**Practical No. 01**

**Aim: A) Program to calculate a number of samples required for the image.**

**Code:**

clc;

close;

m=4;

n=6;

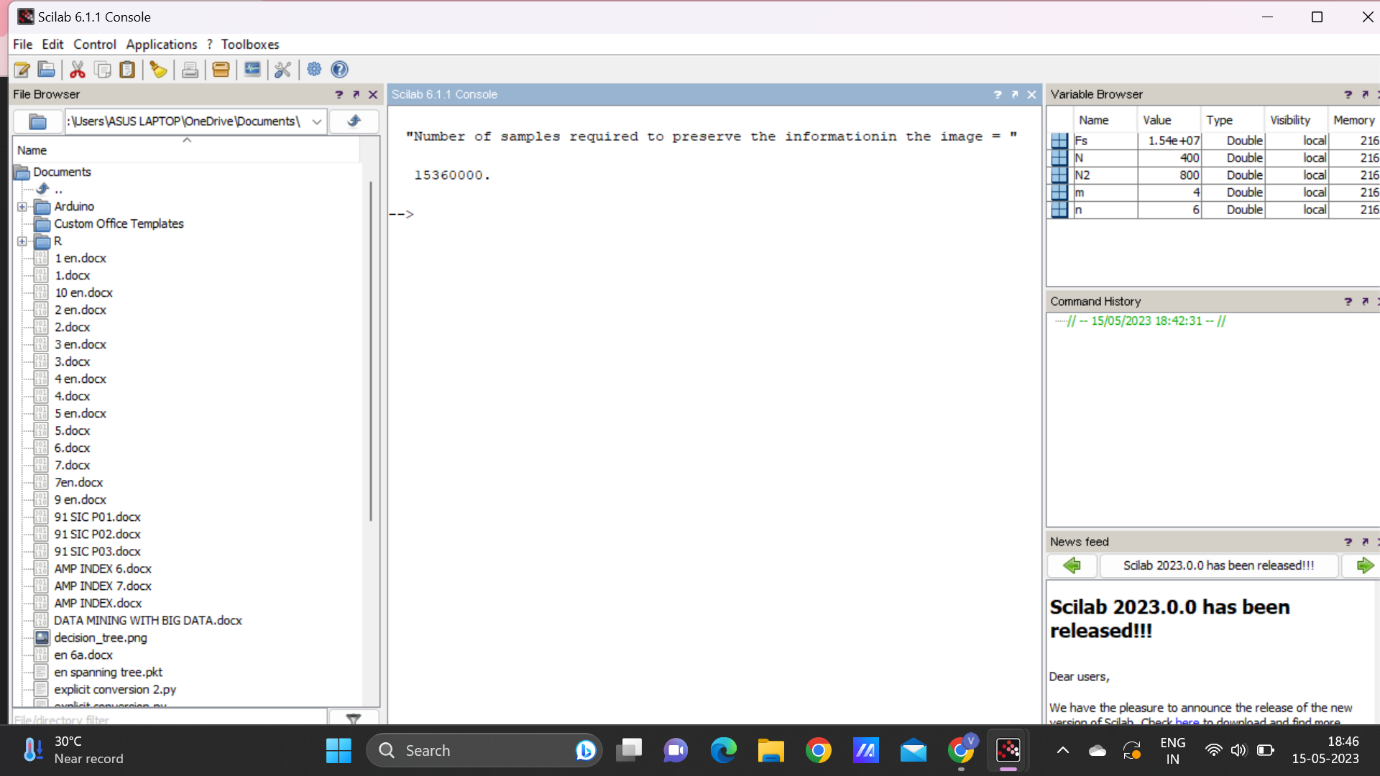
N=400;

N2=2\*N;

Fs=m\*N2\*n\*N2;

disp('Number of samples required to preserve the informationin the image = ',Fs);

**Output:**



**Aim: B) Program to study the effects of reducing the spatial resolution of a digital image.**

**Code:**

n = input('Enter the input samples');

img=rgb2gray(imread('D:\damon.jpeg'));

a=size(img);

w=a(2);

h=a(1);

im=zeros(100);

for i=1:n:h

for j=1:n:w

for k=0:n-1

for l=0:n-1

im(i+k,j+l)=img(i,j);

end

end

end

end

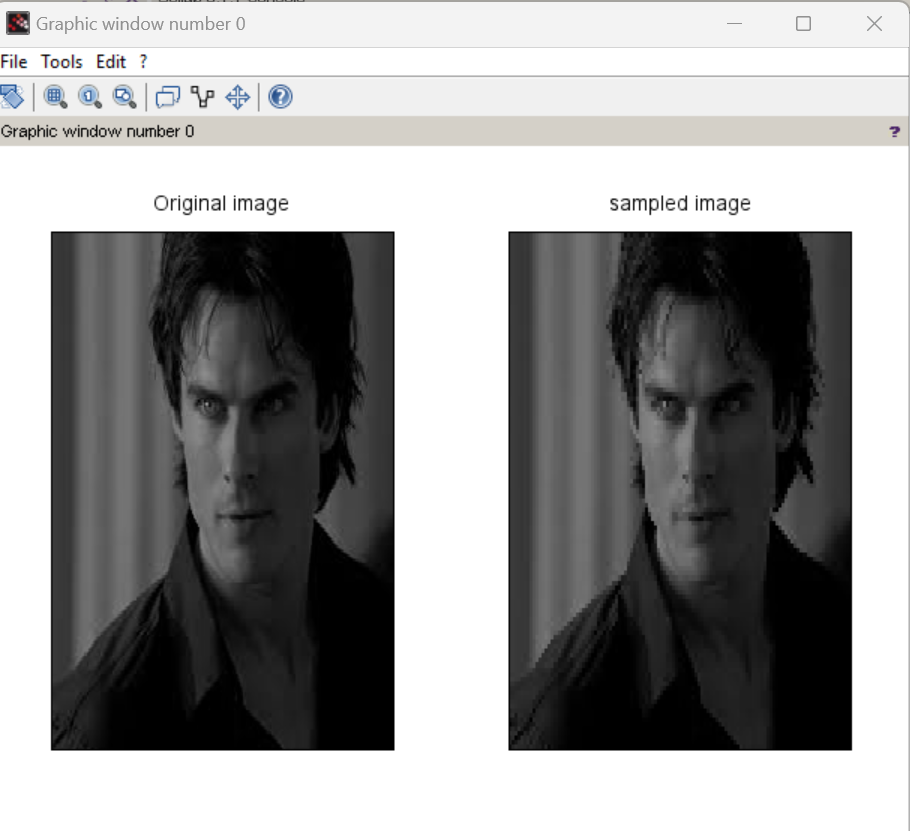
subplot(1,2,1);

imshow(uint8(img));title('Original image');

subplot(1,2,2);

imshow(uint8(im));title('sampled image');

**Output:**



**Practical No. 02**

**Aim: WAP to study the effect of reducing the quantization values and spatial resolution.**

**1) Quantization**

**Code:**

a=imread('D:\damon.jpeg');

[m,n]=size(a);

for i=1:m

for j=1:n

b(i,j)=(a(i,j))/255\*63;

c(i,j)=(a(i,j))/255\*127;

d(i,j)=(a(i,j))/255\*191;

end

end

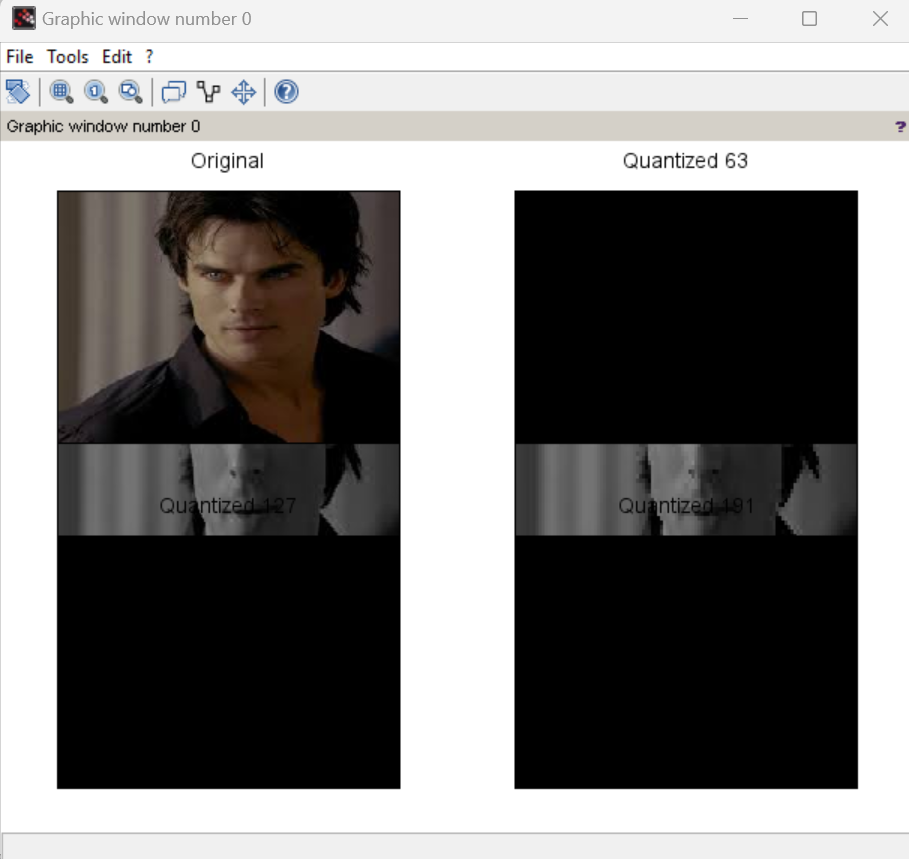
subplot(2,2,1),imshow(a),title('Original');

subplot(2,2,2),imshow(b),title('Quantized 63');

subplot(2,2,3),imshow(c),title('Quantized 127');

subplot(2,2,4),imshow(d),title('Quantized 191');

**Output:**



**2) Spatial Resolution**

**Code**:

i=imread('D:\damon.jpeg');

a=imresize(i,0.8);

b=imresize(i,0.6);

c=imresize(i,0.4);

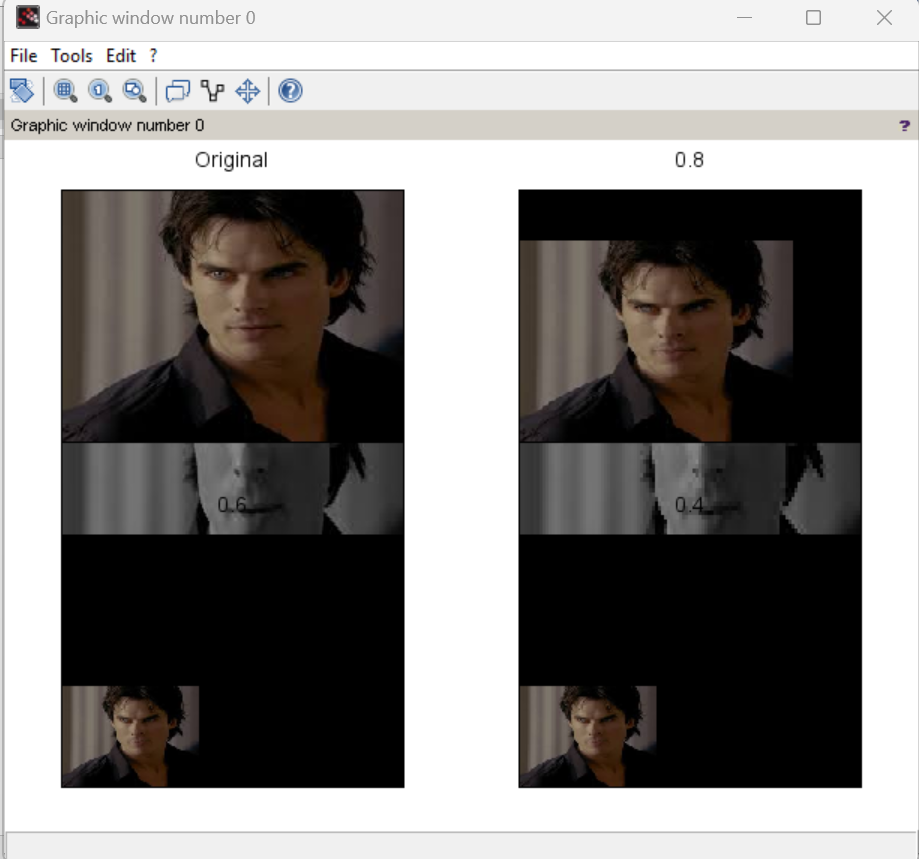
subplot(2,2,1),imshow(i),title('Original');

subplot(2,2,2),imshow(a),title('0.8');

subplot(2,2,3),imshow(c),title('0.6');

subplot(2,2,4),imshow(c),title('0.4');

**Output:**



**Practical No. 03**

**Aim: Image Enhancement**

**A) Thresholding**

**code:**

a=imread('D:\damon.jpeg');

[m,n]=size(a);

for i=1:m

for j=1:n

x=a(i,j);

if x >= 128

b(i,j)=a(i,j)+70;

c(i,j)=a(i,j)+80;

d(i,j)=a(i,j)+100;

else

b(i,j)=a(i,j)-70;

c(i,j)=a(i,j)-80;

d(i,j)=a(i,j)-100;

end

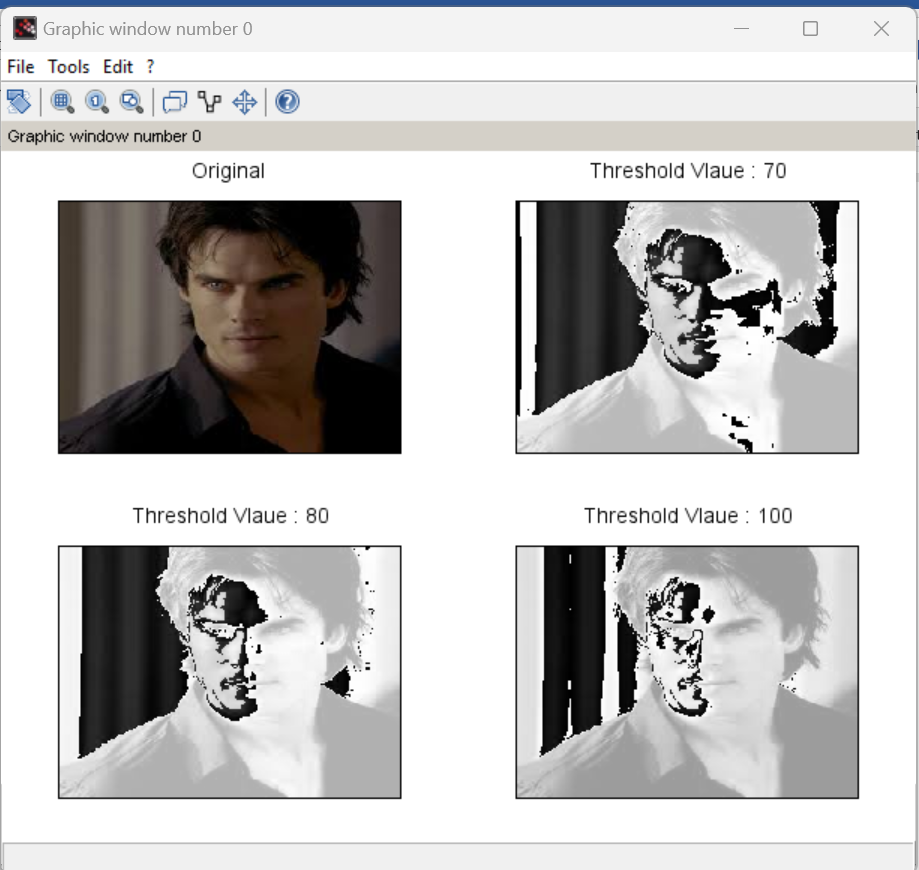
end

end

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

subplot(2,2,2),imshow(b),title('Threshold Vlaue : 70'); subplot(2,2,3),imshow(c),title('Threshold Vlaue : 80'); subplot(2,2,4),imshow(d),title('Threshold Vlaue : 100');

**Output:**



**B) Contrast Adjustment:**

**Code:**

a=imread('D:\damon.jpeg');

r1=100;

r2=140;

s1=150;

s2=240;

l=s1/r1;

m=(s2-s1)/(r2-r1);

n=(255-s2)/(255-r2);

s=size(a);

for i=1:s(1)

for j=1:s(2)

if ((a(i,j) > 0) && (a(i,j) < r1))

b(i,j) = a(i,j)\*l;

end

if ((a(i,j) > r1) && (a(i,j) < r2))

b(i,j) = (m\*(a(i,j)-120))+s1;

end

if ((a(i,j) > r2) && (a(i,j) < 256))

b(i,j) = (n\*(a(i,j)-150))+s2;

end

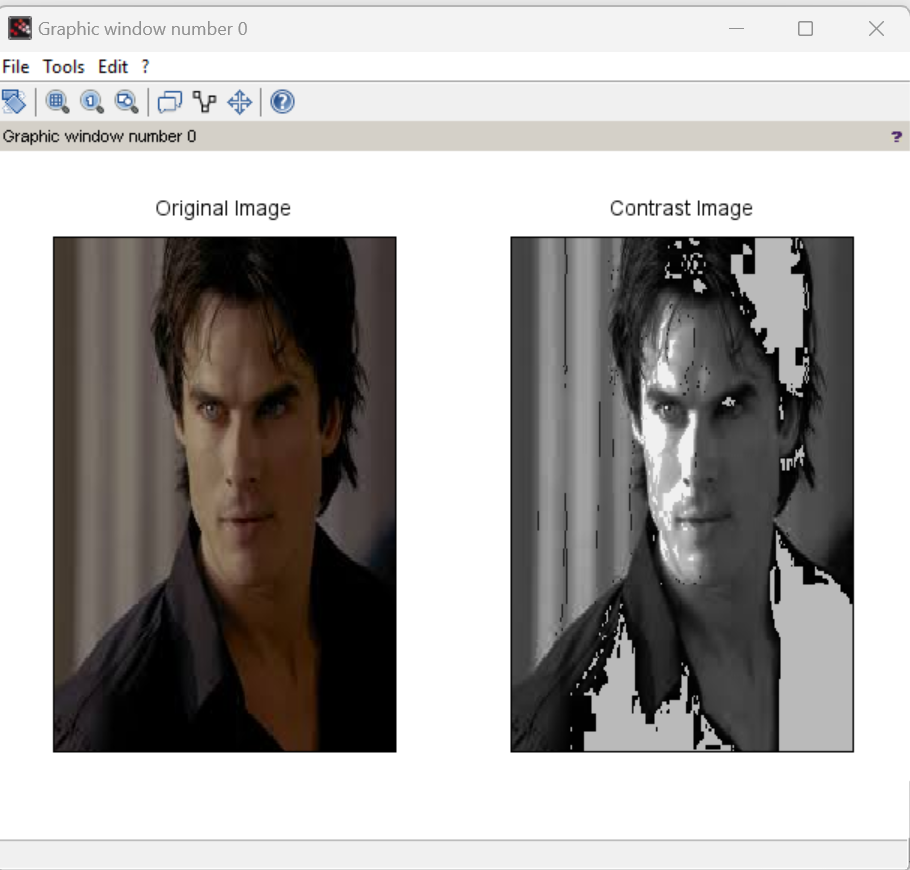
end

end

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

subplot(1,2,2),imshow(uint8(b)),title('Contrast Image');

**Output:**



**C) Brightness Adjustment:**

**Code:**

a=imread('D:\damon.jpeg');

[m,n]=size(a);

for i=1:m

for j=1:n

b(i,j)=a(i,j)-50;

c(i,j)=a(i,j)-100;

d(i,j)=a(i,j)+50;

end

end

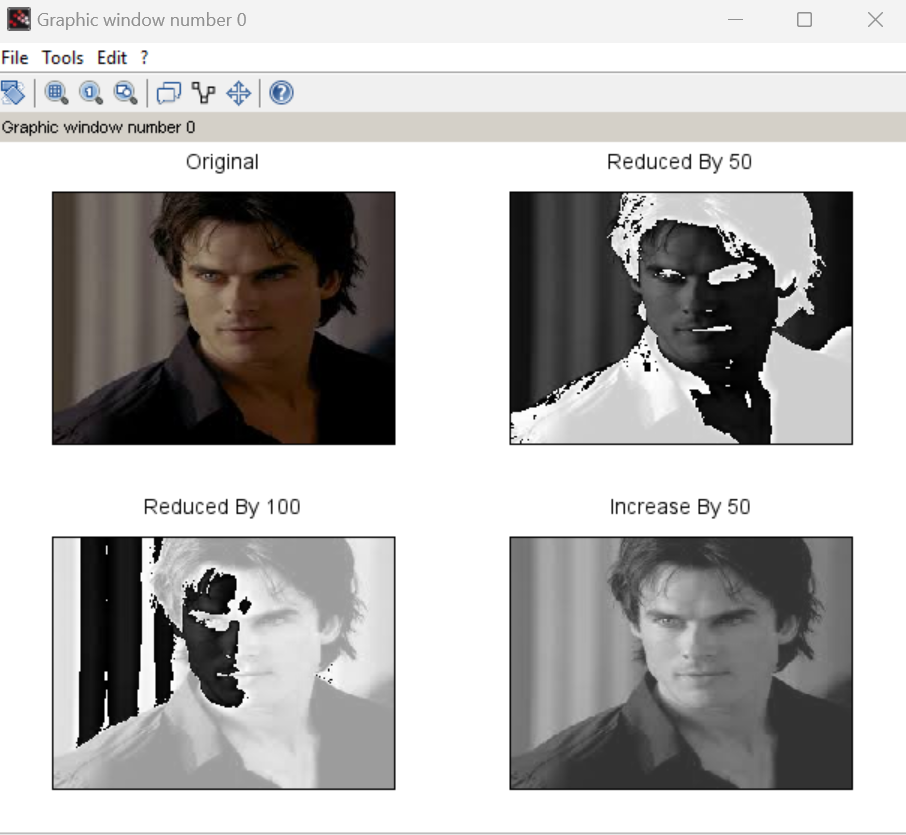
subplot(2,2,1),imshow(a),title('Original');

subplot(2,2,2),imshow(b),title('Reduced By 50');

subplot(2,2,3),imshow(c),title('Reduced By 100');

subplot(2,2,4),imshow(d),title('Increase By 50');

**Output:**



**D) Gray Level Slicing:**

**code:**

a=imread('D:\damon.jpeg');

[m,n]=size(a);

min = 100;

max= 200;

for i=1:m

for j=1:n

x=a(i,j);

if x > min && x < max

b(i,j)=a(i,j);

elseif x > max

b(i,j)=255;

else

b(i,j)=0;

end

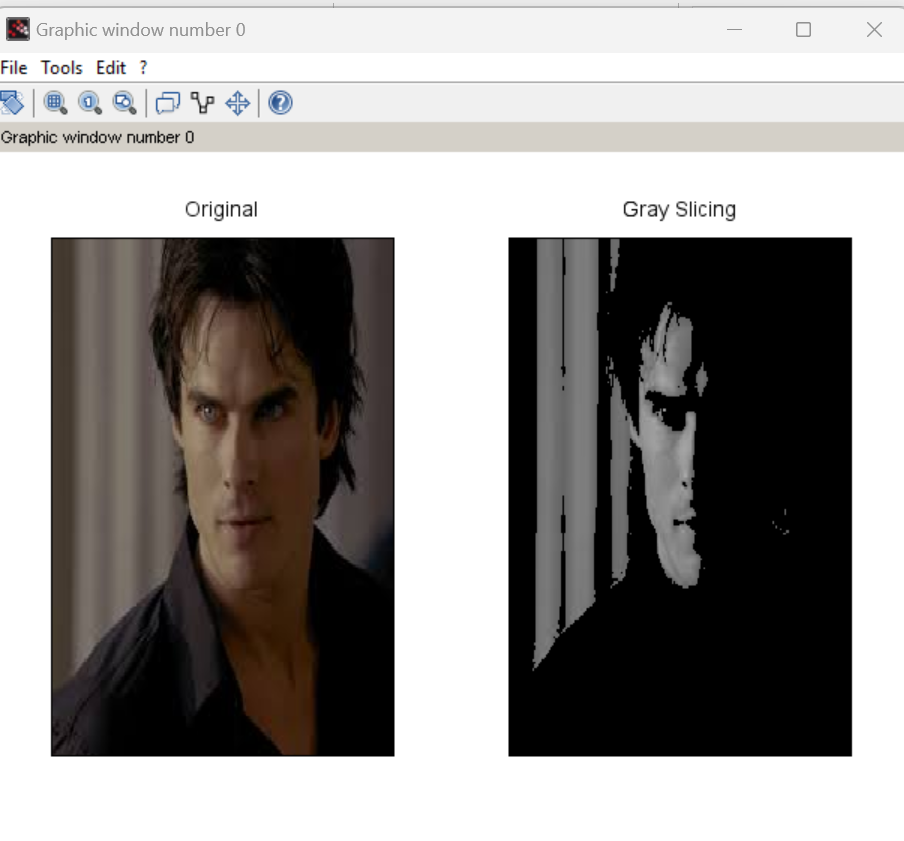
end

end

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

subplot(1,2,2),imshow(b),title('Gray Slicing');

**Output:**



**Practical No. 04**

**Aim: Basic Transformation**

**A) Log Transformation:**

**Code**:

a=imread('D:\klaroline.jpeg');

[m,n]=size(a);

for i=1:m

for j=1:n

x=a(i,j);

b(i,j)=20\*log(1+double(x));

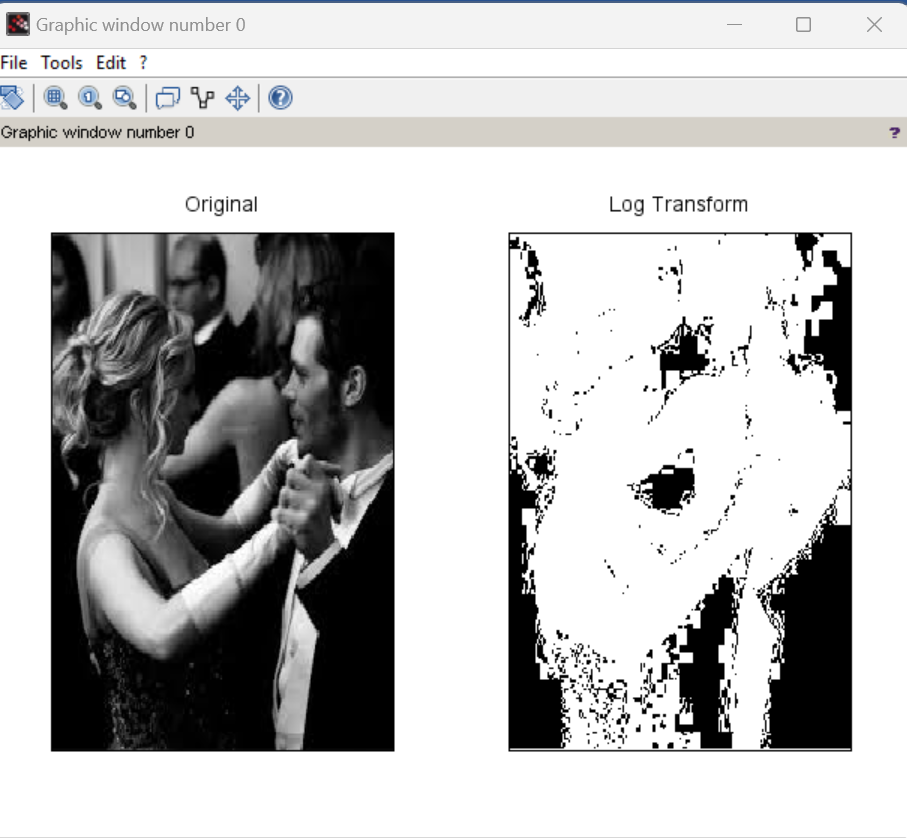
end

end

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

subplot(1,2,2),imshow(b),title('Log Transform');

**Output:**



**B) Power Law Transformation:**

**code:**

a=imread('D:\cameraman.jpeg');

[m,n]=size(a);

for i=1:m

for j=1:n

x=double(a(i,j));

b(i,j)=20\*(x^0.4);

c(i,j)=20\*(x^0.6);

d(i,j)=20\*(x^0.9);

end

end

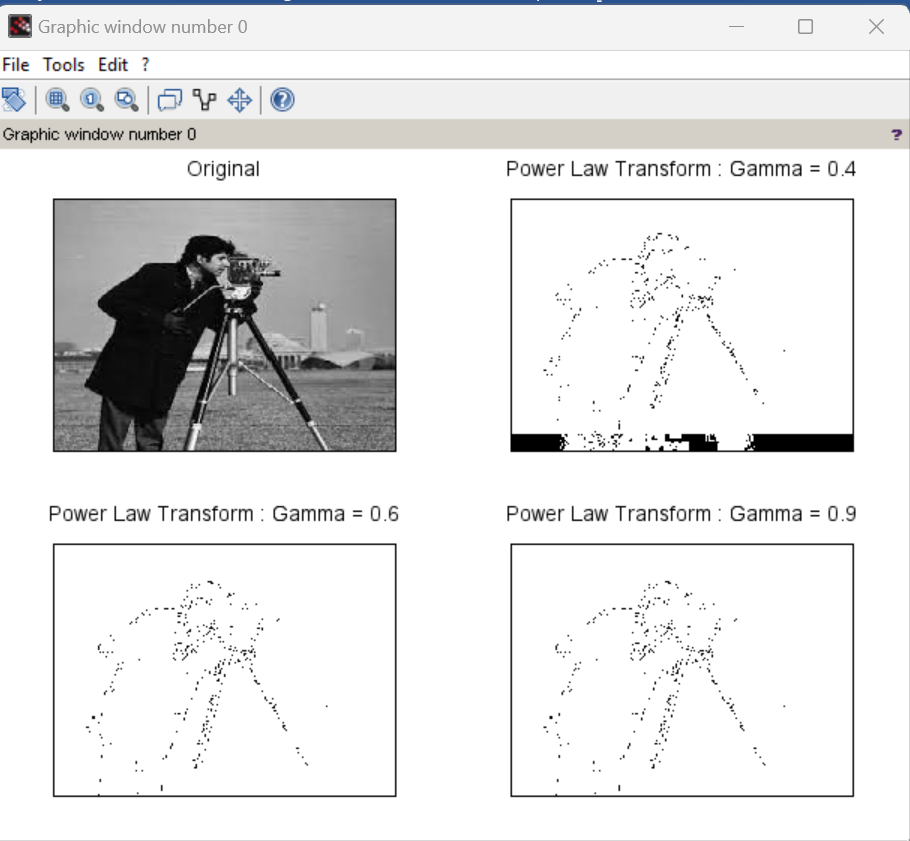
subplot(2,2,1),imshow(a),title('Original');

subplot(2,2,2),imshow(b),title('Power Law Transform : Gamma = 0.4');

subplot(2,2,3),imshow(c),title('Power Law Transform : Gamma = 0.6');

subplot(2,2,4),imshow(d),title('Power Law Transform : Gamma = 0.9');

**Output:**



**C) Negation code**

**code:**

a=imread('D:\klaroline.jpeg');

[m,n]=size(a);

for i=1:m

for j=1:n

b(i,j)=255 - a(i,j);

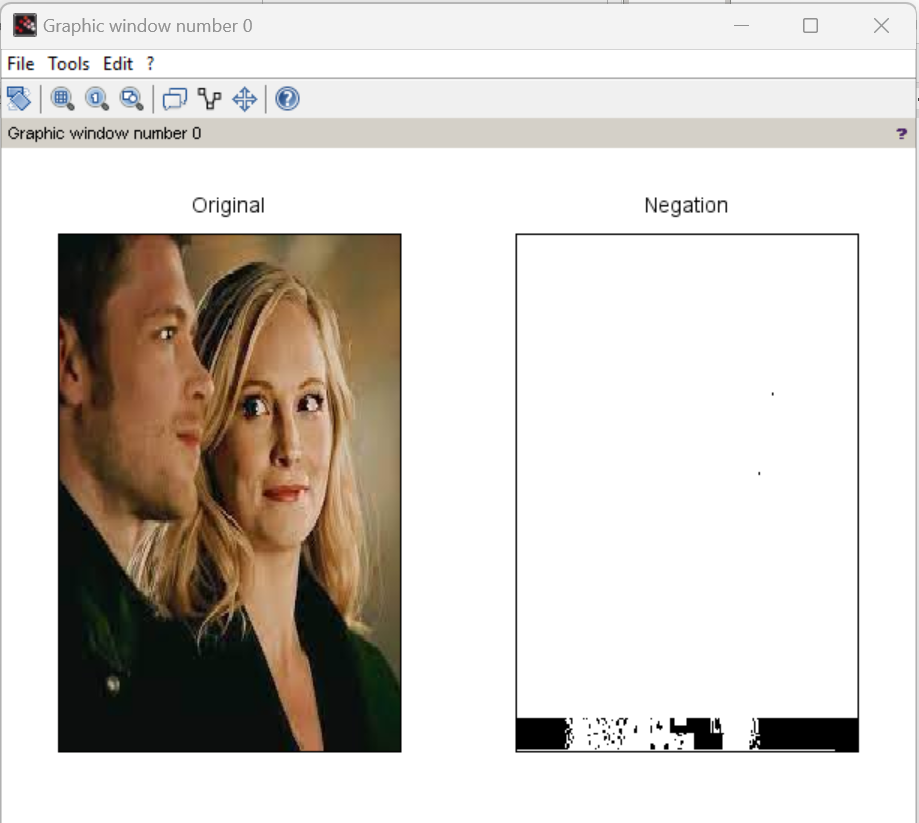
end

end

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

subplot(1,2,2),imshow(b),title('Negation');

**Output:**



**D) Piecewise linear transformations**

**code:**

clc;

clear all;

a=imread('D:\klaroline.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(2,3,5);

imshow(f4);

title('bit 6 Image');

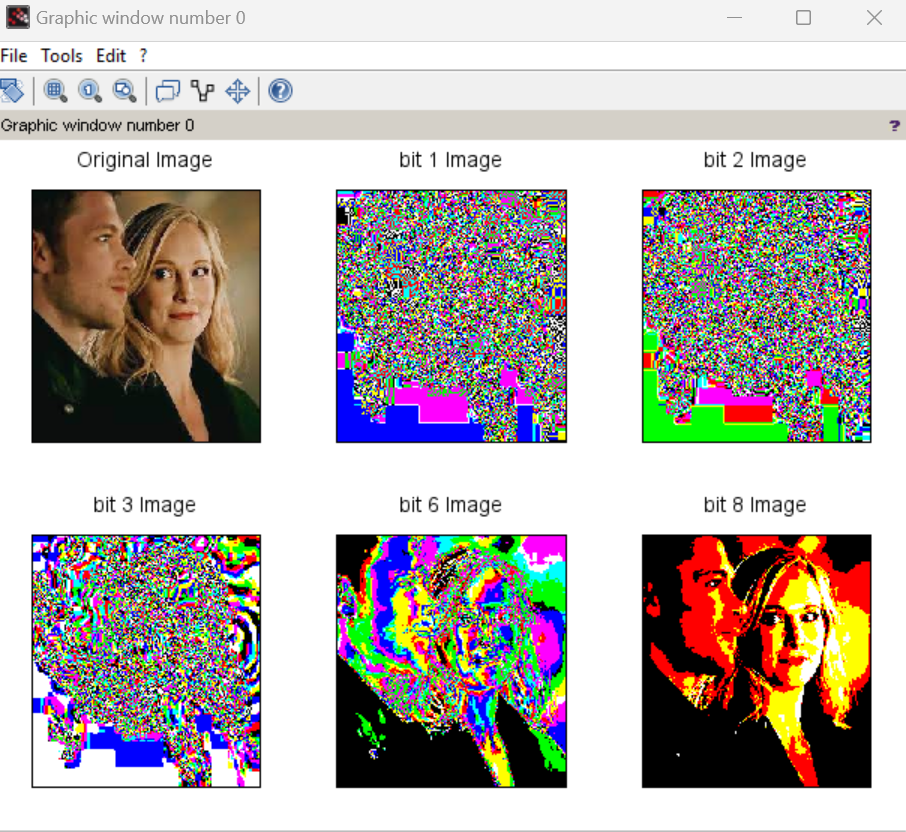
f5=bitget(b,8);

subplot(2,3,6);

imshow(f5);

title('bit 8 Image');

**Output:**



**Practical No. 05**

**Aim: A) Write a program to plot a Histogram for Colour and Grayscale Images.**

**Code:**

a = imread('D:\klaroline.jpeg');

a = double(a);

[row col] = size(a);

h = zeros(1,300);

for n = 1:1:row

for m = 1:1:col

if a(n,m) == 0

a(n,m) = 1;

end

end

end

for n = 1:1:row

for m = 1:1:col

t = a(n,m);

h(t) = h(t)+1;

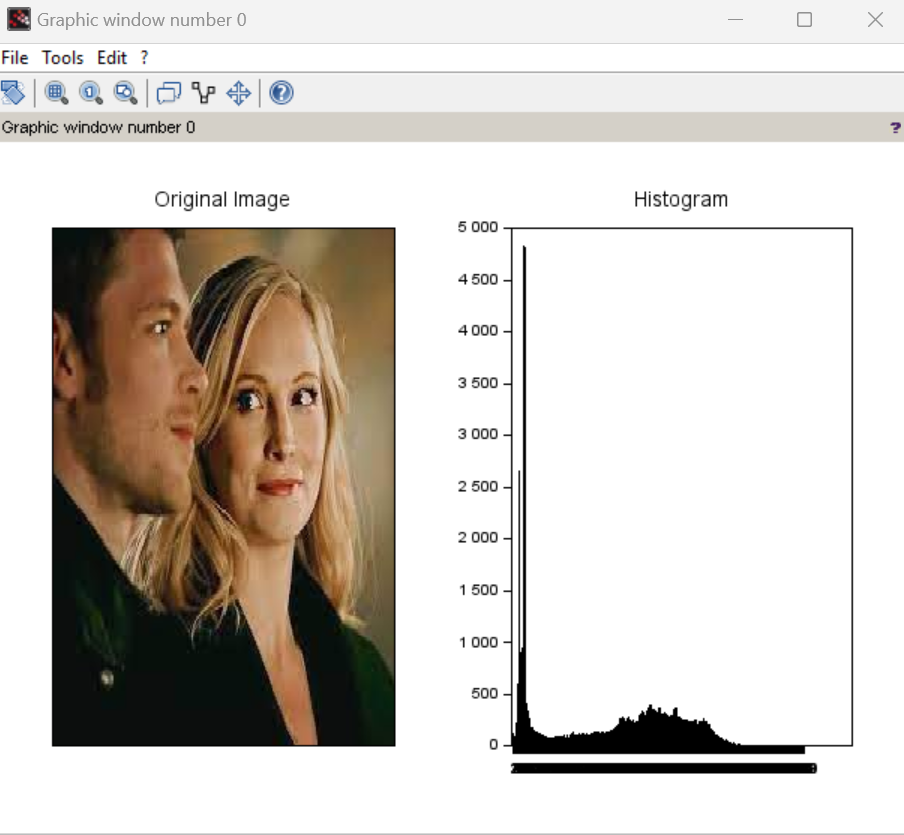
end

end

subplot(1,2,1),imshow(uint8(a)); title('Original Image');

subplot(1,2,2),bar(h),title('Histogram');

**Output:**



**B) Write a program to apply histogram equalization.**

**Code:**

a = imread('D:\klaroline.jpeg');

a = double(a);

big = 256;

[row col d] = size(a);

c = row\*col;

h = zeros(1,300);

z = zeros(1,300);

for e = 1:1:d

for n = 1:1:row

for m = 1:1:col

if a(n,m,e) == 0

a(n,m,e) = 1;

end

end

end

end

for n = 1:1:row

for m = 1:1:col

t = a(n,m);

h(t) = h(t)+1;

end

end

pdf = h/c;

cdf(1) = pdf(1);

for x = 2:1:big

cdf(x) = pdf(x) + cdf(x-1);

end

new = round (cdf\*big);

new = new + 1;

for r = 1:1:d

for p = 1:1:row

for q = 1:1:col

temp = a(p,q,r);

b(p,q,r) = new(temp);

t = b(p,q,r);

z(t) = z(t) + 1;

end

end

end

b = b-1;

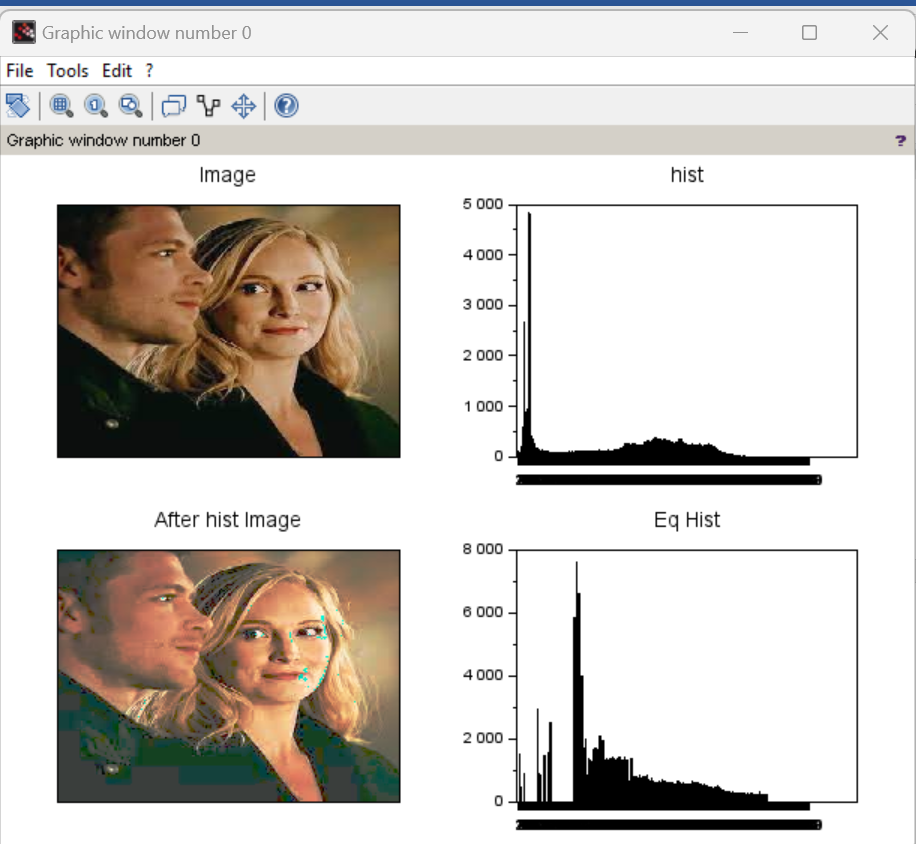
subplot(2,2,1); imshow(uint8(a)); title('Image');

subplot(2,2,2); bar(h); title('hist');

subplot(2,2,3); imshow(uint8(b)); title('After hist Image');

subplot(2,2,4); bar(z); title('Eq Hist');

**Output:**



**Practical No. 06**

**Aim: Write a program to apply Gaussian filter on an image.**

**Code:**

m=input('Enter the Size ');

s=input('Enter the value of sigma ');

sum1=0;

a=m/2;

p=0;q=0;

r=1;

t=1;

w=floor(a);

for i=-w:w

for j=-w:w

p=i\*i;

q=j\*j;

g(r,t)=exp(-(p+q)/(2\*s\*s));

sum1=sum(sum(g(r,t)+sum1));

t=t+1;

end

t=1;

r=r+1;

end

for r=1:m

for t=1:m

h(r,t)=g(r,t)/sum1;

t=t+1;

end

t=1;

r=r+1;

end

im=imread('D:\cameraman.jpeg');

p=double(im);

s1=0;

[M N]=size(p);

for x=0:M-m

for y=0:N-m

for s=1:m

for z=1:m

s1=(h(s,z)\*(p(x+s,y+z)))+s1;

end

end

N\_img(x+1,y+1)=s1;

s1=0;

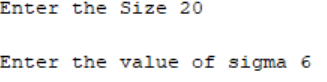
end

end

subplot(1,2,1),imshow(uint8(im)),title('Original Image');

subplot(1,2,2),imshow(uint8(N\_img)),title('Image After Gaussian Filter');

**Output:**





**Practical No. 07**

**Aim: 1) Write a program to apply following morphological operations on the image**.

1. **Opening Code :**

img=imread('cameraman.tif');

se1 = strel('square',11);

im2 = imerode(img,se1);

im3 = imdilate(im2,se1);

subplot(1,2,1),imshow(img),title('orignal image');

subplot(1,2,2),imshow(im3),title('opening image');

**Output:**



1. **Closing Code :**

aa=imread('cameraman.tif');

se1=strel('square',11);

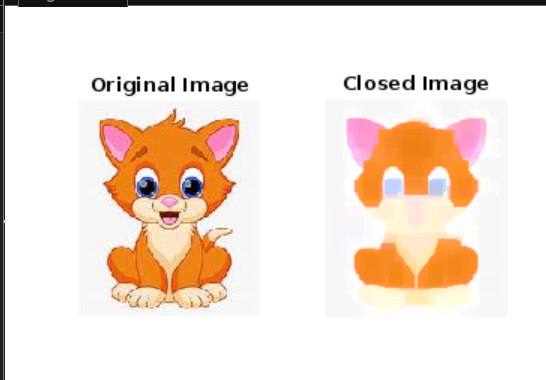
IM2=imdilate(aa,se1);

IM3=imerode(IM2,se1);

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

subplot(1,2,2),imshow(IM3),title('Closed Image');

**Output:**



**C.Morphological Gradient**

**Code :**

img=imread('cameraman.tif');

se1=strel('square',12);

im1=imdilate(img,se1);

im2=imerode(im1,se1);

g=im1-im2;

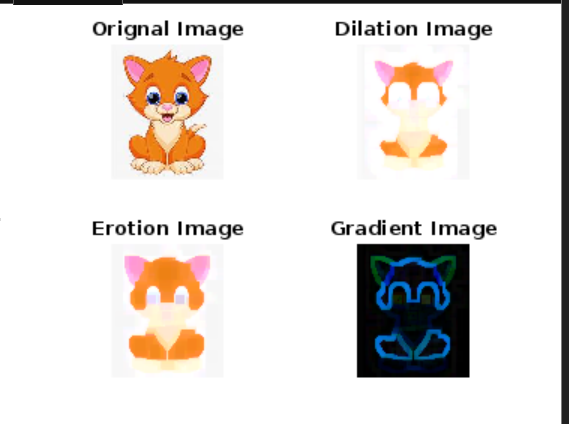
subplot(2,2,1),imshow(img),title('Orignal Image');

subplot(2,2,2),imshow(im1),title('Dilation Image');

subplot(2,2,3),imshow(im2),title('Erotion Image');

subplot(2,2,4),imshow(g),title('Gradient Image');

**Output:**



**D.Top-hat transformation**

**Code:**

i=imread('cameraman.tif');

se1=strel('square',22);

im1=imerode(i,se1);

im2=imdilate(im1,se1);

h=i-im2;

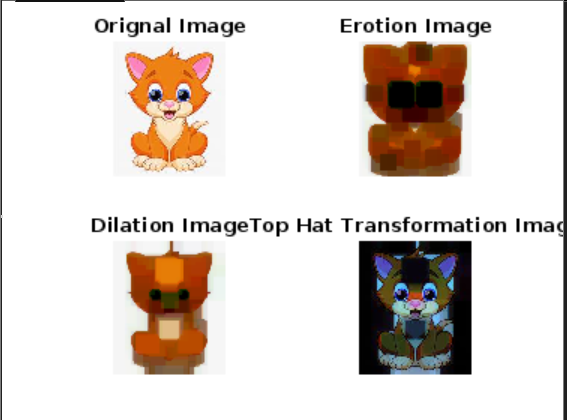
subplot(2,2,1),imshow(i),title('Orignal Image');

subplot(2,2,2),imshow(im1),title('Erotion Image');

subplot(2,2,3),imshow(im2),title('Dilation Image');

subplot(2,2,4),imshow(h),title('Top Hat Transformation Image');

**Output:**



**Aim: 2) Write a program for boundary detection.**

**Code:**

clear all;

clc;

aa=imread('moon.jpeg');

se1=strel('square',11);

subplot(2,1,1),imshow(aa);

m1=imerode(aa,se1);

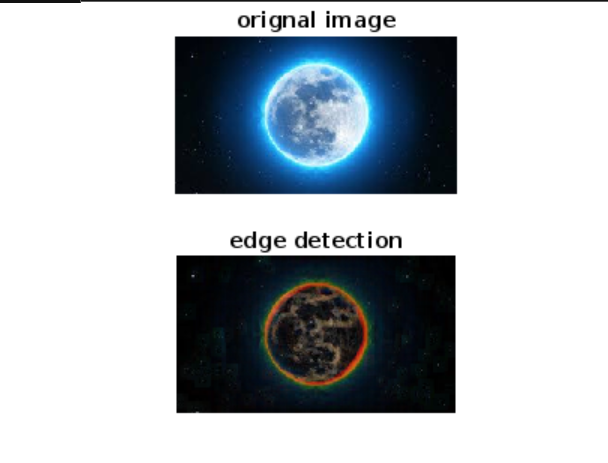
m2=aa-m1;

title('orignal image');

subplot(2,1,2),imshow(m2);

title('edge detection');

**Output:**



**Practical No. 08**

**Aim: A) Write a program to show RGB planes**

**Code:**

original=imread('D:\klaroline.jpeg');

im\_red=original(:,:,1);

im\_green=original(:,:,1);

im\_blue=original(:,:,3);

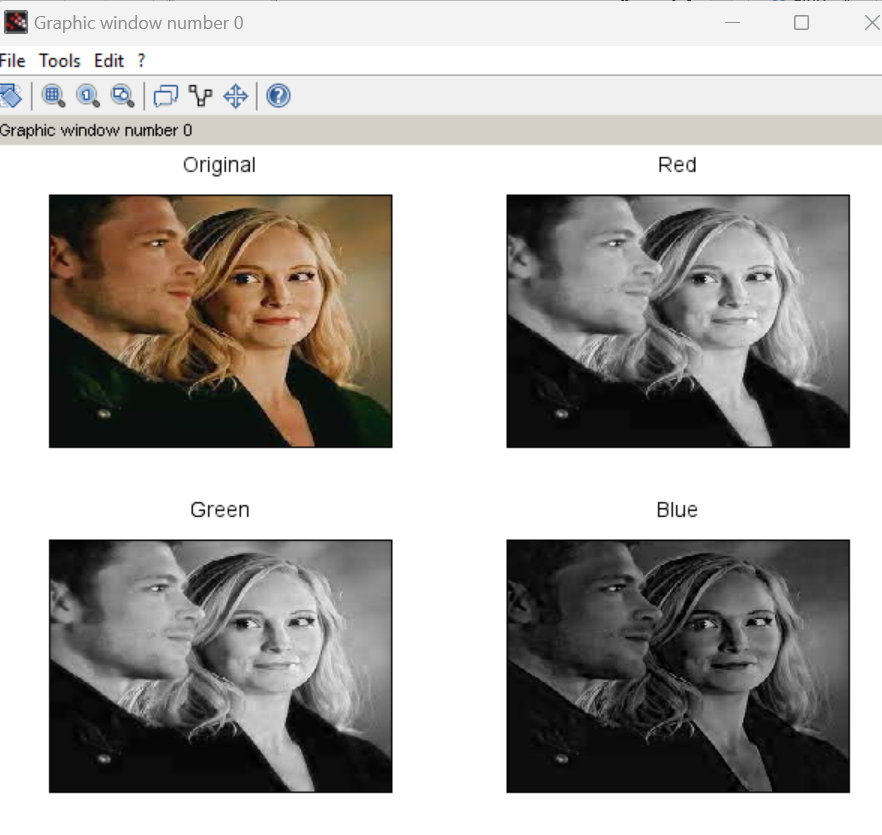
subplot(2,2,1),imshow(original),title('Original');

subplot(2,2,2),imshow(im\_red),title('Red');

subplot(2,2,3),imshow(im\_green),title('Green');

subplot(2,2,4),imshow(im\_blue),title('Blue');

**Output:**



**Aim: B) WAP to convert**

**RGB to NTSC**

**RGB to YCbCr**

**RGB to CMY**

**Code:**

clc;

clear all;

close all;

a = imread('D:\lotus.jpeg');

figure(1),imshow(a);

title('Orignal Image');

k=rgb2ntsc(a);

figure(2),imshow(k);

title('RGB TO NTSC');

l=rgb2ycbcr(a);

figure(3),imshow(l);

title('RGB TO YCbCr');

m=imcomplement(a);

figure(4),imshow(m);

title('RGB TO CMY');

imr=a(:,:,1);

img=a(:,:,2);

imb=a(:,:,3);

figure(5),imshow(imr);

figure(6),imshow(img);

figure(7),imshow(imb);

I=(imr+img+imb)/3;

[m,n]=size(imr);

for c=1:m

for d=1:n

min1=min(imr(c,d),img(c,d));

min2=min(min1,imb(c,d));

S(c,d) = 1-(3/(imr(c,d)+img(c,d)+imb(c,d)))\*min2;

end

end

for c=1:m

for d=1:n

temp= (0.5\*(imr(c,d)-img(c,d))+(imr(c,d)-imb(c,d)))/sqrt(double(imr(c,d)\*imr(c,d)+(imr(c,d)-imb(c,d))\*(img(c,d)-imb(c,d))));

H(c,d)=acos(double(temp));

end

end

for c=1:m

for d=1:n

finali(c,d,1)=I(c,d);

finali(c,d,2)=S(c,d);

finali(c,d,3)=H(c,d);

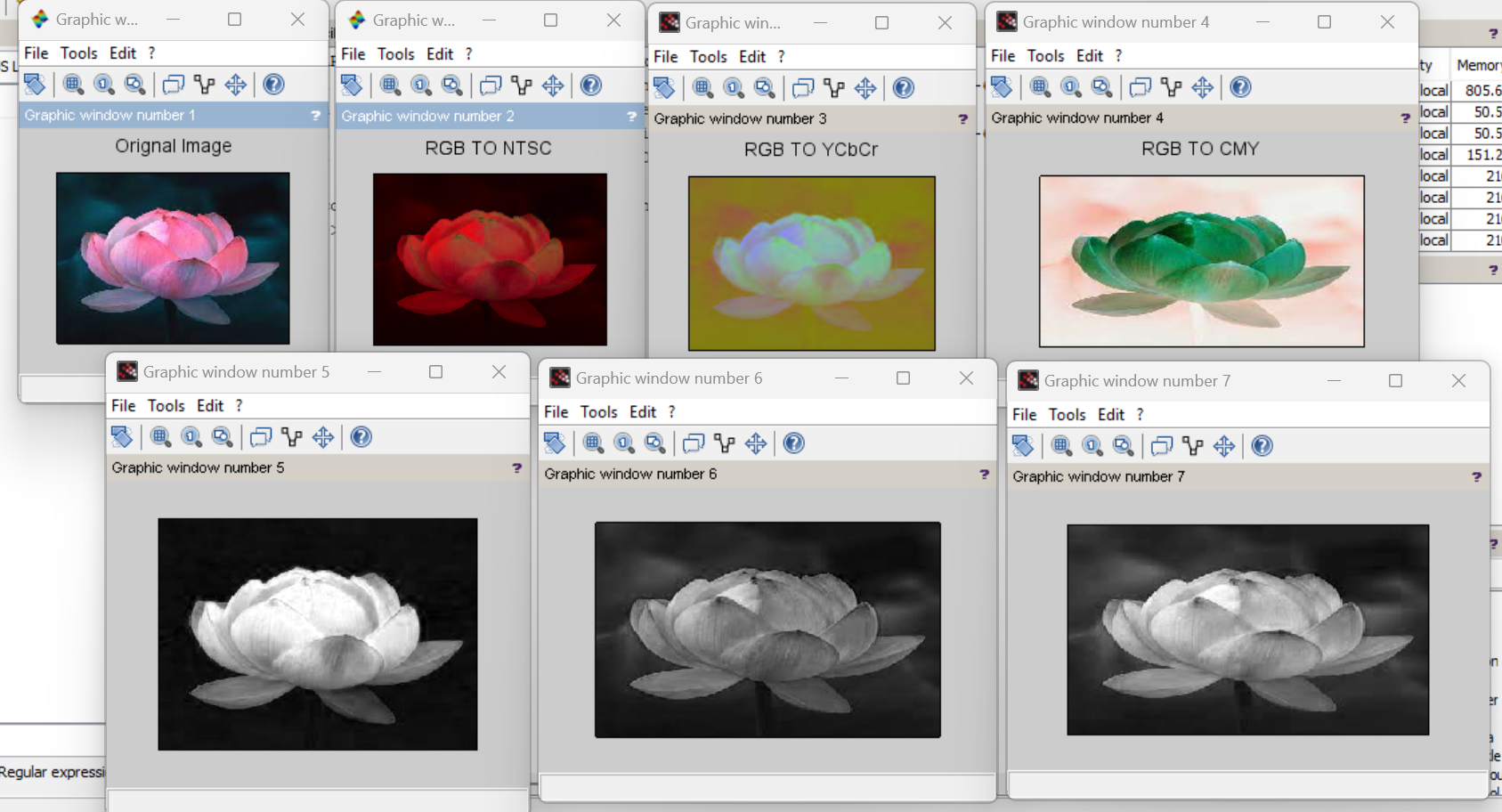
end

end

figure(8),imshow(finali);

title('Final image');

**Output:**



**Practical No. 09**

**Aim: Write a program to achieve Pseudo coloring.**

**Code:**

a=imread('D:\lotus.jpeg');

[l,m,n]=size(a);

for i=1:l

for j=1:m

for k=1:n

if a(i,j)>=0 & a(i,j) < 50

b(i,j,1)=a(i,j,1)+50;

b(i,j,2)=a(i,j,1)+100;

b(i,j,3)=a(i,j,1)+10;

end

if a(i,j)>=50 & a(i,j) < 100

b(i,j,1)=a(i,j,1)+35;

b(i,j,2)=a(i,j,1)+128;

b(i,j,3)=a(i,j,1)+10;

end

if a(i,j)>=100 & a(i,j) < 150

b(i,j,1)=a(i,j,1)+152;

b(i,j,2)=a(i,j,1)+130;

b(i,j,3)=a(i,j,1)+15;

end

if a(i,j)>=150 & a(i,j) < 200

b(i,j,1)=a(i,j,1)+50;

b(i,j,2)=a(i,j,1)+140;

b(i,j,3)=a(i,j,1)+25;

end

if a(i,j)>=200 & a(i,j) < 256

b(i,j,1)=a(i,j,1)+120;

b(i,j,2)=a(i,j,1)+160;

b(i,j,3)=a(i,j,1)+45;

end

end

end

end

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

subplot(1,2,2),imshow(b),title('Pseudo Image');

**Output:**

