PRACTICAL 1: BASICS

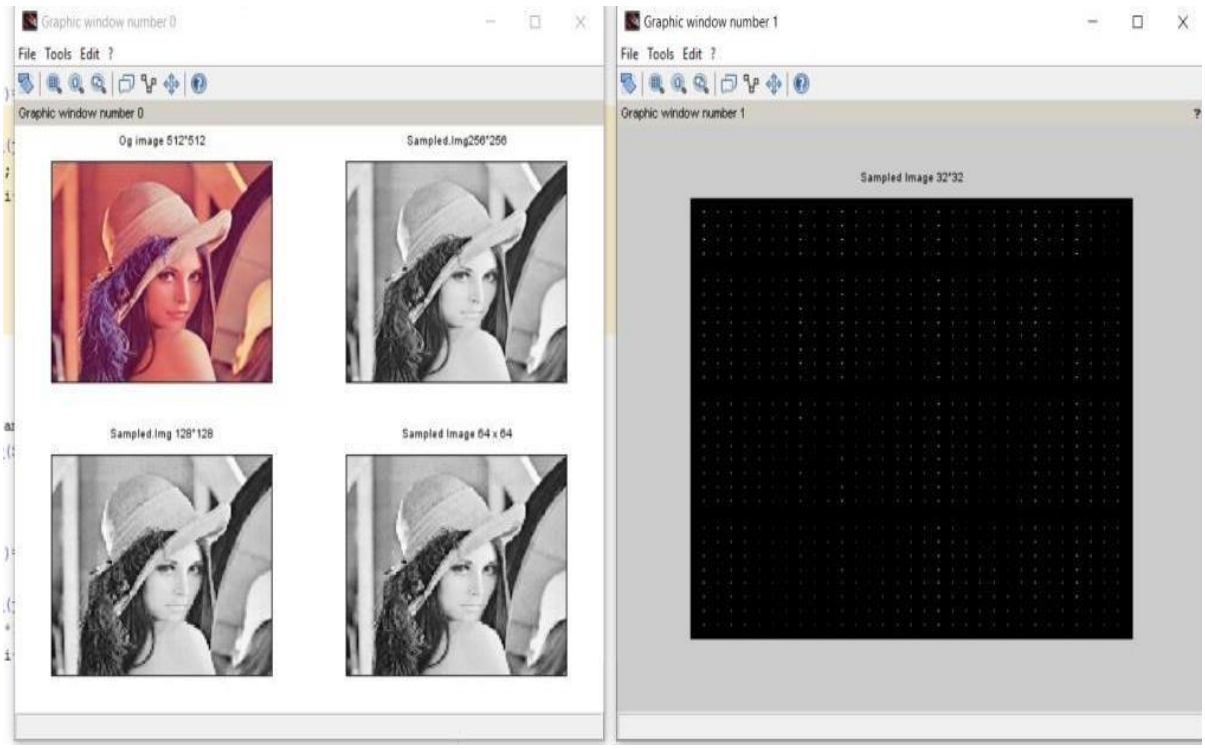
A. Program to study the effects of reducing the spatial resolution of a digital image.

Code:

```
clc; clear
all;
Img=imread('C:\Program Files\scilab-6.1.0\IPCV-
4.1.2src\IPCV\images\lena.png');
subplot(2,2,1),imshow(Img),title('Og image 512*512');
Samp=zeros(256,256);
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          else
m=0;          end      end end
SampImg256=mat2gray(Samp);
subplot(2,2,2),imshow(SampImg256),title('Sampled.Img256*256')
Samp=zeros(128) for
i=1:1:512       for
j=1:1:512           if
modulo(i,4)==0
m=i/4*3;              if
modulo(j,4)==0
n=j/4*3;
                Samp(i-m,j-
n)=Img(i,j);              else n=0;
                  end           else
m=0;           end
    end
 end SampImg128=mat2gray(Samp);
subplot(2,2,3),imshow(SampImg128),title('Sampled.Img 128*128')
Samp=zeros(64) for
i=1:1:512       for
j=1:1:512           if
modulo(i,8)==0
m=i/8*7;              if
modulo(j,8)==0
n=j/8*7;
```

```
                Samp(i-m,j-
n)=Img(i,j);                else
n=0;                end           else
m=0;           end       end
end

SampImg64=mat2gray(Samp);
subplot(2,2,4),imshow(SampImg128),title('Sampled Image 64 x 64'); figure;
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           else
m=0;           end       end end
SampImg32=mat2gray(Samp);
        imshow(SampImg32),title('Sampled Image 32*32');
```

Output:

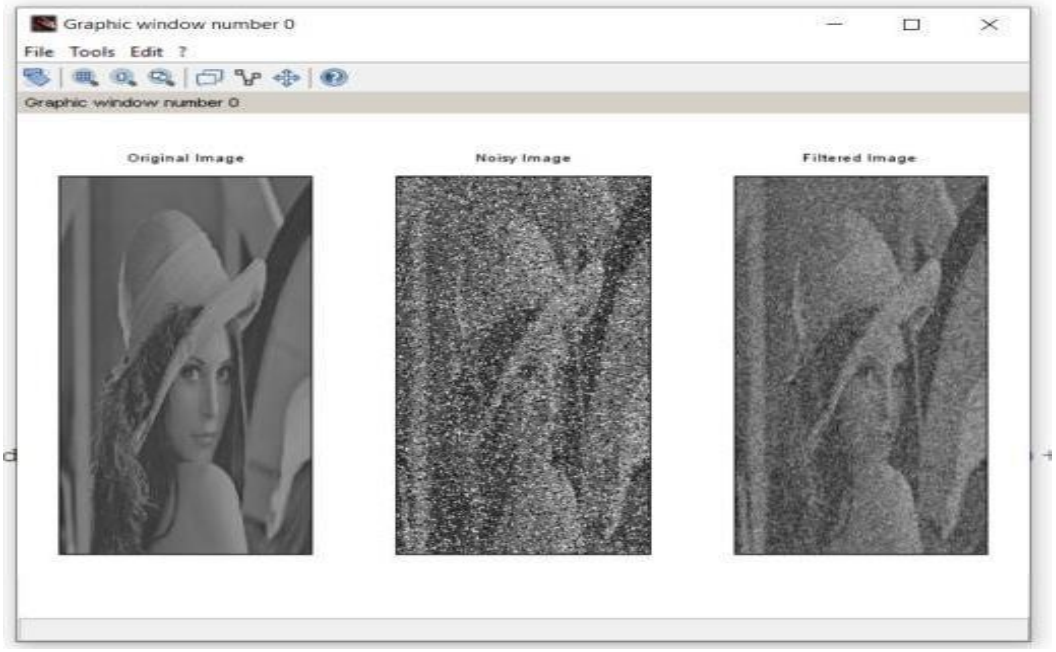


C. Program to perform image averaging (image addition) for noise reduction.

Code:

```
//image
averaging clc;
clear all;
a=uigetfile('*-*','Select the Image:-');
a=imread(a); b=double(a);
c=imnoise(a,'salt &
pepper',0.2); d=double(c);
m=(1/9)*(ones(3,3));
[r1,c1]=size(a); for i=1:r1 for
j=1:c1 new(i,j)=a(i,j);
end end for i=2:1:r1-1 for
j=2:1:c1-1
    new(i,j)=(m(1)*d(i-1,j-1))+(m(2)*d(i-1,j))+(m(3)*d(i-
1,j+1))+(m(4)*d(i,j1))+(m(5)*d(i,j))+(m(6)*d(i,j+1))+(m(7)*d(i+1,j-
1))+(m(8)*d(i+1,j))+(m(9)*d(i+1,j+1));
  end
end
//imshow(uint8(new)); subplot(1,3,1),title('Original
Image'),imshow(a); subplot(1,3,2),title('Noisy
Image'),imshow(c); subplot(1,3,3),title('Filtered
Image'),imshow(uint8(new));
```

Output:



PRACTICAL 2: IMAGE ENHANCEMENT

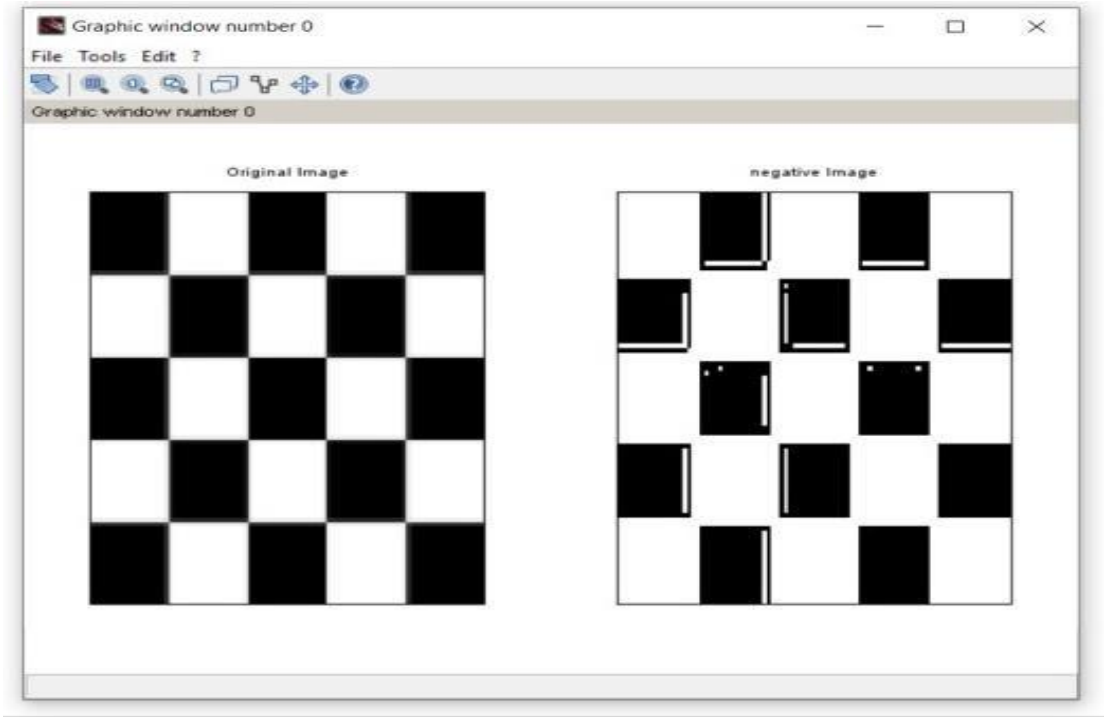
A. Basic Intensity Transformation functions

a. Program to perform Image

negation Code:

```
//image negative clc;
clear all;
a=uigetfile('*.*','Select the Image:-');
a=imread(a); new=255-
a; new=double(new);
//new=uint8(new); if we use uint8 instead of double then we will not get the output as expected subplot(1,2,1),imshow(a),title('Original Image');
subplot(1,2,2),imshow(new);title('negative Image');
```

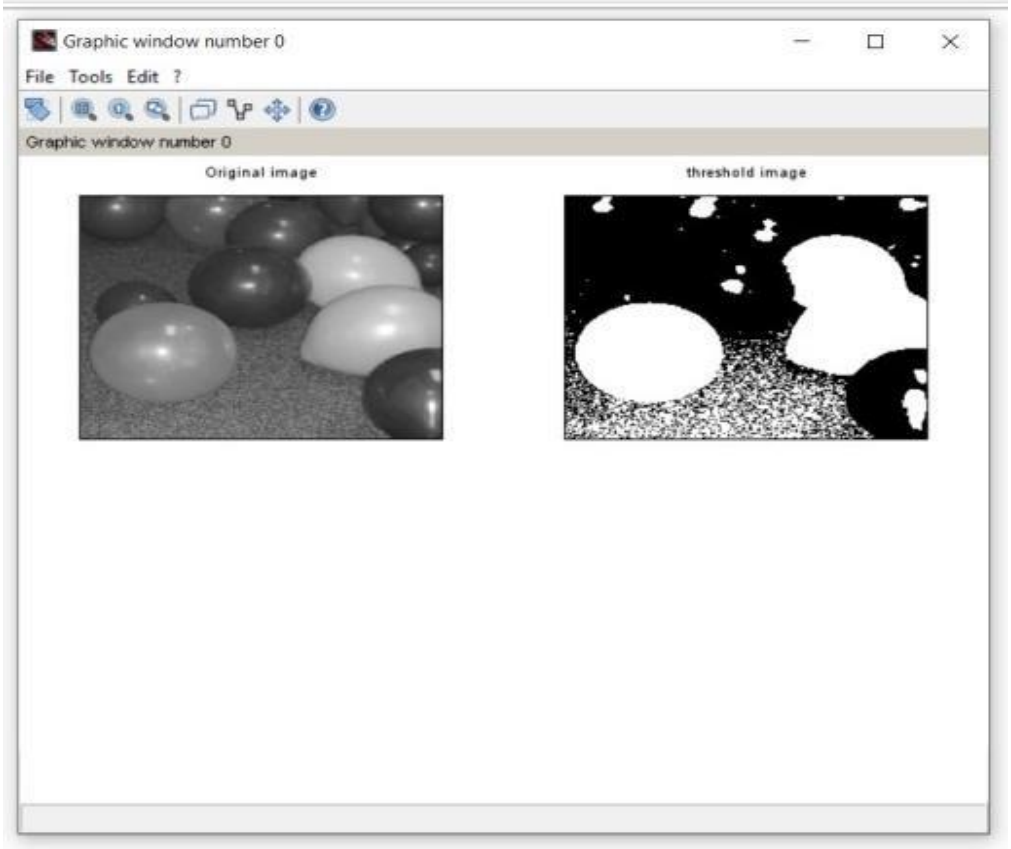
Output:



b. Program to perform threshold on an image. Code:

```
//Thresholding a=uigetfile('*.*','Select the
Image:-'); a=imread(a);
subplot(2,2,1),imshow(uint8(a)),title('Original image');
T=input('Enter the threshold value') //threshhold value [r,c]=size(a);
                        for
i=1:r         for j=1:c
if            (a(i,j)<=T)
x(i,j)=0;             else
x(i,j)=255;
end           end end
x=uint8(x);
subplot(2,2,2),imshow(x),title('threshold image');
```

