Experiment 1

Write a program for reading digital RGB image, access the pixel values and planes. Display each plane independently as a monochrome image.

Objective	Write a program for reading digital RGB image, access the pixel values and planes. Display each plane independently as a monochrome image.
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Reading an image file, accessing pixel value and displaying the image
Description	Reading a image file into MATLAB workspace, access different pixel values and display the Red, Green and Blue planes independently as a monochrome image

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image using imread function
I=imread('peppers.png');
%Defining a figure window
figure(1)
%Displaying the Image
imshow(I);

%Listing first 10*10 pixels in R plane
disp(I(1:10,1:10,1));
%Listing first 10*10 pixels in G plane
disp(I(1:10,1:10,2));
%Listing first 10*10 pixels in B plane
disp(I(1:10,1:10,3));

%Defining a new figure window
figure(2);
%displaying R plane
imshow(I(:,:,1));
%Defining a new figure window
figure(3);
```

```
%displaying G plane
imshow(I(:,:,2));
%Defining a new figure window
figure(4);
%displaying B plane
imshow(I(:,:,3));

% % RGB to Gray Scale Conversion
% I1=rgb2gray(I);
% %Display Gray scale converted image
% imshow (I1)
```

Output:

Original Image

First 10×10 pixels in R plane:

62	63	63	65	66	63	61	63	63	64
63	61	59	64	63	60	61	64	64	63
65	63	63	66	66	62	60	66	64	64
63	67	67	63	64	62	63	68	67	67
63	62	64	65	66	63	65	66	67	66
63	57	59	64	65	64	66	62	62	65
62	61	62	65	64	60	60	61	66	66
65	66	67	65	65	66	65	63	62	62
62	64	63	63	64	64	63	61	62	60
61	62	62	64	66	65	65	63	61	63

First 10×10 pixels in G plane:

29	31	34	30	27	31	31	30	30	31
31	31	32	30	28	31	31	32	31	31
29	30	31	30	31	31	30	31	32	32
29	29	31	31	31	32	33	31	32	33
31	32	33	30	31	32	33	31	31	31
32	32	33	32	31	27	30	31	32	32
30	30	31	32	31	31	32	31	33	32
29	29	31	30	29	32	32	30	32	30
30	30	30	32	33	33	32	30	32	31
29	31	31	31	30	29	30	31	31	32

First 10×10 pixels in B plane:

64	64	64	60	59	62	62	61	59	58
62	64	64	60	59	61	62	62	63	60
60	62	63	61	61	64	63	63	61	64
62	63	63	60	62	62	62	62	64	65
62	63	62	61	61	60	60	61	65	64
61	61	62	59	59	61	65	62	62	65
63	61	60	59	61	63	66	62	63	67
65	60	59	60	62	64	63	61	63	64
66	63	61	61	63	64	64	61	63	60
61	63	63	66	67	63	63	64	61	64

R plane of the Image

G plane of the Image

B plane of the Image

Experiment 2: Computation of image statistics like mean, median, standard deviation, and correlation coefficient.

Objective	Computation of image statistics like mean, median, standard deviation, and correlation coefficient.
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Computation of mean, median, variance, standard deviation, and correlation coefficient
Description	Reading an image into MATLAB into workspace and computation of different statistics of the image.

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image using imread function
I=imread('peppers.png');
%Defining a figure window
figure(1)
%Displaying the Image
imshow(I);

%Converting colour image into grayscale
IG=rgb2gray(I);
%Defdining new figure window
figure(2);
%Displaying the Image
imshow(IG);

% Computation of Mean of all pixels of the image
M=mean2(IG);

%Computation of Median of all pixels of the image
Med=median(IG(:));

%Computation of Standard Deviation of all pixels of the image
SD=std2(IG);

%Computation of Variance of all pixels of the image
variance=(SD)^2;

%Computation of Correlation Coefficient between the gray-scale image with
%itself
r1=corr2(IG,IG);

%Computation of Correlation Coefficient between the gray-scale image with
%R component of original image
r2=corr2(IG,I(:,:,1));
```

Output:

Mean of the Image = 81.65
Median of Image = 60
Standard deviation = 46.57
Correlation coefficient (Grayscale image with itself) = 1
Correlation coefficient (Grayscale image with its Red component) = 0.8458

\

Experiment_3

Write a program to convert a color image into greyscale image and display it. Threshold the grayscale image into a binary image and display it. Assume the Thorold as 128.

