

Digital Image Processing

Practical File

Name: - Kuldeep Saini

College Roll No.: 20201415

Exam Roll No.: 20020570017

Semester: 5th

Course: BSc. (Hons) Computer Science

Question 1: Write program to read and display digital image using MATLAB or SCILAB

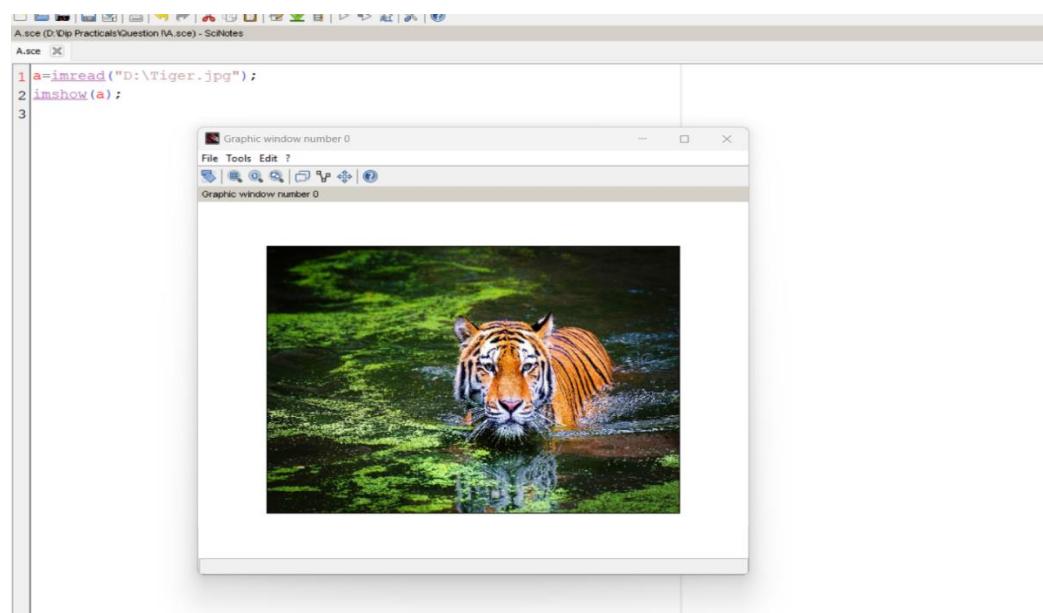
- a. Become familiar with SCILAB/MATLAB Basic commands
- b. Read and display image in SCILAB/MATLAB
- c. Resize given image
- d. Convert given color image into gray-scale image
- e. Convert given color/gray-scale image into black & white image
- f. Draw image profile
- g. Separate color image in three R G & B planes
- h. Create color image using R, G and B three separate planes
- i. Flow control and LOOP in SCILAB
- j. Write given 2-D data in image file

Solution:

Code -

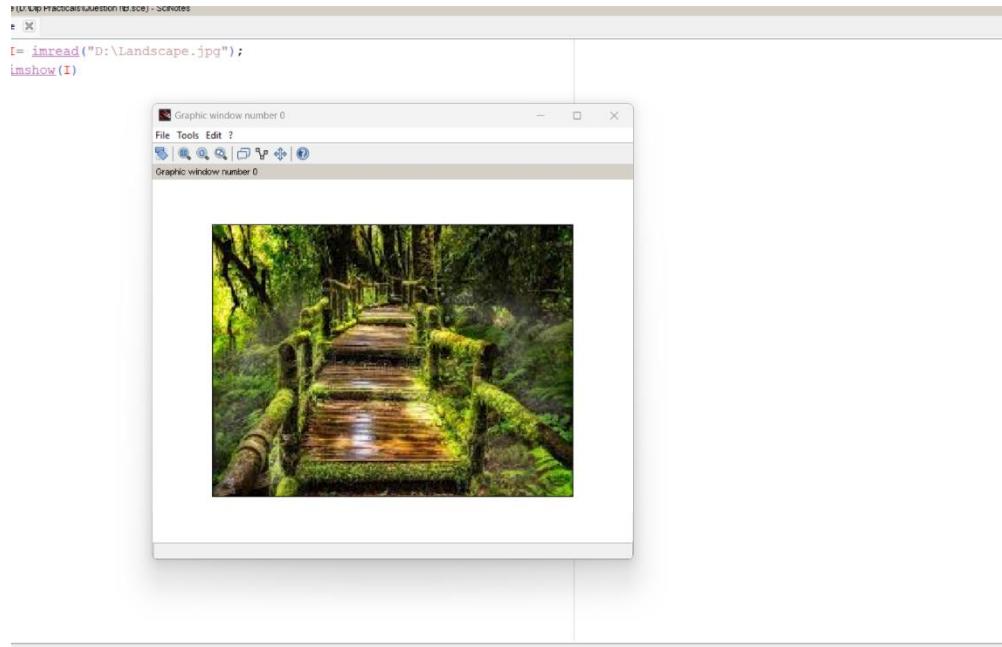
(a)

```
a=imread("D:\ Tiger.jpg");  
imshow(a);  
title("original image");
```



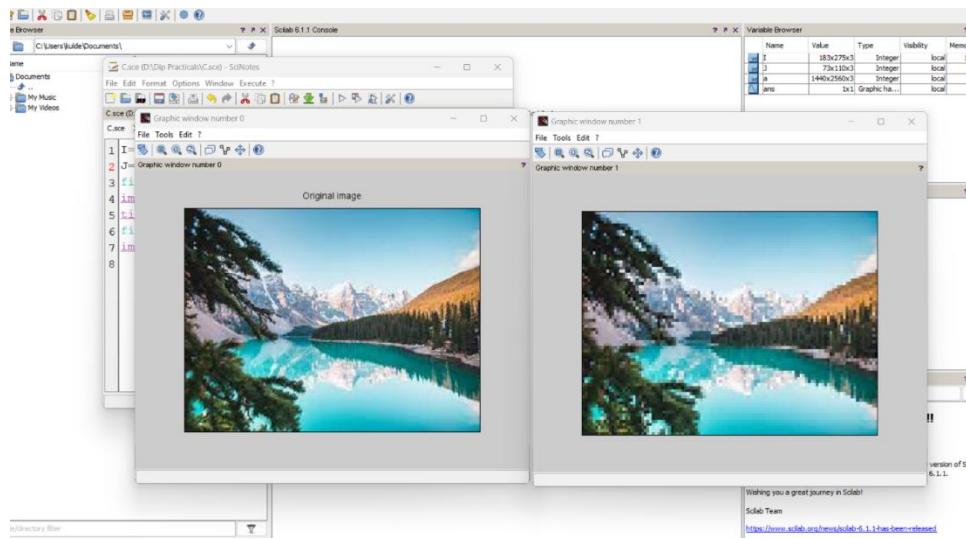
(b)

```
a=imread("D:\ Landscape.jpg ");
imshow(a);
title("original image");
```



(c)

```
I= imread("D:\ Pic.jpg");
J= imresize(I,0.4);
figure
imshow(I);
title("Original image");
figure
imshow(J);
title("Resized image");
```

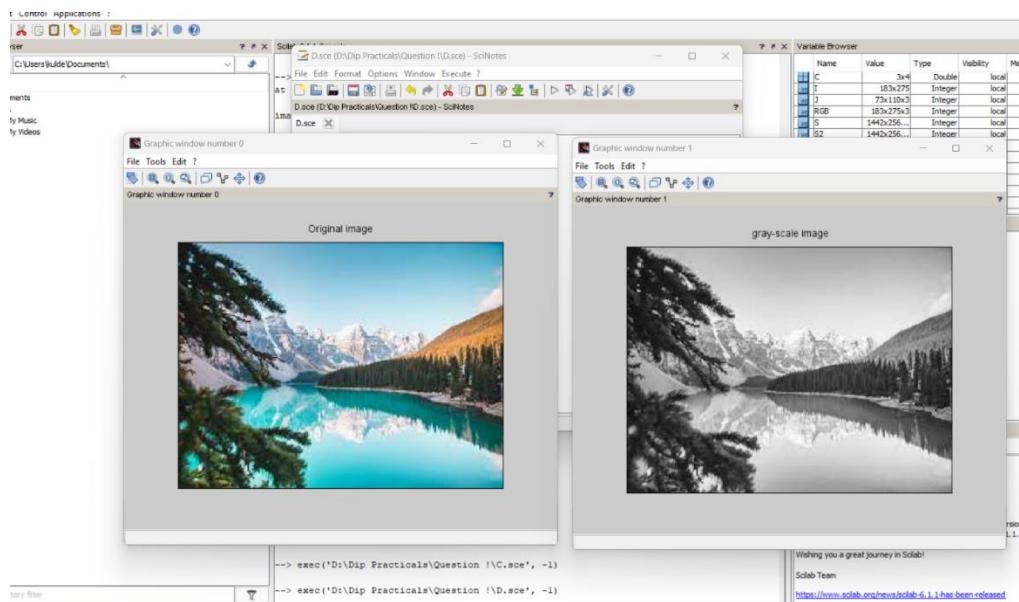


(d)

```

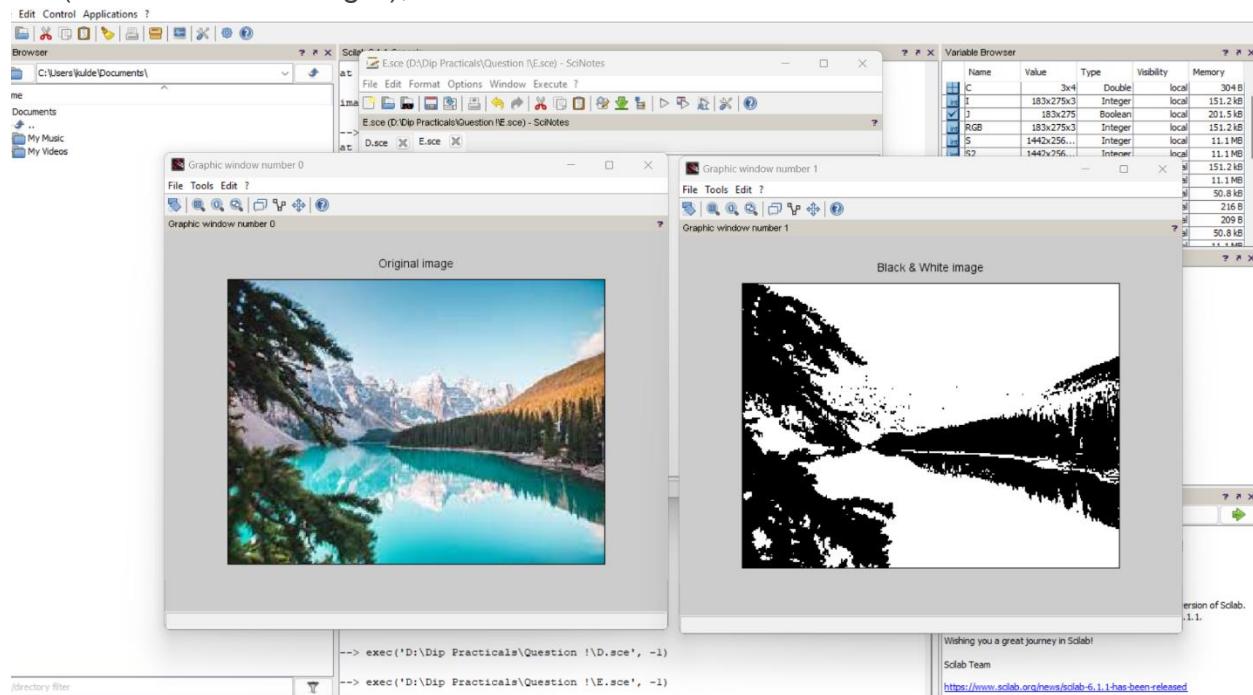
RGB = imread("D:\Pic.jpg");
figure
imshow(RGB);
title("Original image");
I = rgb2gray(RGB);
figure
imshow(I);
title("gray-scale image");

```



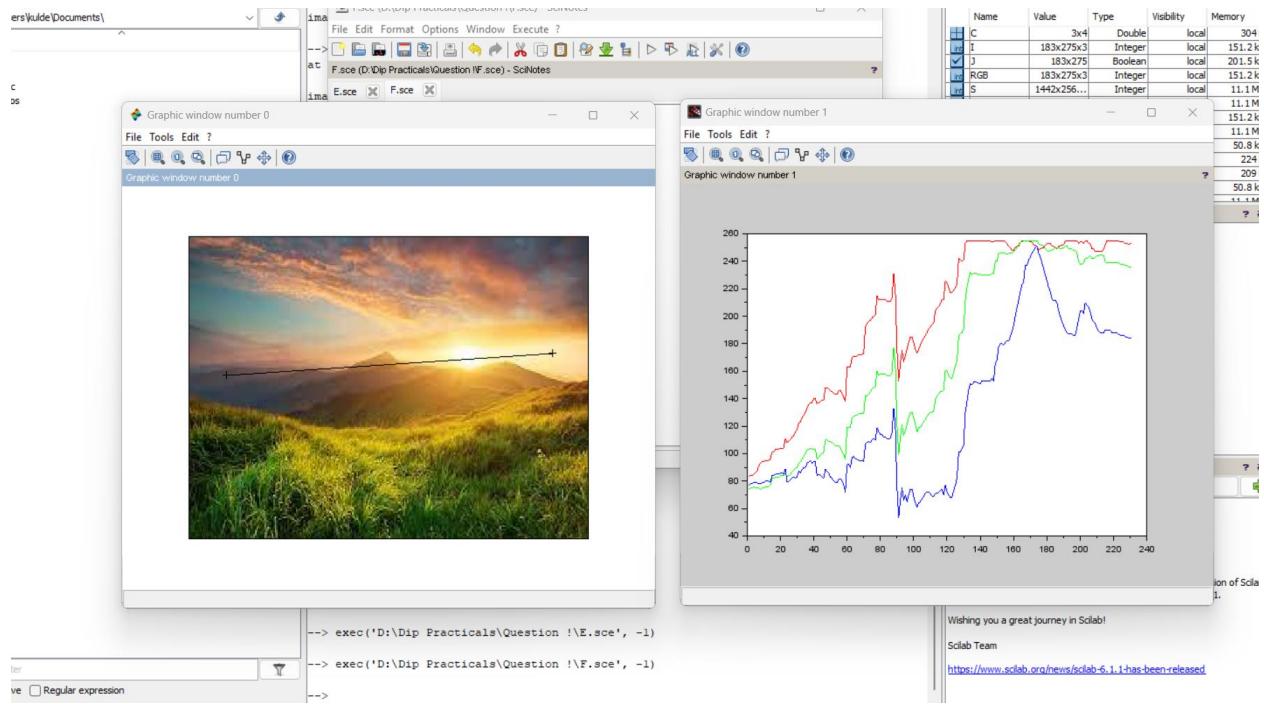
(e)

```
I = imread("D:\ Pic.jpg ");
figure
imshow(I);
title("Original image");
J = im2bw(I,0.5);
figure
imshow(J);
title("Black & White image");
```



(f)

```
i = imread("D:\ Nature.jpg ");
improfile(i);
```



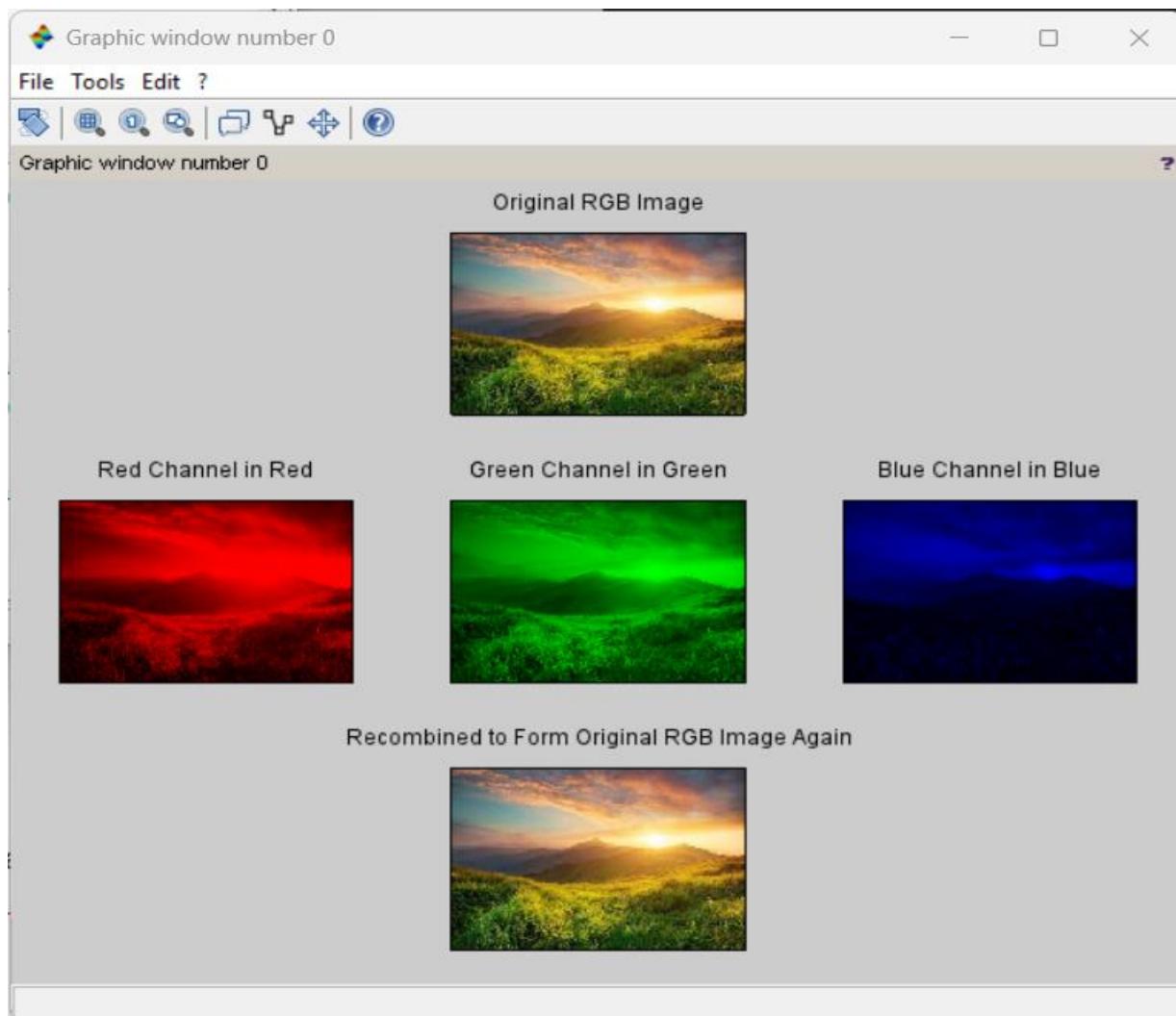
(g),(h)

```
i = imread("D:\Nature.jpg ");
redChannel = i(:,:,1); // Red channel
greenChannel = i(:,:,2); // Green channel
blueChannel = i(:,:,3); // Blue channel
// Create an all black channel.
allBlack = zeros(size(i, 1), size(i, 2), 'uint8');
// Create color versions of the individual color channels.
just_red = cat(3, redChannel, allBlack, allBlack);
just_green = cat(3, allBlack, greenChannel, allBlack);
just_blue = cat(3, allBlack, allBlack, blueChannel);
// Recombine the individual color channels to create the original RGB image again.
recombinedRGBImage = cat(3, redChannel, greenChannel, blueChannel);
// Display them all.
figure
subplot(3, 3, 2);
imshow(i);
fontSize = 2;
title('Original RGB Image', 'FontSize', fontSize)
subplot(3, 3, 4);
imshow(just_red);
```

```

title('Red Channel in Red', 'FontSize', fontSize)
subplot(3, 3, 5);
imshow(just_green)
title('Green Channel in Green', 'FontSize', fontSize)
subplot(3, 3, 6);
imshow(just_blue);
title('Blue Channel in Blue', 'FontSize', fontSize)
subplot(3, 3, 8);
imshow(recombinedRGBImage);
title('Recombined to Form Original RGB Image Again', 'FontSize', fontSize)

```



(i)

```

num = input("Enter a Number ");
% Flow Control

```

```

% Program to find a Number is even or Odd
if mod(num,2) == 0
    disp('The Number is Even');
elseif mod(num,2) ~=0
    disp('The Number is Odd');
else
    disp('Invalid Number');
end
% Loop
% Program to generate Bipolar signal +1 / -1
mat = rand(1,10,'single');
binary=zeros(size(mat));
for count = 1:1:length(mat)
    if mat(count) >= 0
        binary(count) =1;
    else
        binary(count) =-1;
    end
end
disp("Bipolar Signal ")
disp(binary)

```

Output :

```

>> ques1i
Enter a Number  4
The Number is Even
Bipolar Signal
      1      1      1      1      1      1      1      1      1

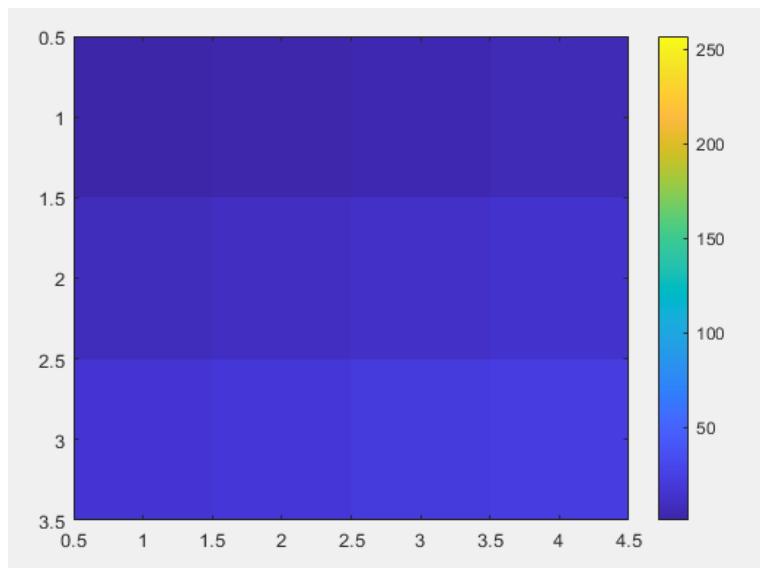
```

(j)

```

C = [0 2 4 6; 8 10 12 14; 16 18 20 22];
image(C)
colorbar

```



Question 2: To write and execute image processing programs using point processing method

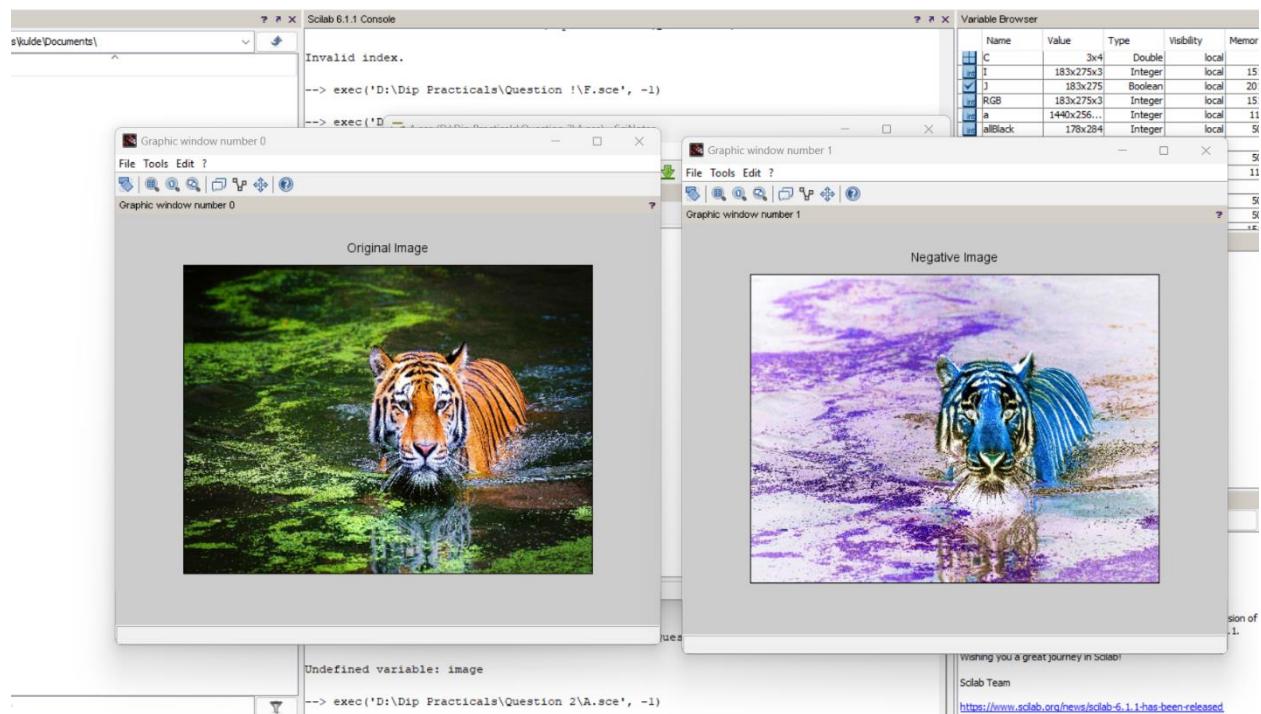
- a. Obtain Negative image
- b. Obtain Flip image
- c. Thresholding
- d. Contrast stretching

Solution:

Code -

(a)

```
a=imread("D:\Tiger.jpg");
figure
imshow(a);
title("Original Image")
d=255-a;
figure
imshow(d);
title("Negative Image")
```

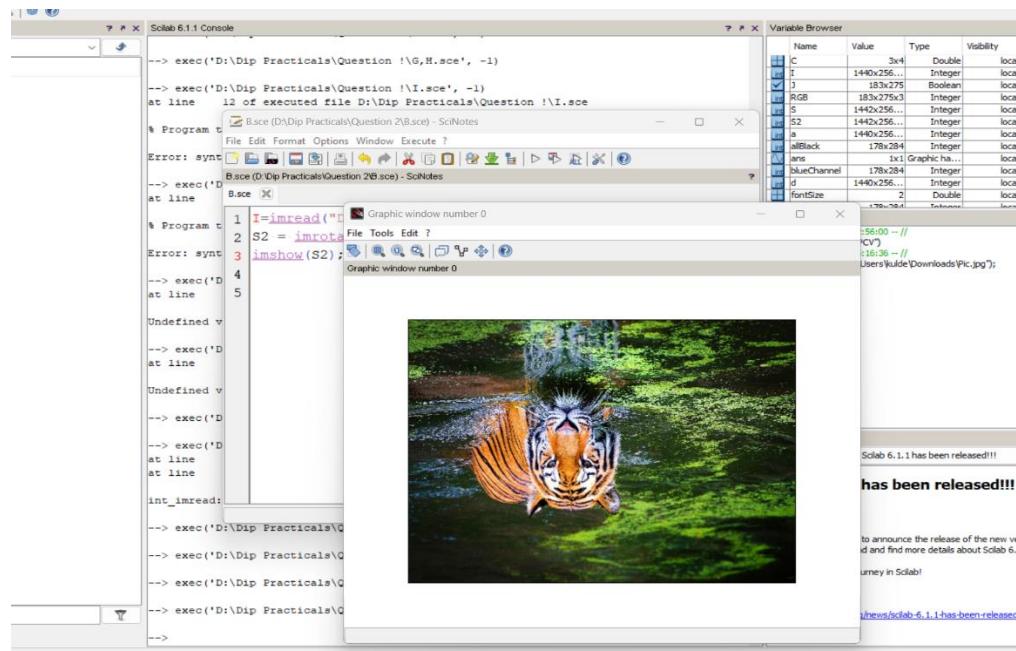


(b)

```

I=imread("D:\Tiger.jpg ");
S2 = imrotate(I,180);
imshow(S2);

```



(c)

```

I = imread("D:\einstein.jpg");
% Convert the image into a binary image.
BW = imbinarize(I);
% Display the original image next to the binary version.
figure
imshowpair(I,BW,'montage')

```

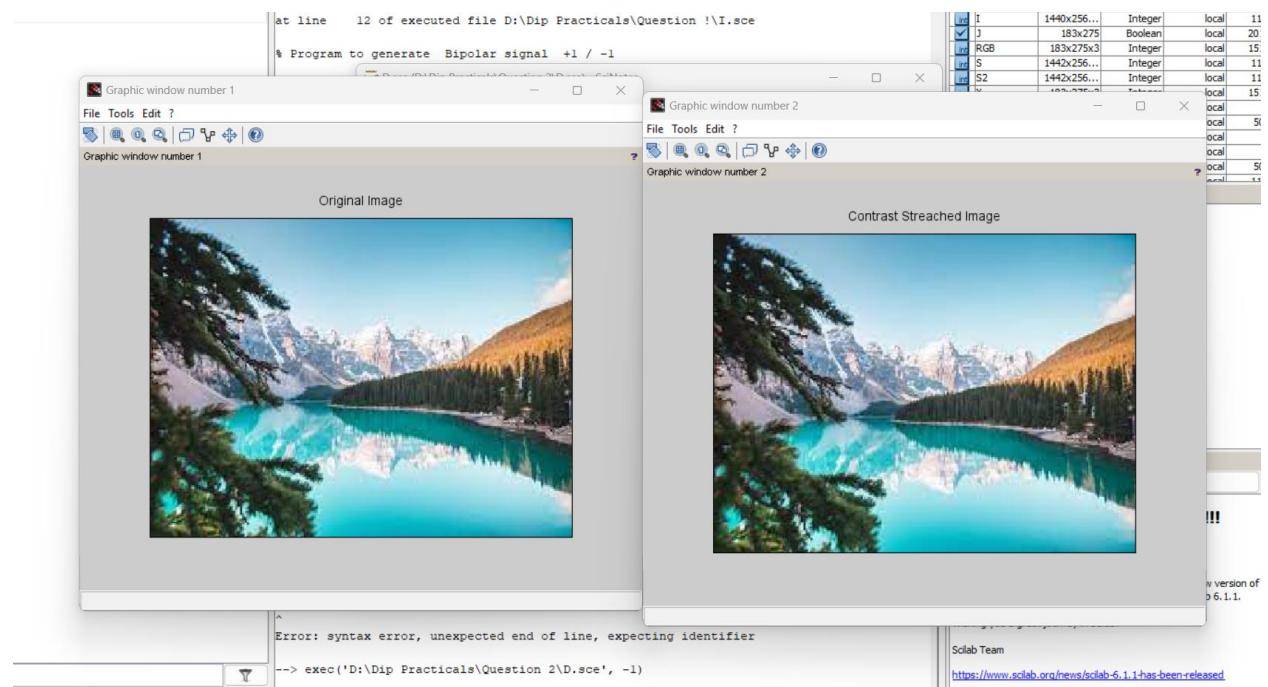


(d)

```

X=imread("D:\Pic.jpg");
//X = imread('image.jpg'); // %reading a grayscale image
figure(1);
imshow(X);
title('Original Image')
a = min(X(:)); // %minimum pixel of image X
b = max(X(:)); // %maximum pixel of image X
X= (X-a).*(255/(b-a)); // %just using the formula above
figure(2);
imshow(X);
title('Contrast Streached Image')

```



Question 3: To write and execute programs for image arithmetic operations

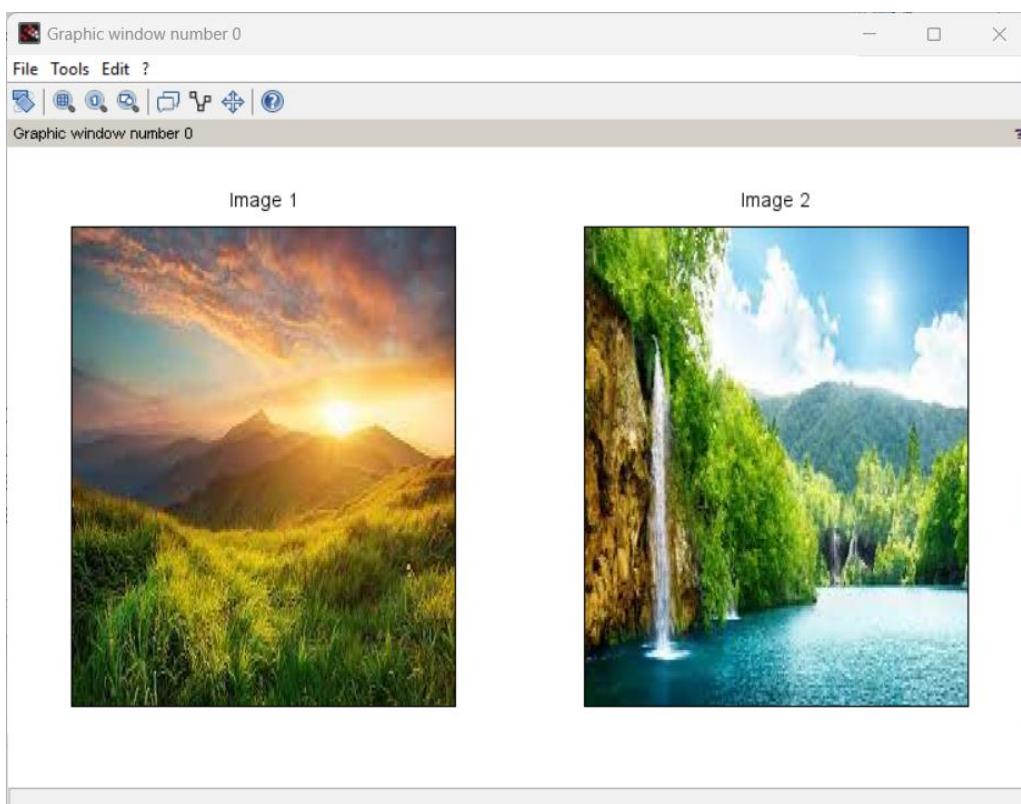
- a. Addition of two images
- b. Subtract one image from other image
- c. Calculate mean value of image

Solution:

Code -

(a)

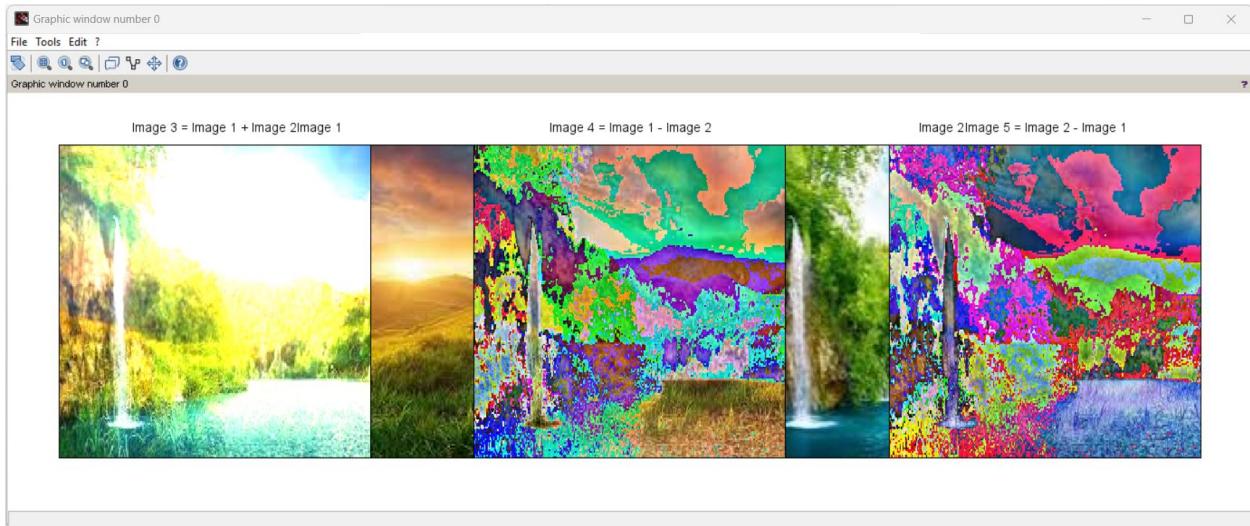
```
%Read two grayscale uint images into the workspace.  
im1=imread("D:\Nature.jpg");  
im2=imread("D:\Nature1.jpg");  
  
subplot(1,2,1), title('Image 1'), imshow(im1);  
subplot(1,2,2), title('Image 2'), imshow(im2);  
  
im3 = imadd(im1, im2);  
imshow(im3);
```



(b)

```
im4 = im1 - im2;  
im5 = im2 - im1;
```

```
subplot(1,3,1), title('Image 3 = Image 1 + Image 2'), imshow(im3);  
subplot(1,3,2), title('Image 4 = Image 1 - Image 2'), imshow(im4);  
subplot(1,3,3), title('Image 5 = Image 2 - Image 1'), imshow(im5);
```



(c)

```
printf('Mean of the image 1 : %.3f\n',mean(im2double(im1)));  
printf('Mean of the image 2 : %.3f',mean(im2double(im2)));
```

```
--> exec('D:\Dip Practicals\Question 3\C.sce', -1)  
Mean of the image 1 : 0.422  
Mean of the image 2 : 0.536  
-->
```

Question 4: To write and execute programs for image logical operations

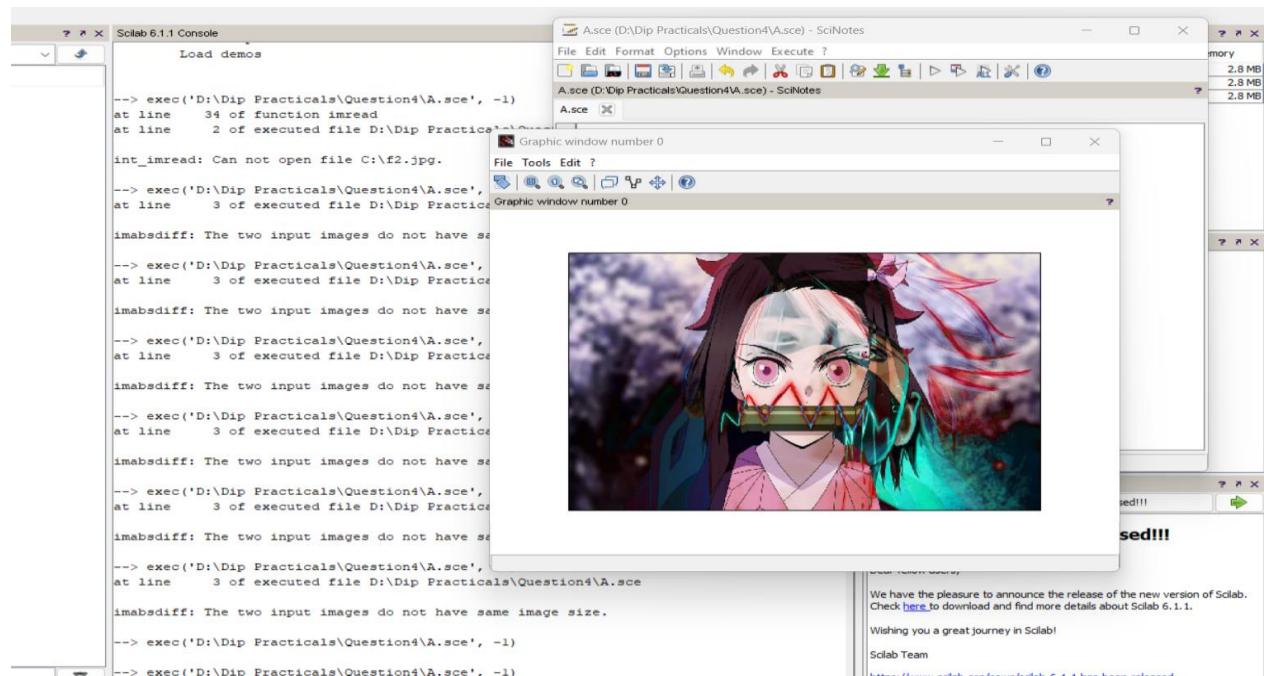
- a. AND operation between two images
- b. OR operation between two images
- c. Calculate intersection of two images
- d. NOT operation (Negative image)

Solution:

Code -

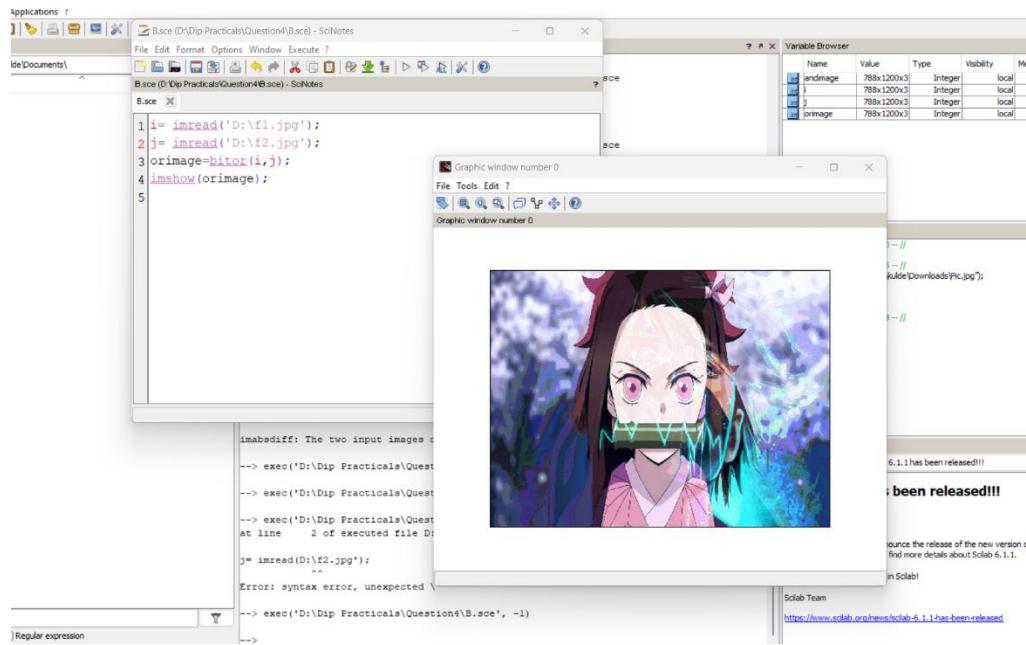
(a)

```
i=imread('D:\f1.jpg');
j=imread('D:\f2.jpg');
andimage=imabsdiff(i,j);
imshow(andimage);
```



(b)

```
i=imread('D:\f1.jpg');
j=imread('D:\f2.jpg');
orimage=bitor(i,j);
imshow(orimage);
```



(c)

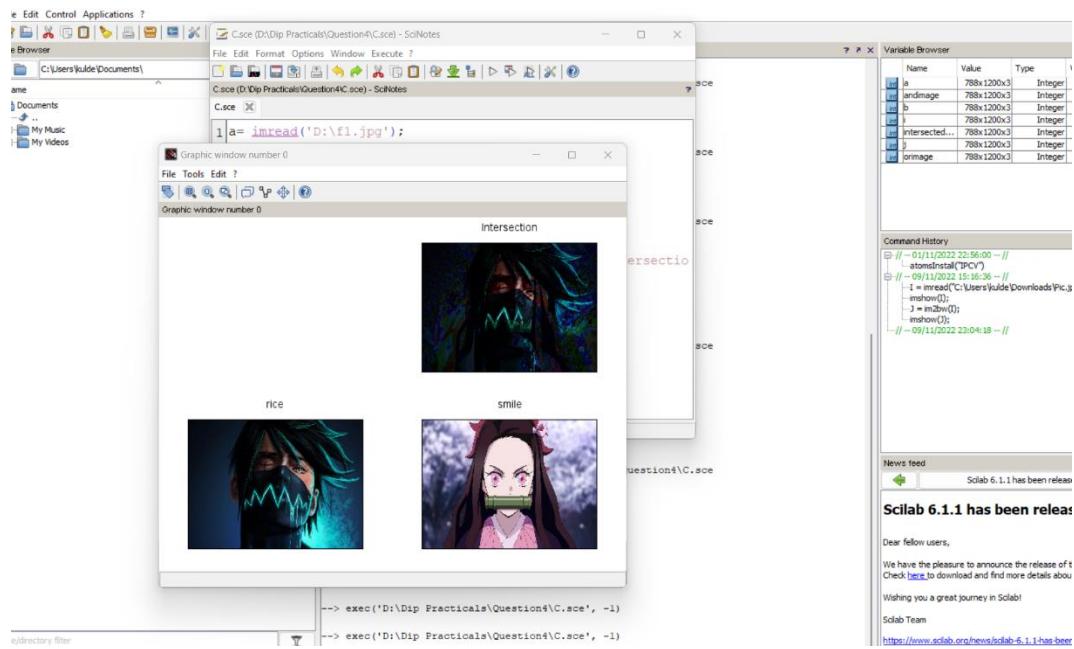
a= imread('D:\f1.jpg');

b=imread(D:\f2.jpg');

```

intersectedImage=bitand(a,b);
subplot(2,2,3), imshow(a), title('rice');
subplot(2,2,4), imshow(b), title('smile');
subplot(2,2,2), imshow(intersectedImage), title('Intersection');

```

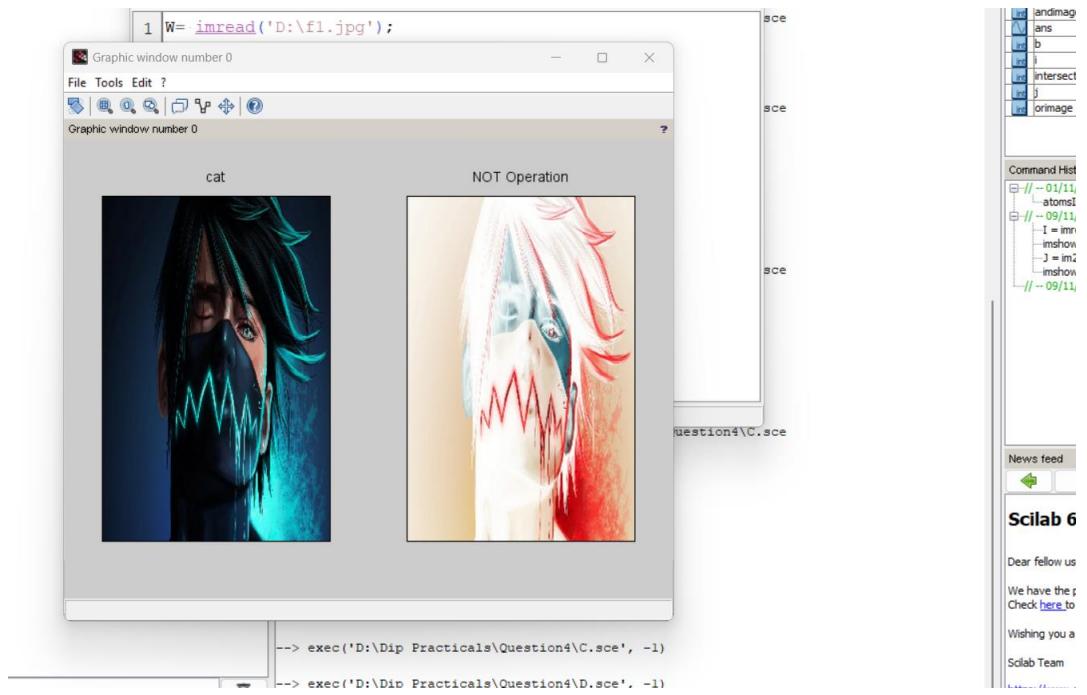


(d)

```

W= imread('D:\f1.jpg');
NotW= bitcmp(W);
figure
subplot(1,2,1)
imshow(W)
title("cat")
subplot(1,2,2)
imshow(NotW)
title("NOT Operation")

```



Question 5: To write a program for histogram calculation and equalization using

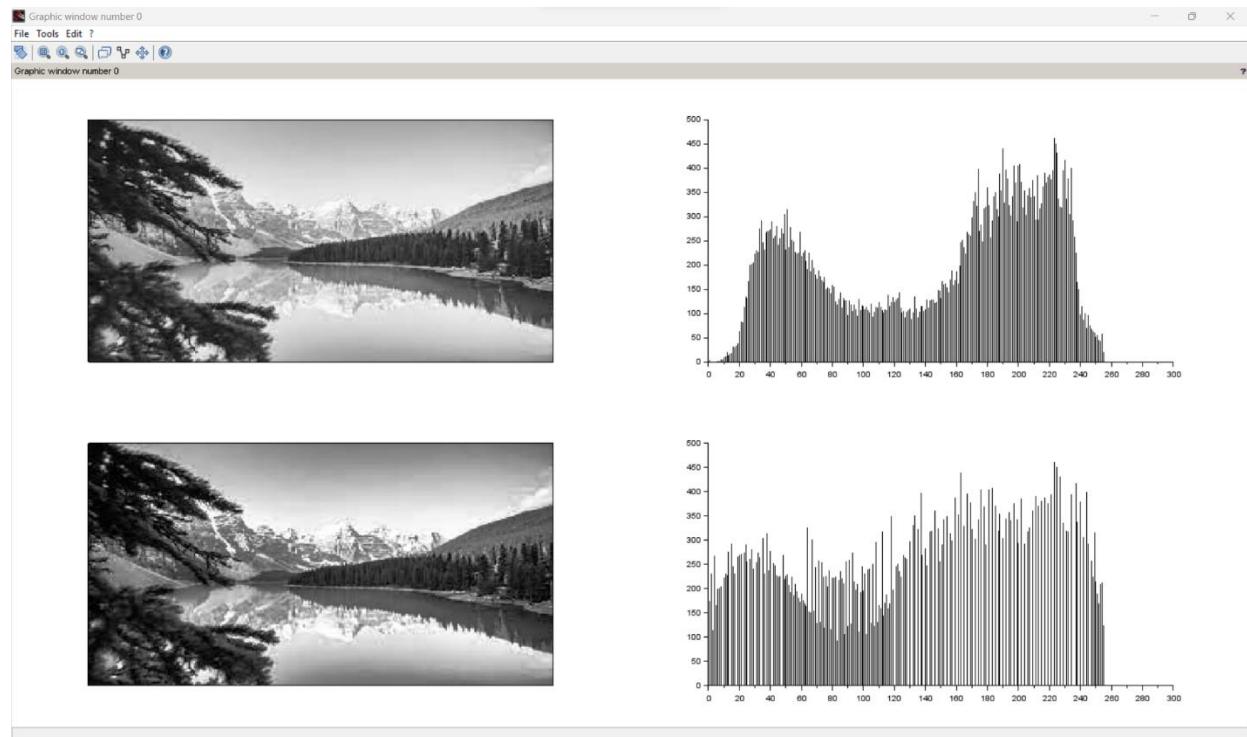
- Standard MATLAB function
- Program without using standard MATLAB functions

Solution:

Code -

(a)

```
I= rgb2gray(imread("D:\Pic.jpg"));
subplot(2,2,1);
imshow(I);
subplot(2,2,2);
imhist(I,[], 1);
J=imhisteq(I);
subplot(2,2,3);
imshow(J);
subplot(2,2,4);
imhist(J,[], 1);
```



(b)

```
GIm=imread('D:\penguin.jpg');
numofpixels=size(GIm,1)*size(GIm,2);
figure
imshow(GIm);
title('Original Image');
HIm=uint8(zeros(size(GIm,1),size(GIm,2)));
freq=zeros(256,1);
probF=zeros(256,1);
probC=zeros(256,1);
cum=zeros(256,1);
output=zeros(256,1);
//freq counts the occurrence of each pixel value.
//The probability of each occurrence is calculated by probf.
for i=1:size(GIm,1)
    for j=1:size(GIm,2)
        value=GIm(i,j);
        freq(value+1)=freq(value+1)+1;
        probF(value+1)=freq(value+1)/numofpixels;
    end
end
sum=0;
no_bins=255;
//The cumulative distribution probability is calculated.
for i=1:size(probF)
    sum=sum+freq(i);
    cum(i)=sum;
    probC(i)=cum(i)/numofpixels;
    output(i)=round(probC(i)*no_bins);
end
for i=1:size(GIm,1)
    for j=1:size(GIm,2)
        HIm(i,j)=output(GIm(i,j)+1);
    end
end
figure,imshow(HIm);
title('Histogram equalization');
```

Original Image



Histogram equalization



Question 6: To write and execute program for geometric transformation of image

- a. Translation
- b. Scaling
- c. Rotation
- d. Shrinking
- e. Zooming

Solution:

Code -

(a)

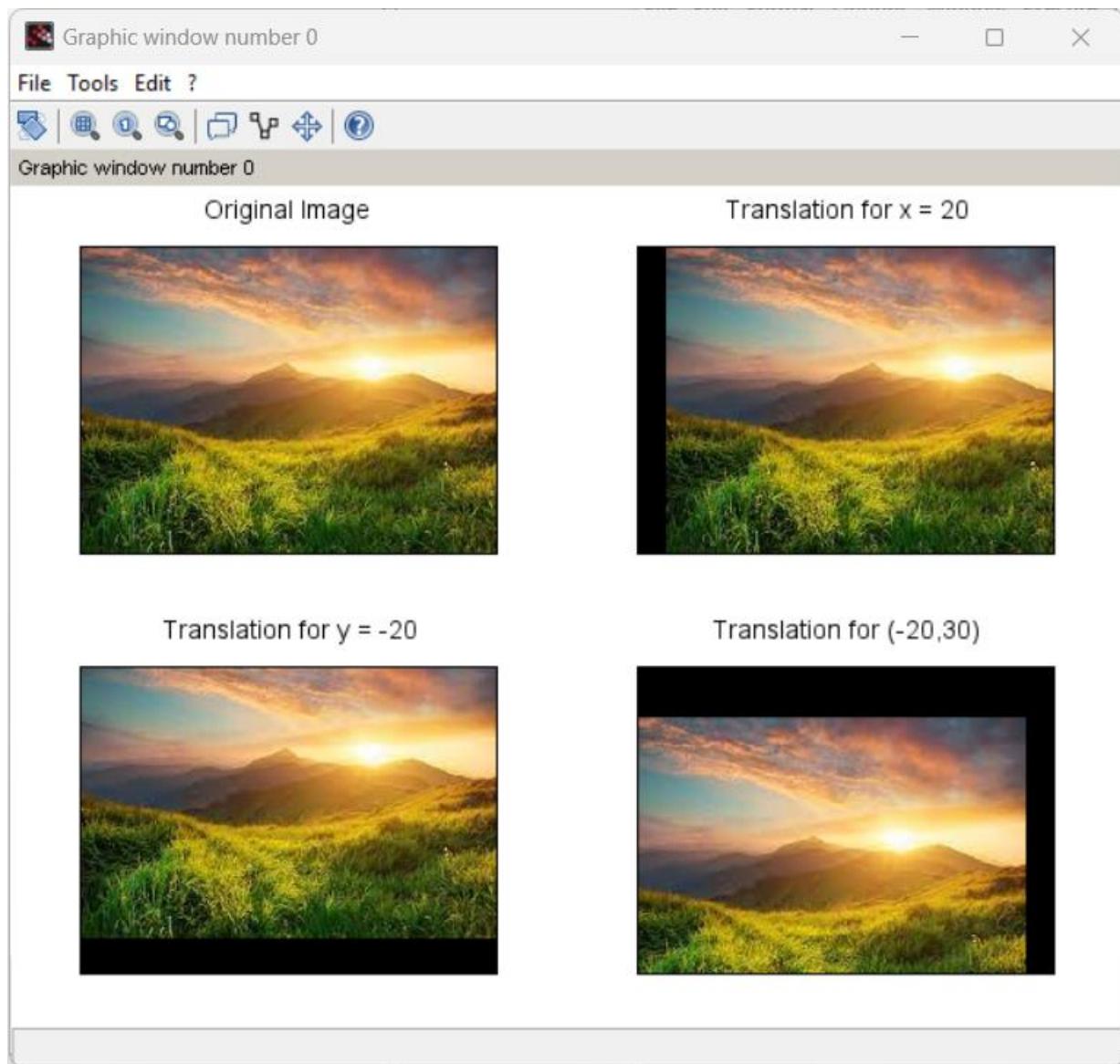
```
S1=imread("D:\Nature.jpg");
```

```
mat = [ 1 0 0;...  
       0 1 0;...  
       20 0 1];  
S2 = imtransform(S1,mat,'affine');
```

```
mat = [ 1 0 0;...  
       0 1 0;...  
       0 -20 1];  
S3 = imtransform(S1,mat,'affine');
```

```
mat = [ 1 0 0;...  
       0 1 0;...  
      -20 30 1];  
S4 = imtransform(S1,mat,'affine');
```

```
subplot(2,2,1), title('Original Image'), imshow(S1);  
subplot(2,2,2), title('Translation for x = 20'), imshow(S2);  
subplot(2,2,3), title('Translation for y = -20'), imshow(S3);  
subplot(2,2,4), title('Translation for (-20,30)'), imshow(S4);
```



(b)

```
s_img=imread("D:\f2.jpg ");
```

```
width = size(s_img, 'c');  
height = size(s_img, 'r');
```

```
w = 2;  
h = 1;
```

```
mat = [ w 0;  
        0 h;  
        0 0];  
  
sc1 = imtransform(s_img, mat, 'affine', width*w, height*h);
```

```
w = 1;  
h = 2;
```

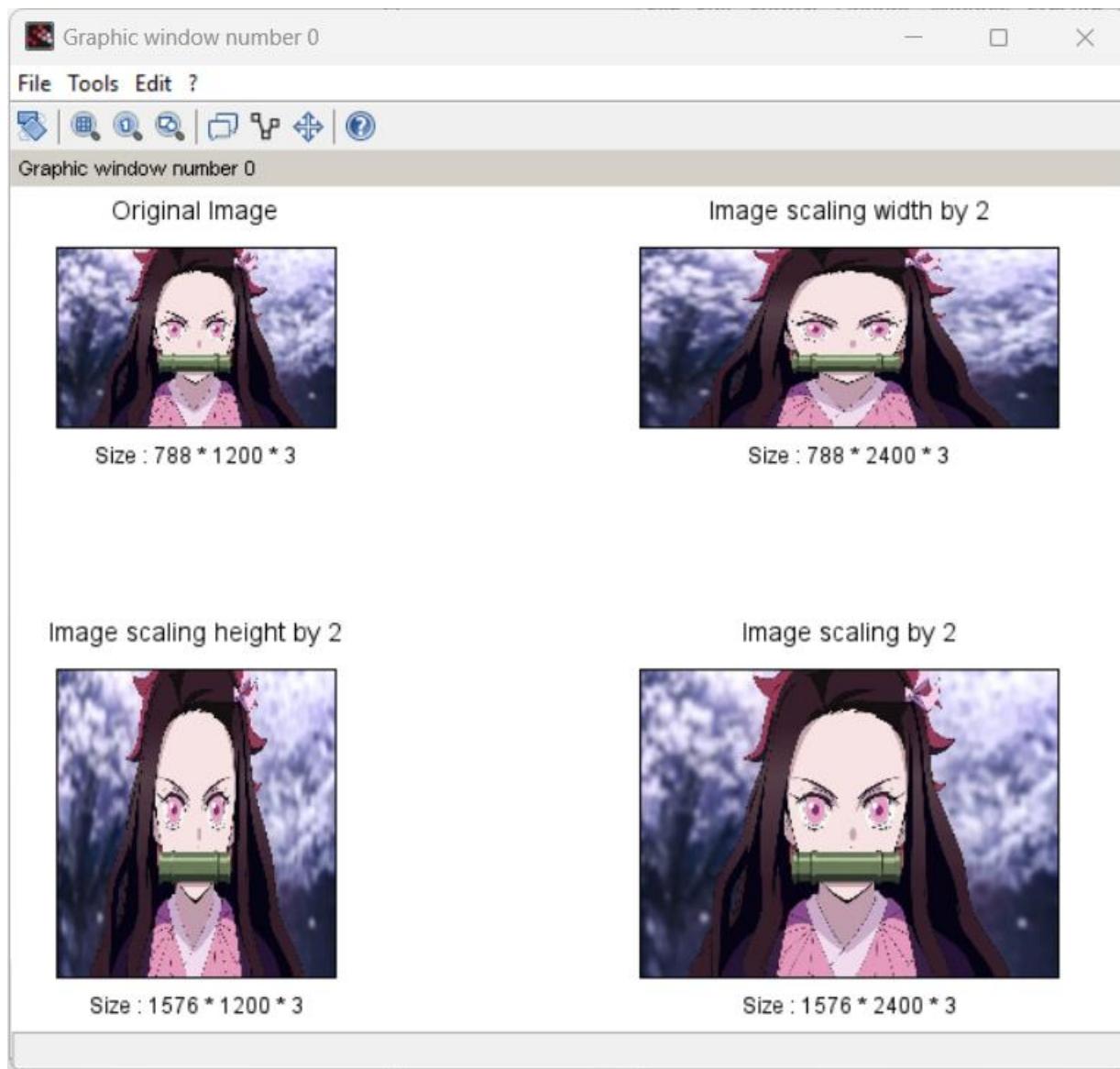
```
mat = [ w 0;  
        0 h;  
        0 0];  
  
sc2 = imtransform(s_img, mat, 'affine', width*w, height*h);
```

```
w = 2;  
h = 2;
```

```
mat = [ w 0;  
        0 h;  
        0 0];  
  
sc3 = imtransform(s_img, mat, 'affine', width*w, height*h);
```

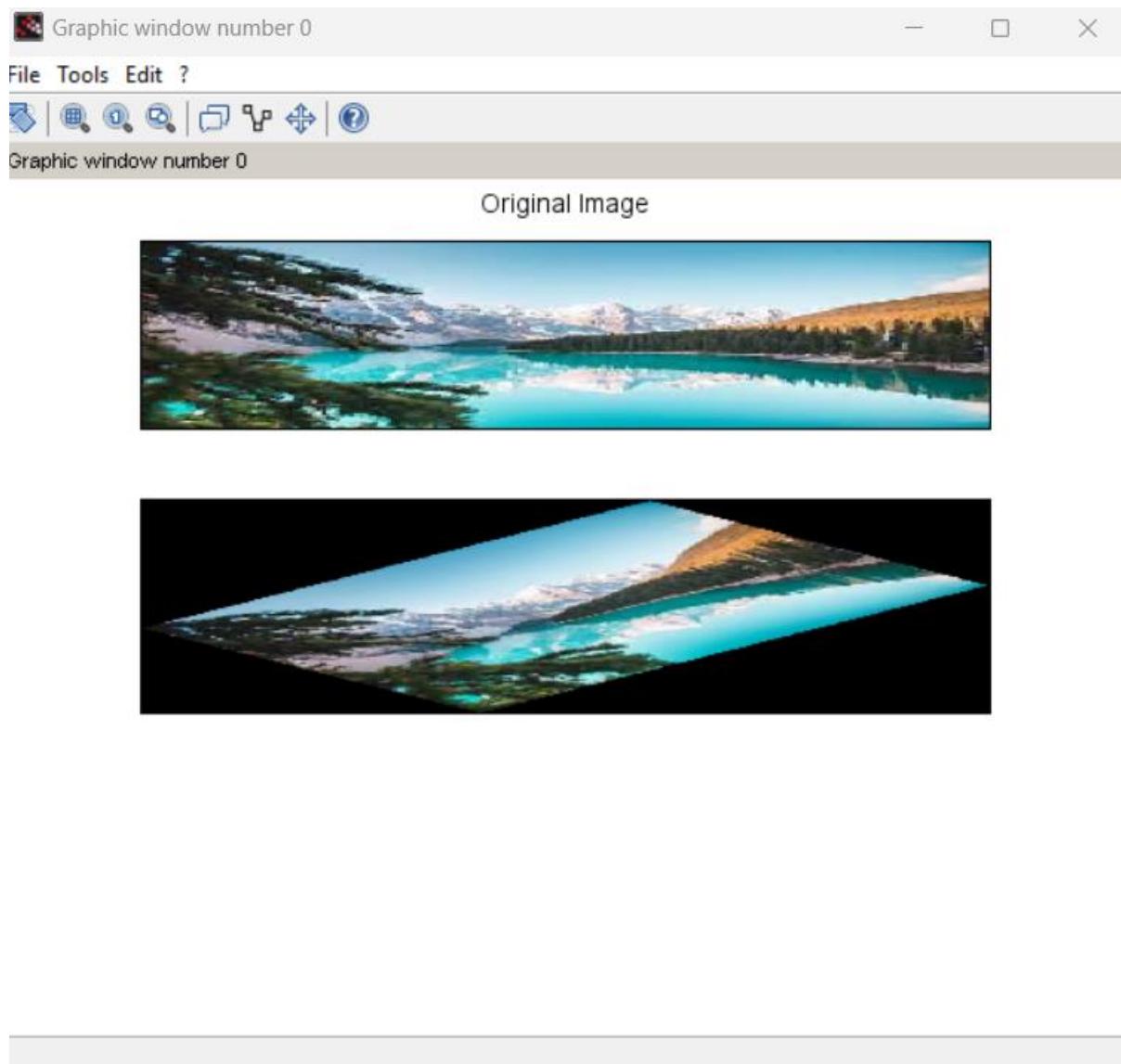
```
function s = str(img)  
    s = 'Size : ' + strcat(string(size(img)), ' * ');\nendfunction;
```

```
subplot(3,3,1), title('Original Image'), xlabel(str(s_img)), imshow(s_img);  
subplot(3,2,2), title('Image scaling width by 2'), xlabel(str(sc1)), imshow(sc1);  
subplot(2,3,4), title('Image scaling height by 2'), xlabel(str(sc2)), imshow(sc2);  
subplot(2,2,4), title('Image scaling by 2'), xlabel(str(sc3)), imshow(sc3);
```



(c)

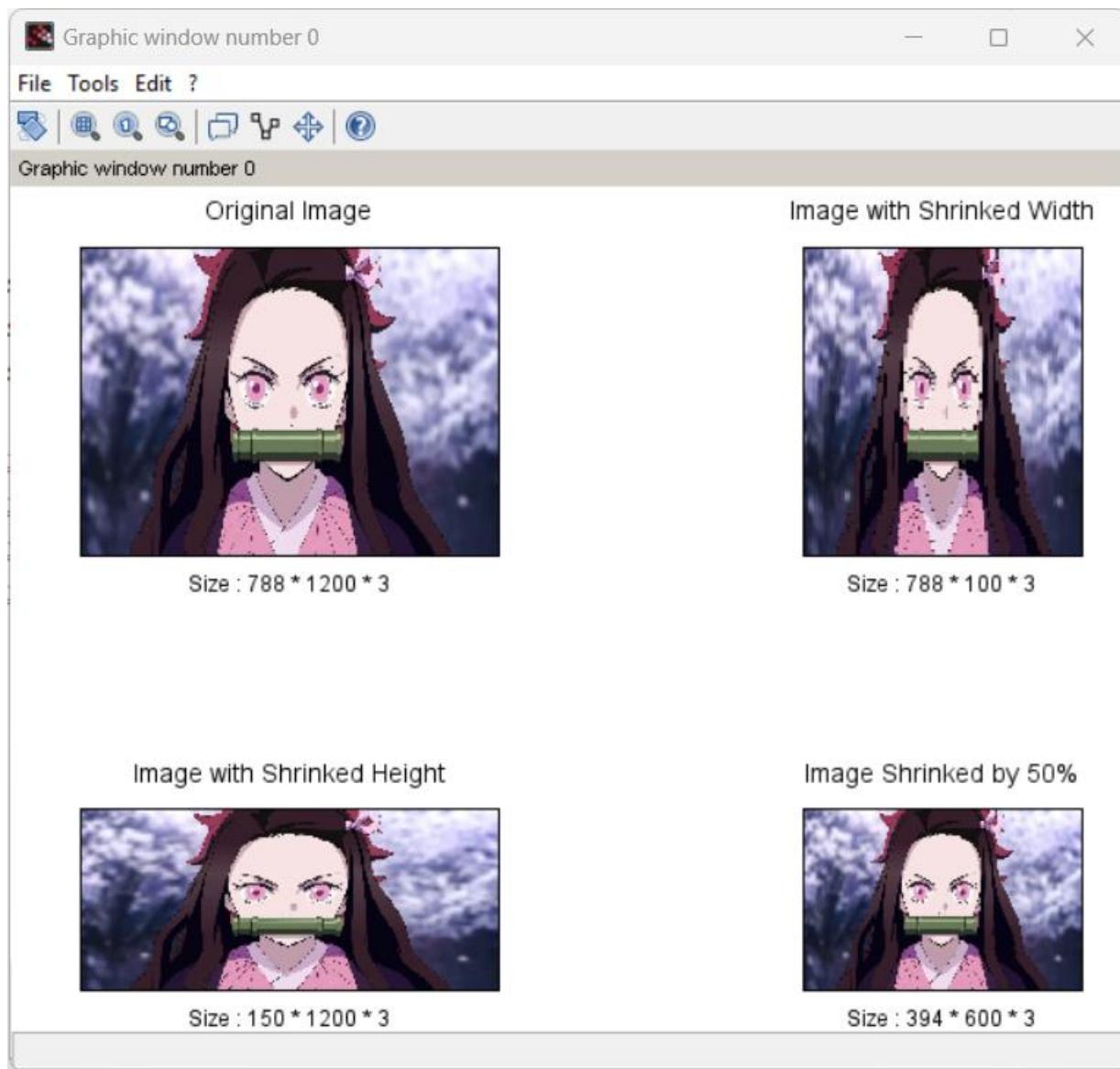
```
img1 =imread("D:\Pic.jpg");
i =imrotate(img1,[45]);
//%Original image
subplot(3,1,1);
imshow(img1);
title("Original Image");
subplot(3,1,2);
imshow(i);
title("Rotate Image");
```



(d)

```
im_50 = imresize(s_img, 0.5);
im_sw = imresize(s_img, [height, 100]);
im_sh = imresize(s_img, [150, width]);

subplot(2,2,1), title('Original Image'), xlabel(str(s_img)), imshow(s_img);
subplot(2,3,3), title('Image with Shrunked Width'), xlabel(str(im_sw)), imshow(im_sw);
subplot(3,2,5), title('Image with Shrunked Height'), xlabel(str(im_sh)), imshow(im_sh);
subplot(3,3,9), title('Image Shrunked by 50%'), xlabel(str(im_50)), imshow(im_50);
```



(e)

```

img6=imread("D:\Cat.jpg");
[m,n,colormap]=size(img6);
if colormap==3
x=img6(:,:,1);
y=img6(:,:,2);
z=img6(:,:,3);
end

c=[];
k =1;

```

```

l=1;
f=input('Enter the replication factor by which the image is to be Zoomed:');
for i=1:m
for t=1:f
for j=1:n
for t=1:f
if colormap==3
c1(k,l)=x(i,j);
c2(k,l)=y(i,j);
c3(k,l)=z(i,j);
else
c(k,l)=b(i,j);
end
l=l+1;
end
end
l=1;
k=k+1;
end
end

if colormap==3
c1(:,:,1)=c1;
c2(:,:,2)=c2;
c3(:,:,3)=c3;
end

//%Original image
figure;
imshow(img6);
title("Original Image(256,256)");

//%Zoomed image
figure;
imshow(c);
title('Zoomed Image(512,512)');

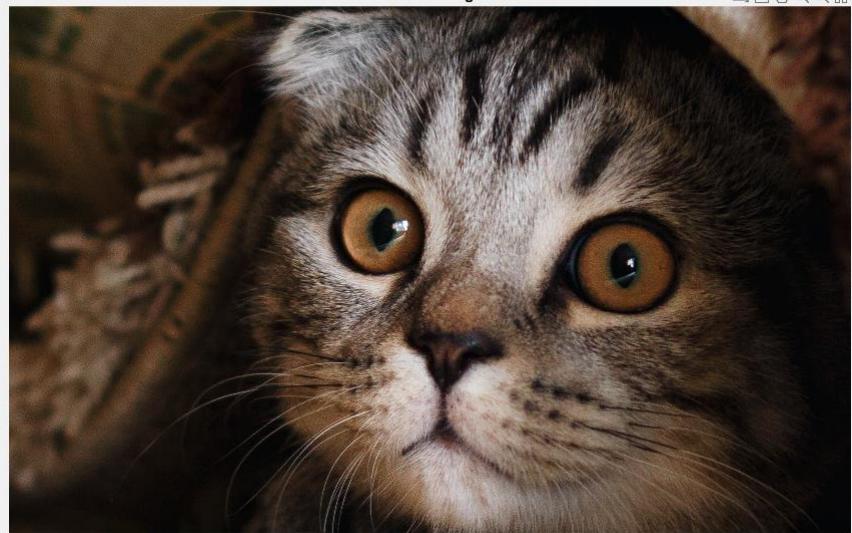
```

Original image



▲ □ ○ × ⊞ ⊛ ⊜ ⊝ ⊞

Zoomed image



▲ □ ○ × ⊞ ⊛ ⊜ ⊝ ⊞

Question 7: To understand various image noise models and to write programs for

- a. image restoration
- b. Remove Salt and Pepper Noise
- c. Minimize Gaussian noise
- d. Median filter

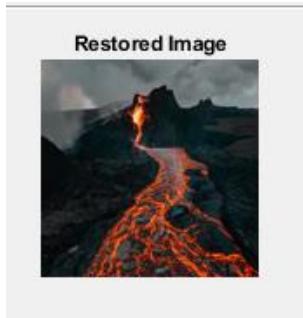
Solution:

Code -

(a)

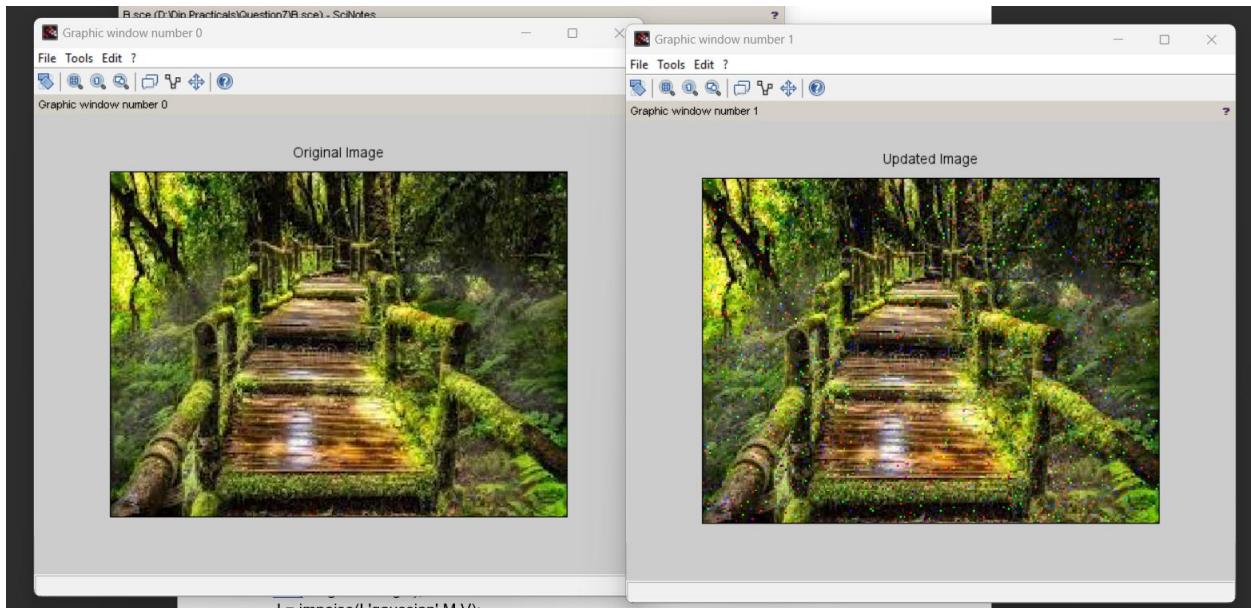
```
im1=imread("D:\lava.jpg");
im= imresize(im1,[125 125]);
S= convn(im1,ones(125,125)/15625,'same');
[Dx,Dy]=gradient(S);
figure,imshow(im1);
title("Original Image");
figure,imshow(im);
title("Restored Image");
```





(b)

```
I=imread("D:\Landscape.jpg");
figure
imshow(I)
title("Original Image")
J = imnoise(I,'salt & pepper',0.02);
figure
imshow(J)
title("Updated Image")
```



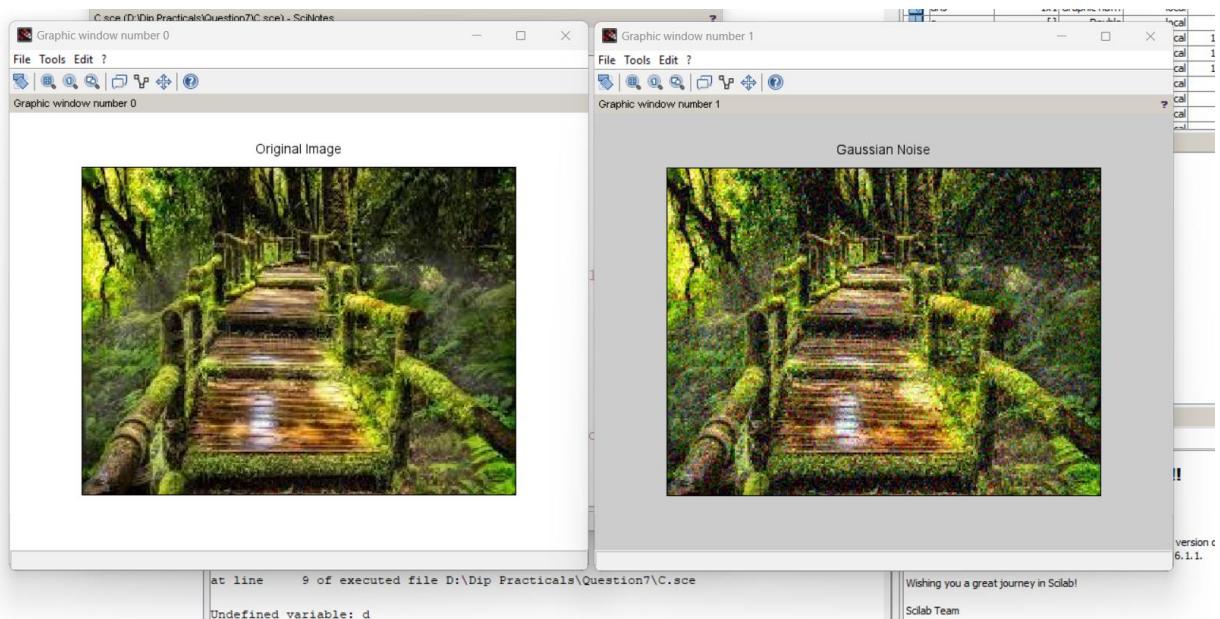
(c)

```
I=imread("D:\Landscape.jpg");
M =0;
V=0.01;
imshow(I);
```

```

title('Original Image');
J = imnoise(I,'gaussian',M,V);
figure,imshow(J);
title('Gaussian Noise');

```

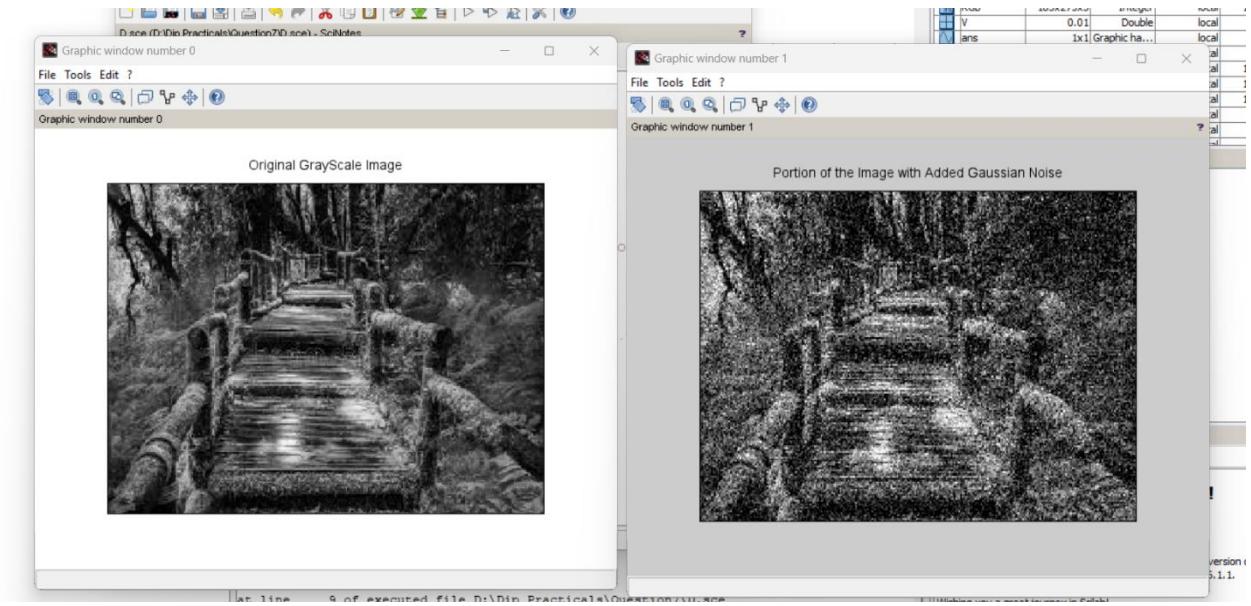


(d)

```

RGB=imread("D:\Landscape.jpg");
I = rgb2gray(RGB);
imshow(I);
title('Original GrayScale Image');
J = imnoise(I,'gaussian',0,0.025);
figure();
imshow(J);
title('Portion of the Image with Added Gaussian Noise');
K = wiener2(J,[5 5]);
figure();
imshow(K);
title('Portion of the Image with Noise Removed by Wiener Filter');

```

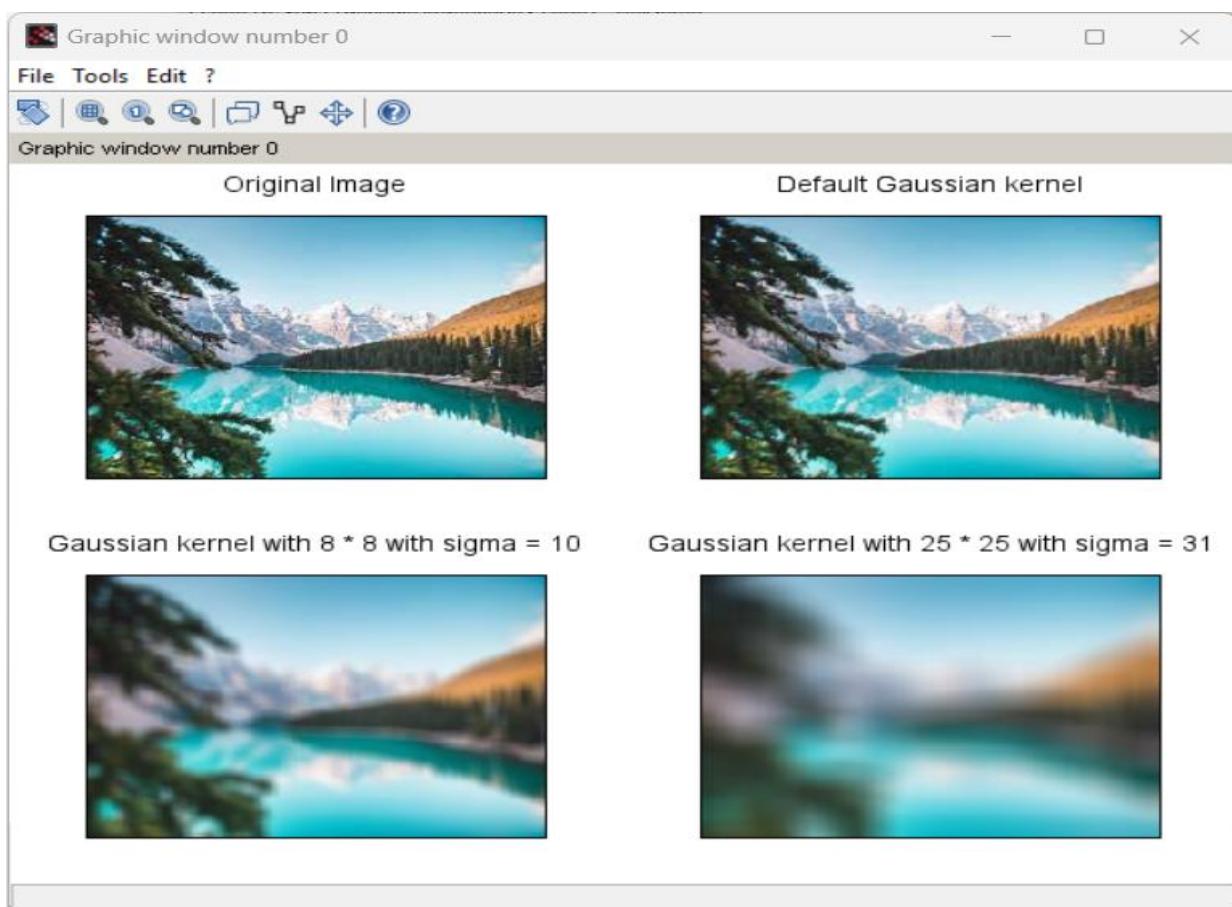


Question 8: Write and execute programs to use spatial low pass and high pass filters.

Solution:

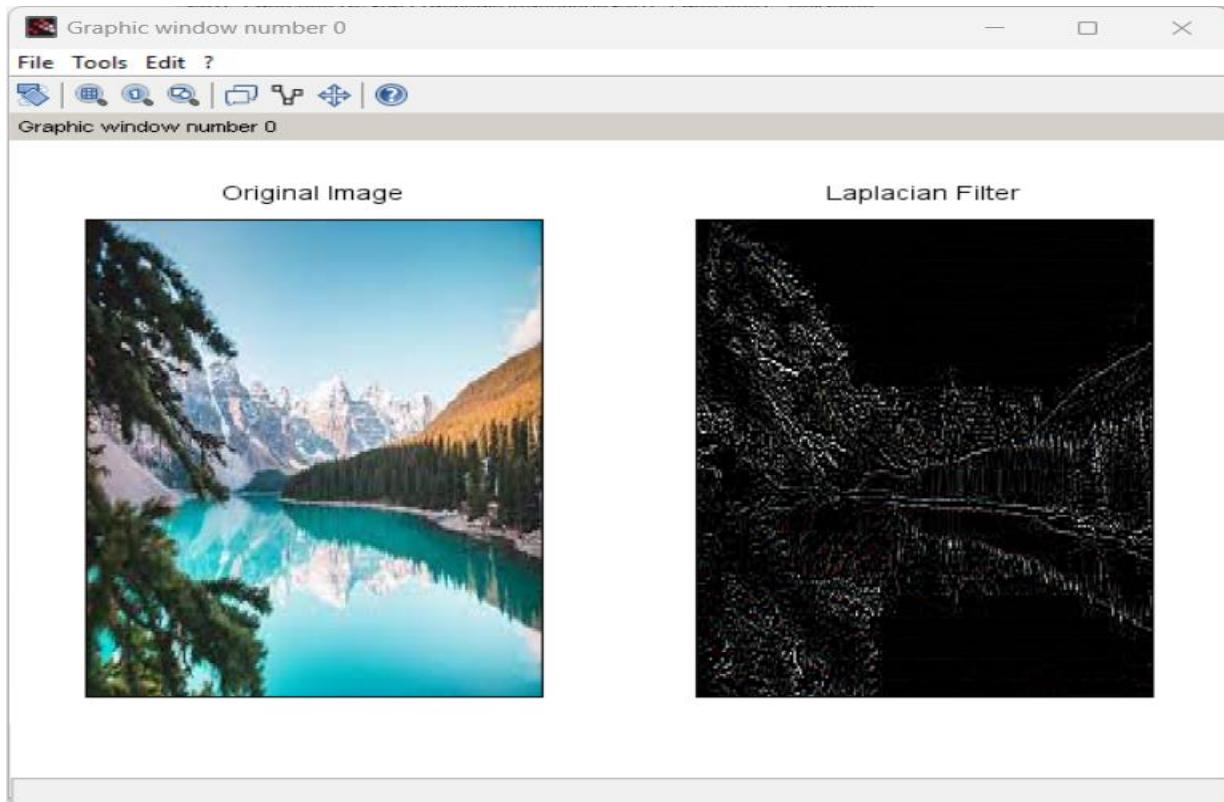
Code -

```
//High pass
i1= imread('D:\Pic.jpg');
g_filter = fspecial('gaussian');
i2 = imfilter(i1, g_filter);
g_filter2 = fspecial('gaussian', [8,8], 10);
i3 = imfilter(i1, g_filter2);
g_filter3 = fspecial('gaussian',[25,25], 31);
i4 = imfilter(i1, g_filter3);
subplot(2,2,1), title('Original Image'), imshow(i1);
subplot(2,2,2), title('Default Gaussian kernel'), imshow(i2);
subplot(2,2,3), title('Gaussian kernel with 8 * 8 with sigma = 10'), imshow(i3);
subplot(2,2,4), title('Gaussian kernel with 25 * 25 with sigma = 31'), imshow(i4);
```



```
//low pass
```

```
i1=imread("D:\Pic.jpg");  
  
l_filter = fspecial('laplacian');  
i2 = imfilter(i1, l_filter);  
  
subplot(1,2,1), title('Original Image'), imshow(i1);  
subplot(1,2,2), title('Laplacian Filter'), imshow(i2);
```



Question 9: Write and execute programs for image frequency domain filtering

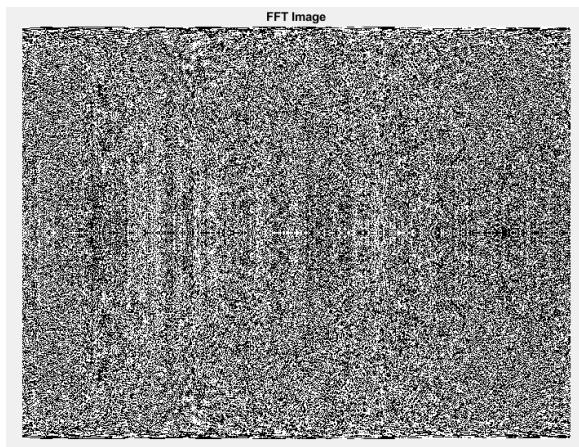
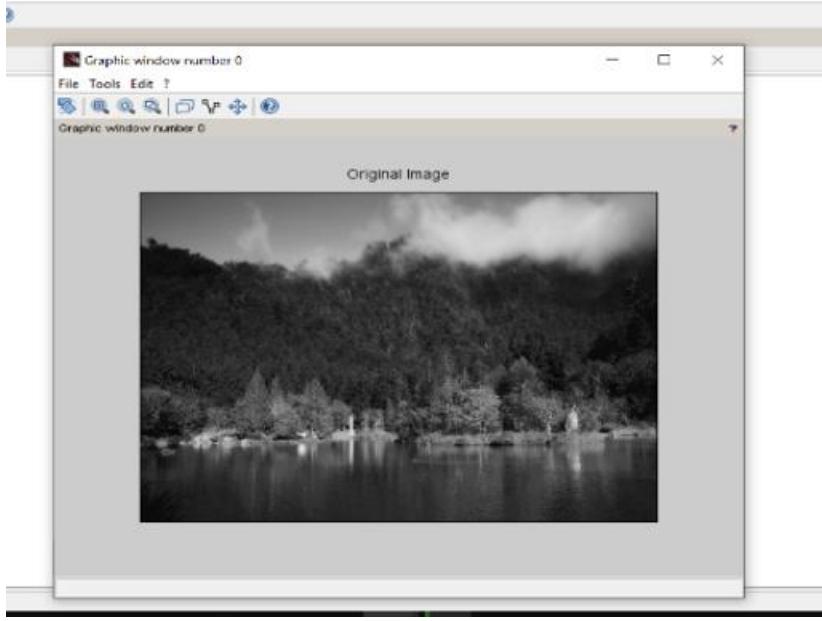
- a. Apply FFT on given image
- b. Perform low pass and high pass filtering in frequency domain
- c. Apply IFFT to reconstruct image

Solution:

Code -

(a)

```
// Read in image.  
I=imread("D:\Nature1.jpg");  
grayImage = rgb2gray(I);  
figure,imshow(grayImage);  
title("Original Image");  
FF = fft2(grayImage);  
figure,imshow(FF);  
title("FFT Image");
```



(b)

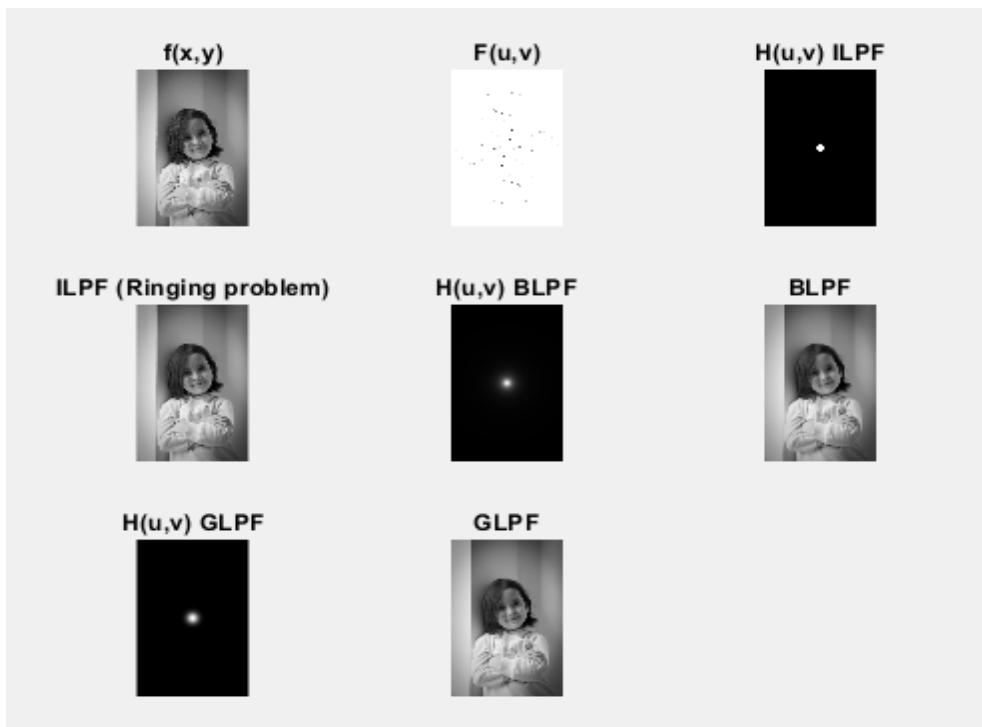
(i) Low Pass Filtering

```
img=imread("D:\\girl.jpg");
gimg=rgb2gray(img);
subplot(331)
imshow(gimg)
title('f(x,y)')
//%% Ideal low pass
P=size(gimg);
M=P(1);N=P(2);
F=fft2(gimg,M,N);
subplot(332)
```

```

//%imshow(uint8(abs(fftshift(F))));
imshow(uint8(abs((F))));
title('F(u,v)')
(u>M/2);
u(idx)=u(idx)-M;
idy=find(v>N/2);
v(idy)=v(idy)-N; u0=100; %remove freq greater than u0
u=0:(M-1);
v=0:(N-1);
idx=find
[V,U]=meshgrid(v,u);
D=sqrt(U.^2+V.^2);
H=double(D<=u0);
//% display
subplot(333)
imshow(abs(fftshift(H)),[]);
title('H(u,v) ILPF')
subplot(334)
G=H.*F;
g=(ifft2(G));

```



(ii) High Pass Filtering

```

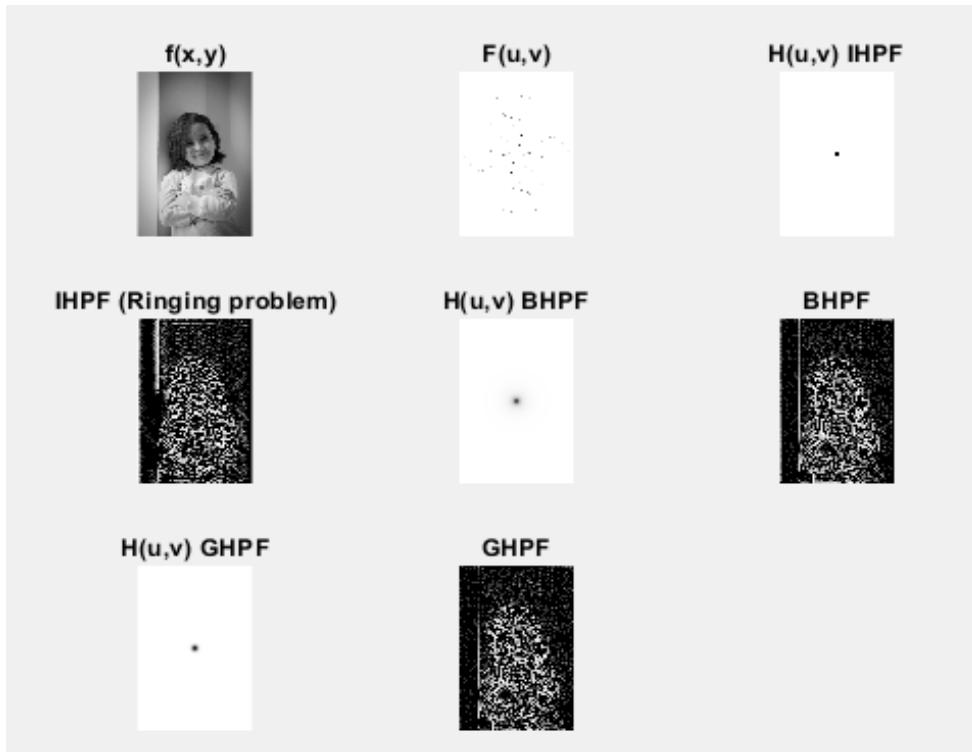
img=imread('girl.jpg');
gimg=rgb2gray(img);
subplot(331)
imshow(gimg)
title('f(x,y)')
%% Ideal high pass
P=size(gimg);
M=P(1);N=P(2);
F=fft2(gimg,M,N);
subplot(332)
%imshow(uint8(abs(fftshift(F))));
imshow(uint8(abs(F)));
title('F(u,v)')
u0=50; %remove freq greater than u0
u=0:(M-1);
v=0:(N-1);
idx=find(u>M/2);
u(idx)=u(idx)-M;
idy=find(v>N/2);
v(idy)=v(idy)-N;
[V,U]=meshgrid(v,u);
D=sqrt(U.^2+V.^2);
H=1-double(D<=u0);
% display
subplot(333)
imshow(abs(fftshift(H)),[]);
title('H(u,v) IHPF')
subplot(334)
G=H.*F;
g=ifft2(G);
imshow(histeq(uint8(g)),[])
title('IHPF (Ringing problem)')
%% Butterworth high pass
n=1; %Butterworth filter of order n
H=1-(1./(1 + (D./u0).^(2*n)));
subplot(335)
imshow(abs(fftshift(H)));
title('H(u,v) BHPF')
subplot(336)
G=H.*F;

```

```

g=(ifft2(G));
imshow(histeq(uint8(g),[]))
title('BHPF')
%% Gaussian high pass filter
H =1- (exp(-(D.^2)./(2*(u0^2)))); 
subplot(337)
imshow(abs(fftshift(H)));
title('H(u,v) GHPF')
subplot(338)
G=H.*F;
g=(ifft2(G));
imshow(histeq(uint8(g),[]))
title('GHPF')

```



(c)

```

//Read in the image you want

I = imread('penguin.jpg');
figure, imshow(I);
title("Original Image")
//Extract and binarize the relevant colour channels

redChannel=imbinarize(I(:,:,1), 0.8);
greenChannel=imbinarize(I(:,:,2), 0.4);
//Filter the image

filtered = medfilt2(redChannel+greenChannel);
//Convert to binary mask

mask=imbinarize(filtered,0.3);
//Apply the mask to the colour channels
Ir = I(:,:,1); Ir(~mask)=0;
Ig = I(:,:,2); Ig(~mask)=0;
Ib = I(:,:,3); Ib(~mask)=0;
//Compile the output image
I_coloured(:,:,1) = Ir;
I_coloured(:,:,2) = Ig;
I_coloured(:,:,3) = Ib;
figure
imshow(I_coloured)
title("Reconstructed Image")
Finding IFFT
IFF = ifft(I_coloured);
figure,imshow(IFF)
title("iFFT image")

```

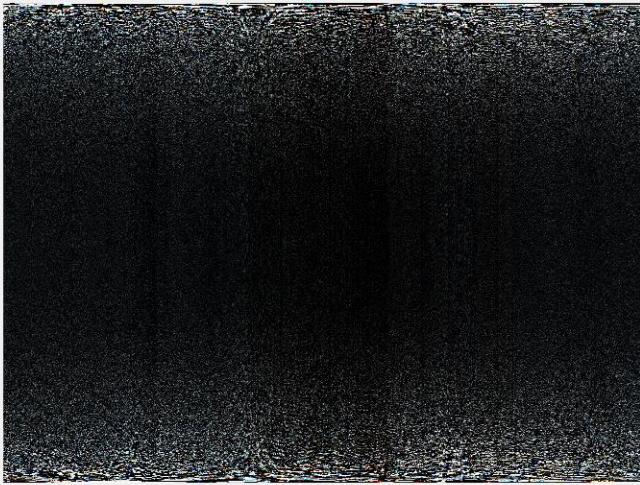
Original Image



Reconstructed Image



IFFT image

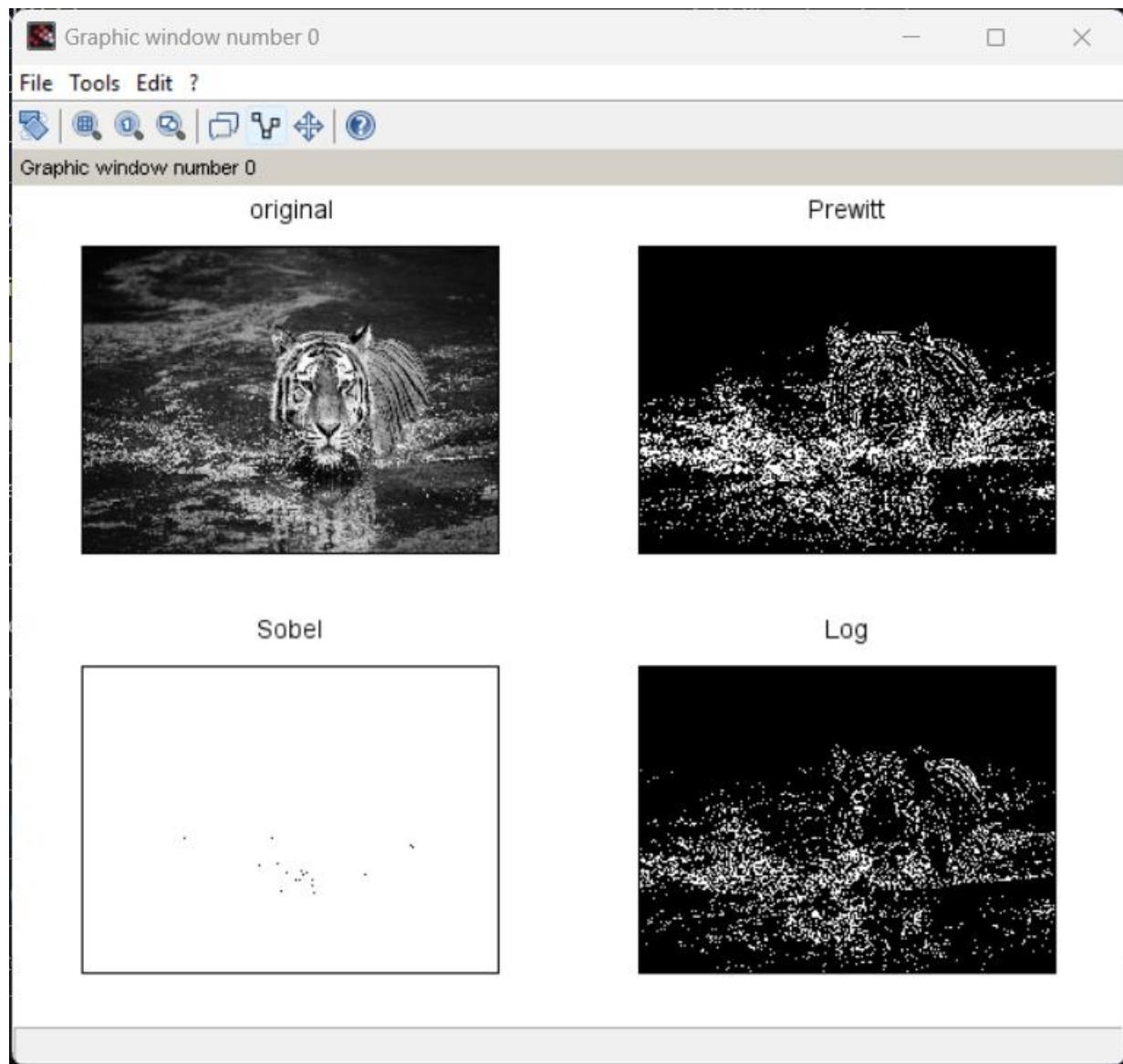


Question 10: Write a program in C and MATLAB/SCILAB for edge detection using different edge detection mask.

Solution:

Code -

```
i=imread('D:\Tiger.jpg');
I=rgb2gray(i);
BW1=edge(I,"prewitt");
BW2=edge(I,"sobel");
BW3=edge(I,"roberts");
subplot(2,2,1);
imshow(I);
title("original");
subplot(2,2,2);
imshow(BW1);
title("Prewitt");
subplot(2,2,3);
imshow(BW2);
title("Sobel");
subplot(2,2,4);
imshow(BW3);
title("Roberts");
```



Question 11: Write and execute a program for image morphological operations erosion and dilation.

Solution:

Code -

```
I1= imread('D:\Landscape.jpg');
I=im2bw(I1,0.5);
se=imcretese('cross',3,3);
subplot(2, 3, 1);
imshow(I);
title("Original image");

dilate = imdilate(I, se);
subplot(2, 3, 2);
imshow(dilate);
title("Dilated image");

erode = imerode(I, se);
subplot(2, 3, 3);
imshow(erode);
title("Eroded image");
```