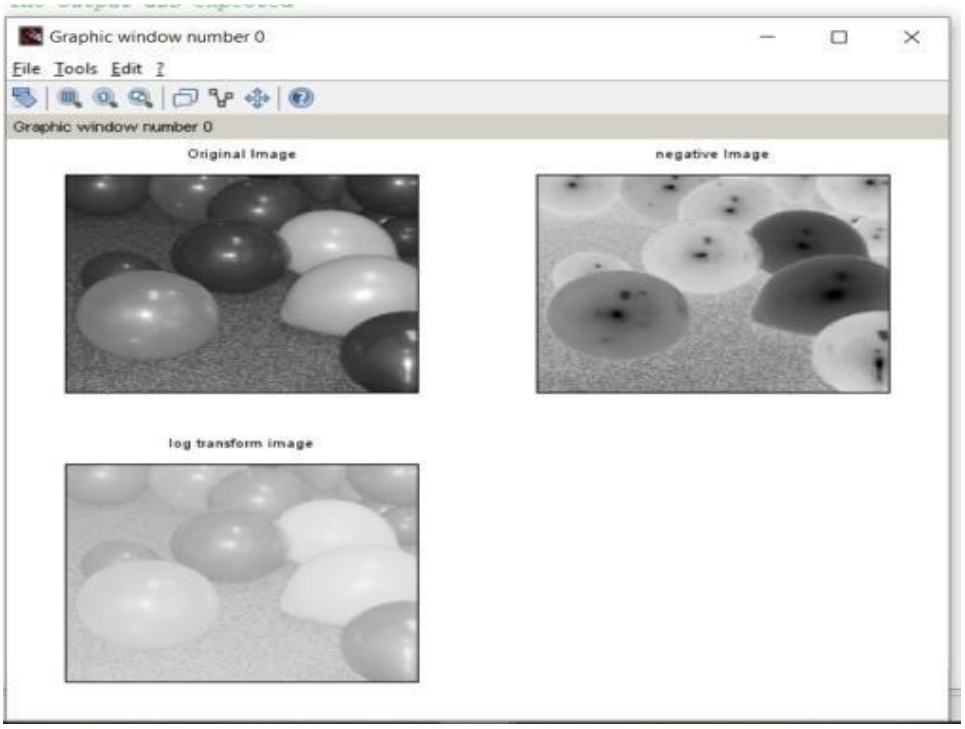
Output:



c. Program to perform Log transformation Code:

```
//image negative clc;
clear all;
a=uigetfile('*.*','Select the Image:-');
a=imread(a); new=255-
a;
//new=double(new); //new=double(new); //if we use uint8 instead of double then we will get the output ass expected
new=uint8(new); subplot(2,2,1),imshow(a),title('Original
                                                Image');
subplot(2,2,2),imshow(new);title('negative Image');

//log transform
c=1;
[r1,c1]=size(a); for
i=1:r1      for j=1:c1
b=double(a(i,j));
s(i,j)=c*log10(1+b);
end end
new=s;
new1=uint8(new*100);
subplot(2,2,3),imshow(new1);title('log transform image');
```

Output:

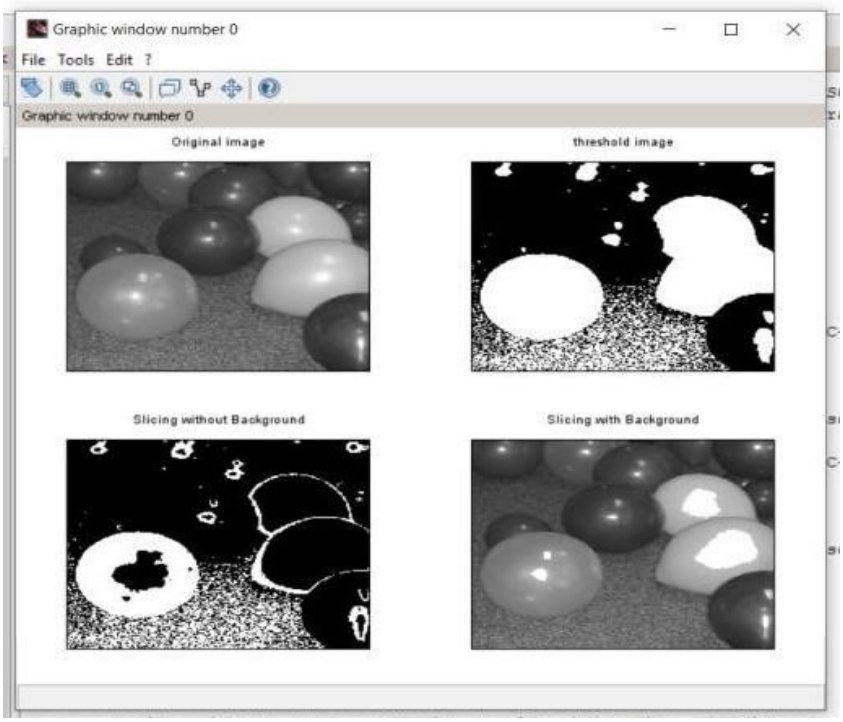


f. Gray-level slicing with and without background. Code:

```
//Thresholding a=uigetfile('*.*','Select the
Image:-'); a=imread(a);
subplot(2,2,1),imshow(uint8(a)),title('Original
image');
T=input('Enter the threshold value') //threshhold value [r,c]=size(a);
for
i=1:r              for
j=1:c                if
(a(i,j)<=T)
x(i,j)=0;
else
x(i,j)=255;
end       end end
x=uint8(x);
subplot(2,2,2),imshow(x),title('threshold image');

//Gray level slicing without Background

a1=input('Enter the lower threshold');
b1=input('Enter the higher threshold');
[r,c]=size(a
);  for i=1:r
for j=1:c
        if (a(i,j)>a1 & a(i,j)<b1)

x(i,j)=255;
else
x(i,j)=0;
end       end end
x=uint8(x);
subplot(2,2,3),imshow(x),title('Slicing without Background');

//Gray Level Slicing With Backgraound

a1=170; //This value is user
defined b1=190; //This value is user
defined
```

```
[r,c]=size(a
); for i=1:r for
j=1:c
        m=a(i,j);          if
((m>a1) & (m>b1))
x(i,j)=255;          else
x(i,j)=a(i,j);          end
end end x=uint8(x);
subplot(2,2,4),imshow(x),title('Slicing with Background');
```

Output:

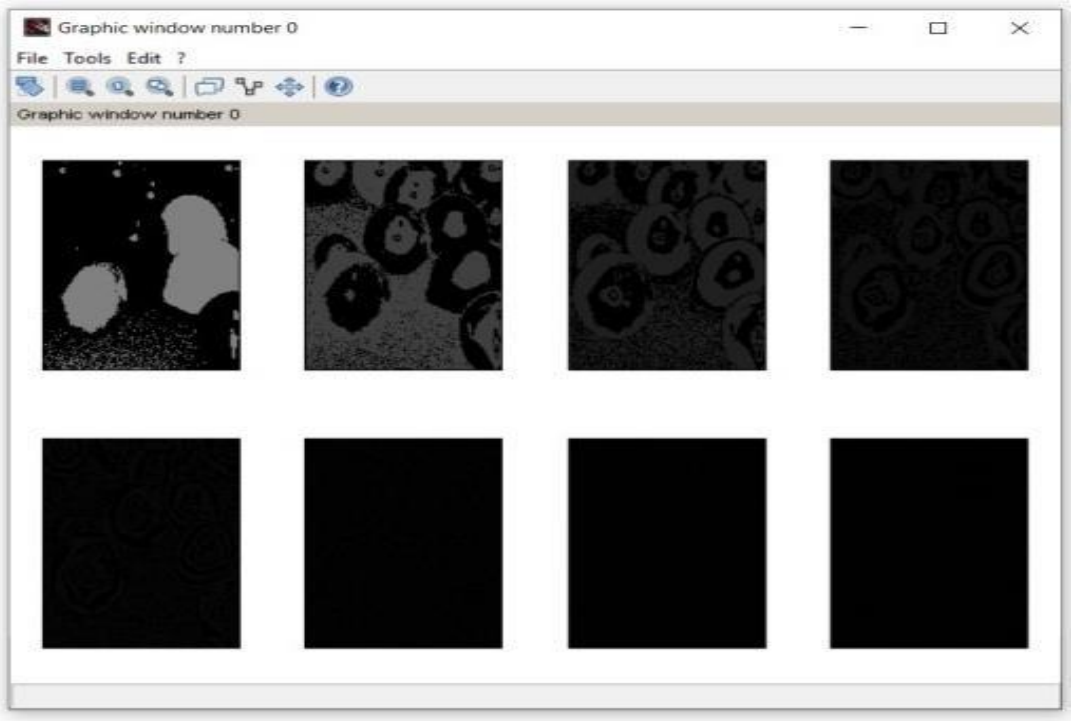


g. Bit-plane slicing

Code:

```
//Bit Plane Slicing
f=uigetfile('*-*','Select the Image:-');
f=imread(f); f=double(f); [r,c]=size(f);
com=[128 64 32 16 8 4 2 1];
for k=1:1:length(com);
for i=1:r;          for j=1:c
            new(i,j)=bitand(f(i,j),com(k));
end
    end subplot(2,4,k),imshow(uint8(new))
end
```

Output:

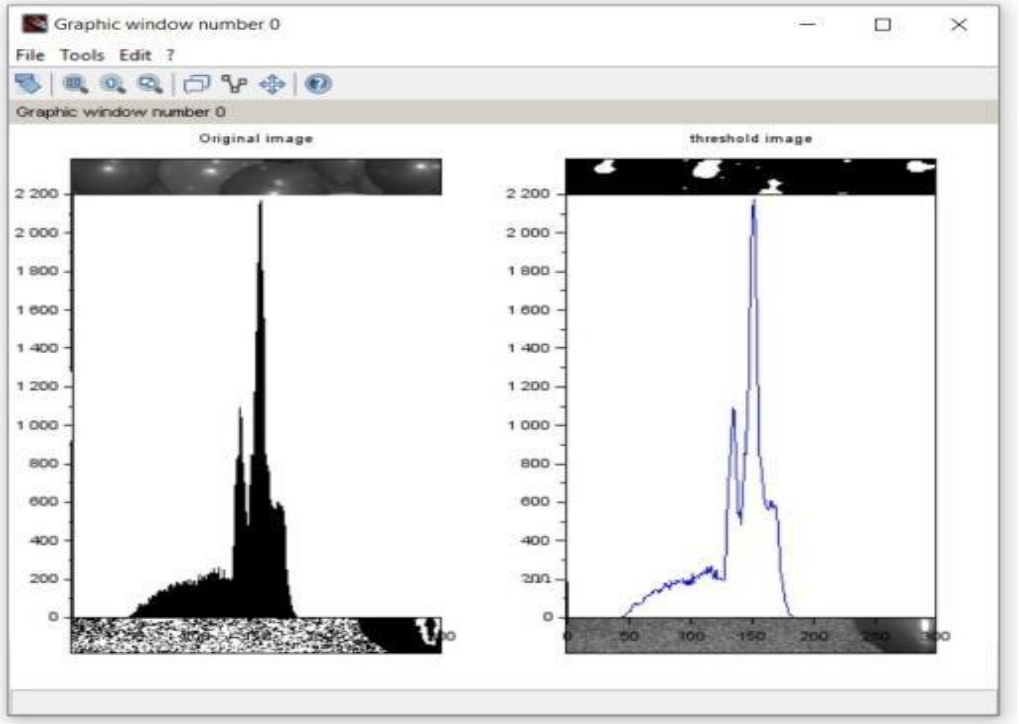


B. Histogram

a. Program to plot the histogram of an image and categorise Code:

```
//Histogram
a=uigetfile('*-*','Select the Image:-');
a=imread(a) subplot(1,2,1),imhist(a,256,0.
5)     r=size(a,1);        c=size(a,2);
h=zeros(1,256);  for  i=1:r  for  j=1:c
                              if  (a(i,j)==0)
a(i,j)=1;  end  k=a(i,j);  h(k)=h(k)+1;
                                  end      end
subplot(1,2,2)
plot(h);
```