Objective	To convert a color image into greyscale image and display it. To threshold the grayscale image into a binary image and display it assuming a threshold of 128.
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	RGB to gray conversion, Image binarization
Description	Reading an image into MATLAB workspace and converting it into grayscale and binary images

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image
I=imread('peppers.png');

%Displaying the Image
imshow(I);

%RGB to Grayscale conversion
I1=rgb2gray(I);
%Displaying the grayscale image
figure;
imshow(I1);
%Finding dimension of the grayscale image
[M, N]= size(I1);

%Finding Binary image with threshold 128
I2=zeros(M,N);
for i=1:M
    for j=1:N
       if I1(i,j)>=128
           I2(i,j)=255;
       else
           I1(i,j)=0;
       end
    end
end

%Display the binary image
figure;
imshow(I2);
```

Output:

Experiment 4:

Write a program to compute negative of an image, display it and demonstrate its advantages.

Write a program to compute logarithm of an image and demonstrate its advantages.

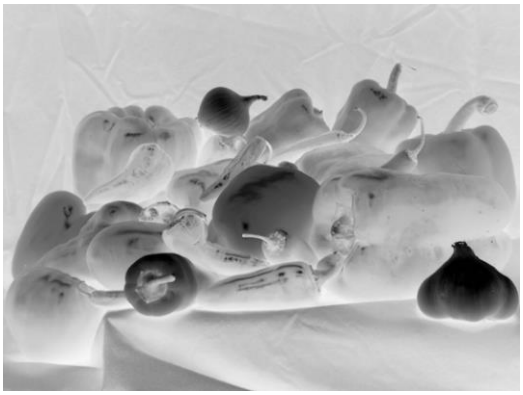
Objective	To Write a program to compute negative of an image, display it and demonstrate its advantages. To write a program to compute logarithm of an image and demonstrate its advantages.
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Negative of image, logarithm of a image
Description	Reading an image into MATLAB workspace and converting it into grayscale and compute its negative and logarithm and display the resultant image

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the Image
I=imread('peppers.png');
%Defining a figure window
figure(1)
%Displaying the Image
imshow(I);

%Converting colour image into grayscale
IG=rgb2gray(I);
%Defdining new figure window
figure(2);
%Displaying the Image
imshow(IG);

%Computing image negative
IN=255-IG;
%Defdining new figure window
figure(3)
%displaying the negative image;
imshow(IN);

%Computing logarithm of a image
% parameter c=1
c=1;
%converting datatype to double for computing logarithm
IG=double(IG);
%Computing the logarithm of image
```

```
IL=c.*log(1+IG);
%As logarithm of integeres are real numbers, hence rounding to nearest
%integer
IL=round(IL);
%Convert the data type as unsigned integer with 8 bit representation
IL=uint8(IL);
%Defdining new figure window
figure(4);
%displaying the logarithm of an image
imshow(IL);
```

Output:



Experiment 5

Write a program to compute and display the histogram of a given image.

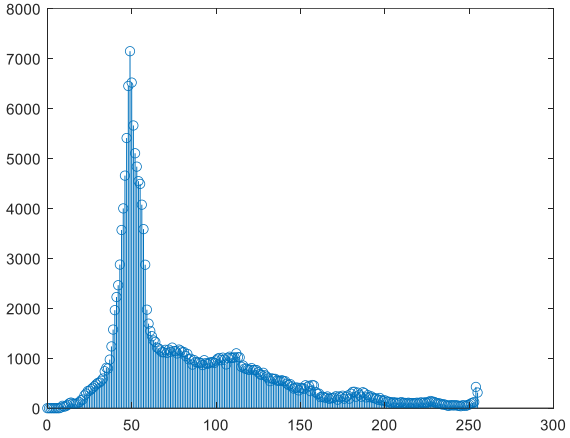
Objective	To write a program to compute and display the histogram of a given image
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Histogram
Description	Reading an image into MATLAB workspace and converting it into grayscale and compute its histogram and display the histogram

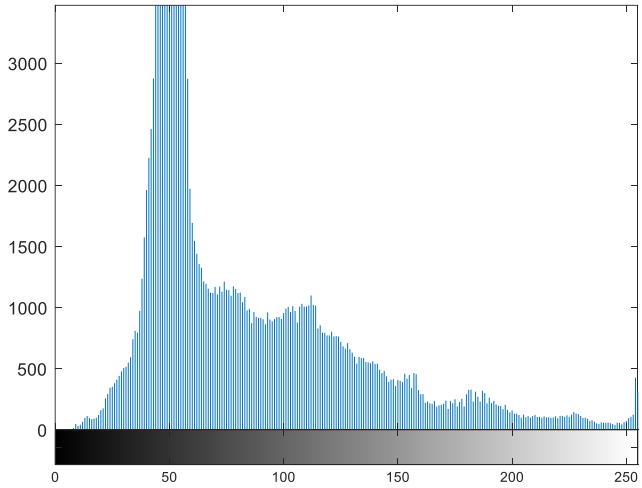
Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image using imread function
I=imread('peppers.png');
%Defining a figure window
figure(1)
%Displaying the Image
imshow(I);

%Converting colour image into grayscale
IG=rgb2gray(I);
%Defining new figure window
figure(2);
%Displaying the Image
imshow(IG);

%Computation of Histogram

%Defining a vector of size 1*256, each element for every possible pixel
%value from 0 to 255
count=zeros(1,256);

%Compute the Image Size
[M N]=size(IG);

%Computing the count of each pixel
for i=1:M
    for j=1:N
        for Gray_Level=0:255
             if IG(i,j)== Gray_Level
                 count(Gray_Level+1)= count(Gray_Level+1)+1;
             end
        end
    end
end
```

```
%Display the contents of count
disp(count)

%Defining new figure window
figure(3);
%Plotting discrete plot of frequency Vs pixel intensity 0 to 255
stem(0:255,count);

%Defining new figure window
figure(4);
%Displaying the Histogram of the Image
imhist(IG);
```

Output:


8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250
300


3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250

Experiment 6

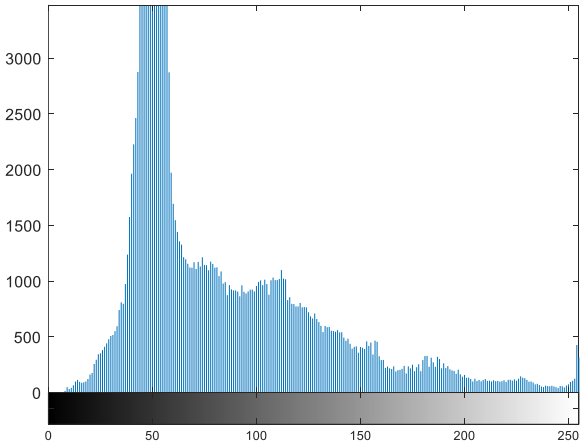
Using histogram equalization technique transform a low contrast image into visually acceptable image.

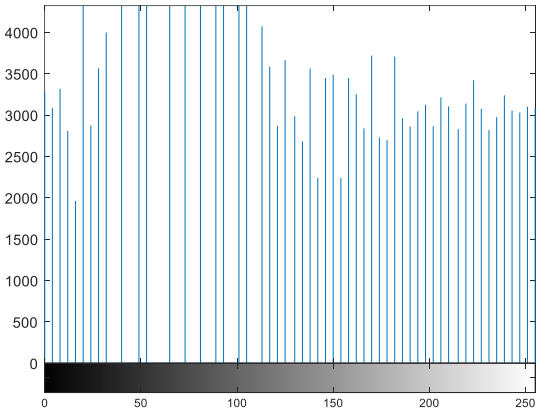
Objective	To write a program to compute and display of histogram of a given image
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Histogram
Description	Reading an image into MATLAB workspace and converting it into grayscale and compute its histogram and display the histogram

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image using imread function
I=imread('peppers.png');
%Defining a figure window
figure(1)
%Displaying the Image
imshow(I);

%Converting colour image into grayscale
IG=rgb2gray(I);
%Defining new figure window
figure(2);
%Displaying the Gray scale Image
imshow(IG);
%Defining new figure window
figure(3);
%Display the histogram of gray scale image
imhist(IG);

%Compute histogram equalized image
IH=histeq(IG);
%Defining new figure window
figure(4);
%Displaying the histogram equalized image
imshow(IH);
%Defining new figure window
figure(5);
%Display the histogram of the histogram equalized image
imhist(IH);
```

Output:




4000
3500
3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250

Experiment 7

Realization of image smoothing using moving average and moving median filter

Objective	Realization of image smoothing using moving average and moving median filter
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Averaging filter, sharpening filter
Description	Reading an image into MATLAB workspace and converting it into grayscale and compute smoothened image and display it.

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image using imread function
I=imread('peppers.png');
%Defining a figure window
figure(1)
%Displaying the Image
imshow(I);

%Converting colour image into grayscale
IG=rgb2gray(I);
%Defining new figure window
figure(2);
%Displaying the Gray scale Image
imshow(IG);

% Define mask for averaging filter
H1=1/9*[1 1 1; 1 1 1; 1 1 1];

%Compute the output image of averaging operation
Y1=filter2(H1,IG);
%Round the values in Y1 with nearest integer and convert the data type into
%uint8 in order to display
Y1=uint8(round(Y1))
%Defining new figure window
figure(3);
%Displaying the Gray scale Image
imshow(Y1);

% Define mask for averaging filter
H2=1/16*[1 2 1; 2 4 2; 1 2 1];

%Compute the output image of averaging operation
Y2=filter2(H2,IG);
%Round the values in Y2 with nearest integer and convert the data type into
```

```
%uint8 in order to display
Y2=uint8(round(Y1))
%Defining new figure window
figure(4);
%Displaying the Gray scale Image
imshow(Y2);
```

Output:

Experiment 8

Realization of image sharpening using moving differences and gradient operators.

Objective	Realization of image sharpening using moving differences and gradient operators.
Software	MATLAB
Dataset	peppers.png
Image Processing Algorithm	Sharpening, gradient operation
Description	Reading an image into MATLAB workspace and converting it into grayscale and compute its sharpened image and display it.

Program:

```
%Clear all the variables
clear all;
%Close all the windows such as figures, GUI etc.
close all;
%Clear the screen
clc;

%Reading the image using imread function
I=imread('peppers.png');
%Defining a figure window
figure(1);
%Displaying the Image
imshow(I);

%Converting colour image into grayscale
IG=rgb2gray(I);
%Defining new figure window
figure(2);
%Displaying the Gray scale Image
imshow(IG);

% Define mask for sharpening filter
H1=[-1 0; 0 1];
%Compute the output image of sharpening operation
Y1=filter2(H1,IG);
%Round the values in Y1 with nearest integer and convert the data type into
%uint8 in order to display
Y1=uint8(round(Y1));
%Defining new figure ;window
figure(3);
%Displaying the Gray scale Image
imshow(Y1);

% Define mask for averaging filter
H2=[0 -1; 1 0];

%Compute the output image of sharpening operation
Y2=filter2(H2,IG);
%Round the values in Y2 with nearest integer and convert the data type into
%uint8 in order to display
Y2=uint8(round(Y2));
```

```
%Defining new figure window
figure(4);
%Displaying the Grayscale Image
imshow(Y2);

% Define mask for sharpening filter
H3=[-1 -2 -1; 0 0 0; 1 2 1];

%Compute the output image of sharpening operation
Y3=filter2(H3,IG);
%Round the values in Y2 with nearest integer and convert the data type into
%uint8 in order to display
Y3=uint8(round(Y3));
%Defining new figure window
figure(5);
%Displaying the Grayscale Image
imshow(Y3);

H4=[-1 0 1;-2 0 2; -1 0 1];
%Compute the output image of sharpening operation
Y4=filter2(H4,IG);
%Round the values in Y2 with nearest integer and convert the data type into
%uint8 in order to display
Y4=uint8(round(Y4));
%Defining new figure window
figure(6);
%Displaying the Grayscale Image
imshow(Y4);
```

Output: