PRACTICAL No. 1

Aim: 2D Linear Convolution, Circular Convolution between two 2D matrices.

2D Linear Convolution

Example 1:

Code:

```
clc;
x=[4,5,6;7,8,9];
h=[1;1;1];
disp(x,"x=");
disp(h,"h=");
y=conv2(x,h);
disp(y, '2D Linear Convolution result: y =' );
```

Output:

```
x=

   4.   5.   6.
   7.   8.   9.

 h=

   1.
   1.
   1.

 2D Linear Convolution result: y =

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

Example 2:

Code:

```
clc;
x=[1,2,3;4,5,6;7,8,9];
h=[1,1;1,1;1,1];
disp(x,"x=");
disp(h,"h=");
y=conv2(x,h);
disp(y, '2D Linear Convolution result: y =' );
```

Output:

```
x=

   1.   2.   3.
   4.   5.   6.
   7.   8.   9.

 h=

   1.   1.
   1.   1.
   1.   1.

 2D Linear Convolution result: y =

   1.    3.    5.    3.
   5.    12.   16.   9.
   12.   27.   33.   18.
   11.   24.   28.   15.
   7.    15.   17.   9.
```

2D Circular Convolution

Example 1:

Code:
```
clc ;
x=[1,2;3,4];
h=[5,6;7,8];
disp(x,'x=');
disp(h,'h=');
X=fft2(x);
H=fft2(h);
Y=X.*H;
y=ifft(Y);
disp(y, '2D Circular Correlation Result: y =' );
```

Output:
```
x=

   1.   2.
   3.   4.

 h=

   5.   6.
   7.   8.

 2D Circular Correlation Result: y =

   70.   68.
   62.   60.
```

Example 2:

Code:

```
clc ;
x=[1,2,3;4,5,6;7,8,9];
h=[1,1,1;1,1,1;1,1,1];
disp(x,'x=');
disp(h,'h=');
X=fft2(x);
H=fft2(h);
Y=X.*H;
y=ifft(Y);
disp(y, '2D Circular Correlation Result: y =' );
```

Output:

```
x=

   1.   2.   3.
   4.   5.   6.
   7.   8.   9.

 h=

   1.   1.   1.
   1.   1.   1.
   1.   1.   1.

 2D Circular Correlation Result: y =

   45.   45.   45.
   45.   45.   45.
   45.   45.   45.
```

PRACTICAL No. 2

Aim: Circular Convolution expressed as Linear Convolution plus alias.

Code:

```
clc ;
x =[1,2;3,4];
h=[5,6;7,8];
y=conv2(x,h);
y1=[y(:,1)+y(:,$),y(:,2) ];
y2=[y1(1,:)+y1($,:);y1(2,:)];
disp(y, 'Linear Convolution Result: y=');
disp(y2 , 'Circular Convolution expressed as Linear Convolution =' );
```

Output:

```
Linear Convolution Result: y=

   5.     16.    12.
   22.    60.    40.
   21.    52.    32.

 Circular Convolution expressed as Linear Convolution =

   70.    68.
   62.    60.
```

PRACTICAL No. 3

Aim: Linear Cross correlation of a 2D matrix, Circular correlation between two signals and Linear auto correlation of a 2D matrix, Linear Cross correlation of a 2D matrix

A] Linear Cross correlation of a 2D matrix

Code:

```
clc;
x = [3,1;2,4];
h1 = [1,5;2,3];
h2 = h1(:,$:-1:1);
h = h2($:-1:1,:);
y = conv2(x,h)
disp(y, 'Linear cross Correlation result y=')
```

Output:

```
Linear cross Correlation result y=
  9.    9.    2.
  21.   24.   9.
  10.   22.   4.
```

B] Circular correlation between two signals

Code:

```
clc;
 x = [1,5;2,4];
 h = [3,2;4,1];
 h = h(:,$:-1:1);
 h = h($:-1:1,:);
 X = fft2(x);
 H = fft2(h);
 Y = X.*H;
 y = ifft(Y);
 disp(y,'Circular Correlation result y=')
```

Output:

```
Circular Correlation result y=

37.   23.

35.   25.
```

C] Linear auto correlation of a 2D matrix

Code:

```
clc;
 x1 = [1,1;1,1];
 x2 = x1(:,$:-1:1);
 x2 = x2($:-1:1,:);
 x = conv2(x1,x2)
 disp(x,'Linear auto Correlation result x=')
```

Output:

```
Linear auto Correlation result x=

   1.   2.   1.

   2.   4.   2.

   1.   2.   1.
```

D] Linear Cross correlation of a 2D matrix

Code:

```
clc;
x = [1,1;1,1];
h1 = [1,2;3,4];
h2 = h1(:,$:-1:1);
h = h2($:-1:1,:);
y = conv2(x,h)
disp(y, ' Linear cross Correlation result y=')
```

Output:

```
Linear cross Correlation result y=

   4.   7.    3.

   6.   10.   4.

   2.   3.    1.
```

PRACTICAL No. 4

Aim: Perform DFT of a 4x4 gray scale image.

Code:

```
clc;
x=[1,1,1,1;1,1,1,1;1,1,1,1;1,1,1,1];
X=fft(x,-1);
disp(X,"X[k]=");
```

Output:

```
X[k]=

  16.   0.   0.   0.
  0.    0.   0.   0.
  0.    0.   0.   0.
  0.    0.   0.   0.
```

PRACTICAL No. 5

AIM:- Compute discrete cosine transform, Program to perform KL transform for the given 2D matrix.

Code:-

```
clear;
clc;
X=[4,3,5,6;4,2,7,7;5,5,6,7];
[m,n]=size(X);
A=[];
E=[];
for i=1:n
    A=A+X(:,i);
    E=E+X(:,i)*X(:,i)';
end
mx=A/n;
E=E/n;
C=E-mx*mx';
[V,D]=spec(C);
d=diag(D);
[d,i]=gsort(d);
for j=1:length(d)
    T(:,j)=V(:,i(j));
end
T=T'
disp(d,' Eigen Values are U = ')
disp(T,'The eigen vector matrix T =')
disp(T,'The KL tranform basis is =')
for i=1:n
    Y(:,i)=T*X(:,i);
end
disp(Y,'KL transformation of the input matrix Y =')
for i=1:n
    x(:,i)=T'*Y(:,i);
end
disp(x,'Reconstruct matrix of the given sample matrix X =')
```

Output:-

```
Eigen Values are U =

   6.1963372
```

0.2147417

0.0264211

The eigen vector matrix T =

0.4384533	0.8471005	0.3002988
0.4460381	- 0.4951684	0.7455591
- 0.7802620	0.1929481	0.5949473

The KL tranform basis is =

0.4384533	0.8471005	0.3002988
0.4460381	- 0.4951684	0.7455591
- 0.7802620	0.1929481	0.5949473

KL transformation of the input matrix Y =

6.6437095	4.5110551	9.9237632	10.662515
3.5312743	4.0755729	3.2373664	4.4289635
0.6254808	1.0198466	1.0190104	0.8336957

Reconstruct matrix of the given sample matrix X =

4.	3.	5.	6.
4.	2.	7.	7.
5.	5.	6.	7.

PRACTICAL No. 6

AIM:- Brightness enhancement of an image, Contrast Manipulation, image negative.

Install Image Processing and Signal Processing packages and restart scilab.

Run this command on console: atomsRemove('scicv')

Restart scilab

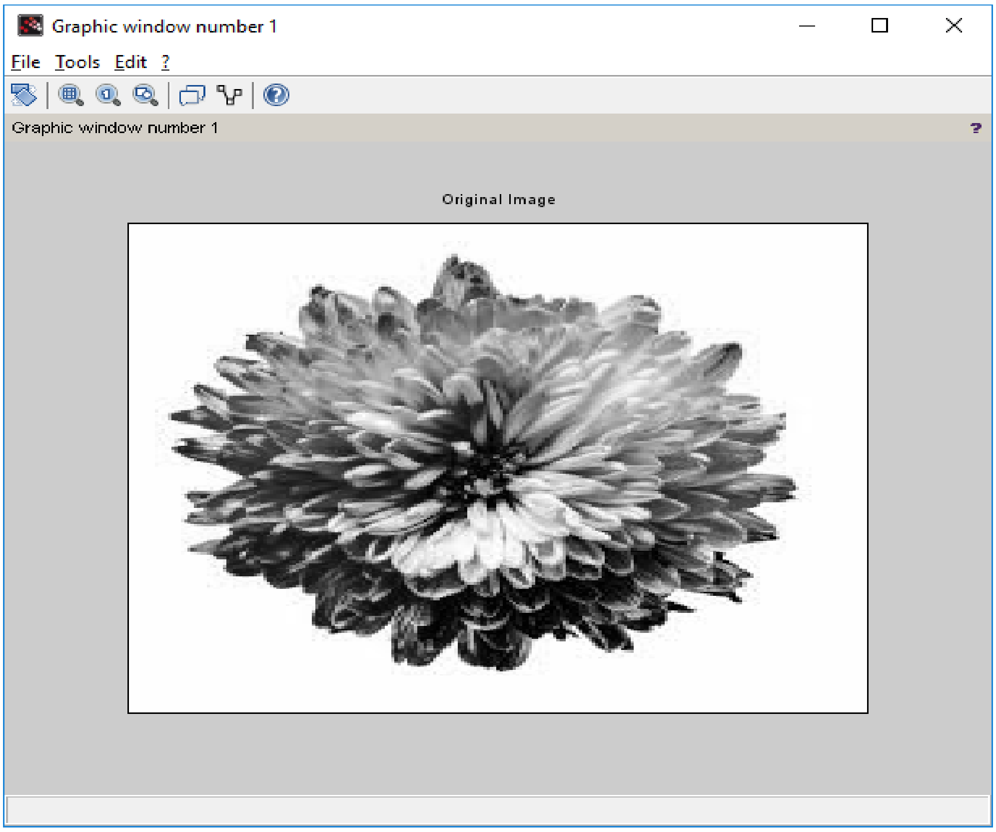
And run code

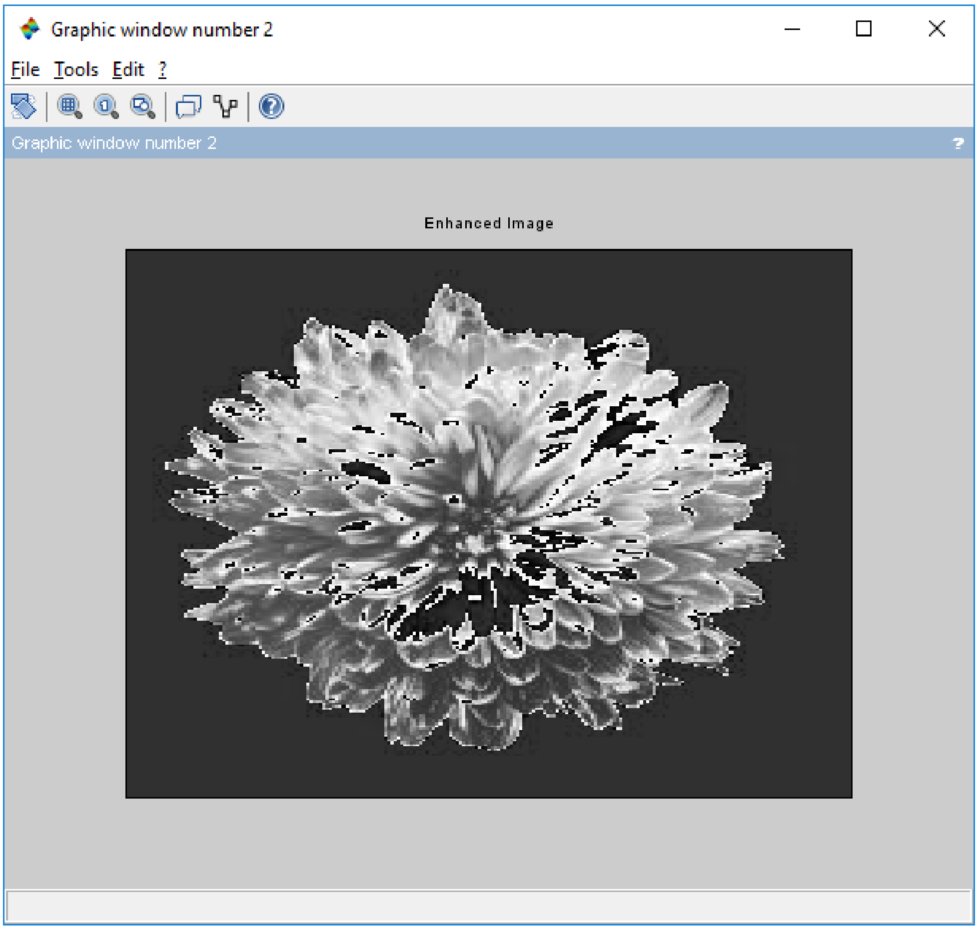
Brightness Enhancement

Code:-