Output:

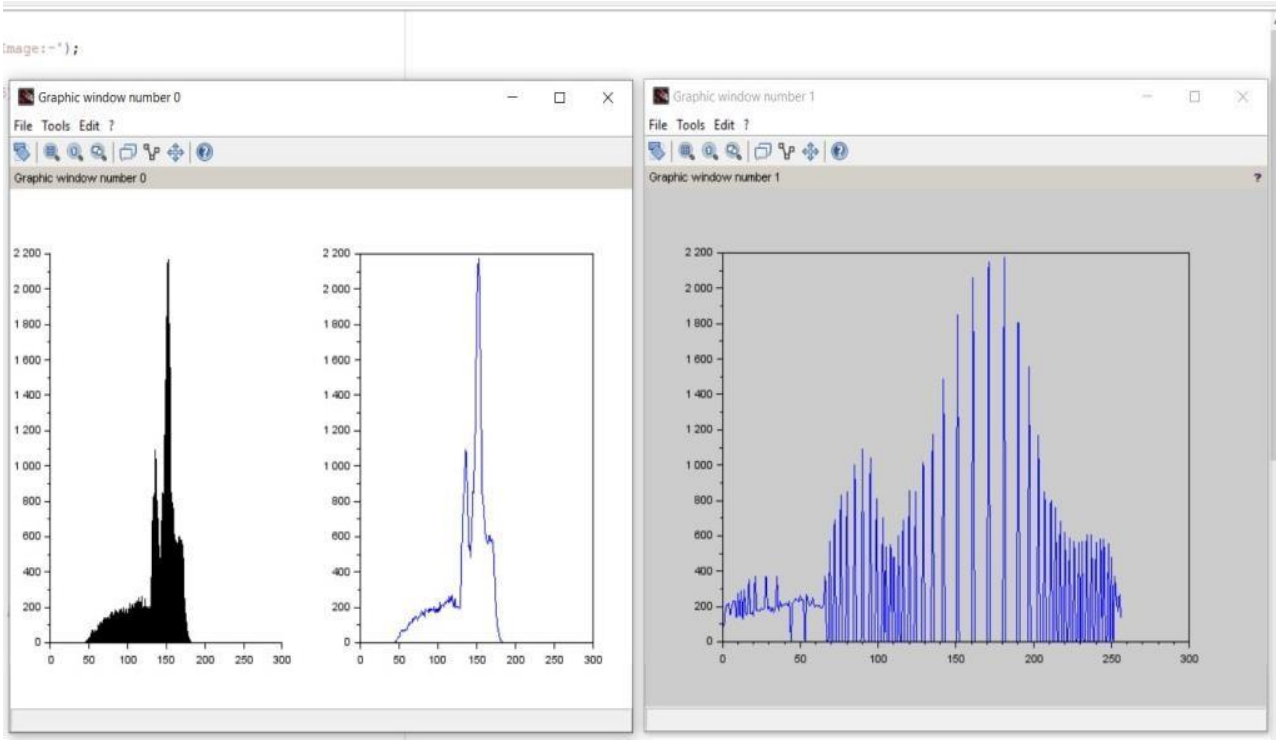


b. Program to apply histogram equalization

Code:

```
//Histogram
a=uigetfile('*-*','Select the Image:-');
a=imread(a) subplot(1,2,1),imhist(a,256,0.
5)      r=size(a,1);          c=size(a,2);
h=zeros(1,256);  for  i=1:r  for  j=1:c
                              if   (a(i,j)==0)
a(i,j)=1;  end  k=a(i,j);  h(k)=h(k)+1;
                                  end      end
subplot(1,2,2) plot(h);

//Histogram Equalization
figure; a=double(a);
big=max(max(a))
; [r,c]=size(a); tot
= r*c;
h=zeros(1,256); //to store the histogram values
z=zeros(1,256); for i=1:1:r              for j=1:1:c
                        if a(i,j)==0                  a(i,j)=1;
end        end end for i=1:1:r              for j=1:1:c
t=a(i,j);  h(t)=h(t)+1;
    end end pdf=h/tot;
cdf(1)=pdf(1); for
i=2:1:big
cdf(i)=pdf(i)+cdf(i-1);
end new=round(cdf*big);
new=new+1;
for i=1:1:r          for
j=1:1:c
temp=a(i,j);
b(i,j)=new(temp);
t=b(i,j);
z(t)=z(t)+1;          end
end plot(z);
```

Output:


Graphic window number 0
File Tools Edit ?
Graphic window number 0
Graphic window number 1
File Tools Edit ?
Graphic window number 1

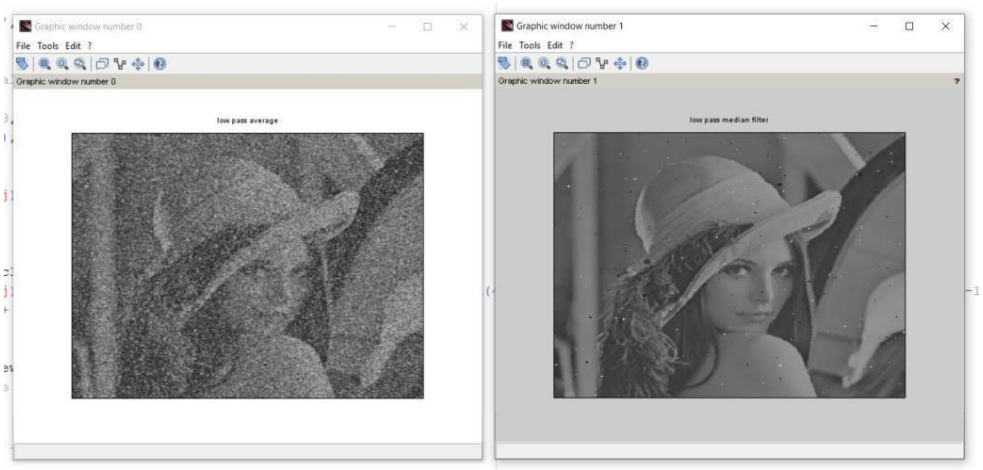
B. Write a program to apply smoothing and sharpening filters on grayscale and color images

a. Low Pass

Code:

```
//wap to apply smoothing and sharpening filter on grayscale and color
//low pass average filter
clear
all;
a=uigetfile('.','select the image:-');
a=imread(a);
b=double(a); c=imnoise(a,'salt
& pepper',0.2); d=double(c);
m=(1/9)*(ones(3,3));
[r1,c1]=size(a); for i=1:r1 for
j=1:c1 new(i,j)=a(i,j); end
end for i=2:1:r1-1 for
j=2:1:c1-1
new(i,j)=(m(1)*d(i-1,j-
1))+(m(2)*d(i-1,j))+(m(3)*d(i-
1,j+1))+(m(4)*d(i,j1))+(m(5)*d(i,
j))+(m(6)*d(i,j+1))+(m(7)*d(i+1,j
-  1))+(m(8)*d(i+1,j))+(m(9)*d(i+1,j+1));
end
end
imshow(uint8(new));
title('low pass average')

//code for the low-pass median filter
for i=2:r1-1 for j=2:c1-1
        a1=[d(i-1,j-1) d(i-1,j) d(i-1,j+1) d(i,j-1) d(i,j) d(i,j+1) d(i+1,j-1)d(i+1,j)
d(i+1,j+1)]; a2=gsort(a1); med=a2(5);
        b(i,j)=med;
end end figure;
imshow(uint8(b))
;
title('low pass median filter');
```

Output:

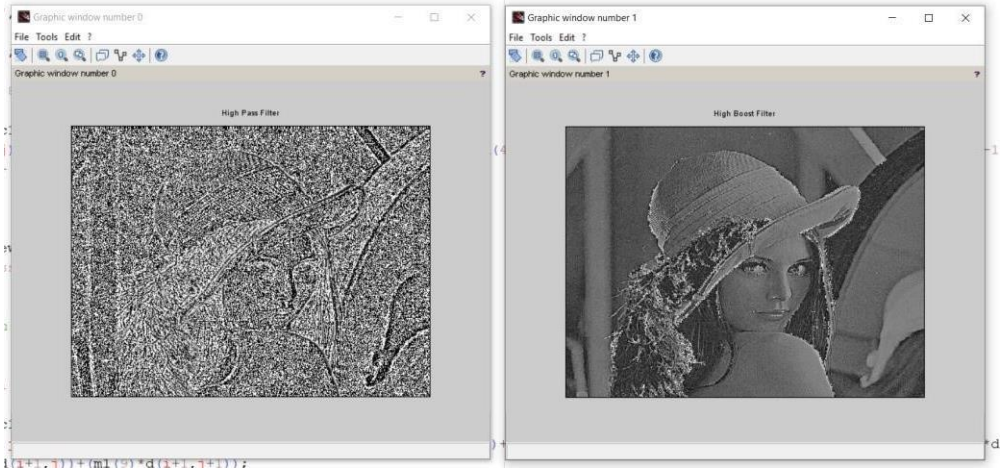

Graphic window number 0
File Tools Edit ?
Graphic window number 0
low pass average
Graphic window number 1
File Tools Edit ?
Graphic window number 1
low pass median filter

b. High Pass

Code:

```
//code for high pass filter
clc;
clear
all;
a=uigetfile('.','select the image:-');
a=imread(a);
[r1,c1]=size(a);
d=double(a);
m=[-1 -1 -1;-1 8 -1;-1 -1 -
1]; for i=2:1:r1-1 for
j=2:1:c1-1
        new(i,j)=(m(1)*d(i-1,j-1))+(m(2)*d(i-1,j))+(m(3)*d(i-
1,j+1))+(m(4)*d(i,j1))+(m(5)*d(i,j))+(m(6)*d(i,j+1))+(m(7)*d(i+1,j-
1))+(m(8)*d(i+1,j))+(m(9)*d(i+1,j+1));
end end
figure;
imshow(uint8(new));
title('High Pass Filter');

//code for high boost filter
A=1.1;
BB=(9*A)-1;
m1=[-1 -1 -1;-1 BB -1;-1 -1 -
1]; for i=2:1:r1-1 for
j=2:1:c1-1
        new1(i,j)=(m1(1)*d(i-1,j-1))+(m1(2)*d(i-1,j))+(m1(3)*d(i-
1,j+1))+(m1(4)*d(i,j-1))+(m1(5)*d(i,j))+(m1(6)*d(i,j+1))+(m1(7)*d(i+1,j-
1))+(m1(8)*d(i+1,j))+(m1(9)*d(i+1,j+1));
end end
figure;
imshow(uint8(new1));
title('High Boost Filter');
```

Output:

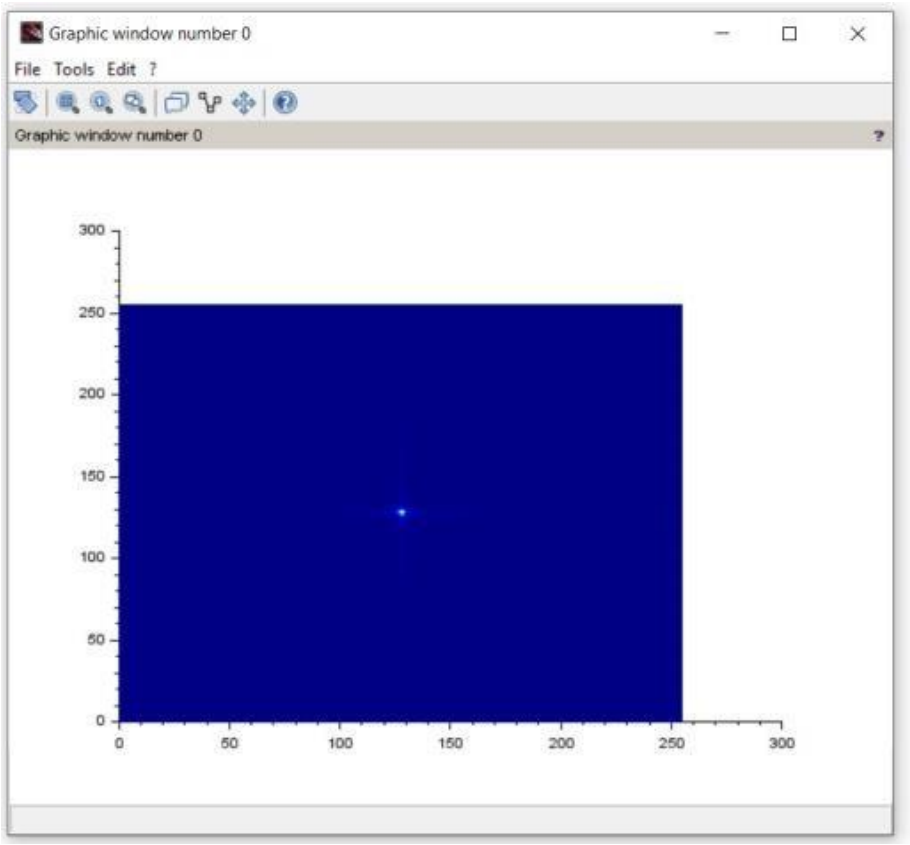

Graphic window number 0
File Tools Edit ?
Graphic window number 0
High Pass Filter
Graphic window number 1
File Tools Edit ?
Graphic window number 1
High Boost Filter

3. Filtering in Frequency Domain

A. Program to apply Discrete Fourier Transform on an image Code:

```
clc;
clear
all;
a=uigetfile('* . *', 'select an Image');
a=imread(a);
a=double(a);
X = fftshift(fft(a));
set(gcf(), "color_map" , jetcolormap(128));
clf;grayplot(0:255, 0:255, abs(X)')
```

Output:

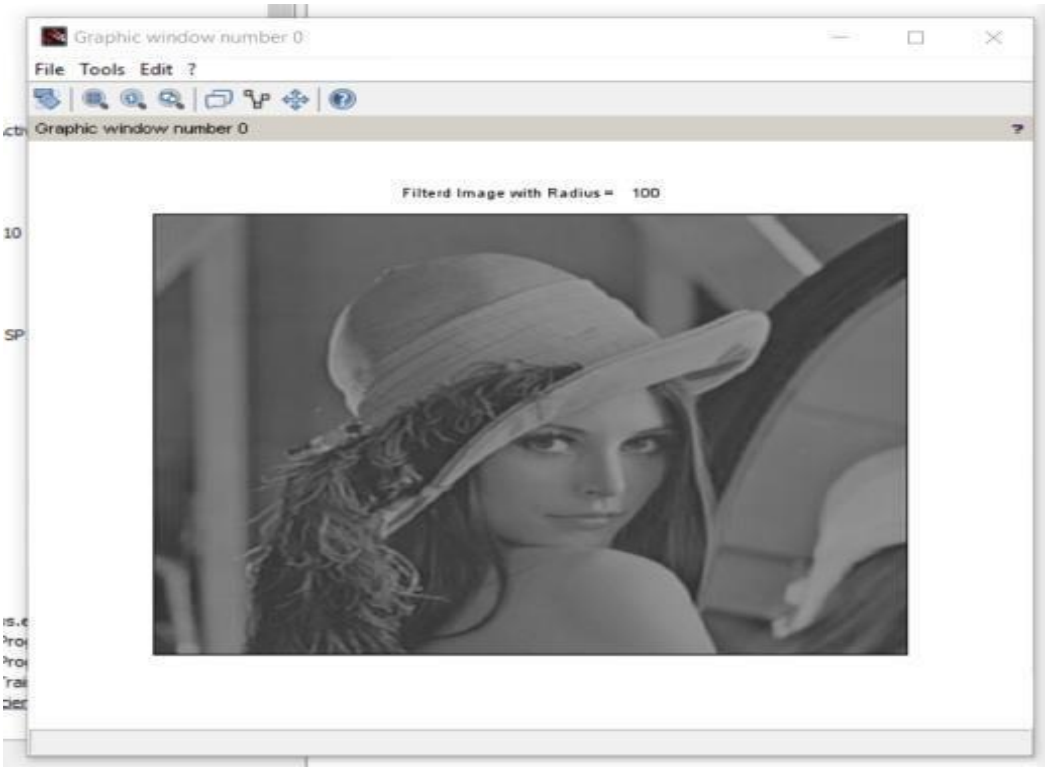


B. Program to apply Low pass and High pass filters in frequency domain

Code: Low Pass Filter Frequency Domain

```
//Ideallowfrequen cy
clc; clear all;
a=uigetfile('* . *', 'select an Image');
a=imread(a);
a=double(a);
r=size(a,1);
c=size(a,2);
d0=input('Enter the cut-off frequency - (Radius) :- ' ); for
u=1:1:r                    for v=1:1:c
d=(((u- (r/2))^2) + ((v-(c/2))^2))^0.5;
if d<d0        h(u,v)=1; else          h(u,v)=0;
end end end
new=zeros(size(h)); b=fft2(a);
c=fftshift(b); c1=uint8(c);
new=c.*h; new2=uint8(new);
new1=abs(ifft(new));
imshow(uint8(new1));title(['Filterd Image with Radius = ' , string(d0)])
```

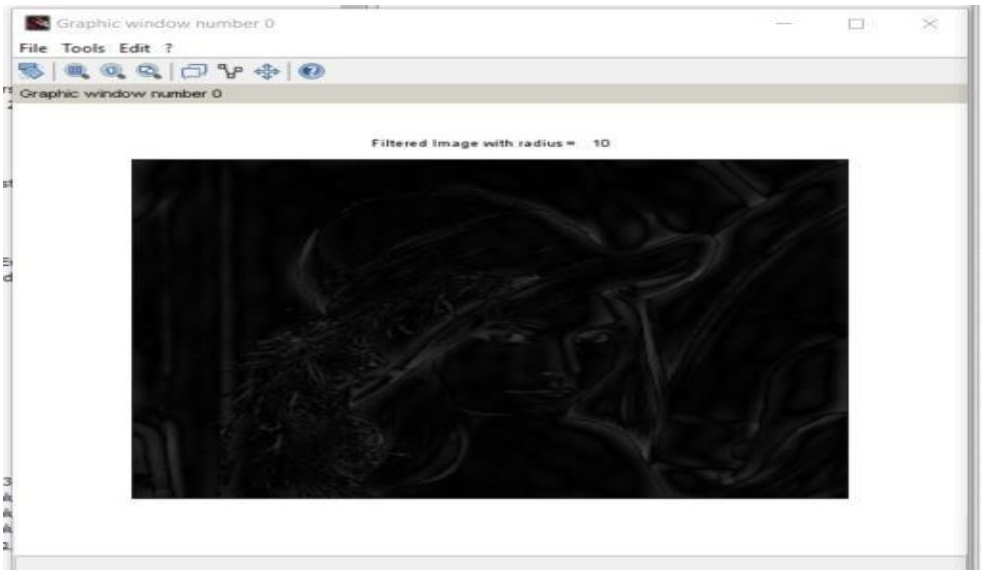
Output:



Code: High Pass Filter Frequency Domain

```
//IdealHighFrequency
clc;
clear
all;
a=uigetfile('*.*','Select the Image');
a=imread(a);
a=double(a);
r=size(a,1);
c=size(a,2);
d0=input('Enter the cut-off frequency -(Radius):- '); for
u=1:1:r                  for v=1:1:c
        d=(((u-(r/2))^2)+                    ((v-
  (c/2))^2))^0.5;                        if   d<=d0
h(u,v)=0;            else              h(u,v)=1;
        end
end
end
new=zeros(size(h));
b=fft2(a);
//p_original=(abs(b))^2+(atan(imag(b),real(b)))^2;
//p_original=sum(sum(p_original))
c=fftshift(b);
c1=uint16(c);
new=c.*h;
new2=uint8(new);
new1=abs(ifft(new));
imshow(uint8(new1));title(['Filtered Image with radius =', string(d0)]);
```

Output:



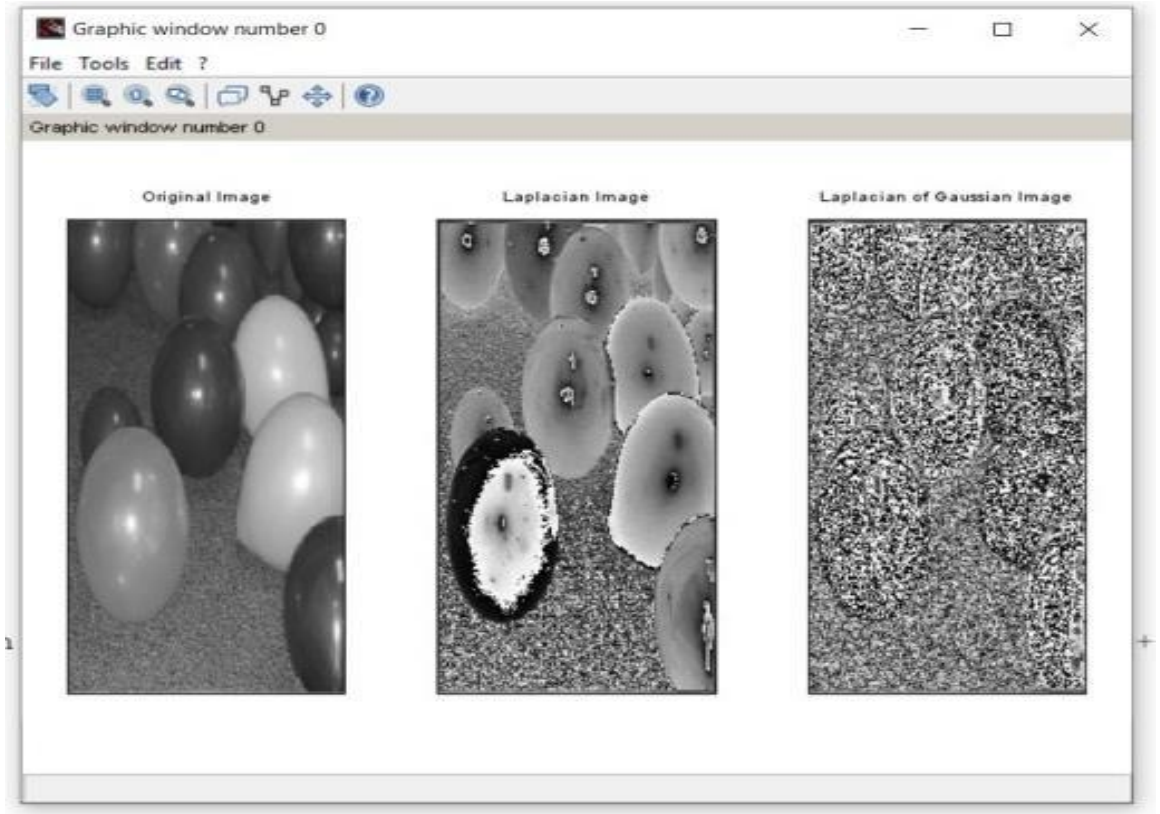
C. Program to apply Laplacian filter in frequency domain

Code:

```
clc;
clear
all;
a=uigetfile('*-*','Select the Image:-');
a=imread(a);
b=double(a);
d=b;
mh=[0,-1,0;1,-4,1;0,1,0]; mv=[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];
[r1,c1]=size(a);
for i=1:r1        for
j=1:c1
LAP(i,j)=a(i,j);
LoG(i,j)=a(i,j);
    end
end
new=double(LA
P);
nem=double(Lo
G); for
i=3:1:r1-2
for j=3:1:c1-2
        new(i,j)=(mh(1)*d(i-1,j-1))+(mh(2)*d(i-1,j))+(mh(3)*d(i-
1,j+1))+(mh(4)*d(i,j-
1))+(mh(5)*d(i,j))+(mh(6)*d(i,j+1))+(mh(7)*d(i+1,j-
1))+(mh(8)*d(i+1,j))+(mh(9)*d(i+1,j
+1));      end end for i=3:1:r1-2 for
j=3:1:c1-2
        nem(i,j)=(mv(1)*d(i-2,j-2))+(mv(2)*d(i-2,j-1))+(mv(3)*d(i-
2,j))+(mv(4)*d(i-2,j+1))+(mv(5)*d(i-2,j+2))+(mv(6)*d(i-1,j-
2))+(mv(7)*d(i-
1,j-1))+(mv(8)*d(i-1,j))+(mv(9)*d(i-1,j+1))+(mv(10)*d(i-
1,j+2))+(mv(11)*d(i,j-2))+(mv(12)*d(i,j-
1))+(mv(13)*d(i,j))+(mv(14)*d(i,j+1))+(mv(15)*d(i,j+2))+(mv(16)*d(i+1
,j2))+(mv(17)*d(i+1,j-
```

```
1))+(mv(18)*d(i+1,j))+(mv(19)*d(i+1,j+1))+(mv(20)*d(i+1,j+2))+(mv(
21)*d(i
+2,j-2))+(mv(22)*d(i+2,j-
1))+(mv(23)*d(i+2,j))+(mv(24)*d(i+2,j+1))+(mv(25)*d(i+2,
j+2));       end end subplot(131);imshow(uint8(a));title('Original
Image'); subplot(132);imshow(uint8(new));title('Laplacian
Image');
subplot(133);imshow(uint8(nem));title('Laplacian of Gaussian Image');
```

