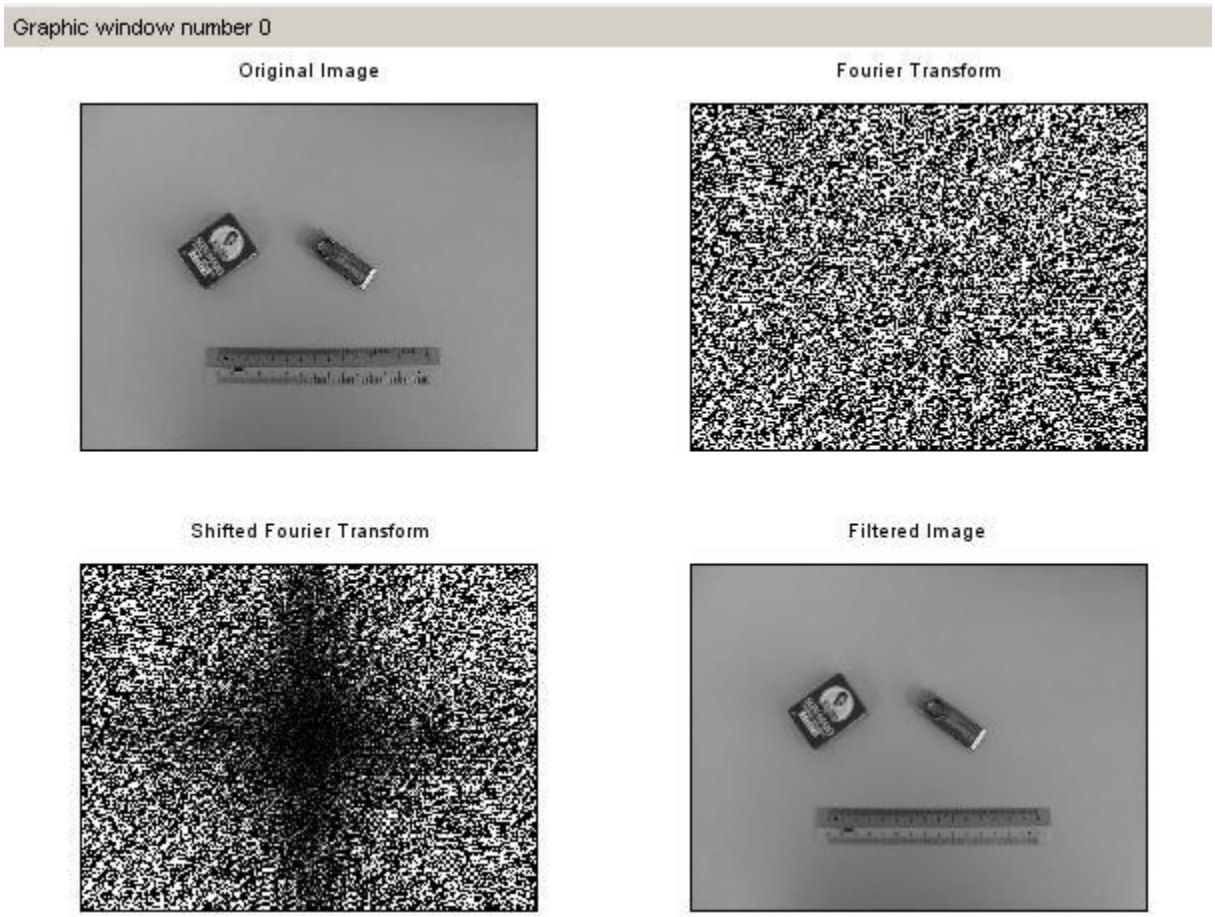
Gaussion filter-

Code-

```
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("Fourier Transform");

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:

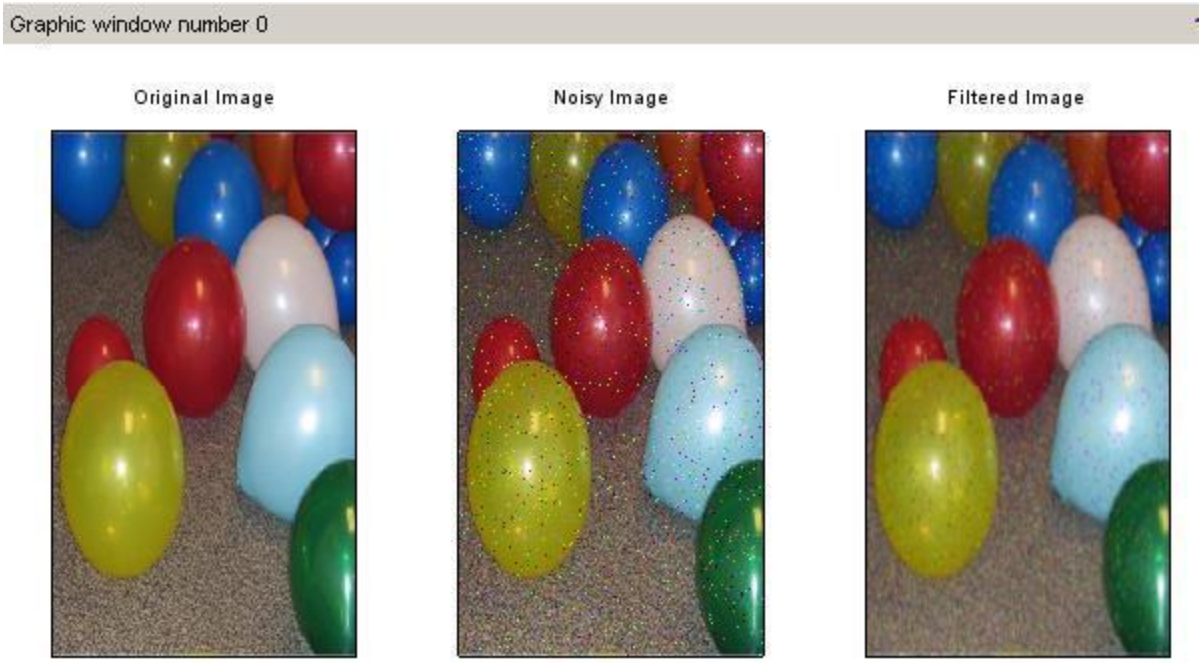
Aim- Program to denoise using spatial mean, median.

Mean-

Code-

```
Image=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\balloons.png'); //Image=rgb2gray(Image);
NoisyImage=imnoise(Image,'salt & pepper', 0.02);
F1=fspecial('average',3);
FilterImage=imfilter(NoisyImage,F1);

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

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

subplot(1,3,3);
imshow(FilterImage);
title('Filtered Image');
```

Output-



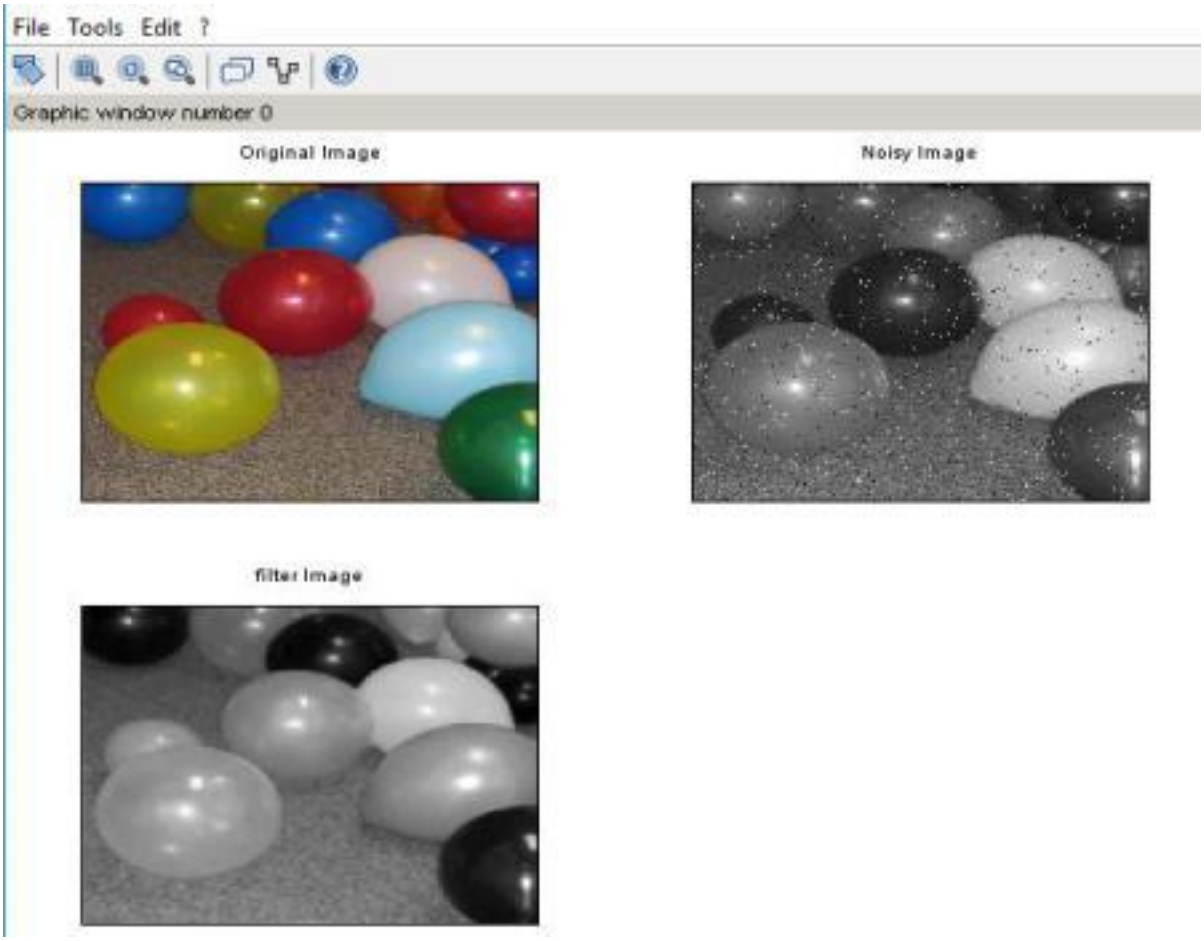
Median-

Code-

```
clc;
clear all;
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\balloons.png');
image=rgb2gray(a);
NoisyImage = imnoise(image,'salt & pepper',0.02);

[m n]=size(a);
for i=2:m-1
   for j=2:n-1
      output(i,j)=median([a(i-1,j+1),a(i,j+1),a(i+1,j+1),a(i-1,j),a(i+1,j),a(i-1,j-1),a(i,j-1),a(i+1,j-
1)]);
   end
end

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

subplot(2,2,2);
imshow(NoisyImage);
title('Noisy Image');

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

title('filter Image');

Output-

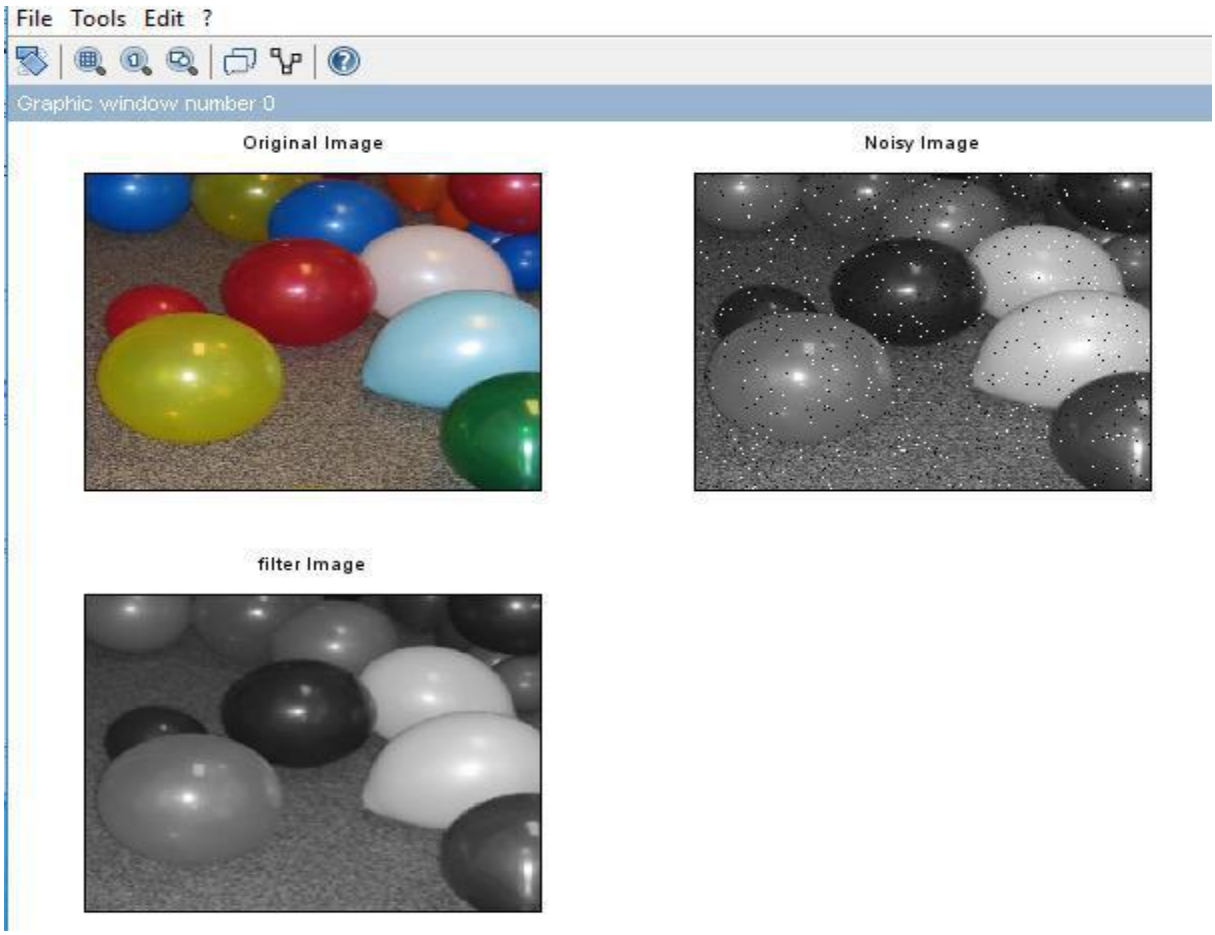


Code-

```
clc;
clear all;
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\balloons.png');
image=rgb2gray(a);

NoisyImage = imnoise(image,'salt & pepper',0.02);

output=immedian(NoisyImage,[3,3]);

subplot(2,2,1);
imshow(a);
title('Original Image');
```

```
subplot(2,2,2);
imshow(NoisyImage);
title('Noisy Image');

subplot(2,2,3);
imshow(output);
title('filter Image');
```

Output-



Part B:

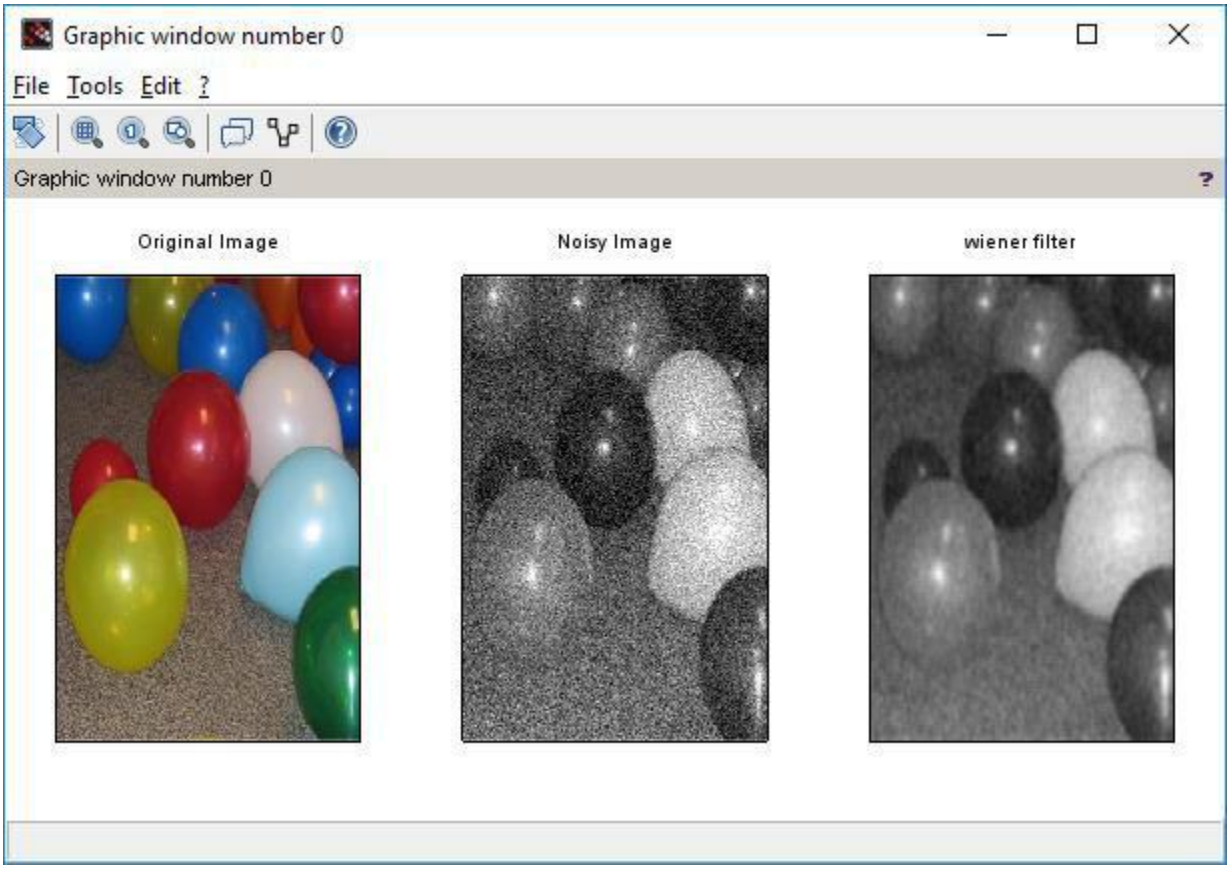
Aim- Program for Image Weiner filters.

Code

```
clc;
clear all;
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\balloons.png');
image=rgb2gray(a);
```

```
NoisyImage = imnoise(image,'gaussian',0.02);

wienerfilter=imwiener2(NoisyImage,[5,5],0.2);

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

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

subplot(1,3,3);
imshow(wienerfilter);
title('wiener filter');
```

Output-



Practical No 5

Aim- Color Image Processing.

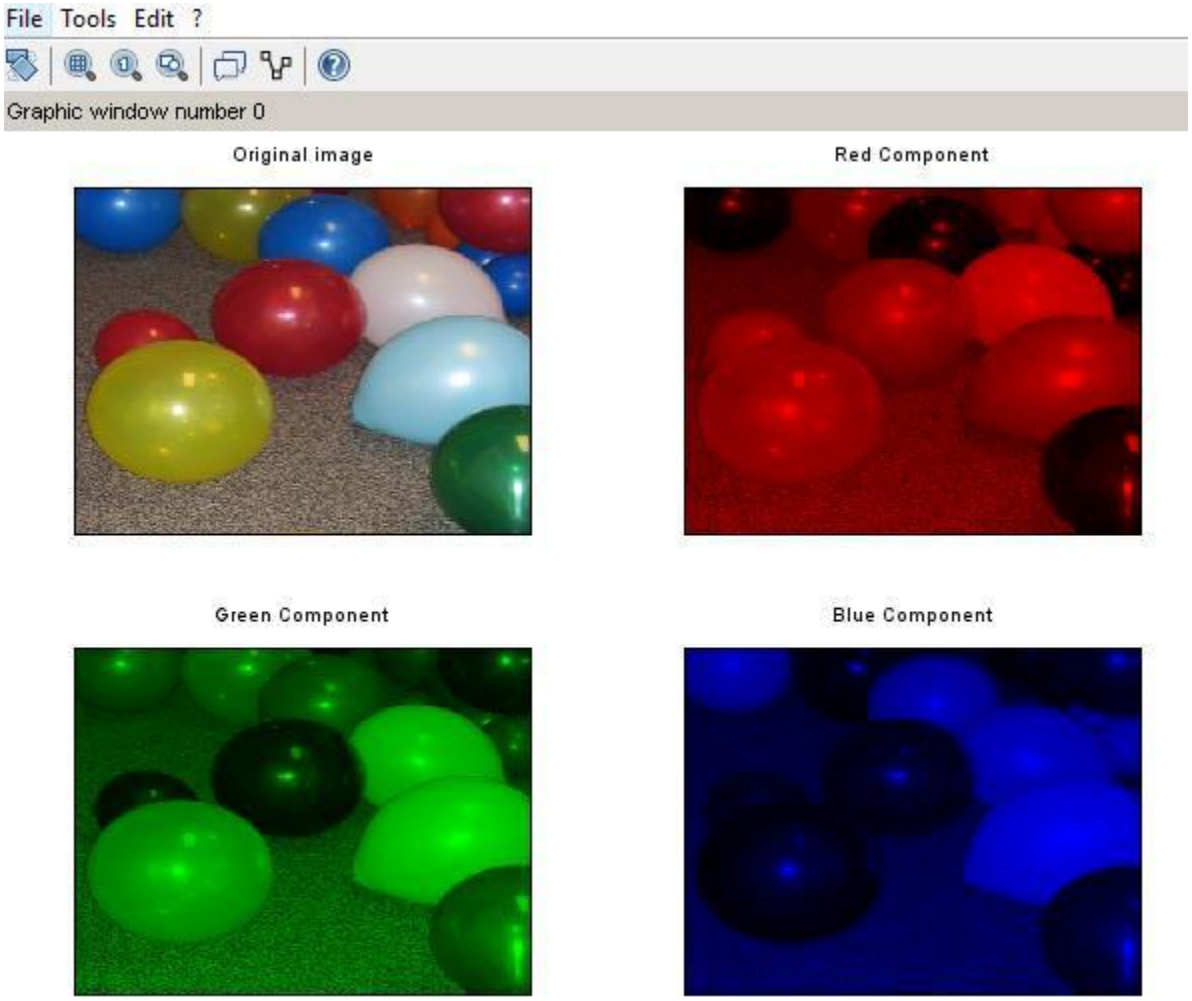
Part A:

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

Code-

```
clc;
I = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\balloons.png');

//rows and columns of 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);

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.

1.

Code-

```
clc;
rgb=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\peppers.png');

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')
```

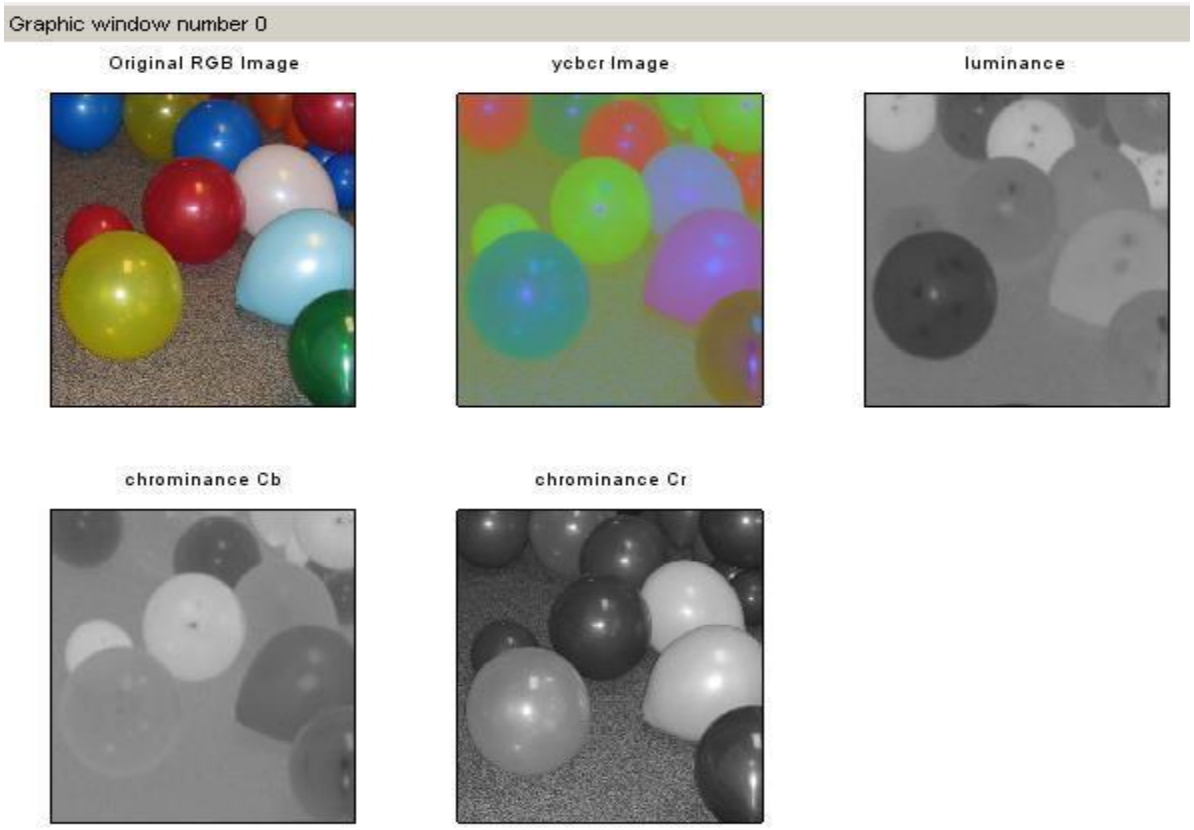
Output-

2.

Code-

```
clc;
rgb = imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\balloons.png');

ycbcr=rgb2ycbcr(rgb);

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

subplot(2,3,2);
imshow(ycbcr);
title('ycbcr Image');

subplot(2,3,3);
imshow(ycbcr(:,:,1));
title('luminance');

subplot(2,3,4);
imshow(ycbcr(:,:,2));
title('chrominance Cb');

subplot(2,3,5);
imshow(ycbcr(:,:,3));
title('chrominance Cr')
```

Output-



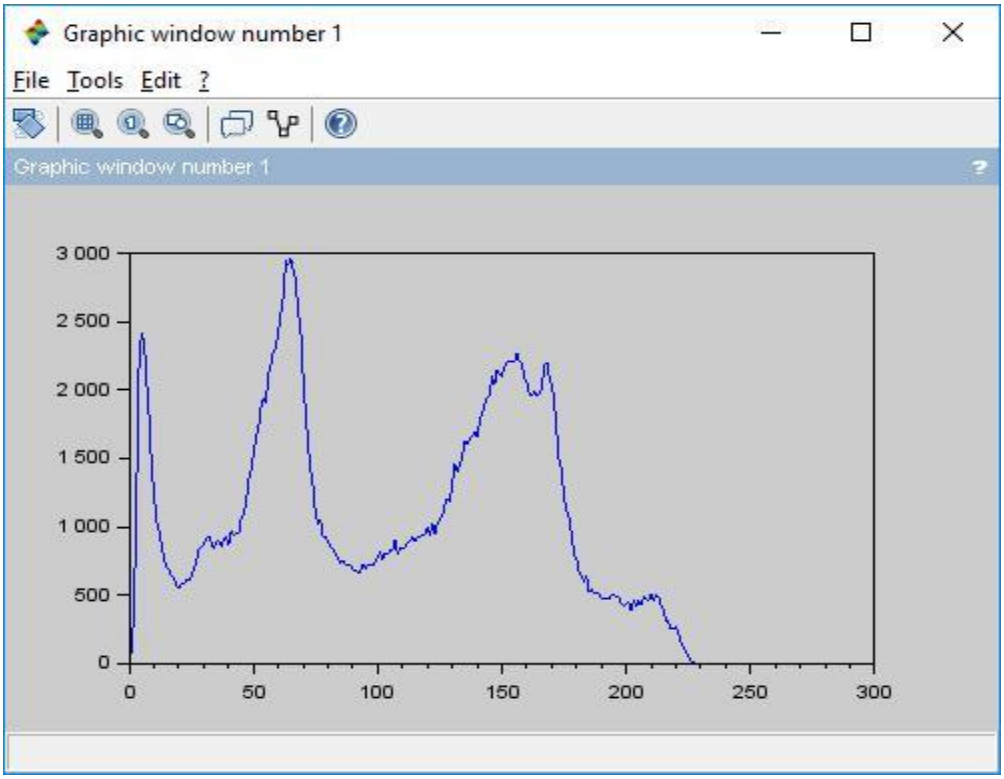
Part C:

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

```
clc;
rgb = imread('C:\Program Files (x86)\scilab-
6.0.2\IPCV\images\peppers.png'); Image = rgb2gray(rgb);

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

subplot(1,3,2);
imshow(Image);
title('Gray RGB Image');

subplot(1,3,3);
```

```
imshow(Image,jetcolormap(256));
title('Pseudo Color Image');

Histogram = imhist(Image);
figure();
plot(0:255,Histogram');
```

Output-


Graphic window number 1
File Tools Edit ?
Graphic window number 1
3 000
2 500
2 000
1 500
1 000
500
0
0
50
100
150
200
250
300

Practical No 6

Aim- Fourier Related Transforms.

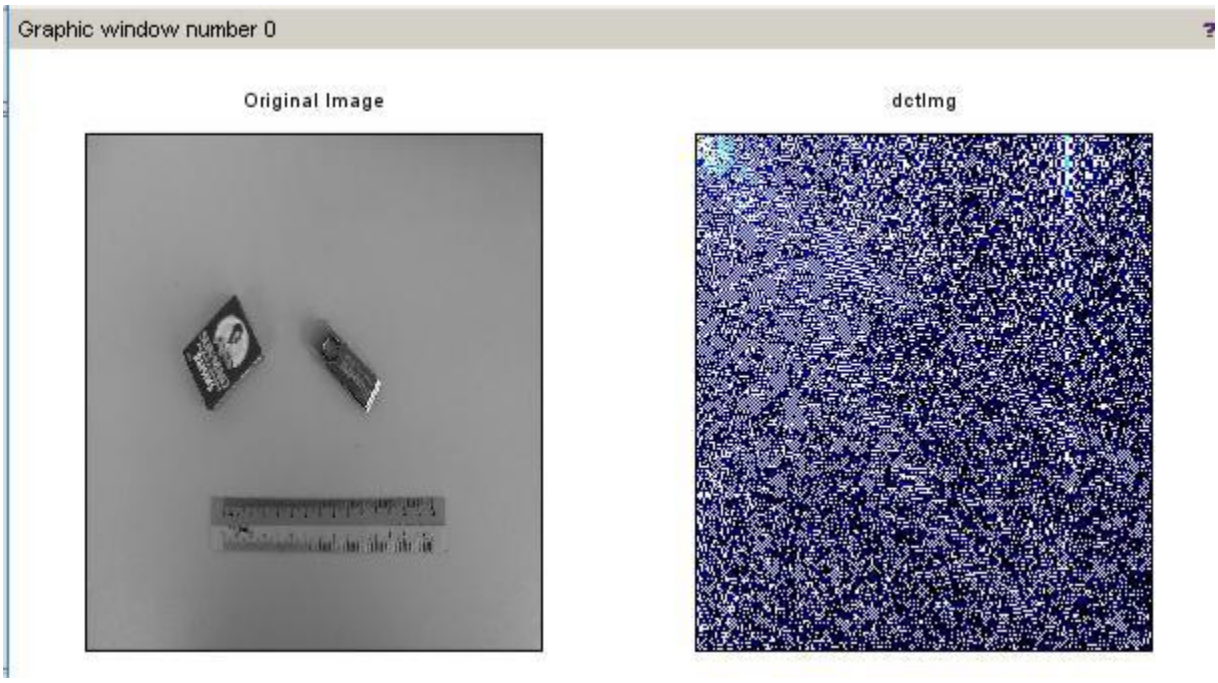
Part A:

Aim- Program to compute Discrete Cosine Transforms.

Code-

```
clear all;
a=imread("C:\Program Files (x86)\scilab-6.0.2\IPCV\images\measure_gray.jpg");
dctImg=imdct(a);
subplot(1,2,1);
imshow(a);
title('Original Image');

subplot(1,2,2);
imshow(dctImg, jetcolormap(256));
title('dctImg');
```

Output-



Practical No 7

Aim- Morphological Image Processing.

Part A:

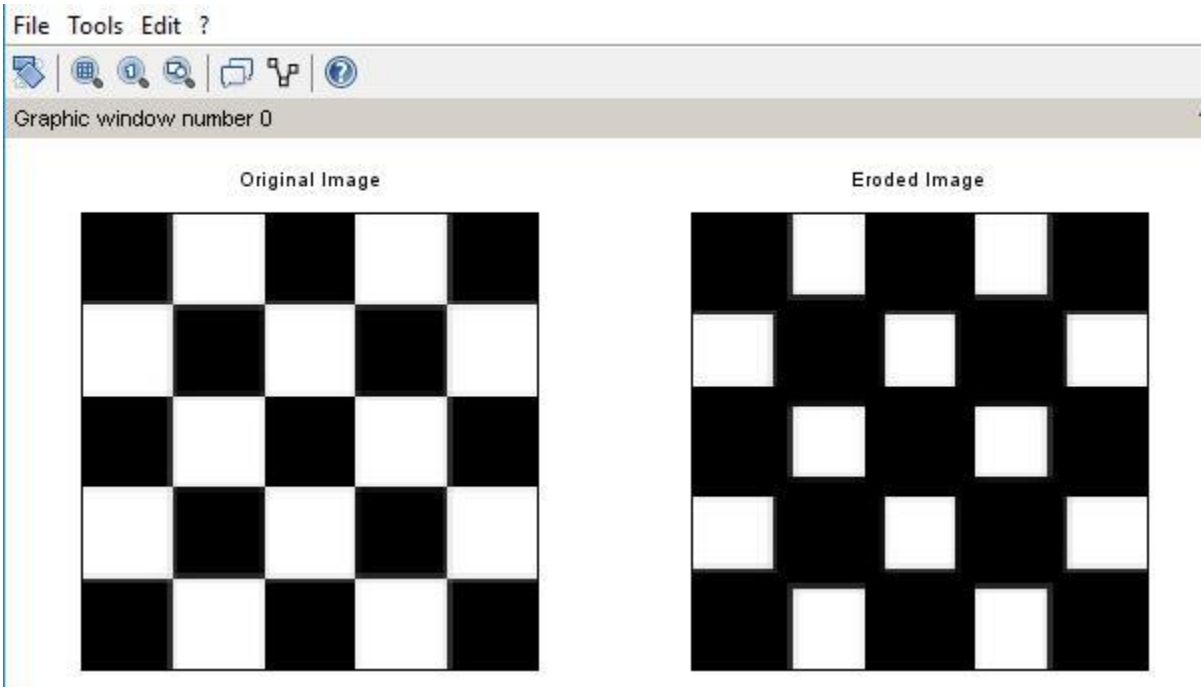
Aim- Program to apply erosion, dilation, opening, closing.

Erosion-

Code-

```
clear all ;
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\checkerbox.png');

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-

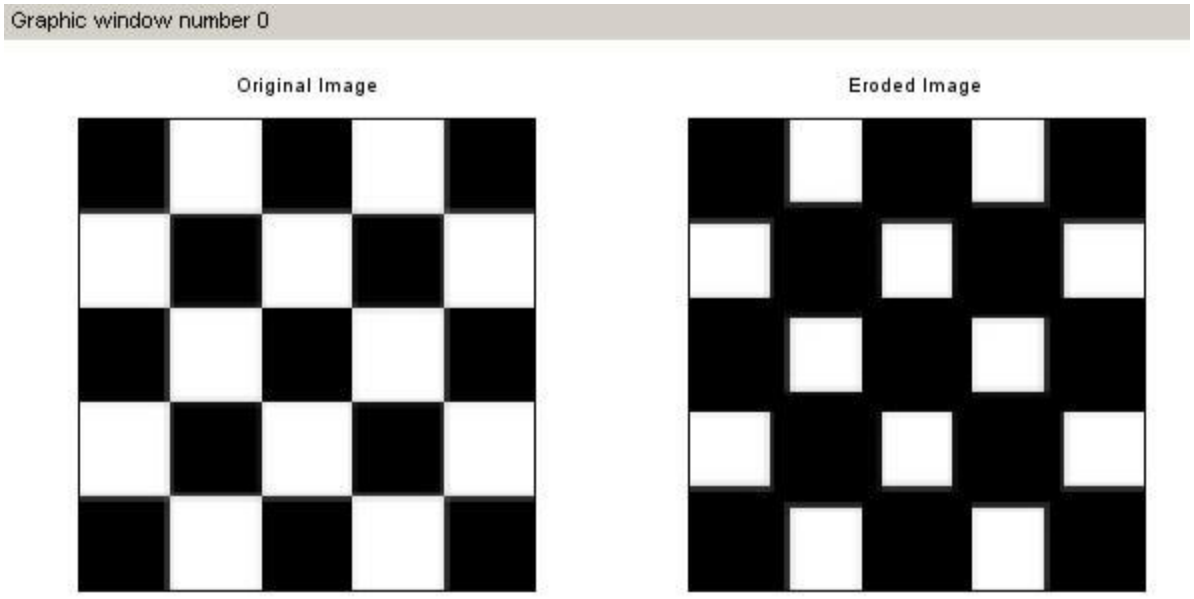


Erosion2-

Code-

```
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\checkerbox.png');

//se = imcreatese('rect',5,5);
se=[1,1,1,1,1;
   1,1,1,1,1;
   1,1,1,1,1;
   1,1,1,1,1;
   1,1,1,1,1];
erosion = imerode(a,se);

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

subplot(1,2,2);
imshow(erosion);
title('Eroded Image');
```

Output-

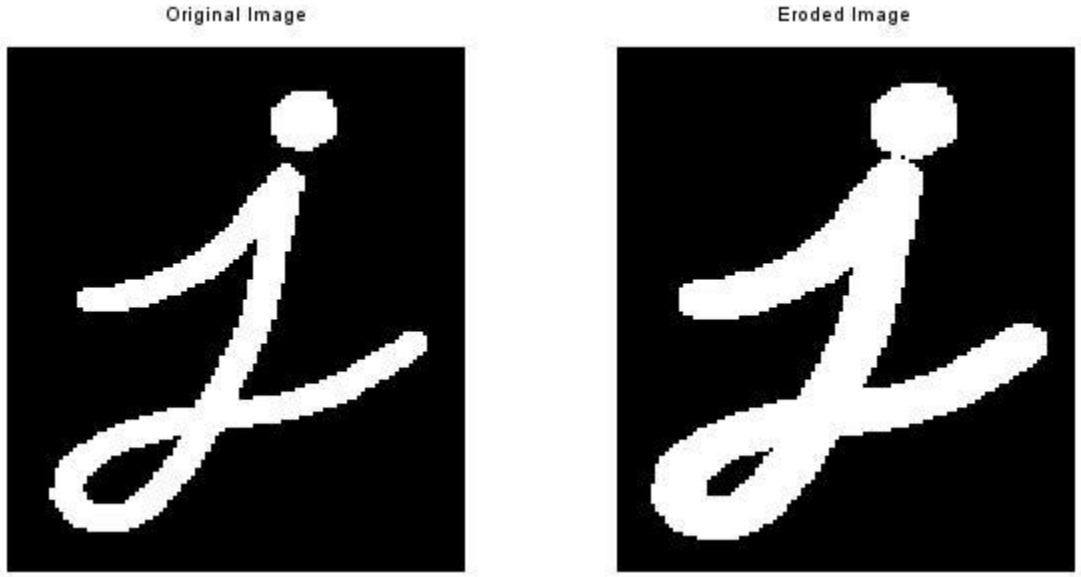


Dilation -

Code-

```
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\morpex.png');

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('Eroded Image');
```

Output-



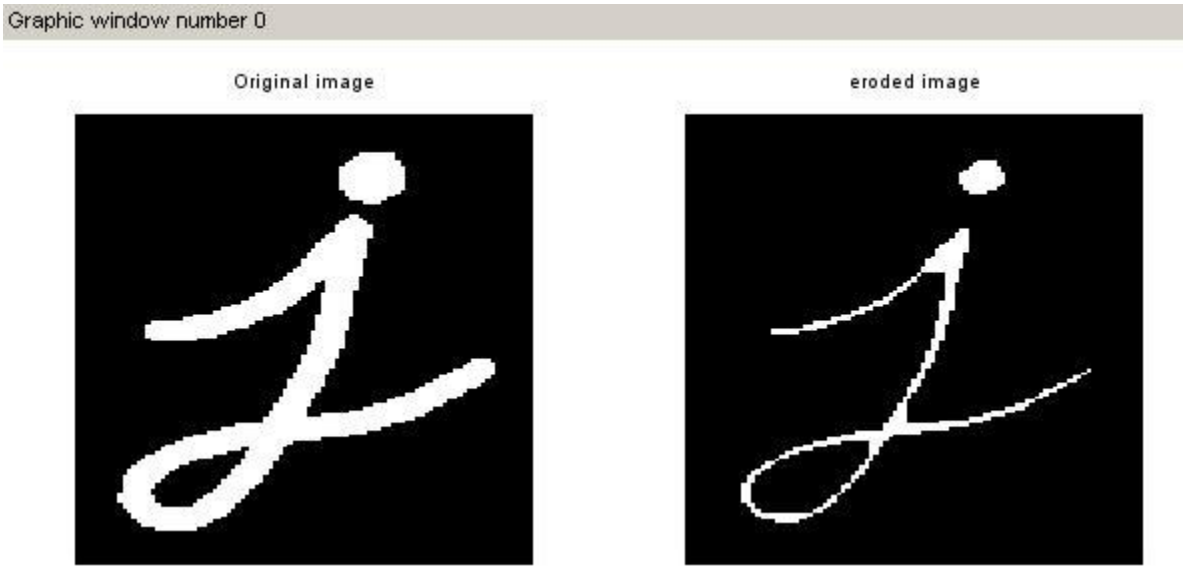
Dilation2 -

Code-

```
clear all;
a=imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\morpex.png')
se=[1,1,1,1,1;
   1,1,1,1,1;
   1,1,1,1,1;
   1,1,1,1,1;
```

```
   1,1,1,1,1]
erosion=imerode(a,se);

subplot(1,2,1);
imshow(a);
title('Original image');

subplot(1,2,2);
imshow(erosion);
title('eroded image');
```

Output-



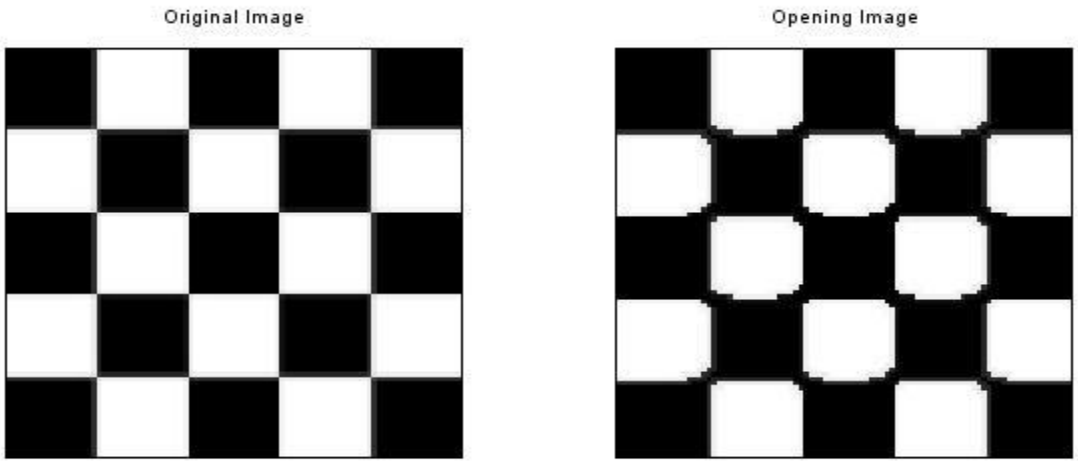
Opening-

Code-

```
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\checkerbox.png');

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-

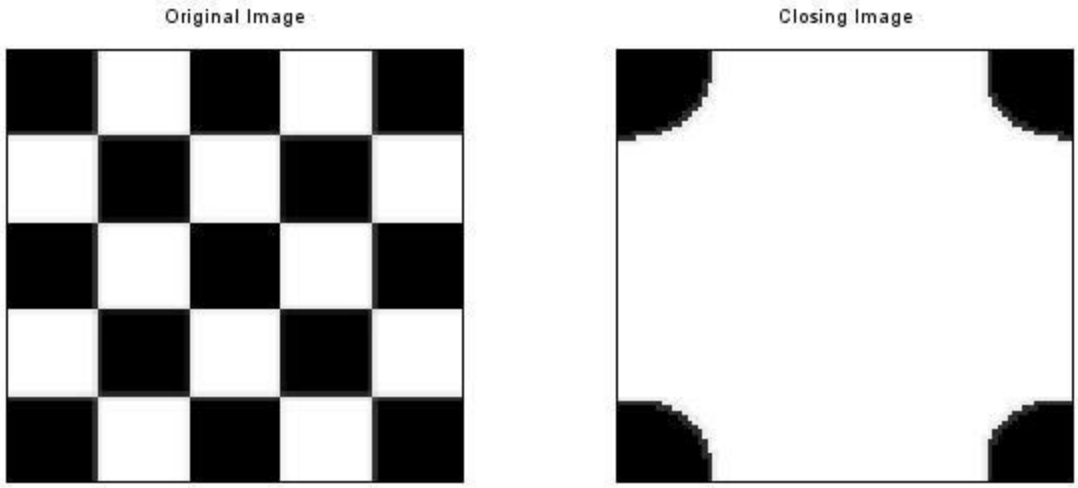


Closing-

Code-

```
clc;
clear all;
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\checkerbox.png');

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-



If I change the image with lena.png then output like this.

Part B:

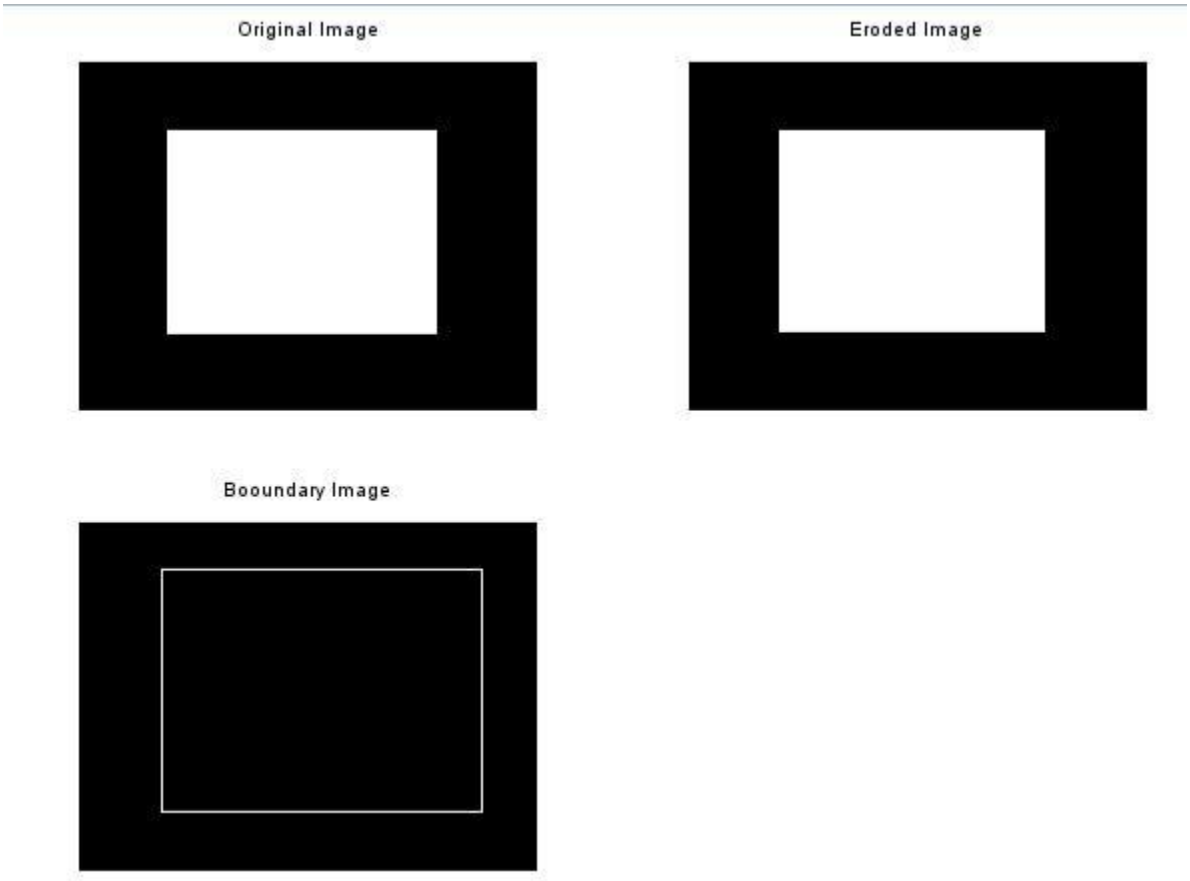
Aim- Program for detecting boundary of an image.

Code-

```
Clear all;
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\big_sq.png');

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('Booundary Image');
```

Output- (Below box if your move the arrow then box is come)



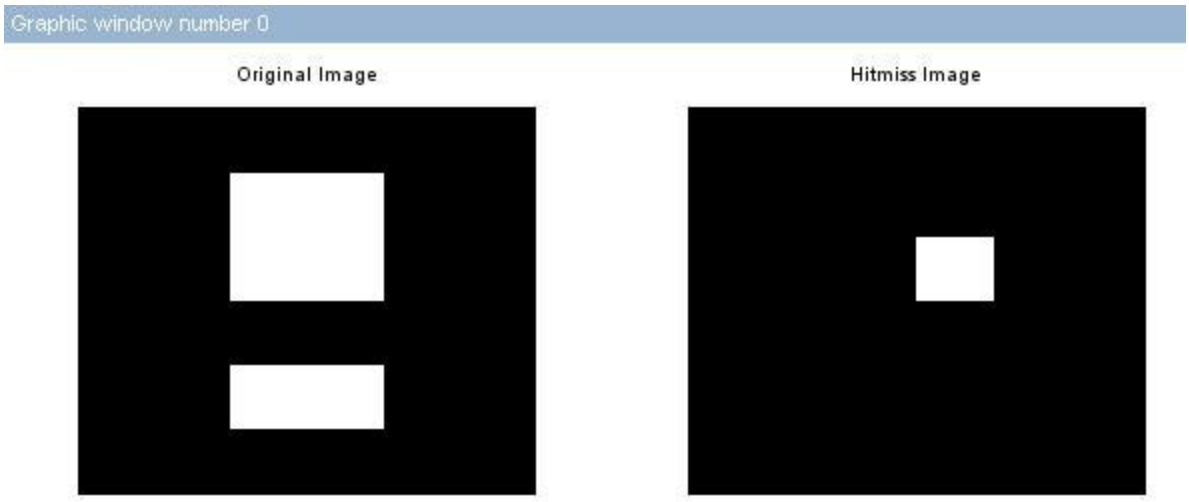
Part C:

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

Code-

```
clc;
clear all;
disp('Practical perform by Ragini')
a=[0 0 0 0 0 0;
   001100;
   001100;
   000000;
   001100;
   000000];

a =im2bw(a,0.5);

se = [0 0 0 0;
      0110;
      0110;
      0000];

//se = increatese('ellipse',5,5);
S2 = imhitmiss(a,se);
//imshow(a);
//imshow(S2);

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

subplot(1,2,2);
imshow(S2);
title('Hitmiss Image')
```

Output-



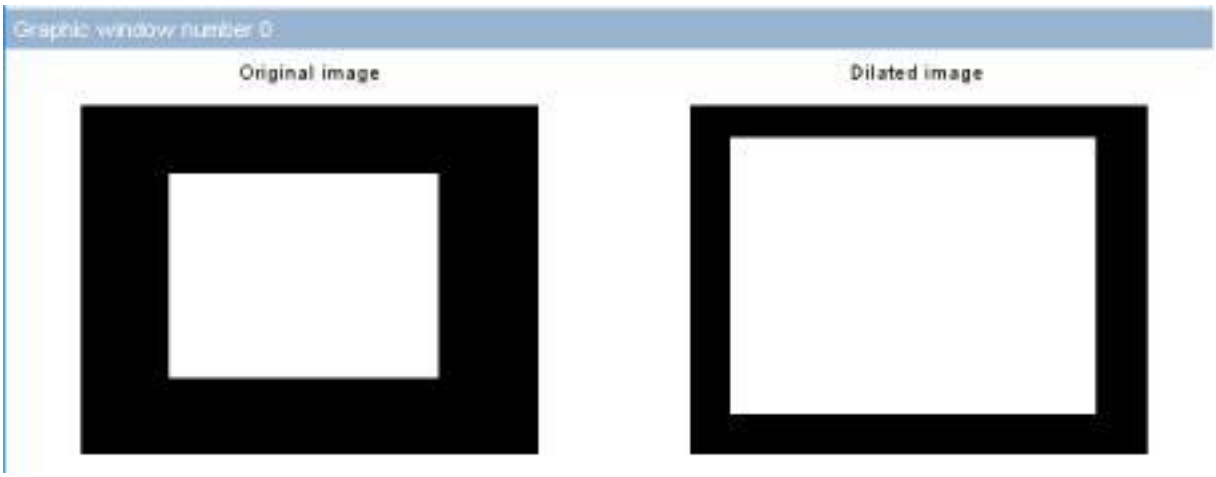
Part D:

Aim- Program to apply morphological gradient on an image.

Code-

```
a=imread("C:\Program Files (x86)\scilab-6.0.2\IPCV\images\big_sq.png");

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("Dilated image");

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

Output-



Part E:

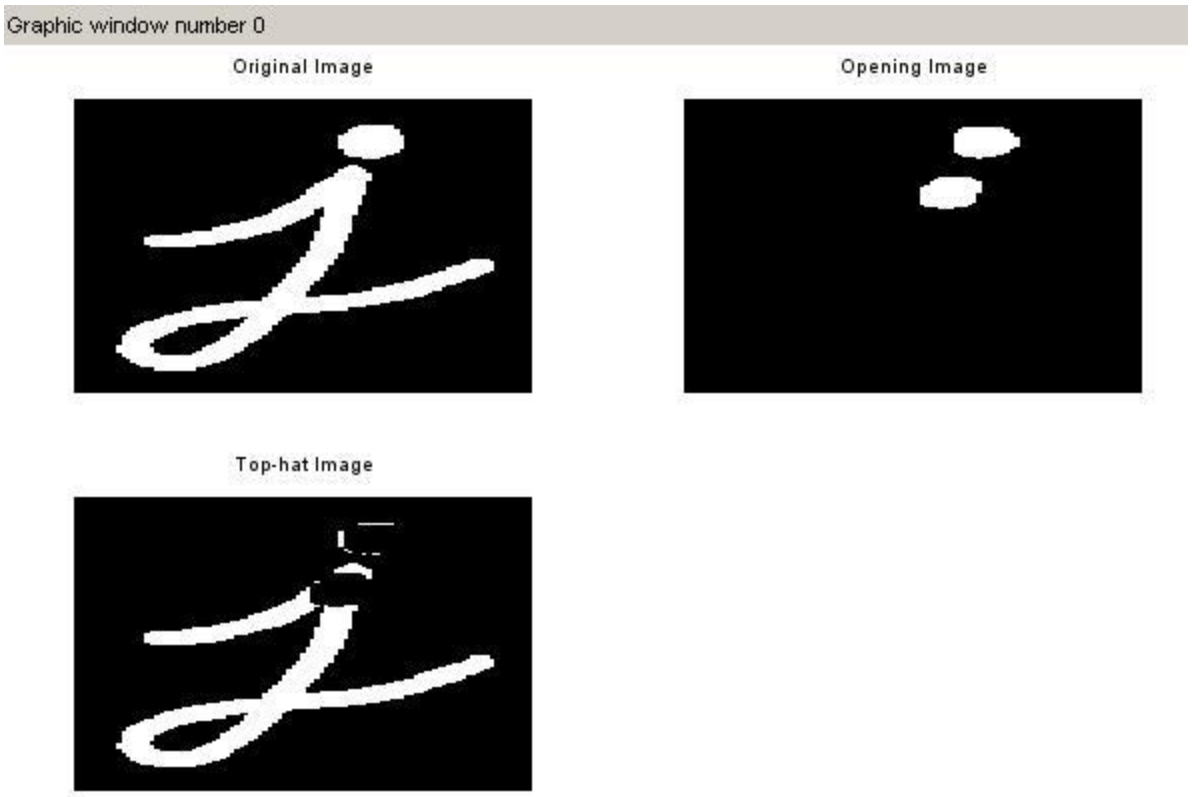
Aim- Program to apply Top-Hat/Bottom-hat Transformations.

Top-Hat-

Code-

```
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\morpex.png');

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-

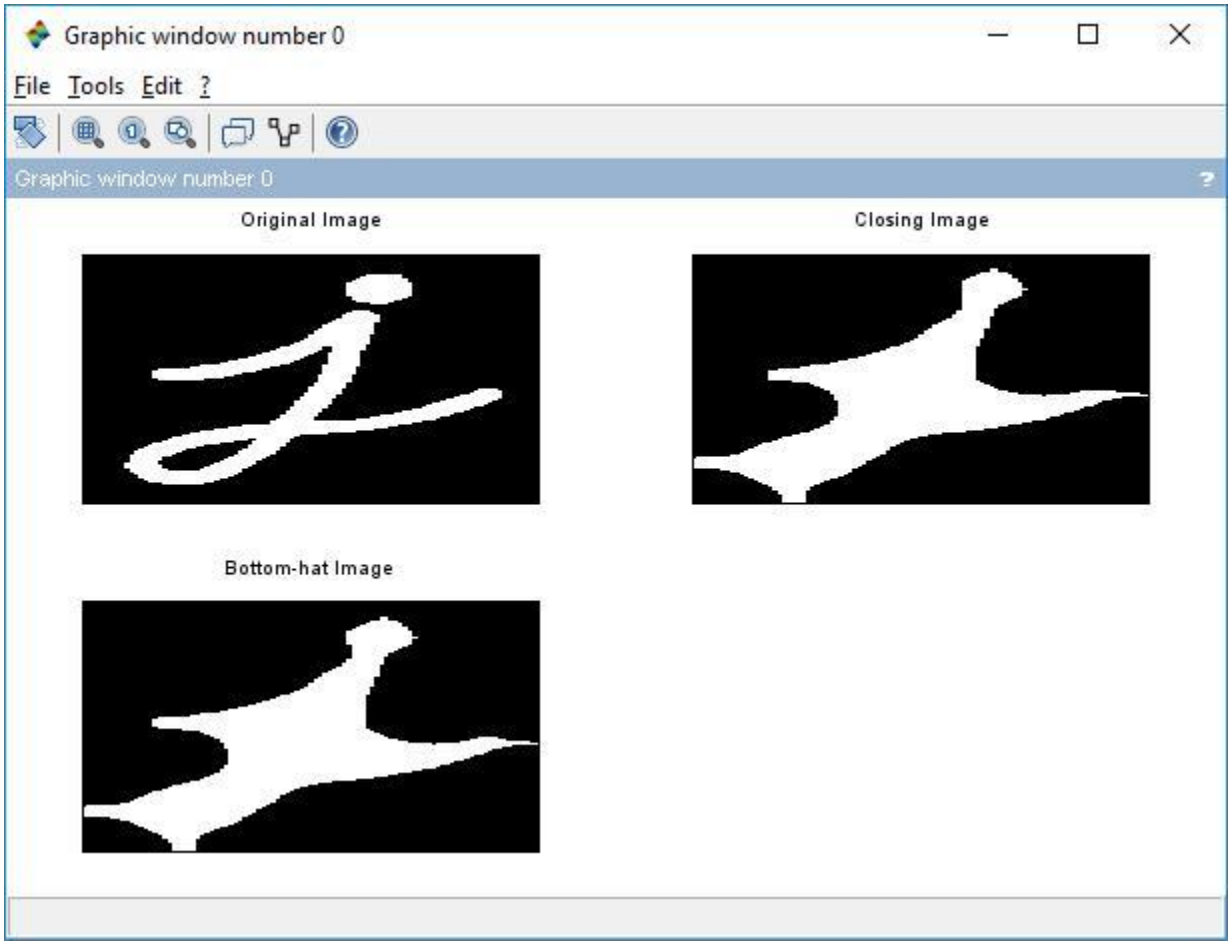


Bottom-hat-

Code-

```
a= imread('C:\Program Files (x86)\scilab-6.0.2\IPCV\images\morpex.png');

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-


Graphic window number 0
File Tools Edit ?
Graphic window number 0
Original Image
Closing Image
Bottom-hat Image

Practical No 8

Aim- Image Segmentation.

Part A:

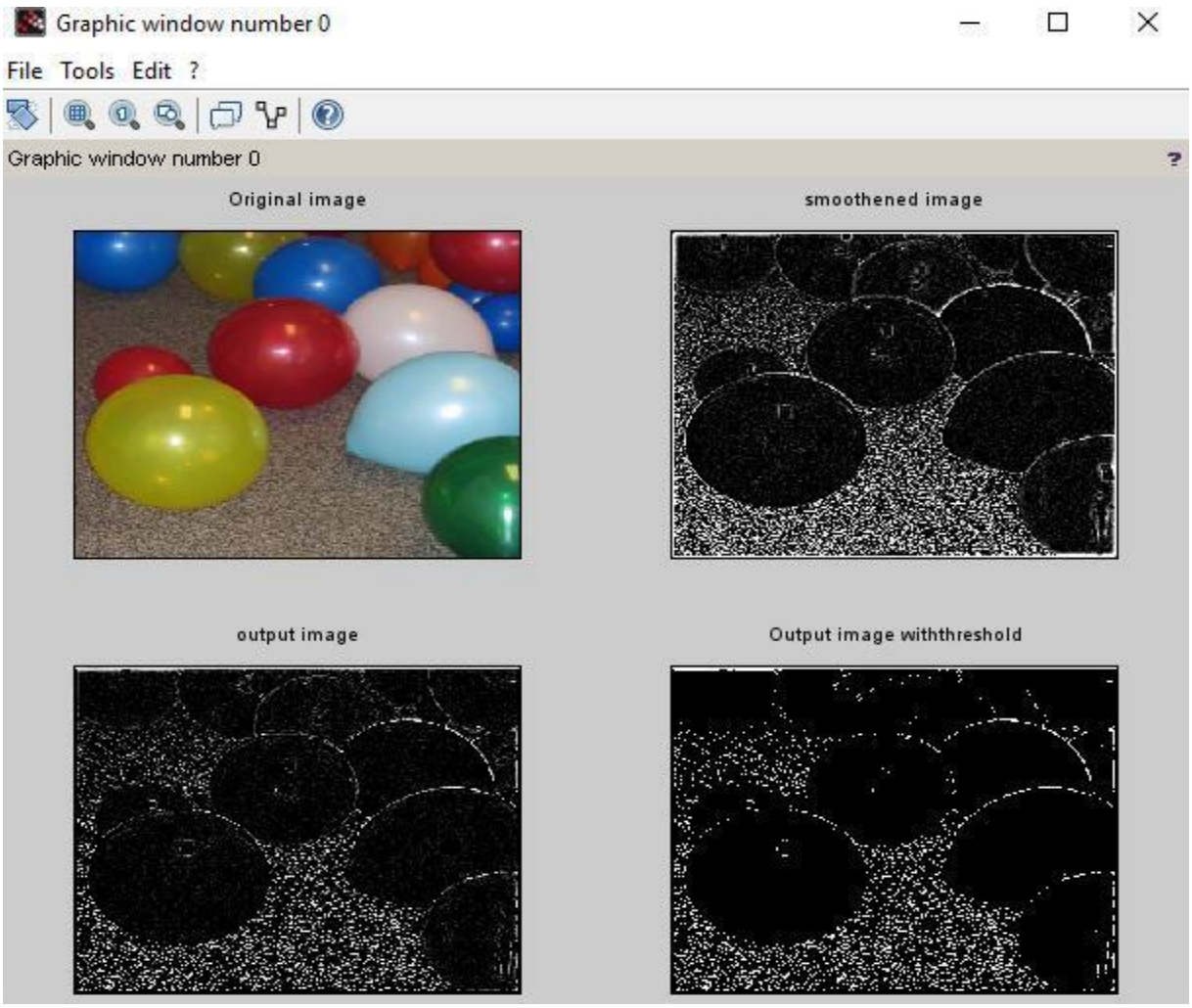
Aim- Program for Edge detection using Sobel, Prewitt.
Edge Detection using Sobel-

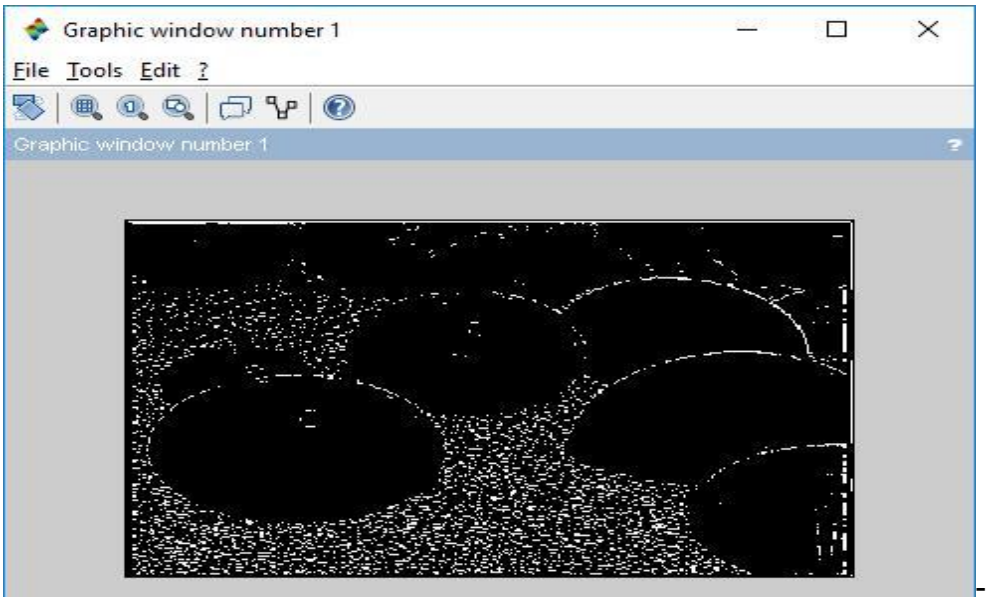
Code-

```
//Marr-Hildreth(Laplassian of Gaussian)
clear all;
im= imread('C:\Program Files (x86)\scilab-
6.0.2\IPCV\images\balloons.png'); im=im2double(im);
//%smoothening the image with a filter
gfilter=[0 0 1 0 0;
         01210;
         12-1621;
         01210;
         00100];
smim=conv2(im,gfilter)
//% finding the zero crossings
[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 && 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
end
otpt=im2uint8(zc);
//%tresholding
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-



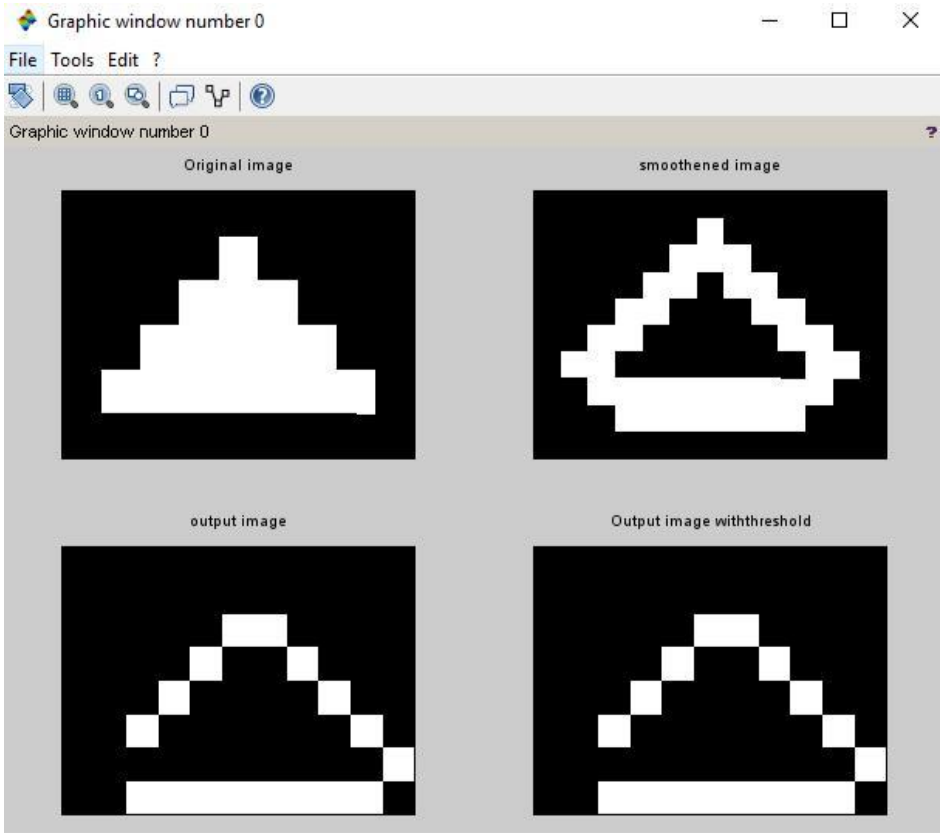


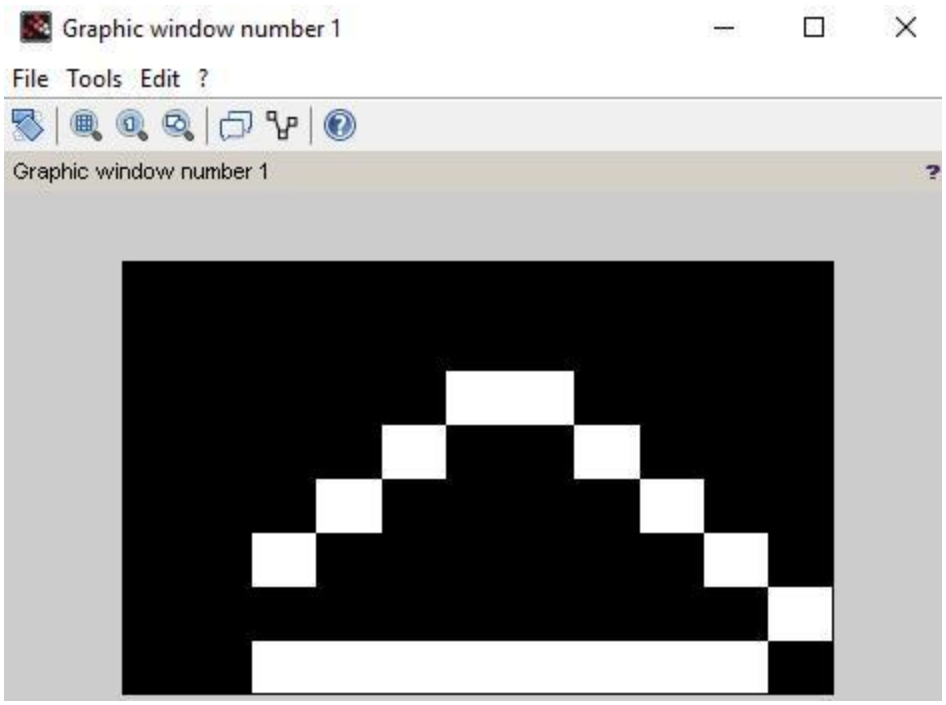
Edge Detection using Prewitt-
Code-

```
//Marr-Hildreth(Laplassian of Gaussian)
clc;
clear all;
//im= imread('C:\Program Files\scilab-6.0.2\IPCV\images\balloons.png');
im = [0 0 0 0 0 0 0 0 0;
     000010000;
     000111000;
     001111100;
     011111110;
     000000000];
im=im2double(im);
//%smoothening the image with a filter
gfilter=[0 0 1 0 0;
          01210;
          12-1621;
          01210;
          00100];
smim=conv2(im,gfilter)
disp(smim)
//% finding the zero crossings
[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
end
otpt=im2uint8(zc);
//%tresholding
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 :




Graphic window number 1
File Tools Edit ?
Graphic window number 1