```
Clc;
close;
a=imread('C:\Users\ADMIN\Desktop\flower.jpg');
a=rgb2gray(a);
b=double(a)+50;
b=uint8(b);
figure(1);
imshow(a);
title("Original Image")
figure(2);
imshow(b);
title("Enhanced Image")
```

Output:-

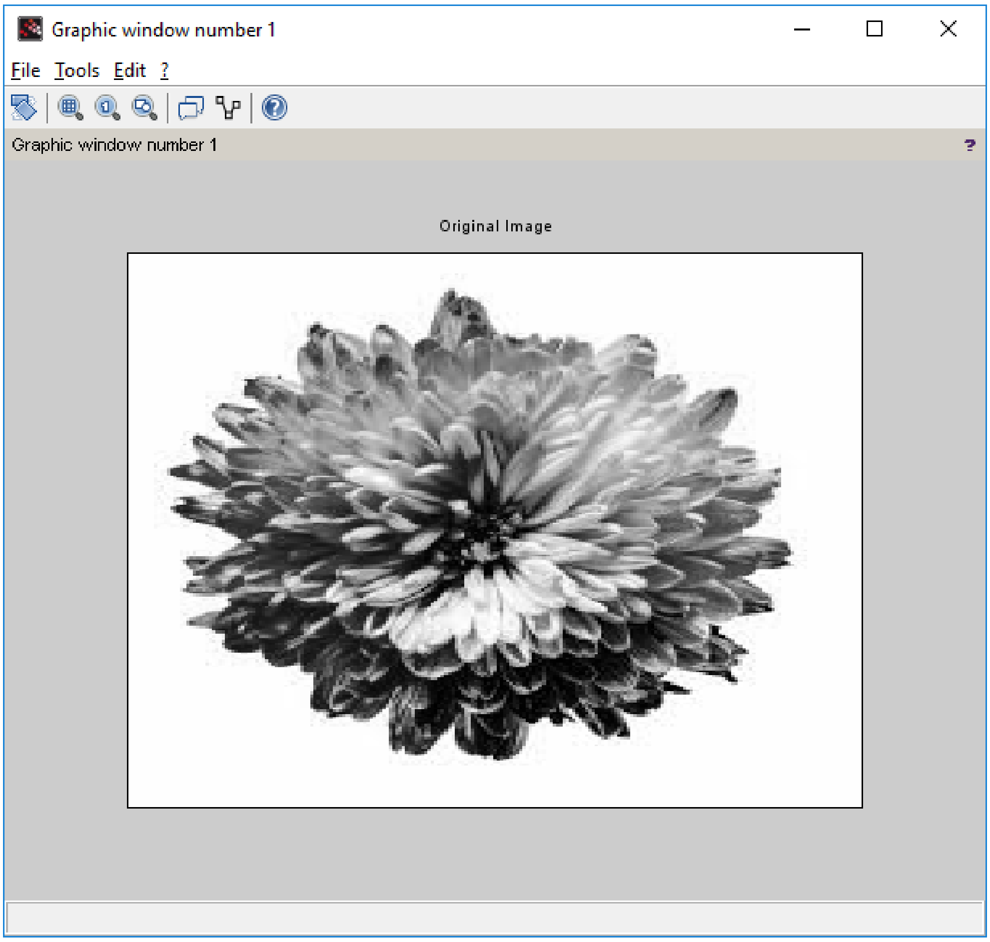

Graphic window number 1
File Tools Edit ?
Graphic window number 1
Original Image

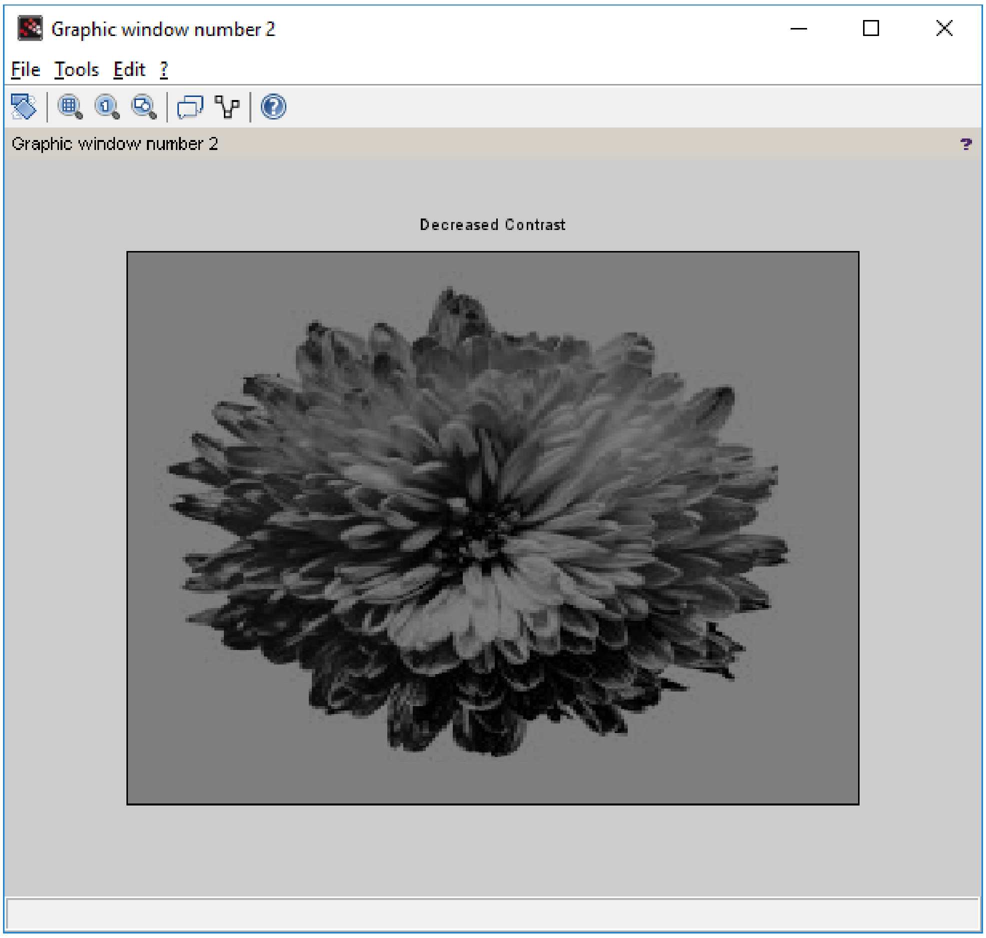

Graphic window number 2
File Tools Edit ?
Graphic window number 2
Enhanced Image

Contrast Manipulation

```
clc ;
close ;
a = imread('C:\Users\ADMIN\Desktop\flower.jpg');
a = rgb2gray(a);
b = double(a)*0.5;
b = uint8 (b)
c = double(b)*2;
c = uint8(c)
figure(1)
imshow(a);
title('Original Image')
figure(2)
imshow(b);
title('Decreased Contrast' )
figure(3)
imshow(c);
title('Increased Contrast')
```

Output:-




Graphic window number 2
File Tools Edit ?
Graphic window number 2
Decreased Contrast

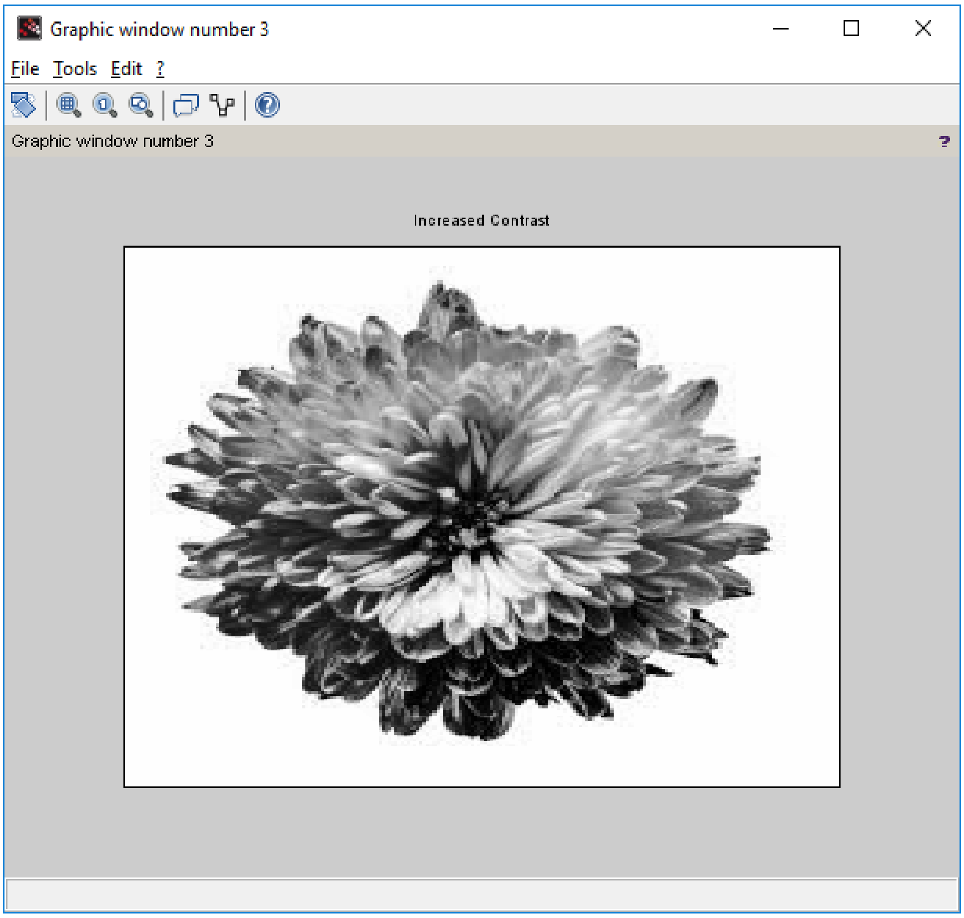
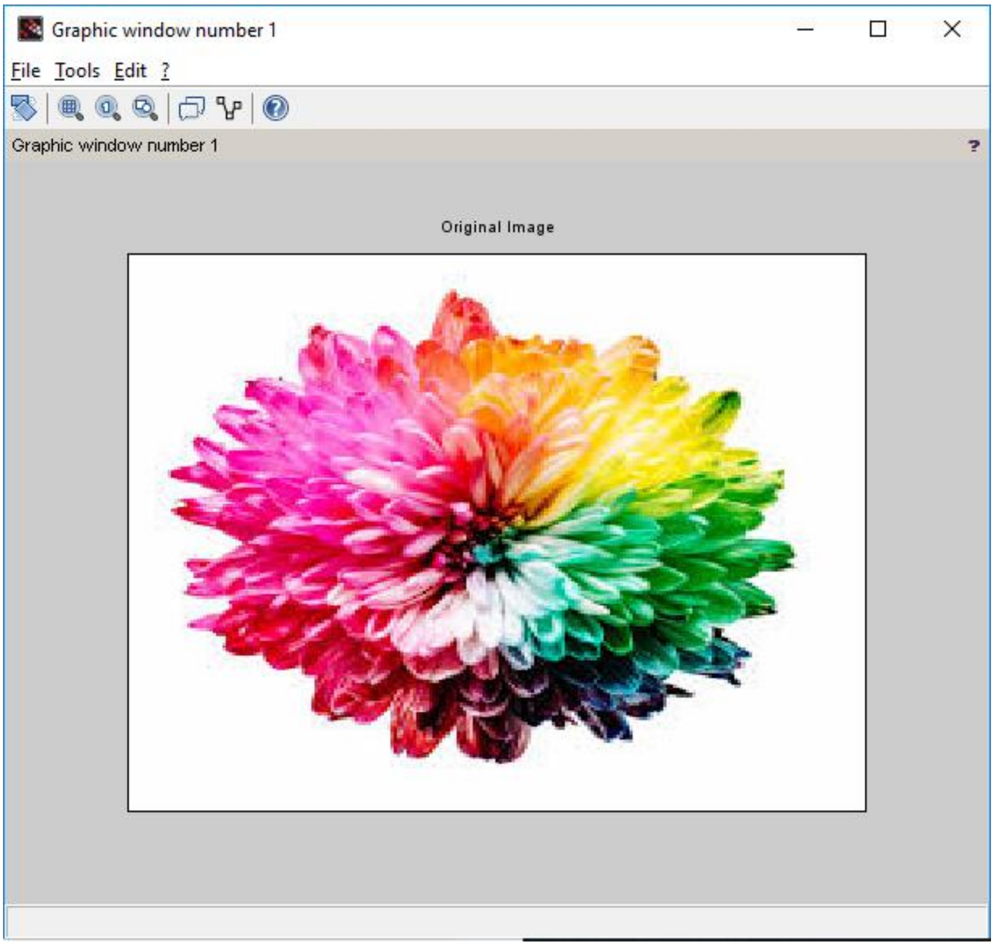

Graphic window number 3
File Tools Edit ?
Graphic window number 3
Increased Contrast

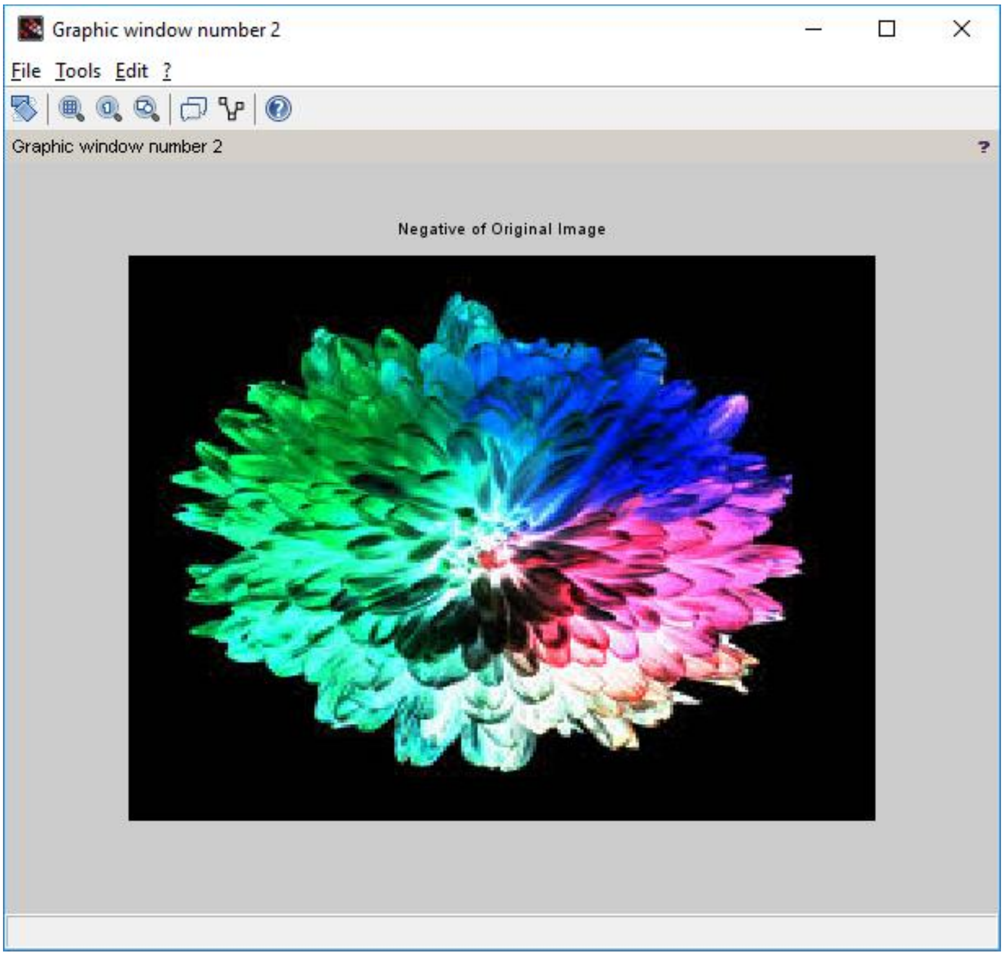
Image Negative

Code:-

```
clc;
close;
a = imread('C:\Users\ADMIN\Desktop\flower.jpg');
k = 255-double(a);
k = uint8(k);
figure(1)
imshow(a);
title('Original Image')
figure(2)
imshow(k);
title('Negative of Original Image')
```

Output:




Graphic window number 2
File Tools Edit ?
Graphic window number 2
Negative of Original Image

PRACTICAL No. 7

AIM:- Perform threshold operation, perform gray level slicing without background.

Install Image Processing and Signal Processing packages and restart scilab.

Run this command on console: atomsRemove('scicv')

Restart scilab

And run code

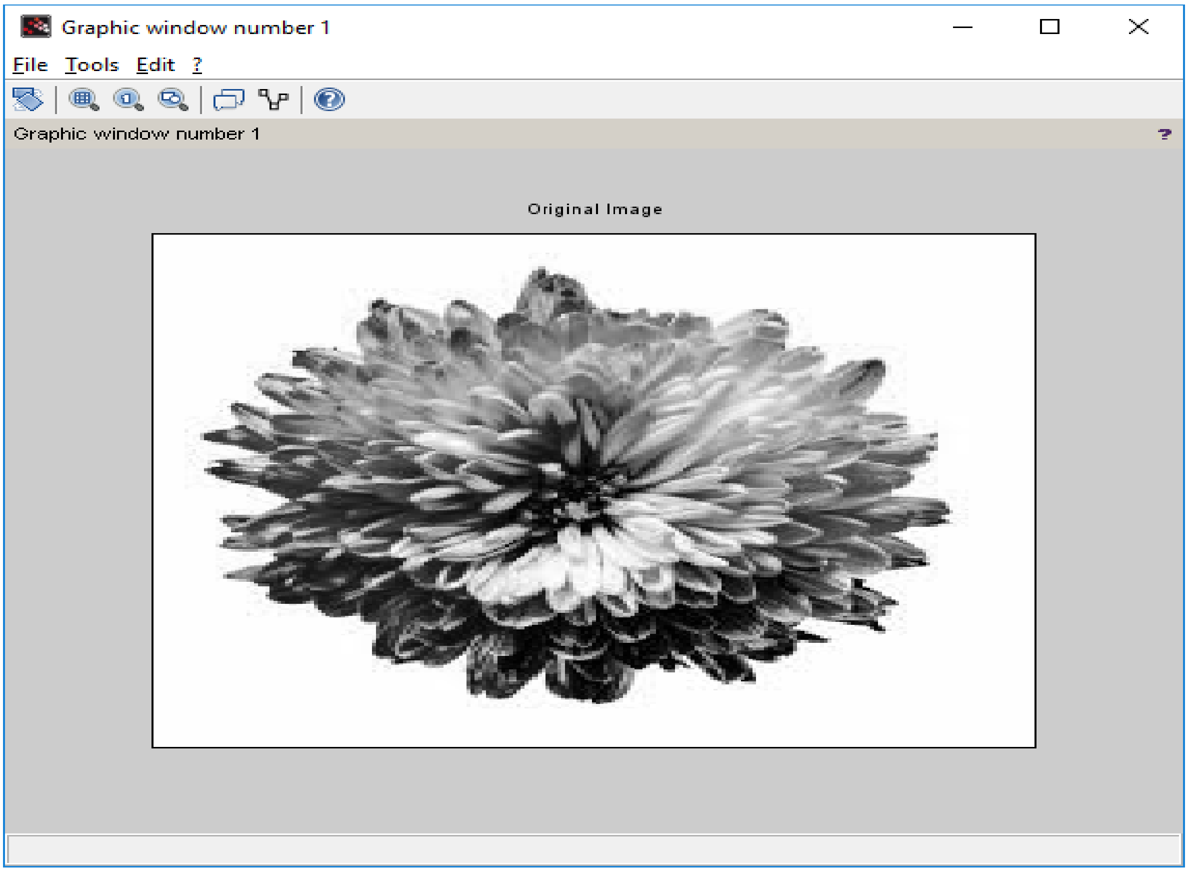
Threshold Operation

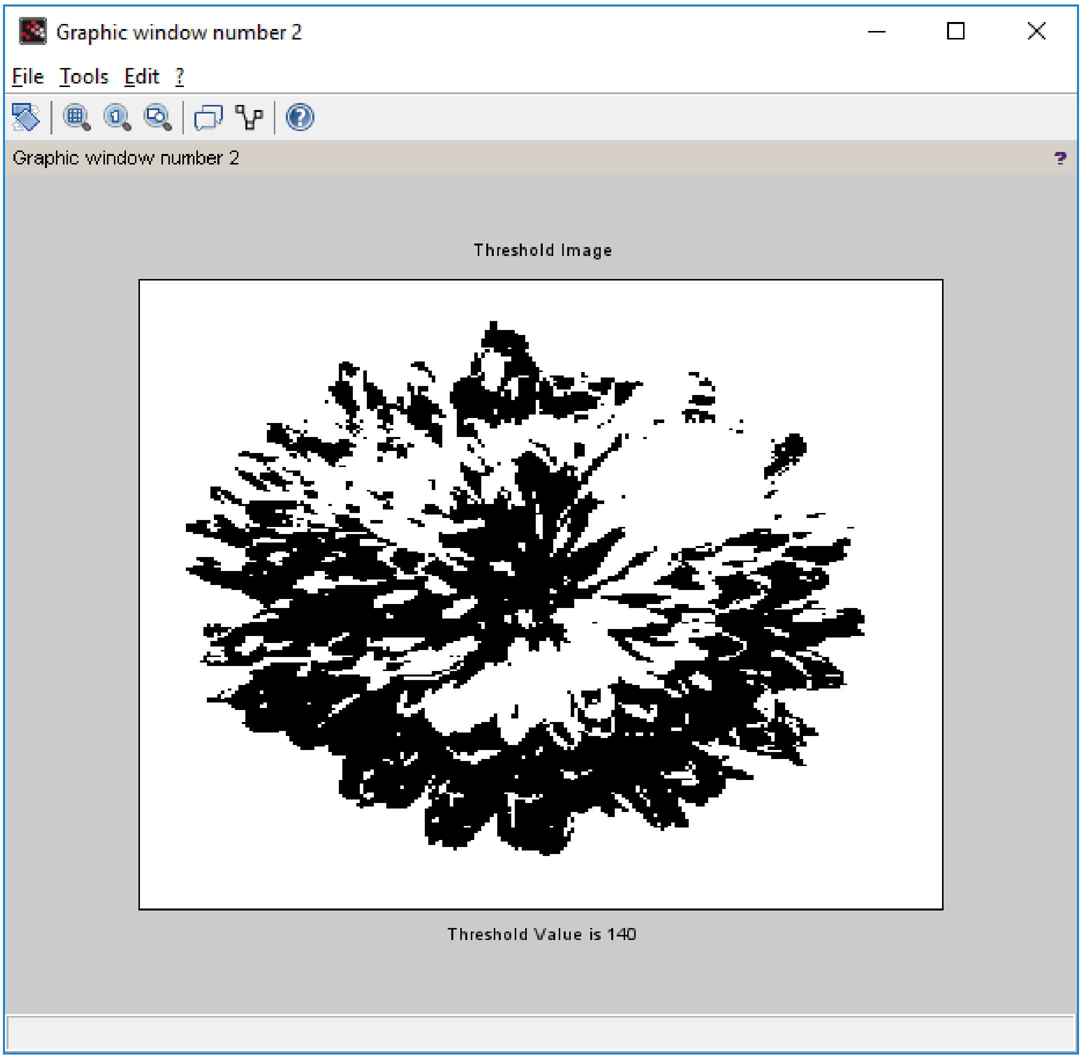
Code:-

```
clc;
close;
a = imread('C:\Users\ADMIN\Desktop\flower.jpg');
a = rgb2gray(a);
[m n] = size(a);
t = input('Enter threshold parameter: ');
for i = 1:m
    for j = 1:n
        if(a(i,j)<t)
            b(i,j)=0;
        else
            b(i,j) =255;
        end
    end
end
figure(1)
imshow(a);
title('Original Image')
figure(2)
imshow(b);
title('Threshold Image')
xlabel(sprintf('Threshold Value is %g ',t))
```

Output:

Enter threshold parameter: 140


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

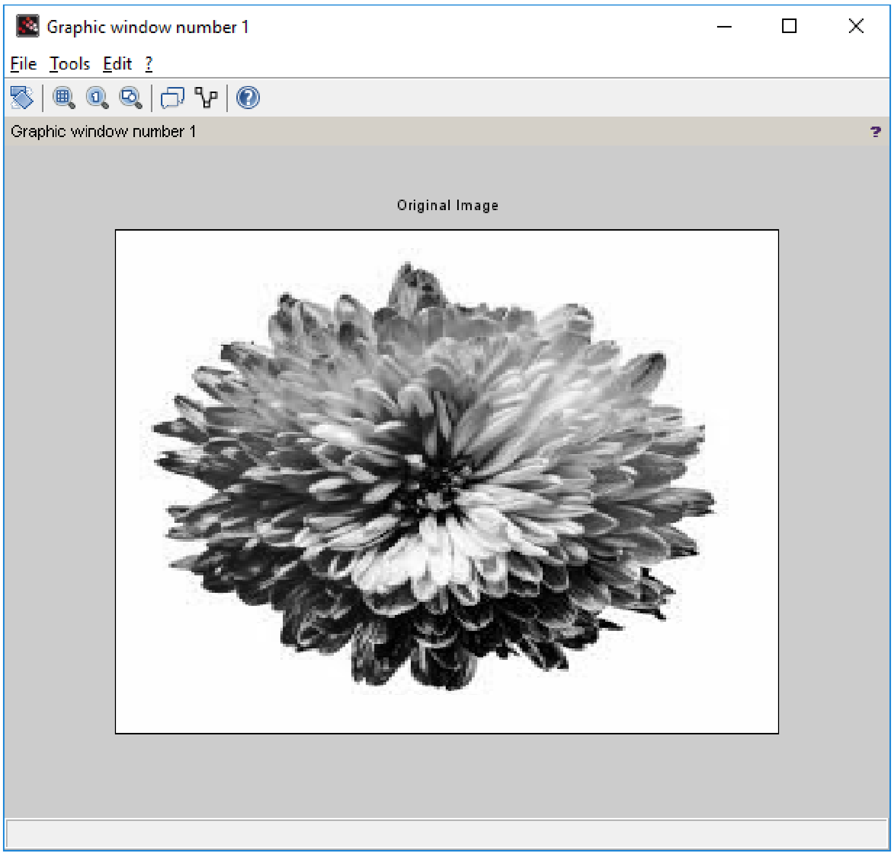

Graphic window number 2
File Tools Edit ?
Graphic window number 2
Threshold Image
Threshold Value is 140

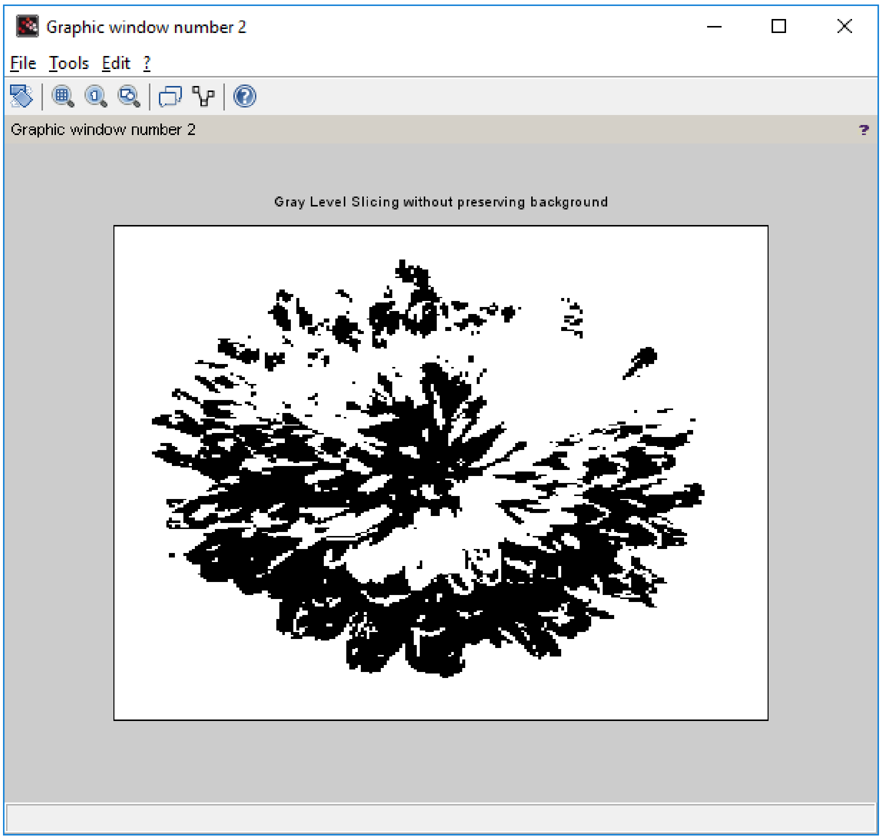
Gray Level Scaling without background.

Code:-

```
clc ;
x = imread('C:\Users\ADMIN\Desktop\flower.jpg');
x = rgb2gray(x);
y = double(x);
[m,n]= size(y);
L = max(max(x));
a = round(L/2) ;
b = L;
for i =1: m
    for j =1: n
        if(y(i,j)>=a & y(i,j)<=b)
            z(i,j) = L;
        else
            z(i,j)=0;
        end
    end
end
z = uint8(z);
figure(1)
imshow(x);
title('Original Image')
figure(2)
imshow(z);
title('Gray Level Slicing without preserving background')
```

Output:


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


Graphic window number 2
File Tools Edit ?
Graphic window number 2
Gray Level Slicing without preserving background

PRACTICAL No. 8

AIM:- Image Segmentation.

Install Image Processing and Signal Processing packages and restart scilab.

Run this command on console: atomsRemove('scicv')

Restart scilab

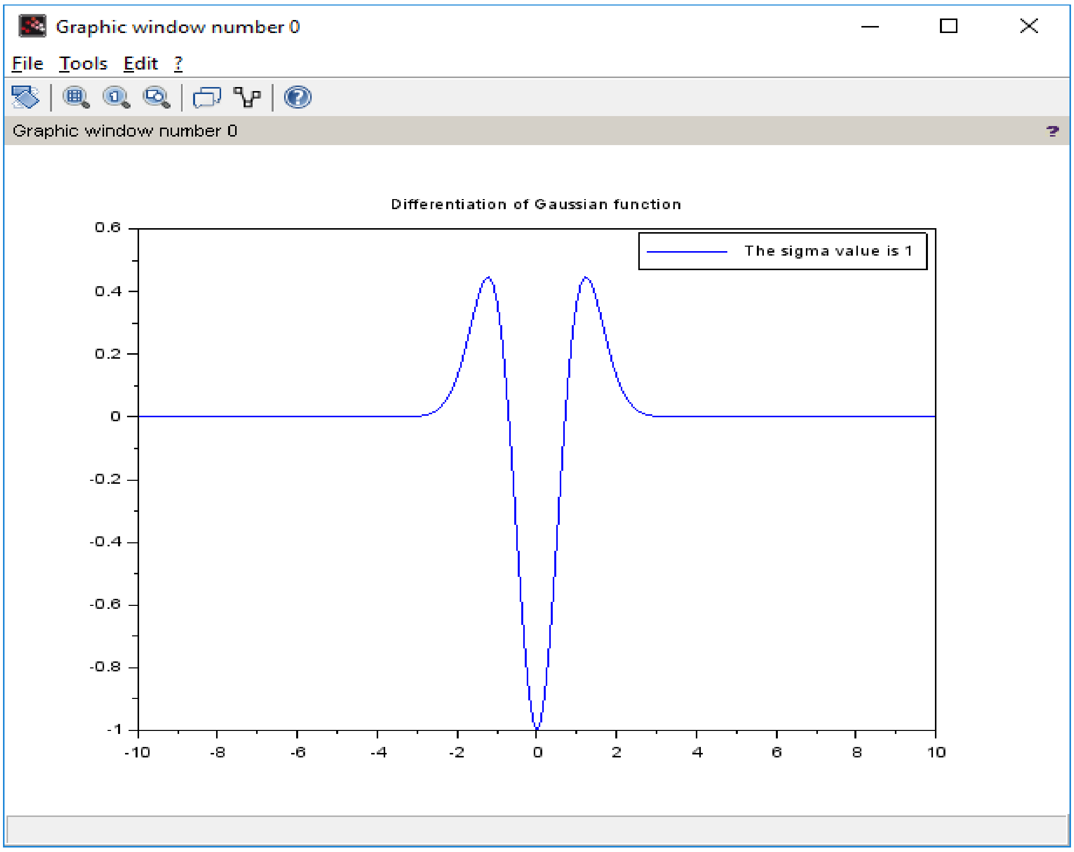
And run code

(a) Differentiation of Gaussian function.

```
clc ;
close ;
sigma = input('Enter the value of sigma: ')
i = -10:.1:10;
j = -10:.1:10;
r = sqrt(i.*i+j.*j);
y = (1/(sigma^2))*(((r.*r)/sigma^2)-1) .* exp(-r.*r/2*sigma^2);
plot(i,y)
legend(sprintf('The sigma value is %g',sigma))
xtitle('Differentiation of Gaussian function')
```

Output:-

Enter the value of sigma: 1



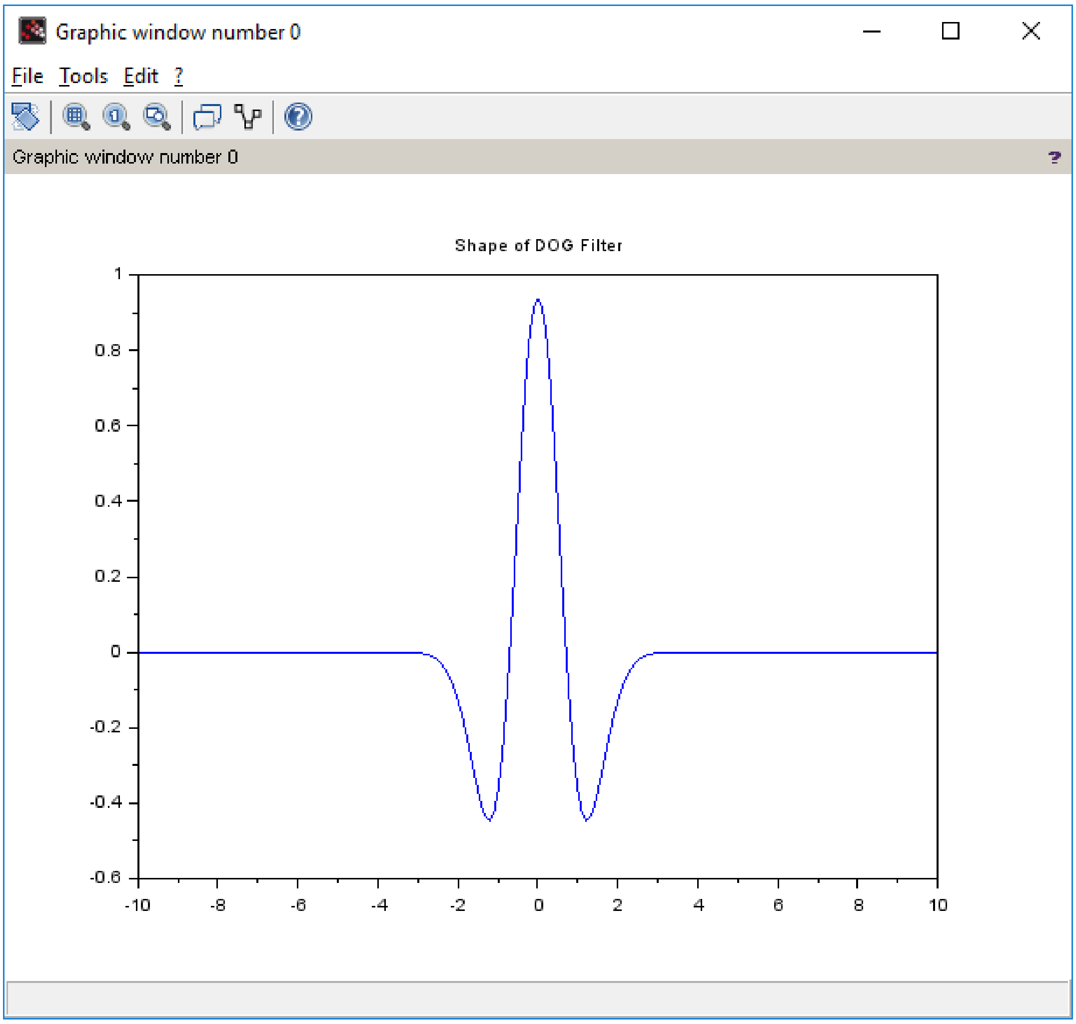
(b) Differentiation of Gaussian Filter function

```
clc ;
close ;
sigma1 = input('Enter the value of sigma1: ')
sigma2 = input ('Enter the value of sigma2: ')
i = -10:.1:10;
j = -10:.1:10;
r = sqrt(i.*i+j.*j);
y1 = (1/( sigma1^2))*(((r.*r)/sigma1^2)-1) .* exp(-r.*r/2*sigma1^2);
y2 = (1/( sigma2^2))*(((r.*r)/sigma2^2)-1) .* exp(-r.*r/2*sigma2 ^2);
y = y1-y2;
plot(i,y)
xtitle('Shape of DOG Filter ')
```

Output:

Enter the value of sigma1: 4

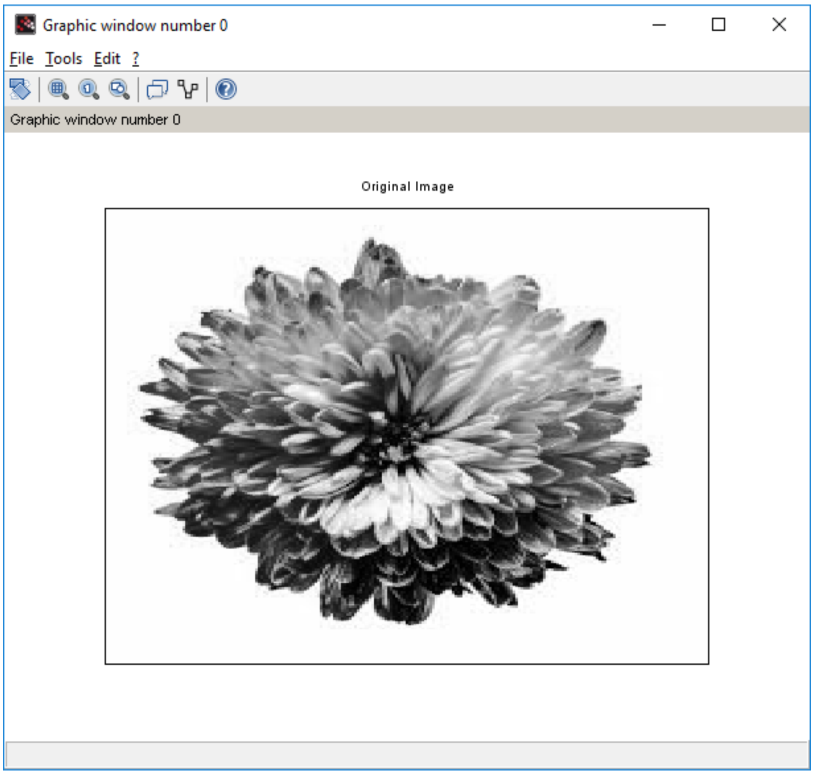
Enter the value of sigma2: 1

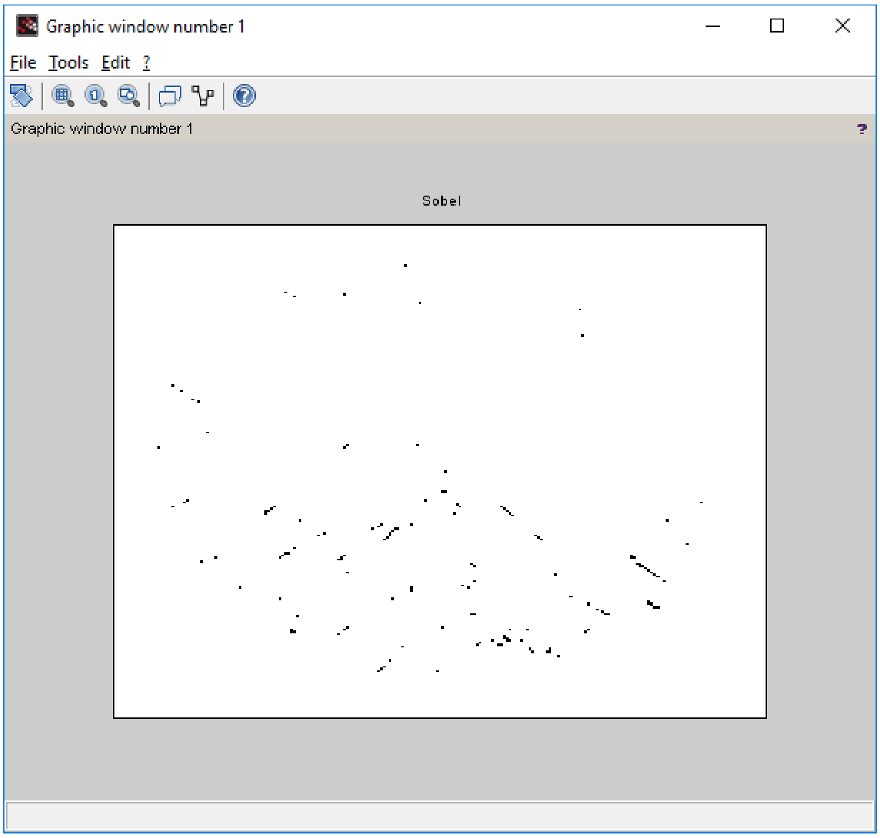


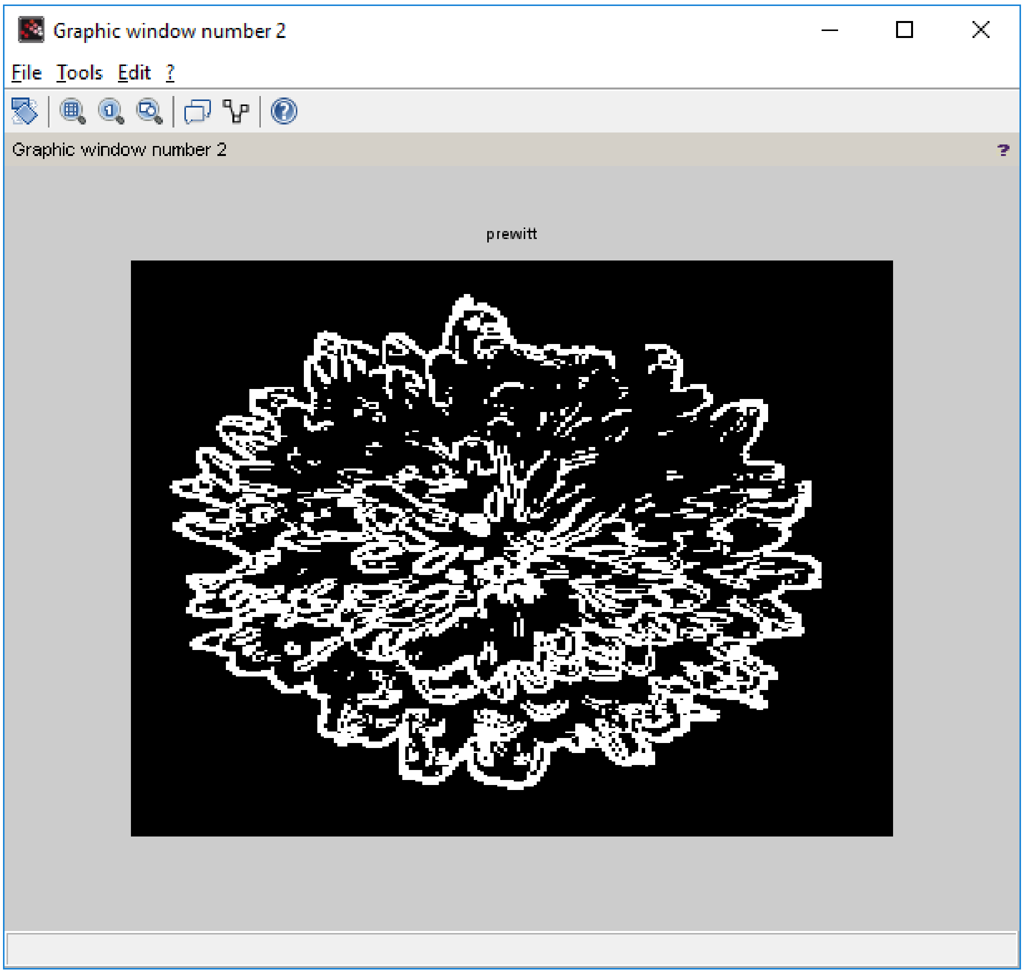
(c) Edge Detection using Different Edge detectors

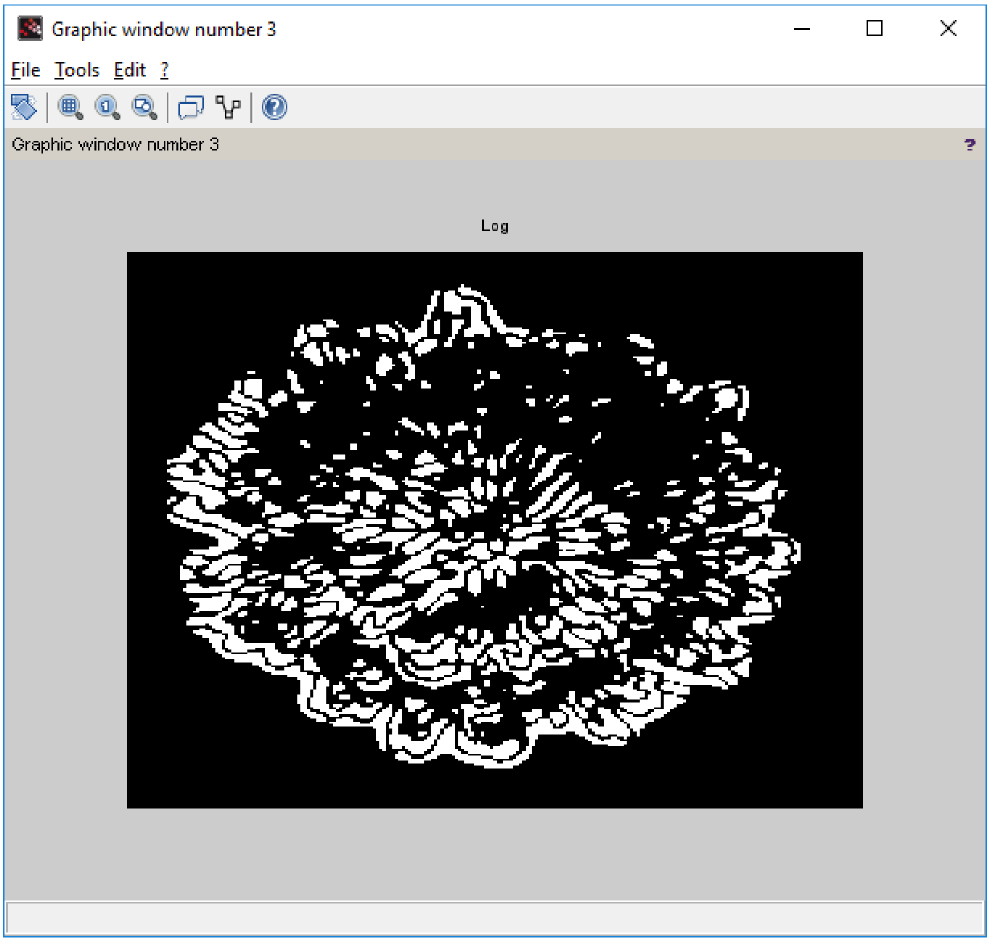
```
close ;
clc ;
a = imread('C:\Users\ADMIN\Desktop\flower.jpg');
a = rgb2gray(a);
c = edge(a,'sobel');
d = edge(a,'prewitt');
e = edge(a,'log');
f = edge (a,'canny');
imshow(a)
title('Original Image')
figure
imshow(c)
title('Sobel')
figure
imshow(d)
title('prewitt')
figure
imshow(e)
title('Log')
figure
imshow(f)
title('Canny')
```

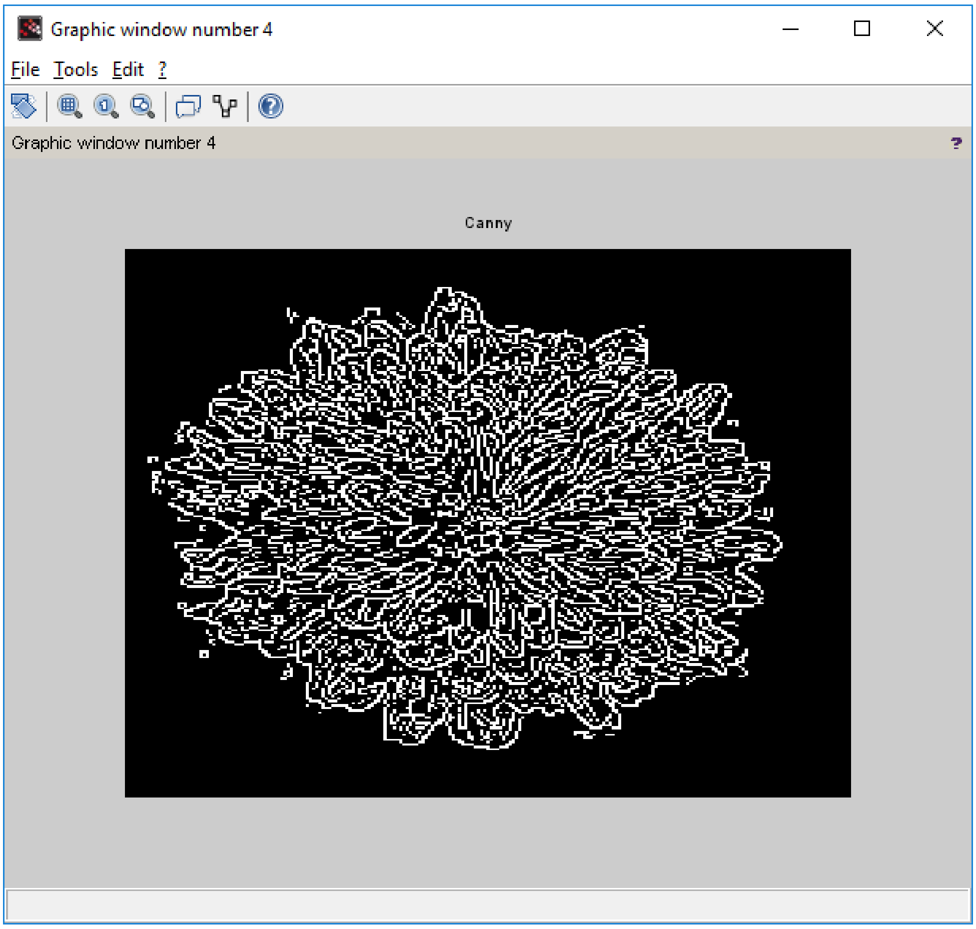
Output:




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


Graphic window number 2
File Tools Edit ?
Graphic window number 2
prewitt


Graphic window number 3
File Tools Edit ?
Graphic window number 3
Log


Graphic window number 4
File Tools Edit ?
Graphic window number 4
Canny

PRACTICAL No. 9

AIM:- Image Compression.

Install Image Processing and Signal Processing packages and restart scilab.

Run this command on console: atomsRemove('scicv')

Restart scilab

And run code

(a) Block Truncation Coding BTC (Output in the form of Matrix).

Code:-

```
close;
clear;
clc;
x=[65,75,80,70;72,75,82,68;84,72,62,65;66,68,72,80];
disp(x,"Original Block is x = " );
[ m1 n1 ] = size(x);
blk = input("Enter the block size: ");
for i = 1:blk:m1
    for j = 1:blk:n1
        y = x(i:i+(blk-1),j:j+(blk -1));
        m = mean(mean(y));
        disp(m,"mean value is m = ");
        sig = stdev(y);
        disp(sig,"Standard deviation of the block is = ");
        b = y>m;
        disp(b,"Binary allocation matrix is B= ");
        K = sum(sum(b));
        disp(K,"number of ones = ");
        if(K~=blk^2)&( K~=0)
            ml = m-sig*sqrt(K/((blk^2)-K));
            disp(ml,"The value of a = ");
            mu = m+sig*sqrt(((blk^2)-K)/K);
            disp(mu,"The value of b = ");
            x(i:i+(blk-1),j:j+(blk-1))=b*mu+(1-b)*ml;
        end
    end
end
disp(round(x),"Reconstructed Block is x = " );
```

Output

Original Block is x =

```
  65.   75.   80.   70.
  72.   75.   82.   68.
  84.   72.   62.   65.
  66.   68.   72.   80.
```

Enter the block size: 4

mean value is m =

72.25

Standard deviation of the block is =

6.6282225

Binary allocation matrix is B=

```
 F T T F
 F T T F
 T F F F
 F F F T
```

number of ones =

6.

The value of a =

67.115801

The value of b =

80.806998

Reconstructed Block is x =

```
  67.   81.   81.   67.
  67.   81.   81.   67.
  81.   67.   67.   67.
  67.   67.   67.   81.
```

PRACTICAL No. 10

AIM:- Binary Image Processing and Colour Image processing.

Install Image Processing and Signal Processing packages and restart scilab.

Run this command on console: atomsRemove('scicv')

Restart scilab

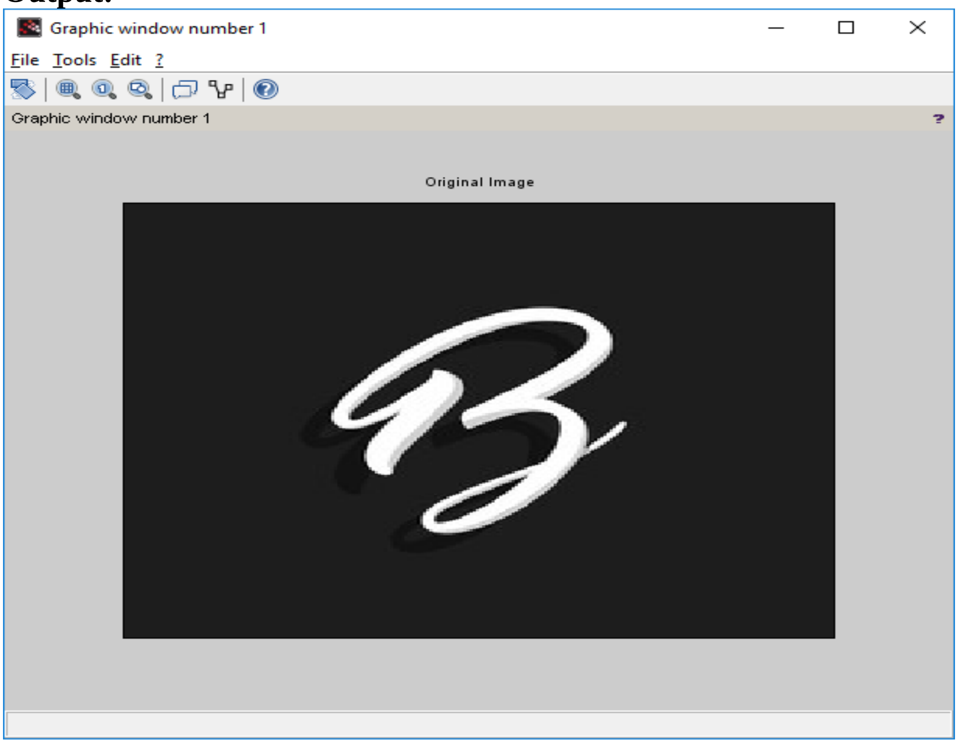
And run code

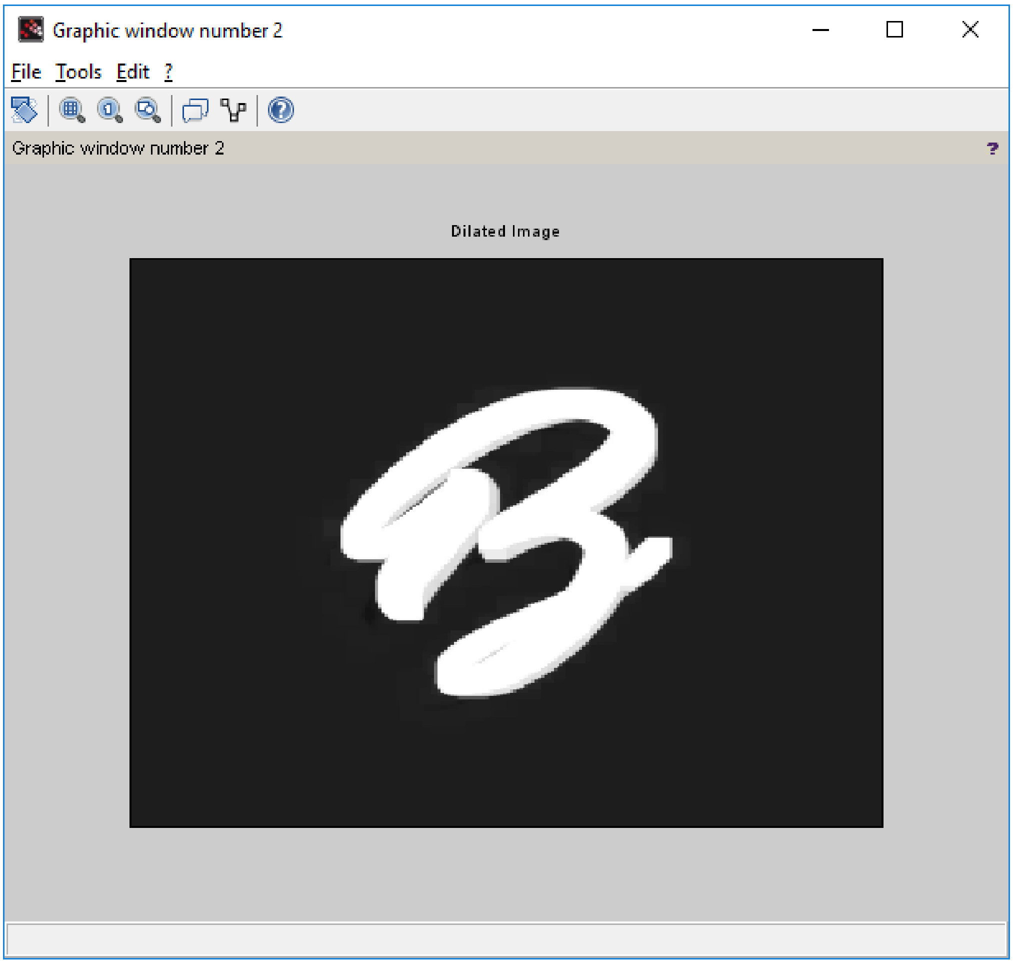
(a) Dilation and erosion process.

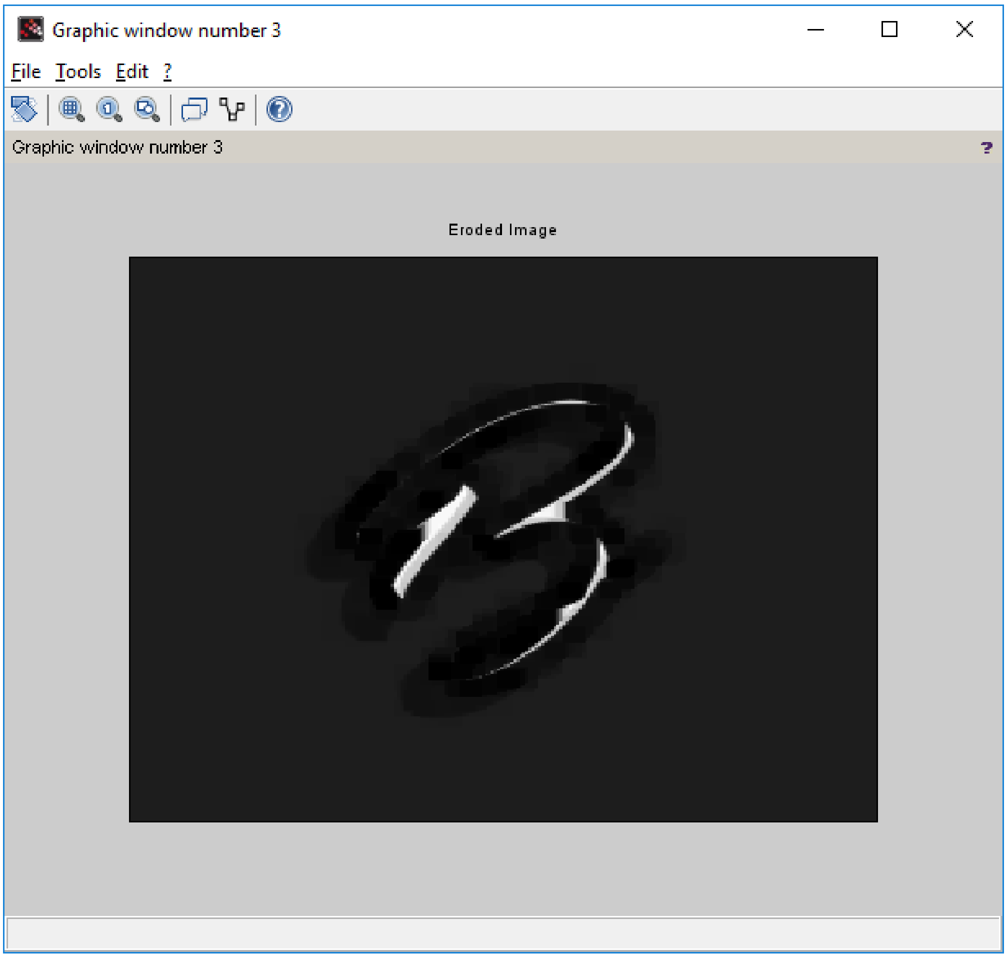
Code:-

```
close ;
clear ;
clc ;
a = imread('C:\Users\ADMIN\Desktop\letter.png');
b = imcreatese('rect',7,7); //Structuring element value can be either rect, ellipse, cross
a1 = imdilate(a,b);
a2 = imerode(a,b);
figure(1)
imshow(a);
title('Original Image')
figure(2)
imshow(a1);
title('Dilated Image')
figure(3)
imshow(a2);
title('Eroded Image')
```

Output:-




Graphic window number 2
File Tools Edit ?
Graphic window number 2
Dilated Image

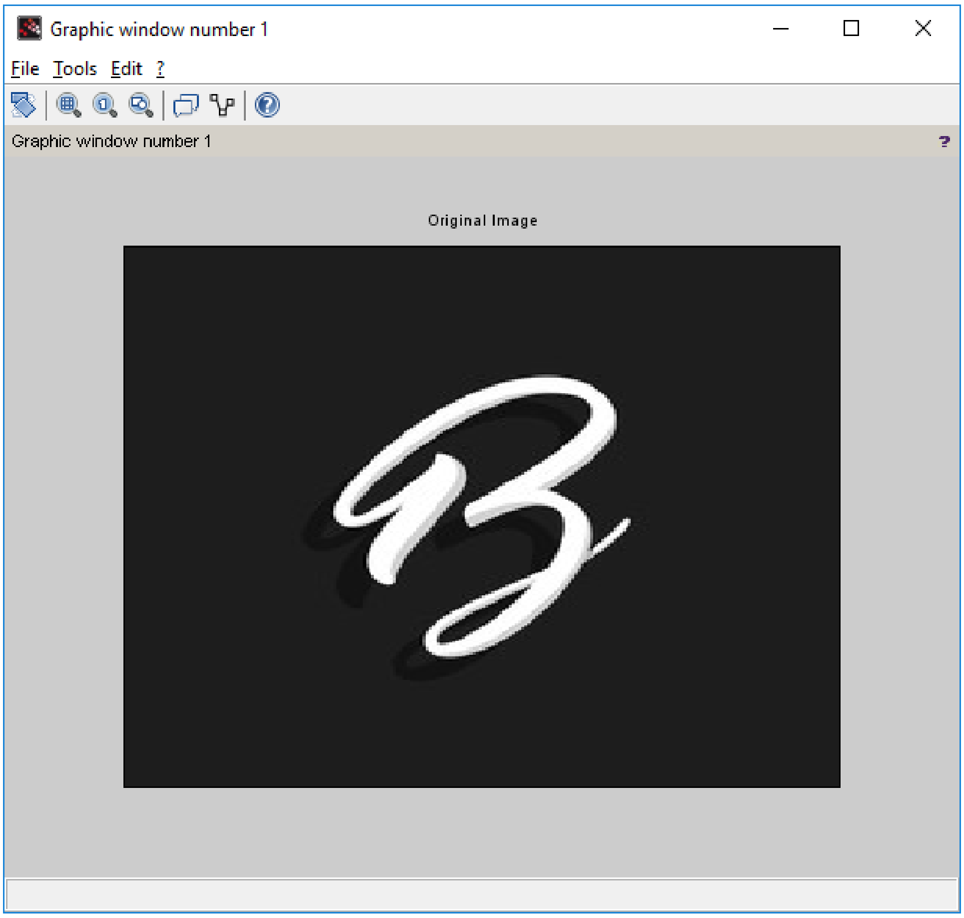

Graphic window number 3
File Tools Edit ?
Graphic window number 3
Eroded Image

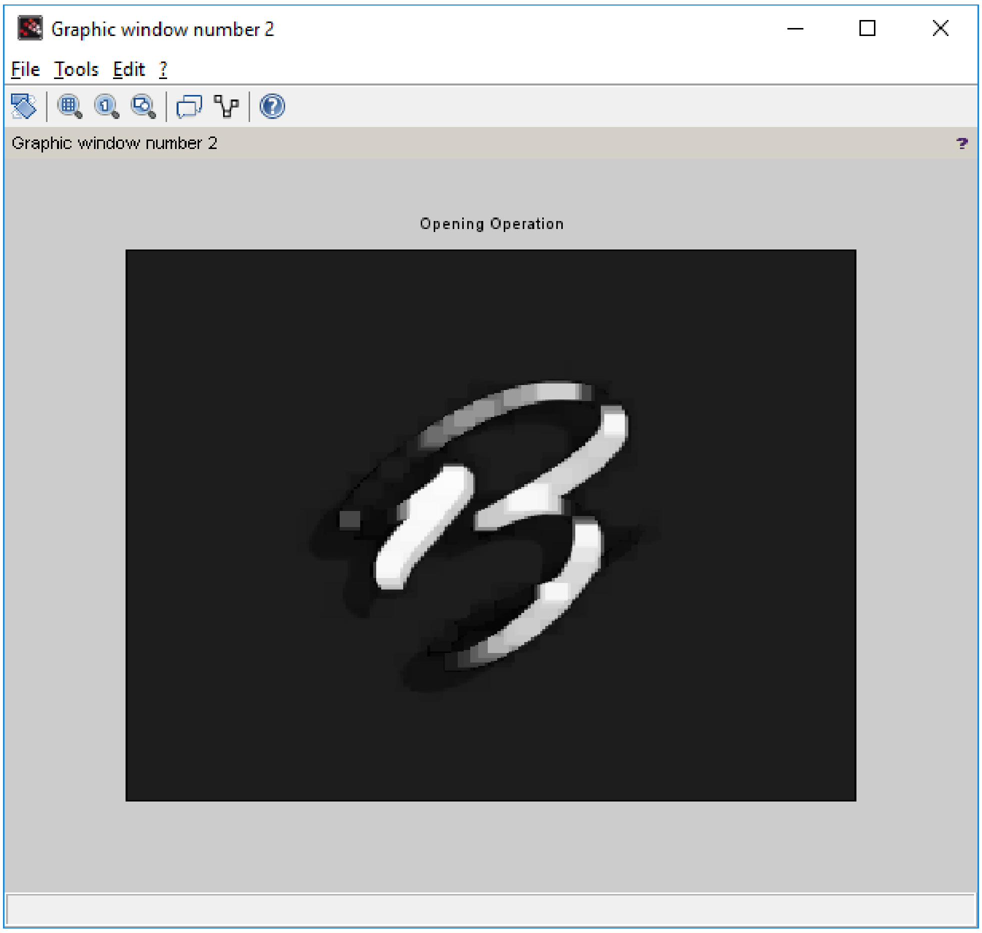
(b) opening and closing operation on the image.

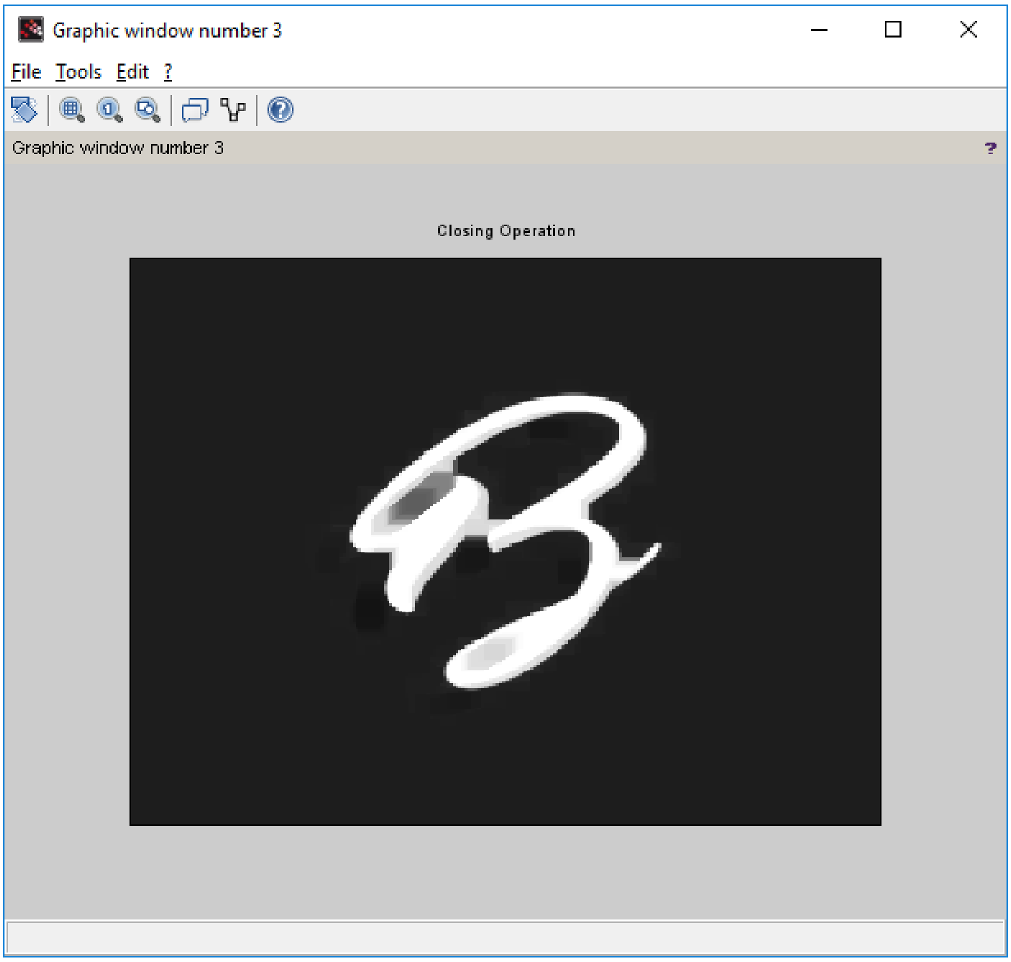
Code:-

```
close ;
clear ;
clc ;
a = imread('C:\Users\ADMIN\Desktop\letter.png');
b = imcreatese('rect',7,7); //Structuring element value can be either rect, ellipse, cross
a1 = imopen(a,b);
a2 = imclose(a,b);
figure(1)
imshow(a);
title('Original Image')
figure(2)
imshow(a1);
title('Opening Operation')
figure(3)
imshow(a2);
title('Closing Operation')
```

Output:-




Graphic window number 2
File Tools Edit ?
Graphic window number 2
Opening Operation

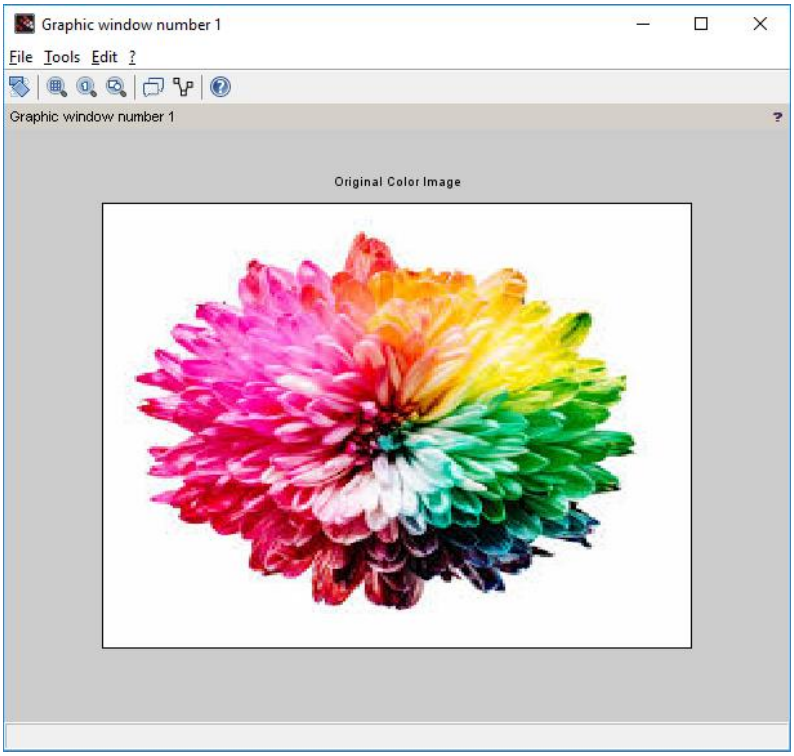

Graphic window number 3
File Tools Edit ?
Graphic window number 3
Closing Operation

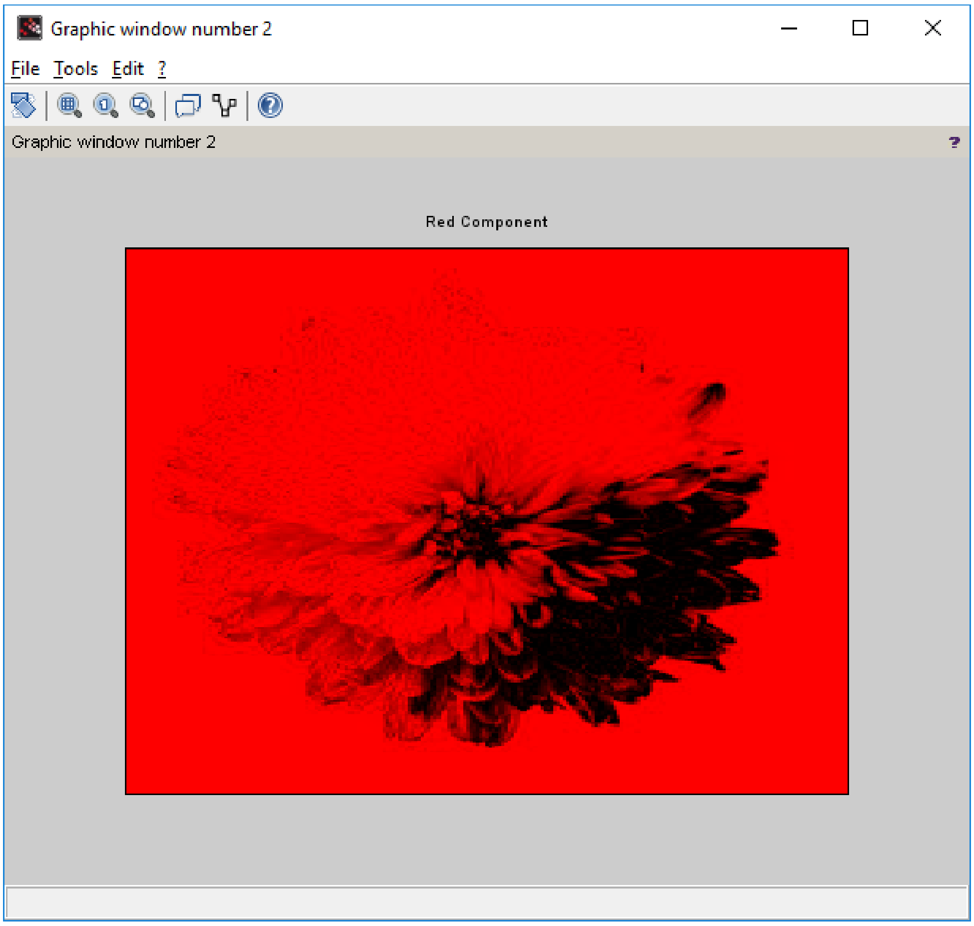
(c) Read an RGB image and extract the three colour components red, green and blue.

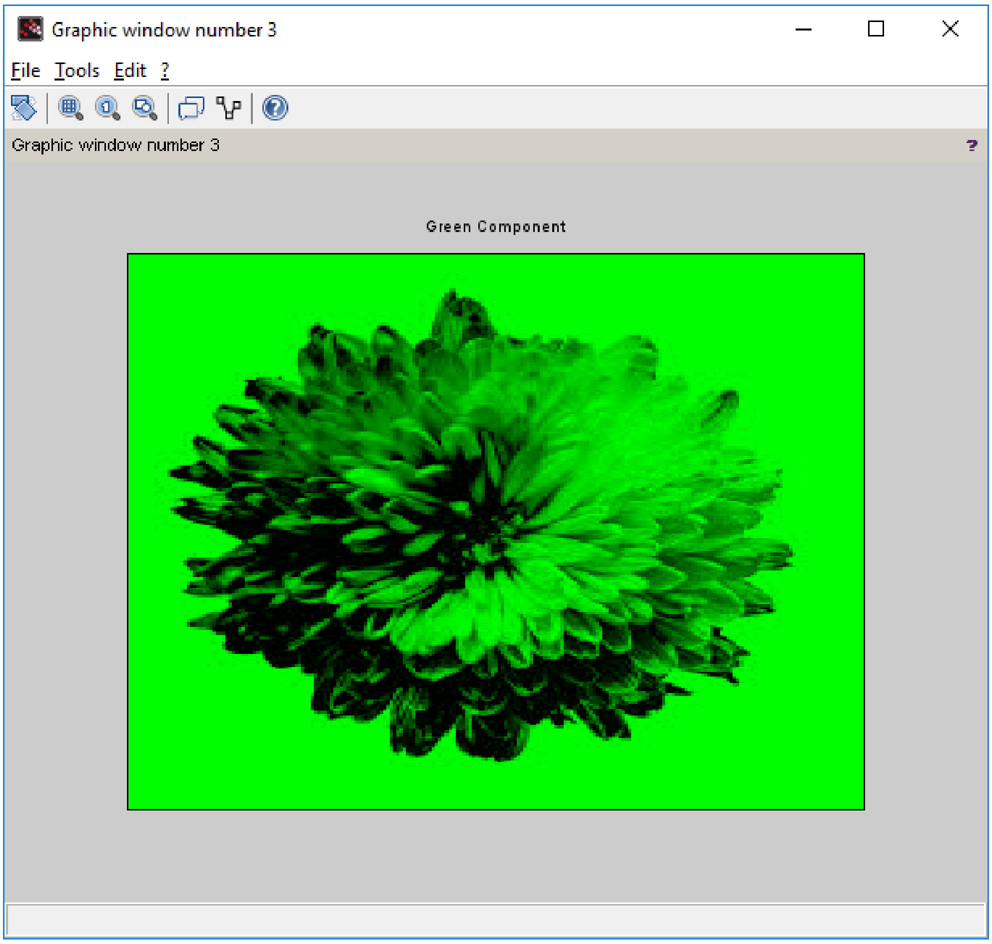
Code:-

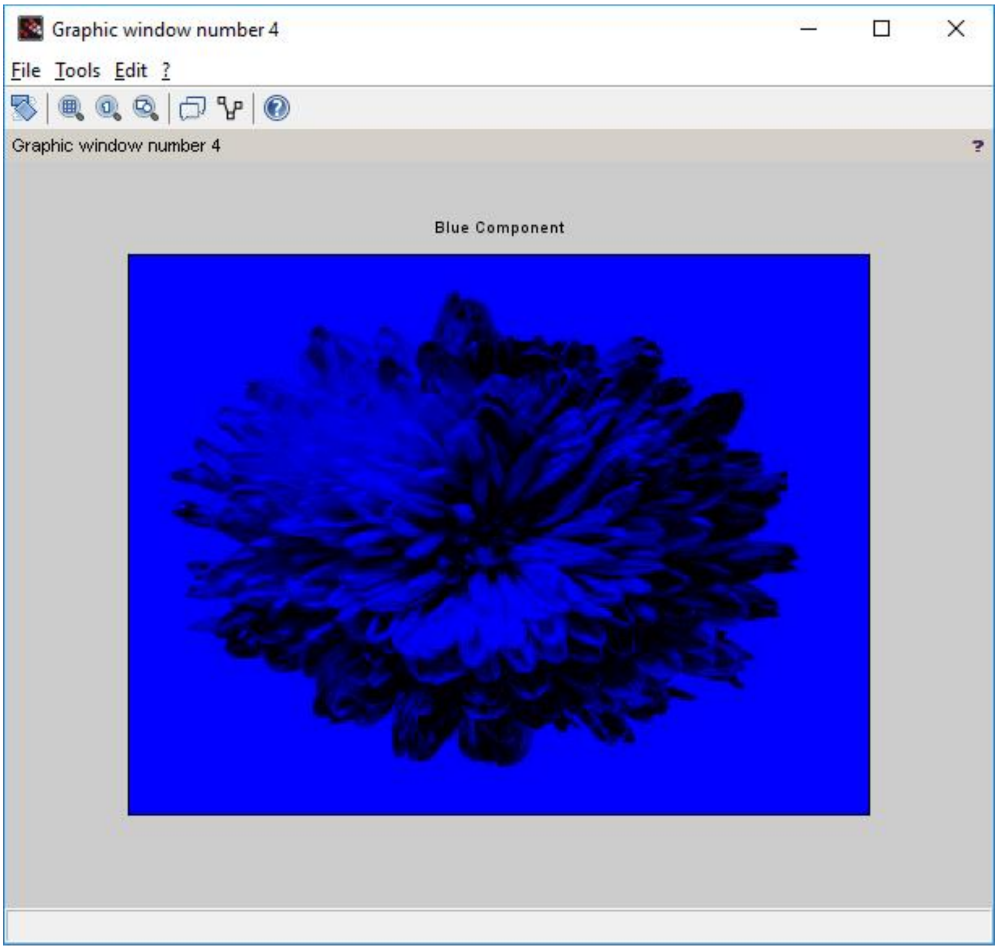
```
clc;
close ;
RGB = imread('C:\Users\ADMIN\Desktop\flower.jpg');
R = RGB;
G = RGB;
B = RGB;
R(: ,: ,2) =0;
R(: ,: ,3) =0;
G(: ,: ,1) =0;
G(: ,: ,3) =0;
B(: ,: ,1) =0;
B(: ,: ,2) =0;
figure(1)
imshow(RGB);
title('Original Color Image');
figure(2)
imshow(R);
title('Red Component');
figure(3)
imshow(G);
title('Green Component');
figure(4)
imshow(B);
title('Blue Component')
```

Output:-




Graphic window number 2
File Tools Edit ?
Graphic window number 2
Red Component


Graphic window number 3
File Tools Edit ?
Graphic window number 3
Green Component

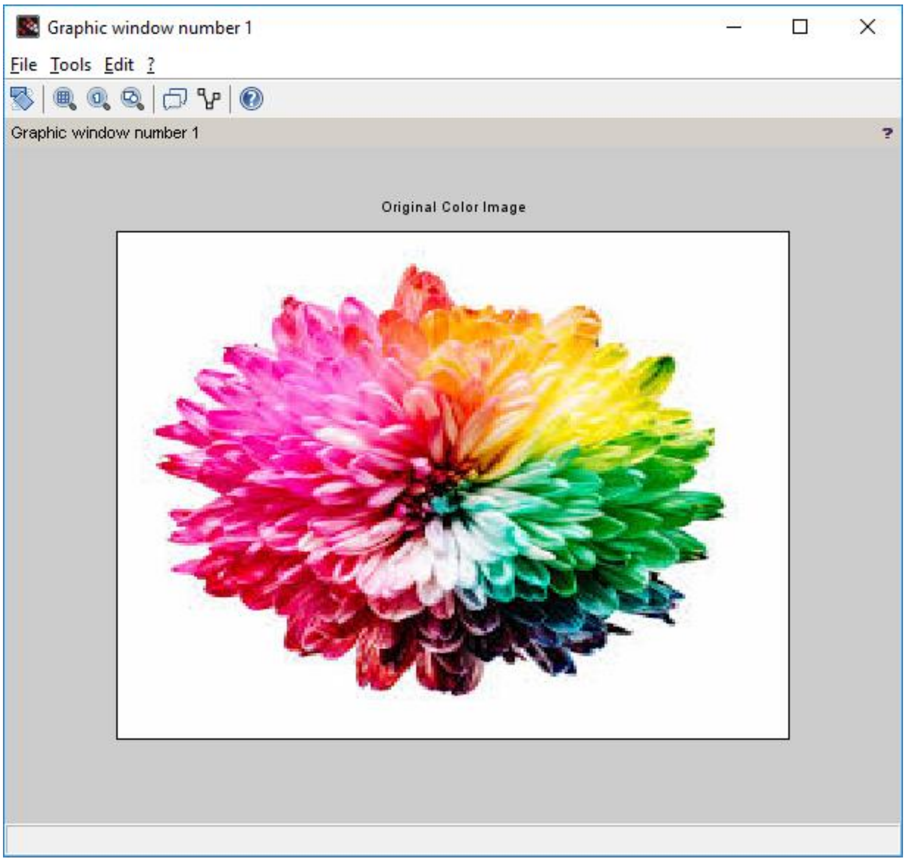


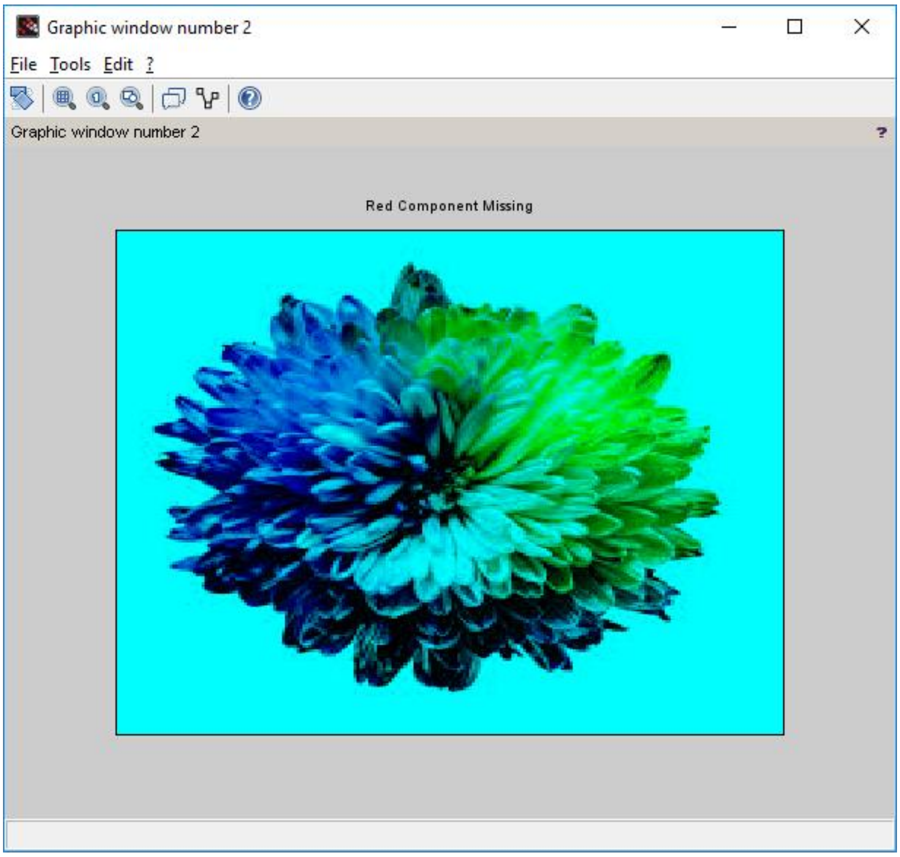
(d) Read a Colour image and separate the colour image into red green and blue planes.

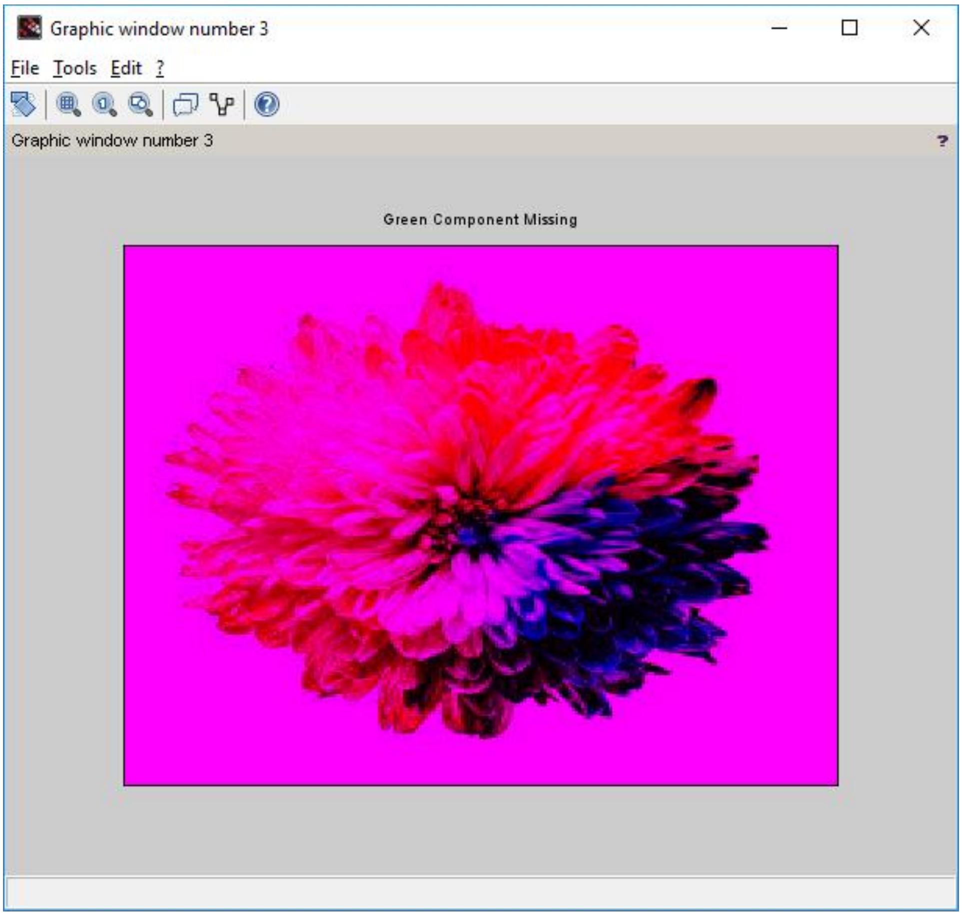
Code:-

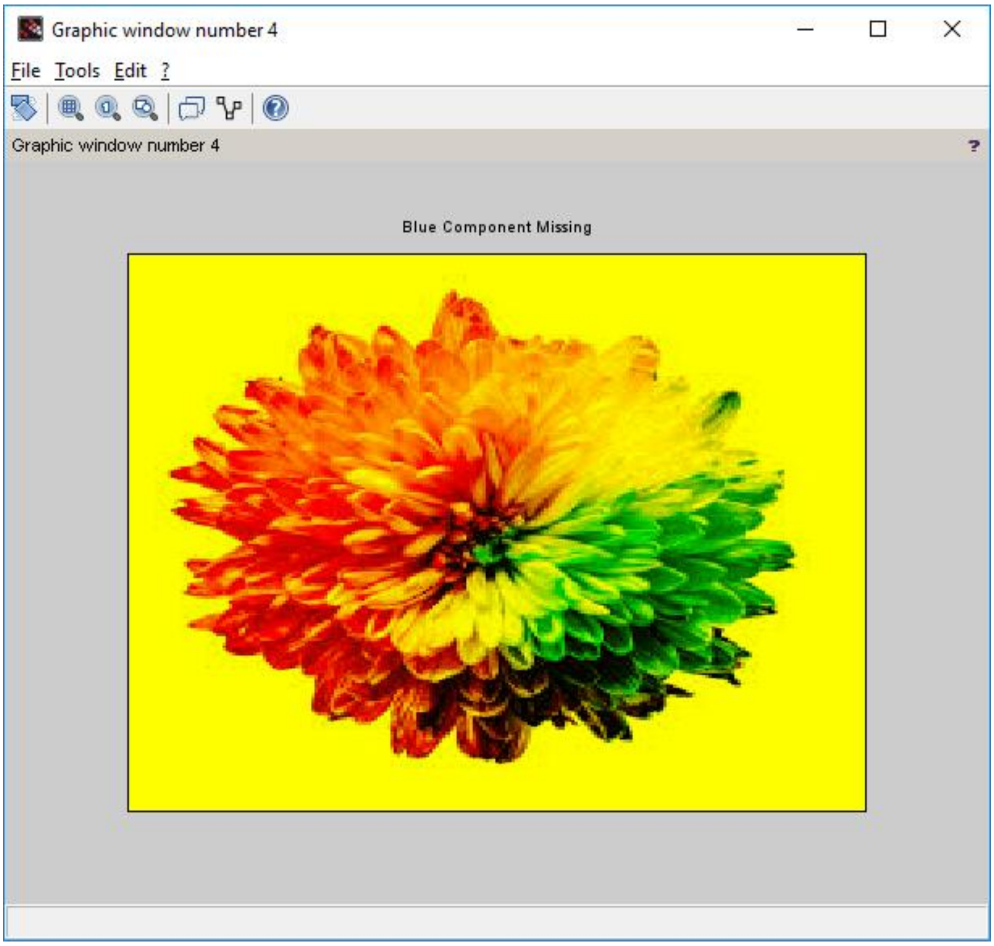
```
clc;
close ;
RGB = imread('C:\Users\ADMIN\Desktop\flower.jpg');
R = RGB;
G = RGB;
B = RGB;
R(: ,: ,1) =0;
G(: ,: ,2) =0;
B(: ,: ,3) =0;
figure(1)
imshow(RGB);
title('Original Color Image');
figure(2)
imshow(R);
title('Red Component Missing');
figure(3)
imshow(G);
title('Green Component Missing');
figure(4)
imshow(B);
title('Blue Component Missing')
```

Output:-


Graphic window number 1
File Tools Edit ?
Graphic window number 1
Original Color Image


Graphic window number 2
File Tools Edit ?
Graphic window number 2
Red Component Missing


Graphic window number 3
File Tools Edit ?
Graphic window number 3
Green Component Missing


Graphic window number 4
File Tools Edit ?
Graphic window number 4
Blue Component Missing