Output:



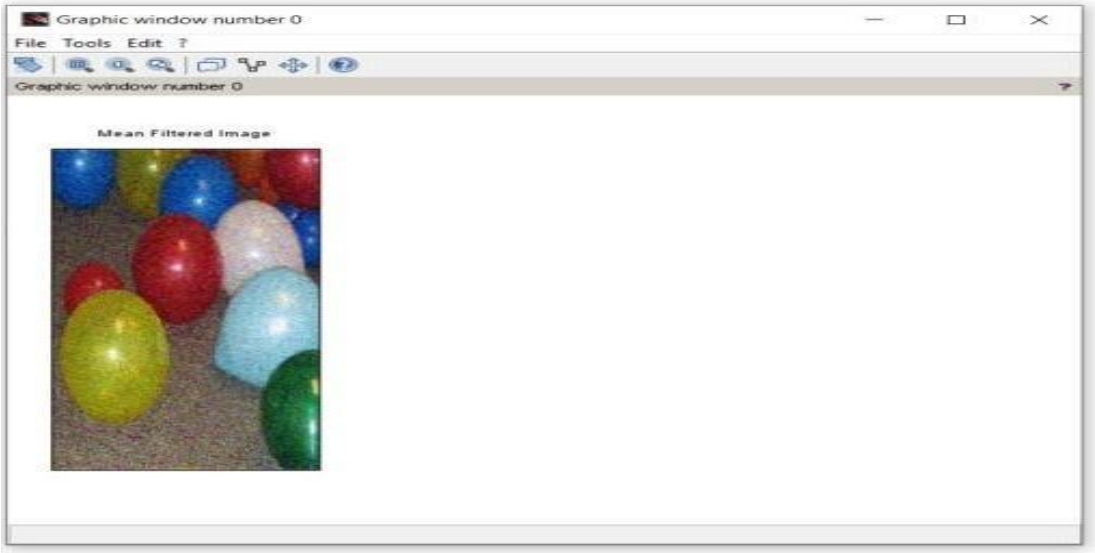
4. Image Denoising

A. Program to denoise using spatial mean, median and adaptive mean filtering

Code:

```
//Mean filter
        clc;
clear all;
a=uigetfile('*.*','Select image');
a=imread(a);
b1=double(a);
c=imnoise(a,'gaussian ');
d=double(c); b=d;
m=(1/9)*(ones(3,3));
[r1,c1]=size(a); for
i=2:r1-1        for j=2:c1- 1
        a1=d(i-1,j-1)+d(i-1,j)+d(i,j-1)+d(i,j)+d(i,j)+d(i,j+1)+d(i+1,j-
1)+d(i+1,j)+d(i+1,j+1);
b(i,j)=a1*(1/9);
        end
end
subplot(1,3,1);
imshow(uint8(b));title('Mean Filtered Image');
```

Output:



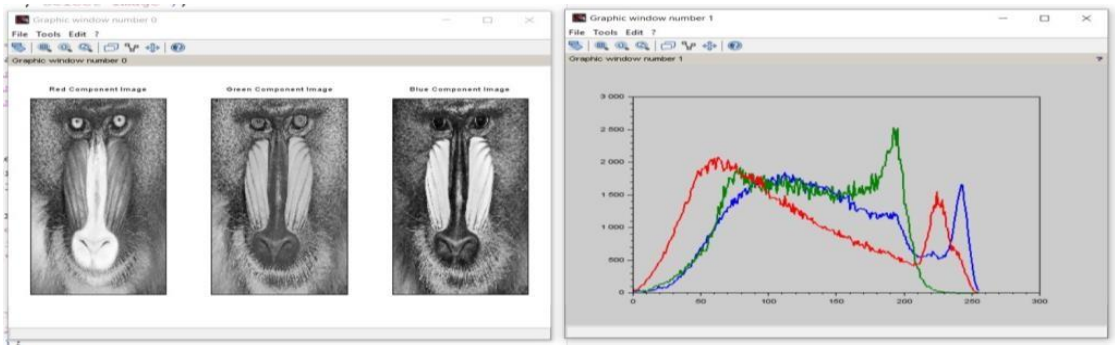
5. Color Image Processing

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

Code:

```
clc;
clear
all;
a=uigetfile('*.*','Select image');
a=imread(a);
Red_Component=a(:,:,1);
Green_Component=a(:,:,2); Blue_Component=a(:,:,3);
subplot(131);imshow(Red_Component);title('Red Component Image');
subplot(132);imshow(Green_Component);title('Green Component Image');
subplot(133);imshow(Blue_Component);title('Blue Component Image'); figure;
nBins = 256;

[yR,x]=imhist(Red_Component,nBins);
[yG,x]=imhist(Green_Component,nBins);
[yB,x]=imhist(Blue_Component,nBins);

plot(x,yR,x,yG,x,yB,"Linewidth",2);
xlable("RGB Intensity"); ylable("No of
Pixles");
set(gca(),"grid",[1,1]);

figure;
CMY = ncomplete(a); imwrite(CMY,'CMY.tif');
subplot(1,3,1);imshow(CMY);title('Image in CMY color space'); HSV =
rgb2hsv(a); imswrite(HSV,'HSV.tif');
subplot(123);imshow(YCC);title('Image in YCB Color space')
```

Output:



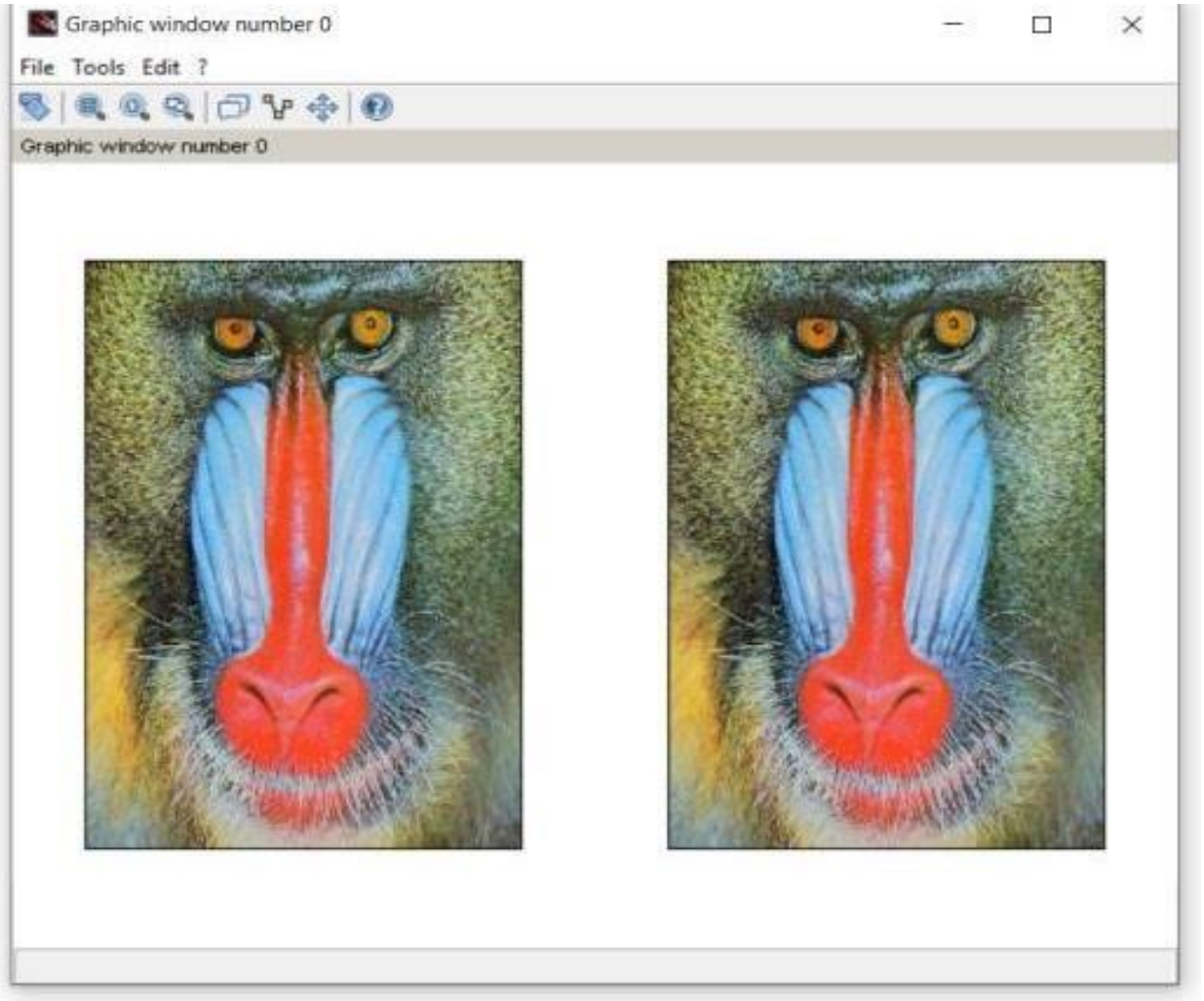
6. Image Compression

Program to apply compression and decompression algorithm on an image

Code:

```
//Compression
//Title: Error-Free Image Compression
clc; clear all; a=uigetfile('*.*','Select the
Image:-'); a=imread(a); im=[];
singvals=20; if typeof(a)=='uint8'
then                                red
= double(a(:,:,1));          green =
double(a(:,:,2));          blue =
double(a(:,:,3));          [u,s,v] =
svd(red,singvals);
imred=uint8(u*s*v');  [u,s,v]
= svd(green,singvals);
imgreen=uint8(u*s*v'); [u,s,v] =
svd(blue,singvals);
imblue=uint8(u*s*v');
im1(:,:,1)=imred;
im1(:,:,2)=imgreen;
im1(:,:,3)=imblue; Image5=a;
                    Image6=im1;
imwrite(a,'Image5.jpg');
imwrite(im1,'Image6.jpg');
subplot(1,2,1);
imshow(Image5);
subplot(1,2,2);
imshow(Image6);

//disp(info);
end
```

Output:


Graphic window number 0
File Tools Edit ?
Graphic window number 0

7. Morphological Image Processing

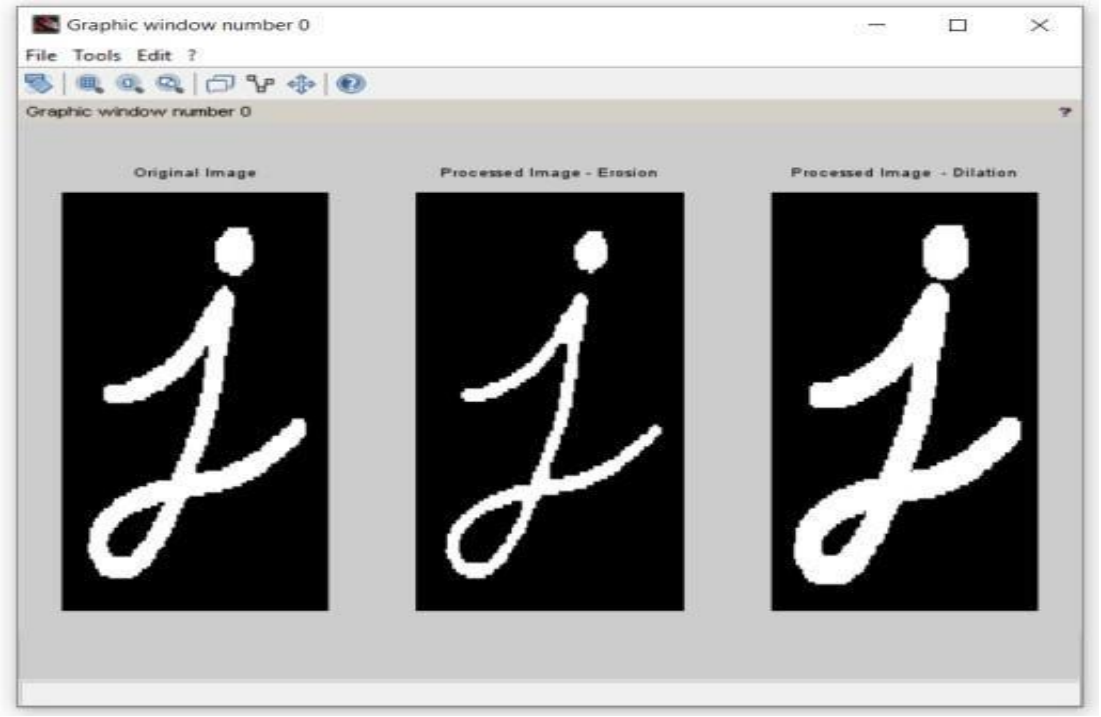
A. Program to apply erosion, dilation, opening, closing ,Thinning and Thinkening

Code:

```
clc;
clear
all;
//Program to implement erosion and dilation
a=uigetfile('*-*','Select the Image'); a=imread(a);
//a=[0 0 0 0 0
0 0 0
//      0 0 1 1 1 1 0 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1
1 0 //       0 0 0 0
0 0 0 0];
d=a;
[r,c]=size(d);
m=ones(3,3);

for i=2:1:r-1 for
    j=2:1:c-1

        new=[(m(1)*d(i-1,j-1)) (m(2)*d(i-1,j)) (m(3)*d(i-1,j+1))
(m(4)*d(i,j-1)) (m(5)*d(i,j)) (m(6)*d(i,j+1))
        (m(7)*d(i+1,j-1)) (m(8)*d(i+1,j)) (m(9)*d(i+1,j+1))];
        A1(i,j)=min(new);
A2(i,j)=max(new);
    end
end
figure;
subplot(1,3,1),imshow(a),title('Original                    Image');
subplot(1,3,2),imshow(A1);title('Processed Image  -
```

Erosion'); subplot(1,3,3),imshow(A2);title('Processed Image - Dilation');

Output:

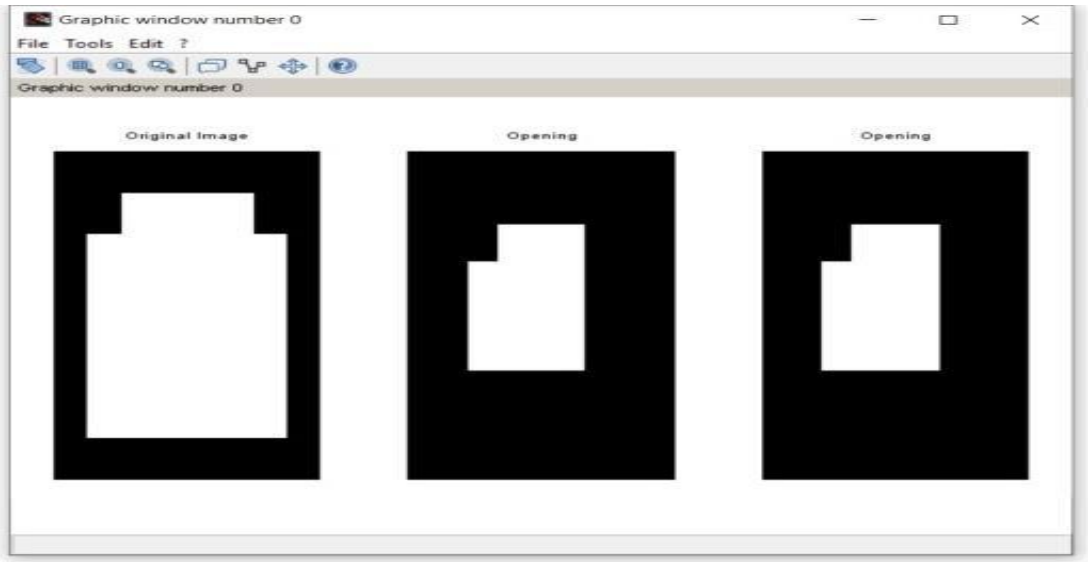


Code:

```
clc; clear all; // opening
d=[0 0 0
0 0 0 0 0 0          0
0 0 0 0 0 0 0 0 0
    0 0 0 1 1 1 0 0 0
    0 0 1 1 1 1 0 0 0
    0 0 1 1 1 1 0 0 0
    0 0 1 1 1 1 0 0 0
    0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0];
// a=uigetfile('.','Select the Image');
// a= imread(a);
//d=1;
//d1=d; [r,c]=size(d);
A2=zeros(r,c);
A1=zeros(r,c); m=[1 1
1;1 1 1;1 1 1]; for
```

```
i=2:1:r-1        for
j=2:1:c-1
new=[(m(1)*d(i-1,j-1)) (m(2)*d(i-1,j))  (m(3)*d(i-1,j+1))

(m(4)*d(i,j-1)) (m(5)*d(i,j))  (m(6)*d(i,j+1))
(m(7)*d(i+1,j-1))  (m(8)*d(i+1,j))       (m(9)*d(i+1,j+1))  ];
A2(i,j)=min(new)
; end end

[r,c]=size(d);
for i=2:1:r-1
    for j=2:1:c-1
new=[(m(1)*A2(i-1,j-1)) (m(2)*A2(i-1,j))  (m(3)*A2(i-1,j+1))

(m(4)*A2(i,j-1)) (m(5)*A2(i,j))  (m(6)*A2(i,j+1))
(m(7)*A2(i+1,j-1))  (m(8)*A2(i+1,j))       (m(9)*A2(i+1,j+1))  ];
A1(i,j)=max(new)
; end end
A3=imdilate(imerode(d,m),m);
subplot(1,3,1);imshow(a);title('Original Image');
subplot(1,3,2);imshow(A1);title('Opening');
subplot(1,3,3);imshow(A3);title('Opening');
```

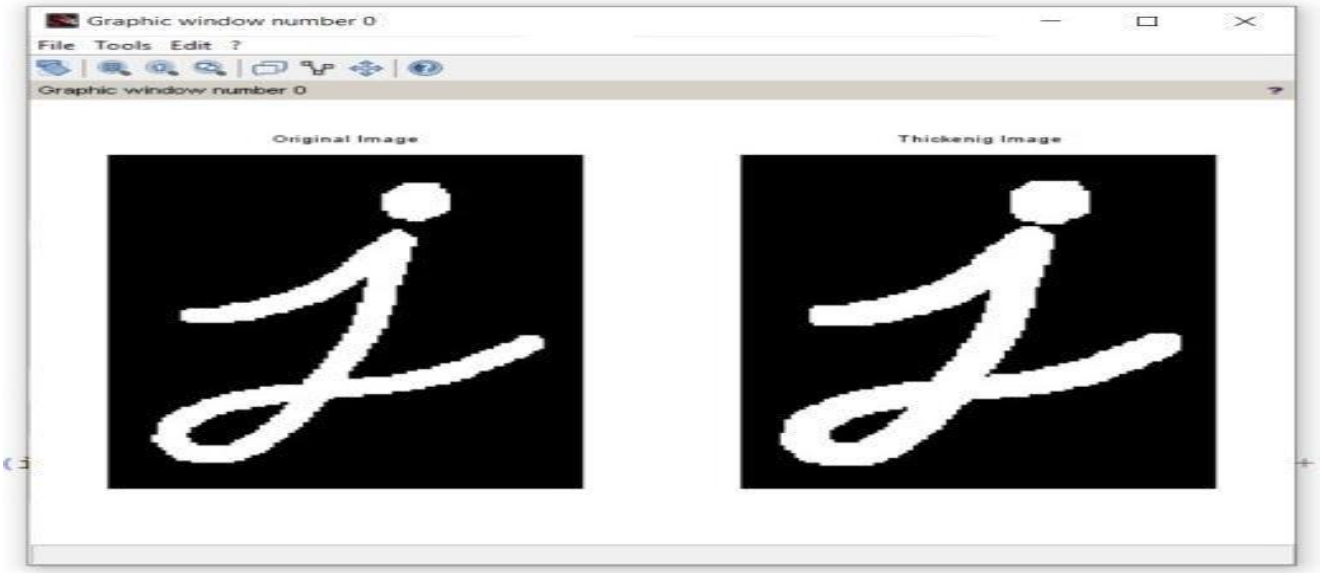
Output:



Code:

```
//Thickeni ng
clc; clear all;
z=uigetfile('*-*','Select an image file'); a=imread(z);
subplot(1,2,1),imshow(a),title('Original Image');
//a=[0 0 0 0 0
//      0 0 1 0 0
//      0 0 1 0 0
//      0 0 0 0 0 //         0
0 0 0 0];
a=uint8(a); s=[1 1
1,1 1 1,1 1 1];
r=size(a,1);
c=size(a,2);
new=zeros(r,c);
new1=zeros(r,c);
for i=2:r-1         for
j=2:c-1
         if((a(i-1,j-1)==s(1)) | a(i-1,j)==s(2)) | (a(i-1,j+1)==s(3)) | (a(i,j-1)==s(4)) |
(a(i,j)==s(5)) | (a(i,j+1)==s(6)) | (a(i+1,j-1)==s(7)) | (a(i+1,j)==s(8)) |
(a(i+1,j+1)==s(9))

              new(i,j)=1;
           else
new(i,j)=0;
end
           new1(i,j)=a(i,j)+new(i,j);
      end
end
//if sum(sum(new)==sum(sum(new1))
//disp 'same';
//else 'different';
//end;
//new1=a+new;

//new1=a+new;
subplot(1,2,2),imshow(new);title('Thickenig Image');
```

Output:

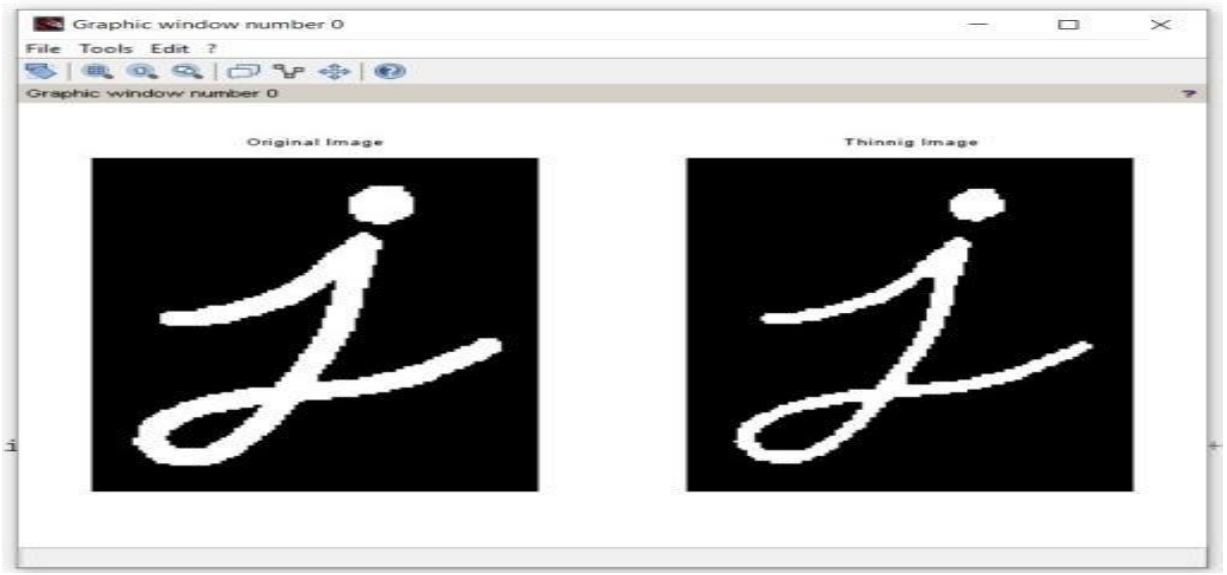


Code:

```
//Thinni ng
clc; clear
all;
z=uigetfile('-','Select an image file'); a=imread(z);
subplot(1,2,1),imshow(a),title('Original Image');
//a=[0 0 0 0 0
//      0 0 1 0 0
//      0 0 1 0 0
//      0 0 0 0 0 //          0 0 0 0 0]; a=uint8(a); s=[1 1 1,1 1 1,1 1 1];
r=size(a,1); c=size(a,2); new=zeros(r,c); new1=zeros(r,c); for i=2:r-1 for j=2:c-
1                          if((a(i-1,j-1)==s(1)) & a(i-1,j)==s(2)) & (a(i-
1,j+1)==s(3)) & (a(i,j1)==s(4)) & (a(i,j)==s(5)) & (a(i,j+1)==s(6)) &
(a(i+1,j-1)==s(7)) &
(a(i+1,j)==s(8)) & (a(i+1,j+1)==s(9))

              new(i,j)=1;
else
new(i,j)=0;
end
          new1(i,j)=1-new(i,j);
      end
end
//if sum(sum(new)==sum(sum(new1))
//disp 'same';
//else 'different';
//end;
//new1=a+new;
```

```
//new1=a+new; subplot(1,2,2),imshow(new);title('Thinnig
Image');
```

Output:



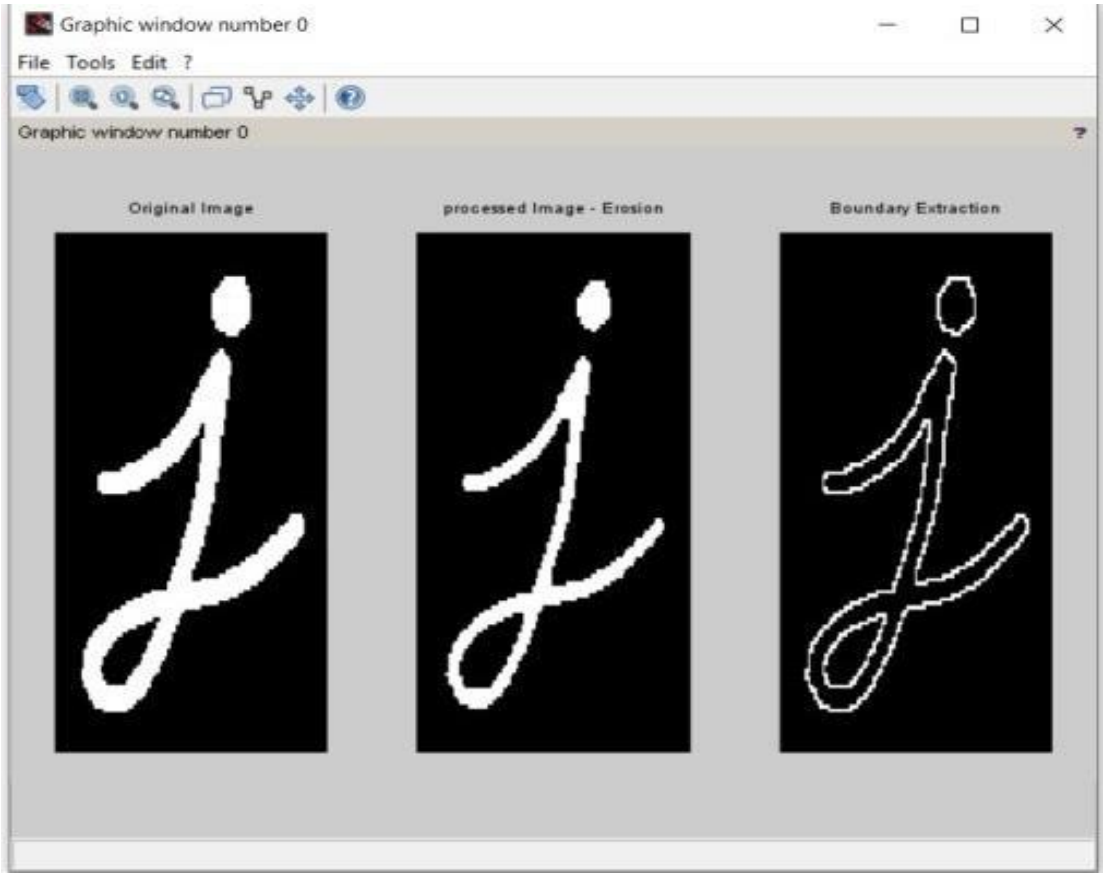
B. Program for detecting boundary of an image

Code:

```
clc;
clear
all;

//boundary Extraction a=uigetfile('*-
*','Select the Image'); a=imread(a);
//a=[0 0 0 0 0
0 0 0
//      0 0 1 1 1 1 0 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1 1 0
//      0 1 1 1 1 1
1 0 //       0 0 0 0
0 0 0 0];
d=a;
[r,c]=size(d);
m=ones(3,3);
A1=zeros(r,c);
for i=2:1:r-1
    for j=2:1:c-1

        new=[(m(1)*d(i-1,j-1)) (m(2)*d(i-1,j)) (m(3)*d(i-1,j+1))
(m(4)*d(i,j-1)) (m(5)*d(i,j)) (m(6)*d(i,j+1))
        (m(7)*d(i+1,j-1)) (m(8)*d(i+1,j)) (m(9)*d(i+1,j+1))];
        A1(i,j)=min(new);

    end end
figure;
A2=d-A1;
subplot(1,3,1),imshow(a),title('Original                Image');
subplot(1,3,2),imshow(A1);title('processed        Image        -
Erosion');
```

```
subplot(1,3,3),imshow(A2);title('Boundary
Extraction');
```

Output:

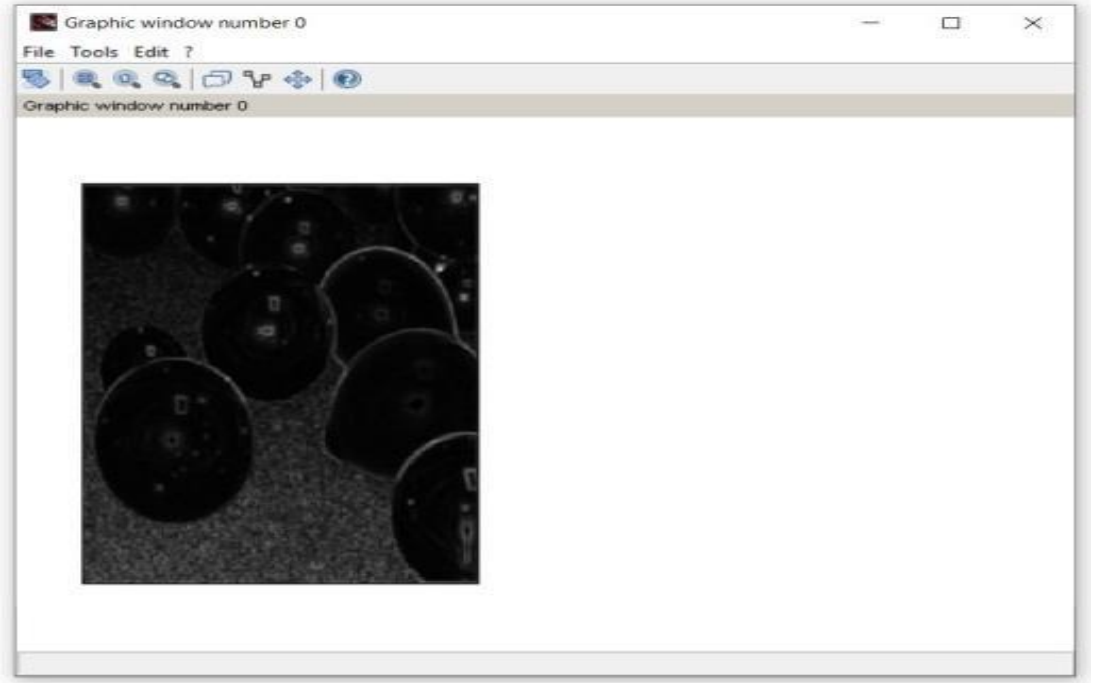


d. Program to apply morphological gradient on an image

Code:

```
a=uigetfile('*-*','Select the Image');
a=imread(a);
se = imcreatese('rect',3,3)
; b = imdilate(a,se);
c=imerode(a,se); Gra
= mtlb_s(b,c);
subplot(1,2,1);

imshow(Gra);

//subplot(1,2,2);
//d=imopen(a,se);
//toph=mtlb_s(a,d);
//imshow(toph);
```

Output:

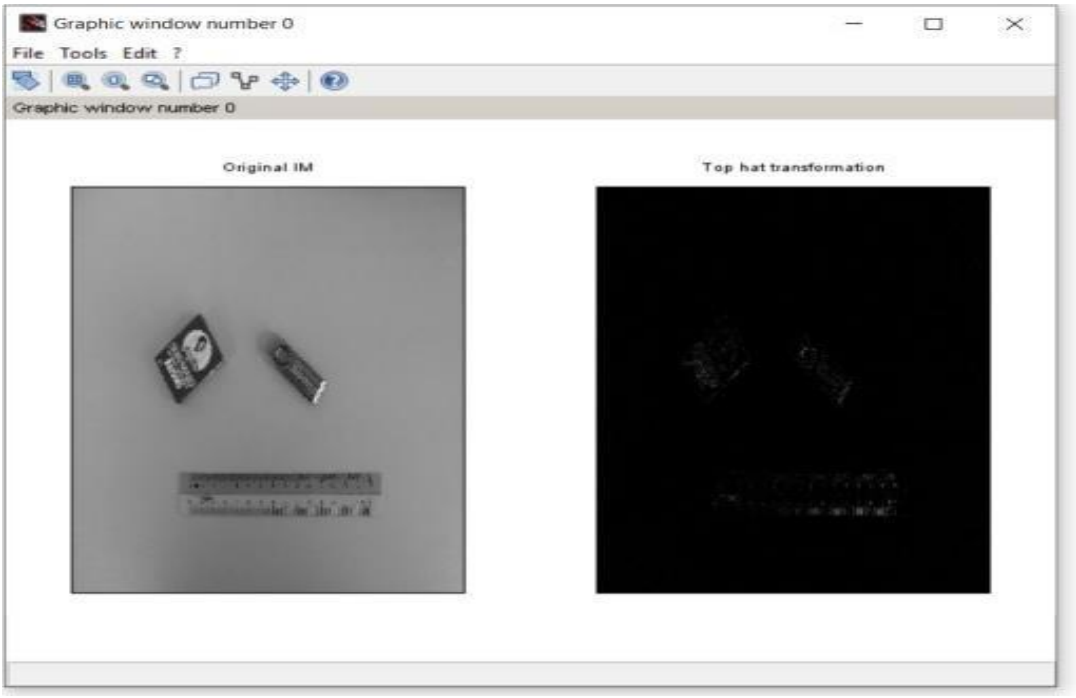


e. Program to apply Top-Hat/Bottom-hat Transformations

Code:

```
//TopHatTransformation clc;
clear
all;
a=uigetfile('*-*','Select the Image');
a=imread(a);
se = imcreatese('rect',3,3);

subplot(1,2,1);
imshow(a),title('Original IM');
d=imopen(a,se) toph=mtlb_s(a,d);
subplot(1,2,2); imshow(toph);
title('Top hat transformation');
```

Output:



8. Image Segmentation

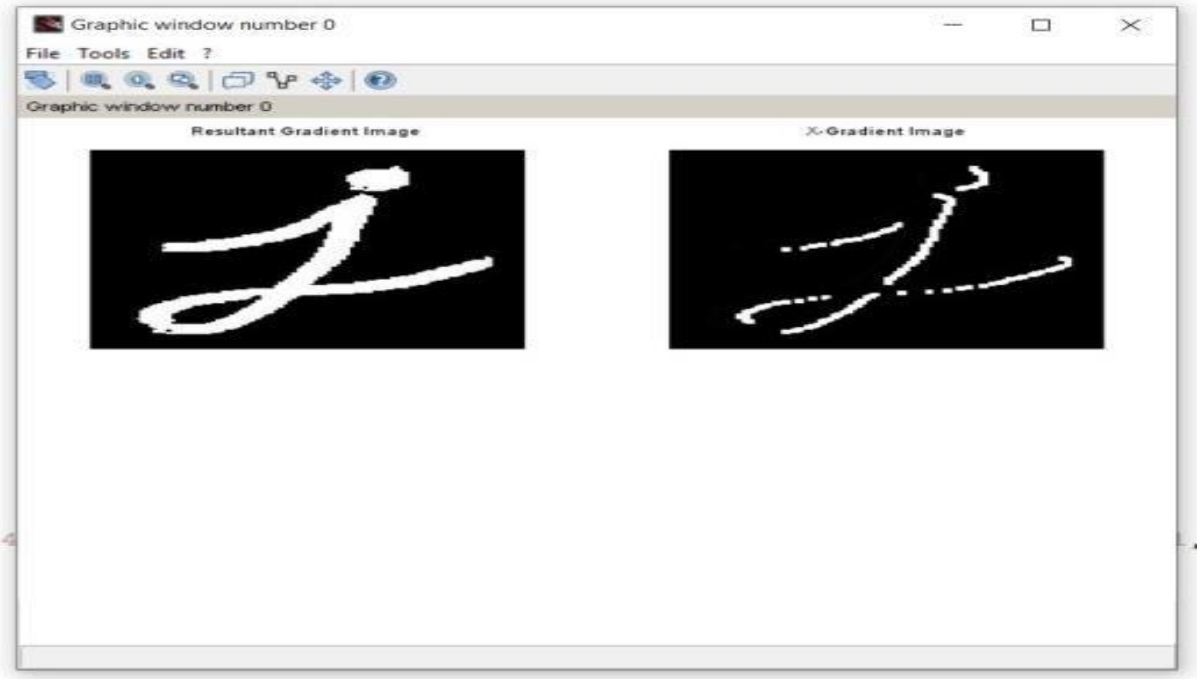
Program for Edge detection using
A. Sobel

Code:

```
clc;
clear
all;
a=uigetfile('*-*','Select the Image');
a=imread(a);
b=double(a); d=b;
mh=[-1 -2 -1;0 0 0;1
2 1]; mv=[-1 0 1;-2 0
2;-1 0 1];
[r1,c1]=size(a);
for i=1:r1        for
j=1:c1
new(i,j)=a(i,j);
nem(i,j)=a(i,j);
end end
new=double(ne
w);
new=double(ne
m); for i=2:1:r1-1
for j=2:1:c1-1
        new(i,j)=(mh(1)*d(i-1,j-1))+(mh(2)*d(i-1,j))+(mh(3)*d(i-
1,j+1))+(mh(4)*d(i,j-
1))+(mh(5)*d(i,j))+(mh(6)*d(i,j+1))+(mh(7)*d(i+1,j-
1))+(mh(8)*d(i+1,j))+(mh(9)*d(i+1,j
+1));      end end for i=2:1:r1-1 for
j=2:1:c1-1
        nem(i,j)=(mv(1)*d(i-1,j-1))+(mv(2)*d(i-
1,j))+(mv(3)*d(i1,j+1))+(mv(4)*d(i,j-
1))+(mv(5)*d(i,j))+(mv(6)*d(i,j+1))+(mv(8)*d(i+1,j))+(mv(9)*d(i+1,j+1)
);      end
end
new2=new+nem;
```

```
subplot(221);imshow(uint8(a));title('Original Image');
subplot(222);imshow(uint8(new));title('X-Gradient Image');
subplot(221);imshow(uint8(nem));title('Y-Gradient Image');
subplot(221);imshow(uint8(new2));title('Resultant Gradient Image');
```

Output:



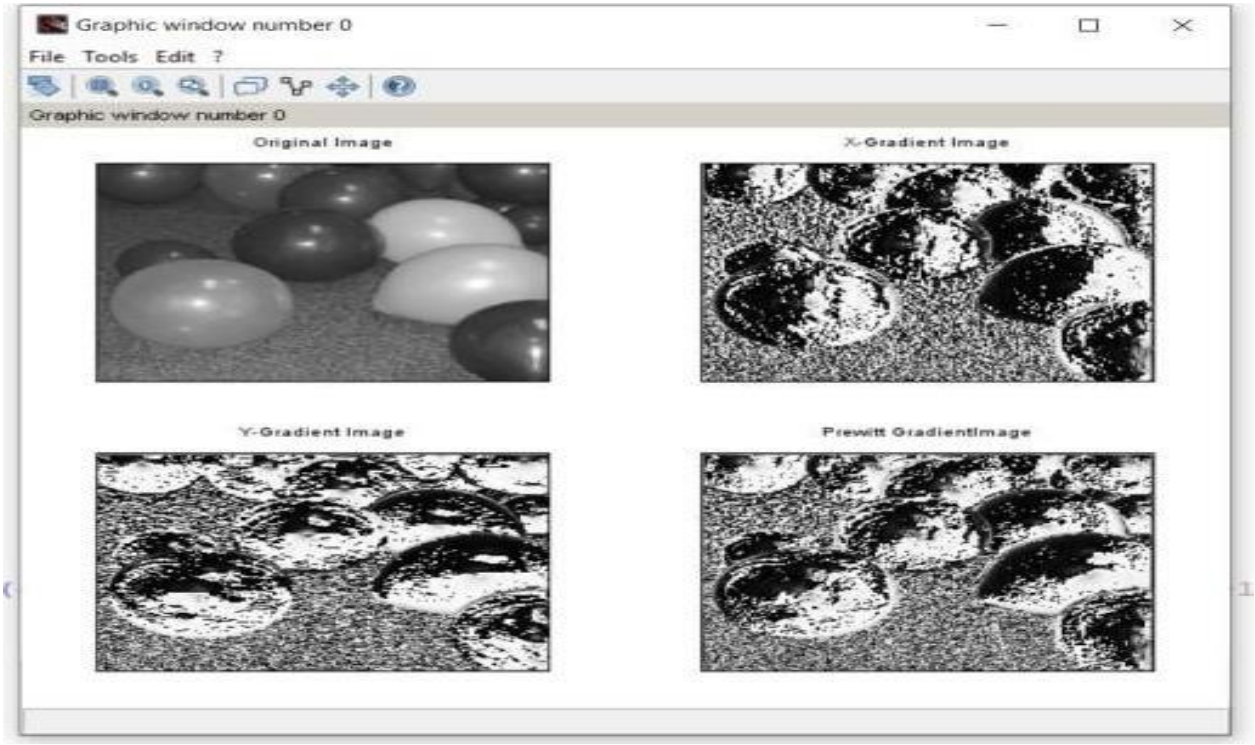
B. Prewitt

Code:

```
clc;
clear;
a=uiget
file('.','
Select an
image
file');
a=imread(a);
b=double(a); d=b;
mh=[-1 -1 -1;0 0 0;1
1 1]; mv=[-1 0 1;-1 0
1;-1 0 1];
[r1,c1]=size(a);
for i=1:r1        for
j=1:c1
new(i,j)=a(i,j);

nem(i,j)=a(i,j);
end end
new=double(ne
w);
nem=double(ne
m); for
i=2:1:r1-1
for j=2:1:c1-1
        new(i,j)=(mh(1)*d(i-1,j-1))+(mh(2)*d(i-1,j))+(mh(3)*d(i-
1,j+1))+(mh(4)*d(i,j-
1))+(mh(5)*d(i,j))+(mh(6)*d(i,j+1))+(mh(7)*d(i+1,j-
1))+(mh(8)*d(i+1,j))+(mh(9)*d(i+1,j
+1)); end end for i=2:1:r1-1              for
j=2:1:c1-1
        nem(i,j)=(mv(1)*d(i-1,j-1))+(mv(2)*d(i-1,j))+(mv(3)*d(i-
1,j+1))+(mv(4)*d(i,j-1))+(mv(5)*d(i,j))+(mv(6)*d(i,j+1))+(mv(7)*d(i+1,j-
1))+(mv(8)*d(i+1,j))+(mv(9)*d(i+1,j+
1)); end end new2=new+nem;
```

```
subplot(221);imshow(uint8(a));title('Original Image');
subplot(222);imshow(uint8(new));title('X-Gradient Image');
subplot(223);imshow(uint8(nem));title('Y-Gradient Image');
subplot(224);imshow(uint8(new2));title('Prewitt GradientImage')
```

Output:

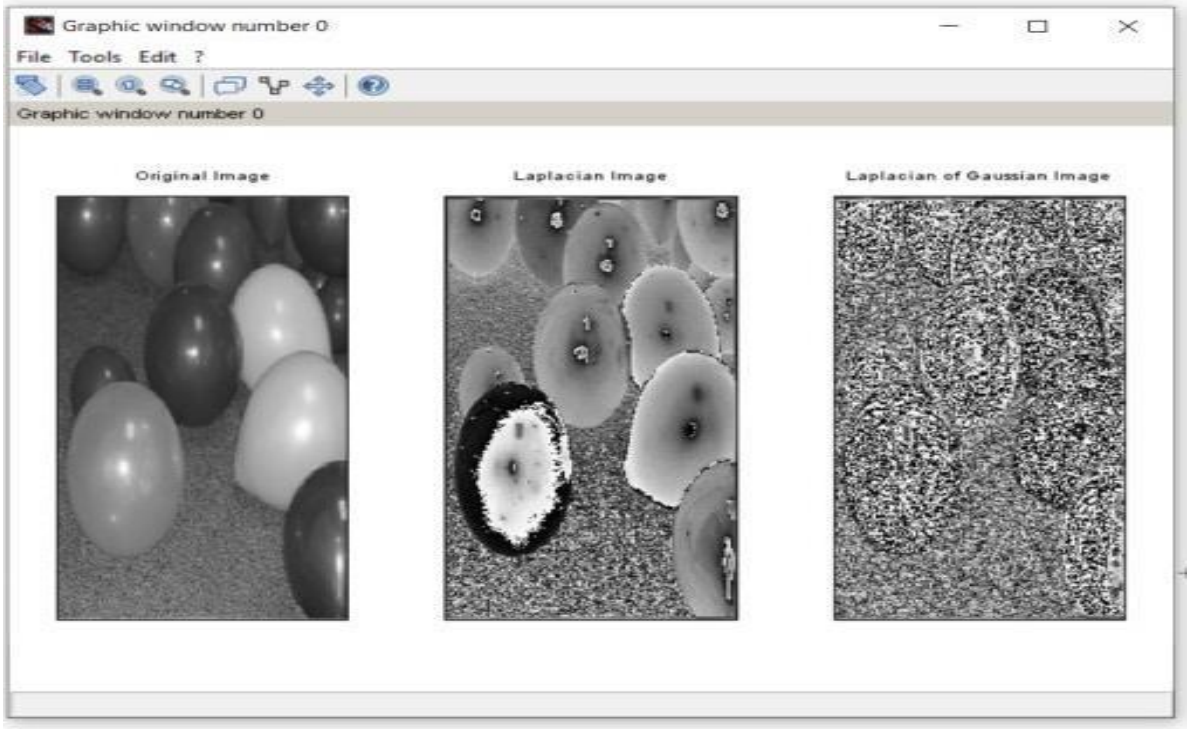


C. Marr-Hildreth

Code:

```
clc;
clear
all;
a=uigetfile('*-*','Select the Image:-');
a=imread(a);
b=double(a);
d=b;
mh=[0,-1,0;1,-4,1;0,1,0];
mv=[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];
[r1,c1]=size(a); for
i=1:r1        for j=1:c1
LAP(i,j)=a(i,j);
LoG(i,j)=a(i,j);
    end
end
new=double(LAP);
nem=double(LoG)
; for i=3:1:r1-2
for j=3:1:c1-2
        new(i,j)=(mh(1)*d(i-1,j-1))+(mh(2)*d(i-1,j))+(mh(3)*d(i-
1,j+1))+(mh(4)*d(i,j-1))+(mh(5)*d(i,j))+(mh(6)*d(i,j+1))+(mh(7)*d(i+1,j-
1))+(mh(8)*d(i+1,j))+(mh(9)*d(i+1,j+1));
    end end for
i=3:1:r1-2
for j=3:1:c1-2
        nem(i,j)=(mv(1)*d(i-2,j-2))+(mv(2)*d(i-2,j-1))+(mv(3)*d(i-
2,j))+(mv(4)*d(i-2,j+1))+(mv(5)*d(i-2,j+2))+(mv(6)*d(i-1,j-2))+(mv(7)*d(i-
1,j-1))+(mv(8)*d(i-1,j))+(mv(9)*d(i-1,j+1))+(mv(10)*d(i-
1,j+2))+(mv(11)*d(i,j-2))+(mv(12)*d(i,j-
1))+(mv(13)*d(i,j))+(mv(14)*d(i,j+1))+(mv(15)*d(i,j+2))+(mv(16)*d(i+1,j2))+(mv(1
7)*d(i+1,j-
1))+(mv(18)*d(i+1,j))+(mv(19)*d(i+1,j+1))+(mv(20)*d(i+1,j+2))+(mv(21)*d(i
+2,j-2))+(mv(22)*d(i+2,j-
1))+(mv(23)*d(i+2,j))+(mv(24)*d(i+2,j+1))+(mv(25)*d(i+2,j+2));
end end subplot(131);imshow(uint8(a));title('Original Image');
subplot(132);imshow(uint8(new));title('Laplacian Image');
subplot(133);imshow(uint8(nem));title('Laplacian of Gaussian Image');
```

Output:


Graphic window number 0
File Tools Edit ?
Graphic window number 0
Original Image
Laplacian Image
Laplacian of Gaussian Image

C. Canny

Code:

```
clc;
clear
all;
a=uigetfile('*-*','Select the Image:-');
a=imread(a);
b=double(a);
thresh=0.4;
sigma=3;
E=edge(a,'canny',thresh, sigma);
imshow(E);
```

Output: