Aim: To demonstrate working of Basic Image Processing Functions.

Code:

```
x=imread('images.png');
imshow(x)
whos x
figure
imhist(x)
imwrite(x,'pout2.png')
imfinfo('pout2.png')
size(x)
imcrop(x,[100,100,100,100])
im2bw(x)
x1=rgb2hsv(x)
imshow(x1)
gray=rgb2gray(x1)
imshow(x1)
imcomplement(x)
```

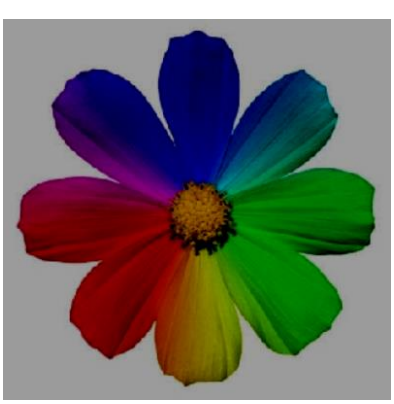
Output:

Aim: Write a MATLAB Script to demonstrate working of Image Arithmetic.

Code:

```
x = imread('flower.jpg');
imshow(x);

i = imcomplement(x);
figure
imshow(i);

y= imadd(x,200);
figure
imshow(y);

z = imsubtract(x,100);
figure
imshow(z);

A = imdivide(x,10);
figure
imshow(A);

B = immultiply(x,5);
figure
imshow(B);
```

Output:



Aim: Write a MATLAB Script to demonstrate working of Affine Transformation.

Code:

```
I = imread("photo.jpg");
imshow(I)
tform1 = affine2d([1 0 0; 1 1 0; 0 0 1]);
X = imwarp(I, tform1);
figure
imshow(X);
```

Output:


Aim: Write a MATLAB Scripts for the following Point Processing Operation 2

1 Image Negative

2 Image Thresholding

3 Image Brightness & Contrast Modification

4 Log Transformation

5 Power Law Transformation

6 Contrast Stretching

7 Intensity Slicing and Bit Plane Slicing.

Code:

Image Negative

```
x = imread("flower.jpg");
imshow(x);
negative_image = 255 - x;
figure
imshow(negative_image);
```

Image Thresholding

```
I = imread("cameraman.tif");
imshow(I)
level = multithresh(I);
seg_I = imquantize(I,level);
```

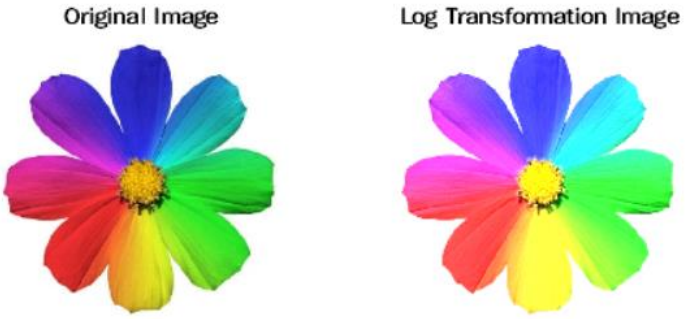
figure imshow(seg_I,[])

Image Brightness & Contrast Modification

```
I = imread("bike.png");
A = I*2;
B = I*0.02;
subplot(1,3,1), imshow(I), title('Original Image');
subplot (1,3,2), imshow(A), title('Increased In Contrast');
subplot (1,3,3), imshow(B), title('Decreased In Contrast');
```

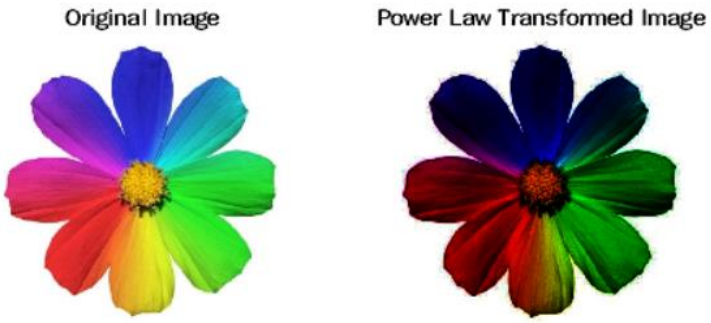
Log Transformation

```
a1 = imread("tree.jpg");%Read the image
a= double (a1)/255; %Normalized Image
c=input('Enter the constant value c=');
f=c*log(1+(a)); %Log Transfor
subplot (1,2,1),imshow(a1),title('Original Image');
subplot (1,2,2),imshow ((f)),title('Log Transformation Image');
```



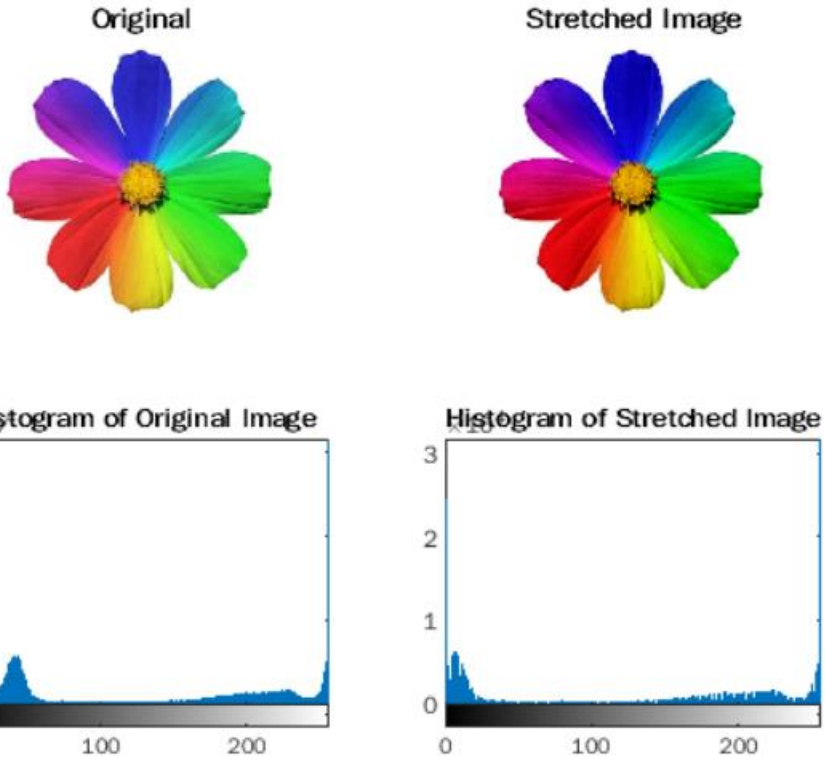
Power Law Transformation

```
itemp = imread("tree.jpg"); %read the image
r=double(itemp)/255; %normalized the image
c=1; %constant
gamma=6; %To make image dark take value of gamma > 1, to make image bright take value of gamma < 1
s= c * (r).^gamma; %formula to implement power law transformation
subplot (1,2,1), imshow(itemp), title('Original Image');
subplot (1,2,2), imshow(s), title('Power Law Transformed Image');
```



Contrast Stretching

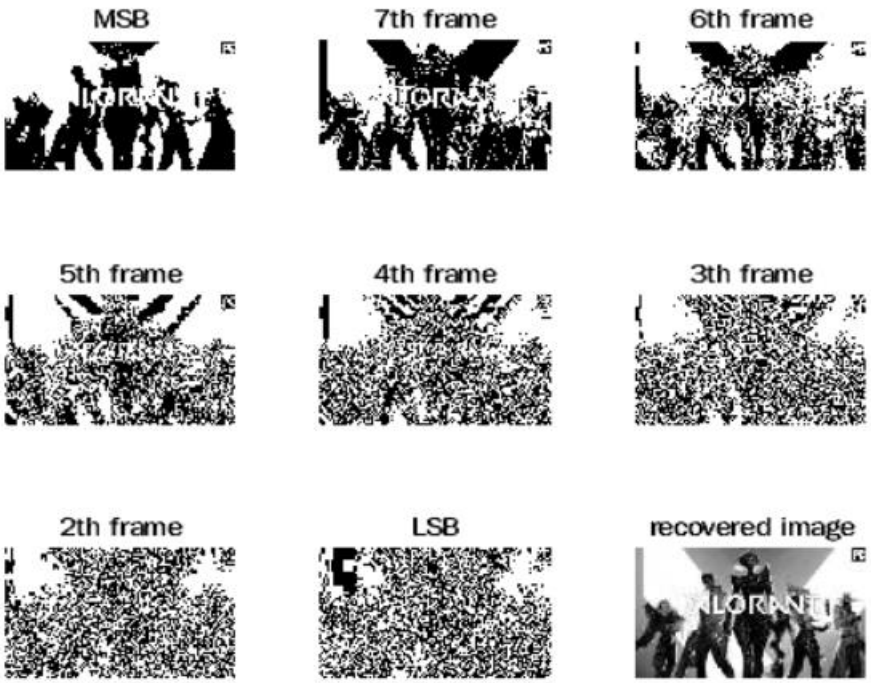
```
i = imread("Car.jpg");
s = imadjust (i, stretchlim(i,[0.05 0.95]), []);
subplot (2,2,1), imshow(i), title('Original');
subplot (2,2,2), imshow(s), title('Stretched Image');
subplot (2,2,3), imhist(i), title('Histogram of Original Image');
subplot(2,2,4), imhist(s), title('Histogram of Stretched Image');
```



Intensity Slicing and Bit Plane Slicing.

```
%Bit plane slicing
it = imread('Car.jpg'); %read the image
itemp = it(:,:,1);
[r,c]=size(itemp); %get the dimensions of image
s=zeros(r,c,8); %pre allocate a variable to store a bit planes of the image
for k=1:8
for i=1:r
for j=1:c
s(i,j,k)=bitget(itemp(i,j),k); %get kth bit from each pixel
end
end
end
figure,imshow(itemp); title('Original Image');%display original image
figure; %display all the 8 bit planes
subplot(3,3,1); imshow(s(:,:,8)), title('8th(MSB) plane');
subplot(3,3,2); imshow(s(:,:,7)), title('7th plane');
subplot(3,3,3); imshow(s(:,:,6)), title('6th plane');
subplot(3,3,4); imshow(s(:,:,5)), title('5th plane');
subplot(3,3,5); imshow(s(:,:,4)), title('4th plane');
subplot(3,3,6); imshow(s(:,:,3)), title('3rd plane');
subplot(3,3,7); imshow(s(:,:,2)), title('2nd plane');
subplot(3,3,8); imshow(s(:,:,1)), title('1st(LSB) plane');
% reconstruct the original image from generated bit planes
rec=s(:,:,1)+s(:,:,2)*2+s(:,:,3)*4+s(:,:,4)*8+s(:,:,5)*16+s(:,:,6)*32+s(:,:,7)*64+s(:,:,8)*128;
```

subplot(3,3,9); imshow(uint8(rec)), title('Recovered Image') %display the reconstructed image

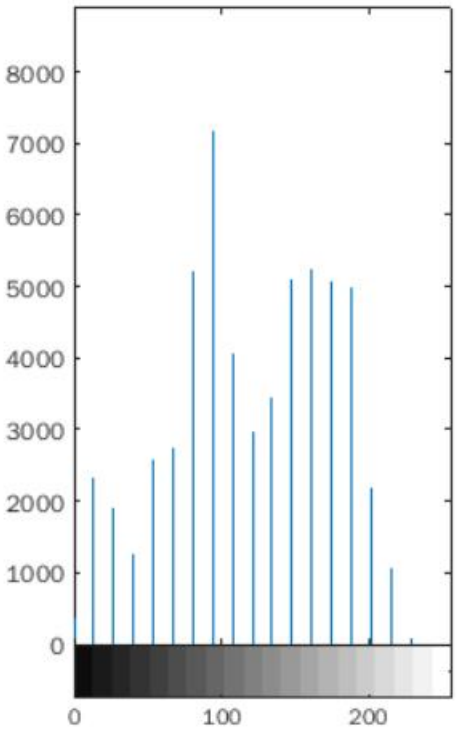


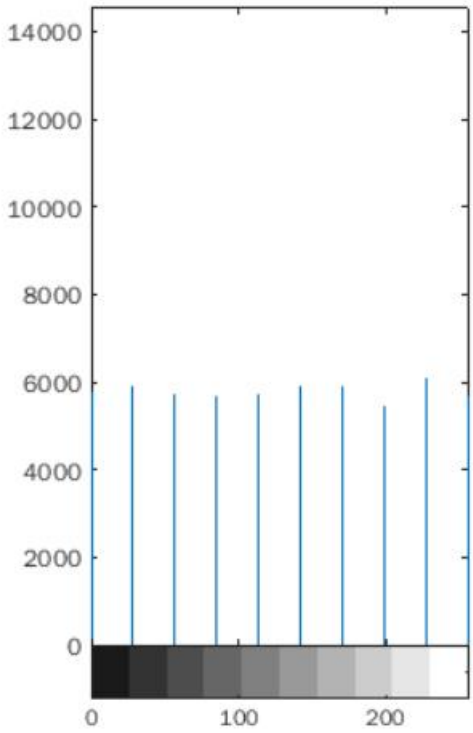
Aim: Write a MATLAB Script for Image Histogram Equalization

Code:

```
img = imread("tree.jpg");

figure
subplot(1,2,1)
imshow(img)
subplot(1,2,2)
imhist(img,5)
j = histeq(img);
figure
subplot(1,2,1)
imshow(j)
subplot(1,2,2)
imhist(j,5)
```

Output:


8000
7000
6000
5000
4000
3000
2000
1000
0
0
100
200

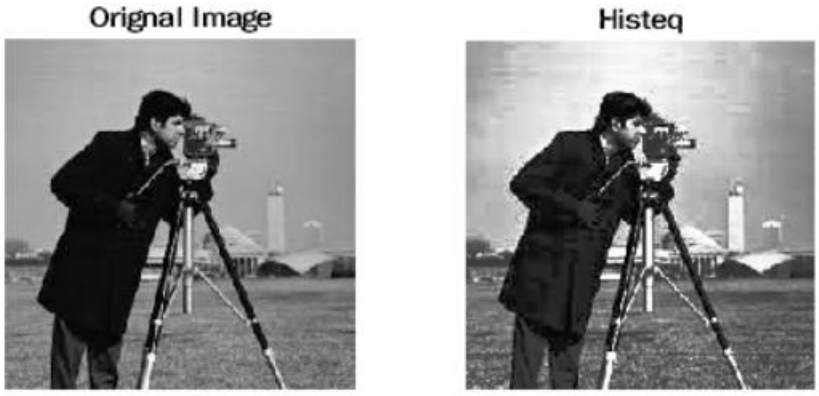

14000
12000
10000
8000
6000
4000
2000
0
0
100
200

(Without histeq() function)

Code:

```
clc;
close all;


orginal = imread('cm.tif');
[rows,columns,~] = size(orginal);
finalResult = uint8(zeros(rows,columns));
pixelNumber = rows*columns;
frequncy = zeros(256,1);
pdf = zeros(256,1);
cdf = zeros(256,1);
cummlative = zeros(256,1);
outpic = zeros(256,1);
for i = 1:1:rows
   for j = 1:1:columns
      val = orginal(i,j);
      frequncy(val+1) = frequncy(val+1)+1;
      pdf(val+1) = frequncy(val+1)/pixelNumber;
   end
end
sum =0 ;
intensityLevel = 255;

for i = 1:1:size(pdf)
   sum =sum +frequncy(i);
   cummlative(i) = sum;
   cdf(i) = cummlative(i)/ pixelNumber;
   outpic(i) = round(cdf(i) * intensityLevel);
end


for i = 1:1:rows
   for j = 1:1:columns
      finalResult(i,j) = outpic(orginal(i,j) + 1);
   end
end
subplot(1,2,1);imshow(orginal);title('Orignal Image');
subplot(1,2,2);imshow(finalResult);title('Histeq');
```

Output:



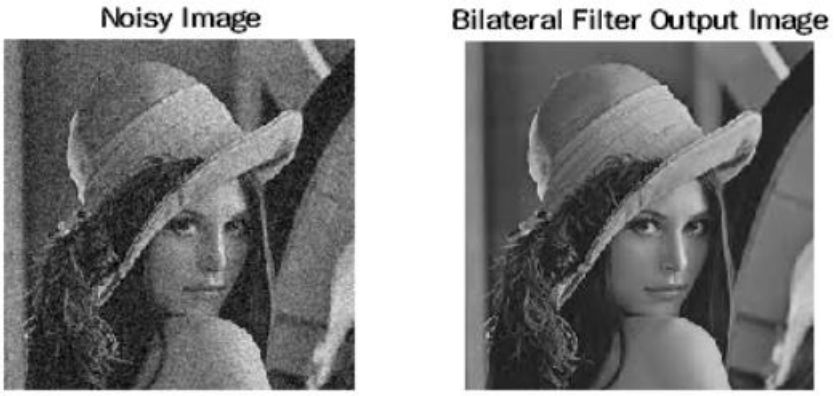
Aim: Write a MATLAB Script for Image Restoration.

Code:

```
clc;
close all;

%Read the Image
a = imread('lena.bmp');
figure;
imshow(a);
b = size(a);

%Convert to grayscale in case it is color
if(size(b,2) == 3)
   a1 = rgb2gray(a);
   figure;
   imshow(a1);
end

%Add noise
a = imnoise(a1, 'gaussian', 0, 0.003998);
figure;
imshow(a);
a = double(a);

%Initialize the parameters
n = 11; %filter size
n1 = ceil(n/2);
vars = 50;  %Special Variance
varr = 25;  %Pixel Value Variance
c = 0;
c1 = 0;
```

```
%Bilateral Filter Loop
for i = n1:b(1) - n1
   for j = n1:b(2) - n1
      for k = 1:n
         for l=1:n
            c=c+gs(sqrt((-n1+k)^2 + (-n1+l)^2),0,vars)*gs(a(i-n1+k,j-n1+1),a(i,j),varr)*a(i-
n1+k,j-n1+1);
            c1=c1+gs(sqrt((-n1+k)^2 + (-n1+l)^2),0,vars)*gs(a(i-n1+k,j-n1+1),a(i,j),varr);
         end
      end
      d(i-n1+1,j-n1+1)=c/c1;
      c=0;
      c1=0;
   end
end
%Convert Output image to unit8
d1 = uint8(d);

%Plotting the Images
figure;
subplot(1,2,1);
imshow(uint8(a));
title('Noisy Image');
%figure;
subplot(1,2,2);
imshow(a1);
title('Bilateral Filter Output Image');
```

Output:



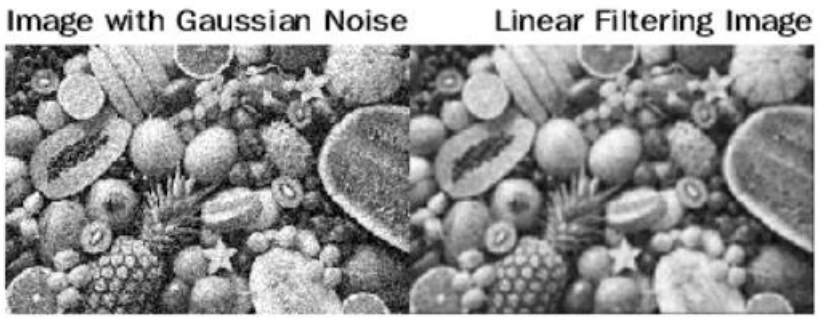
Aim: Write a MATLAB Script for Following Neighbourhood Operations.

Code:

(Linear Filtering)

```
%Linear Filtering
a = imread('fruits.png');
a = rgb2gray(a);
a = im2double(a);

a = imnoise(a, 'gaussian', 0.01);
%a = imnoise(a, 'salt & pepper', 0.02);

w = fspecial('average',[3 3]);
%w = fspecial('average',[5 5]); % size of box filter is increase, noise is
%decrease but image became blur

I = imfilter(a,w);
montage({a, I});
title('Image with Gaussian Noise        Linear Filtering Image');
```

Output:

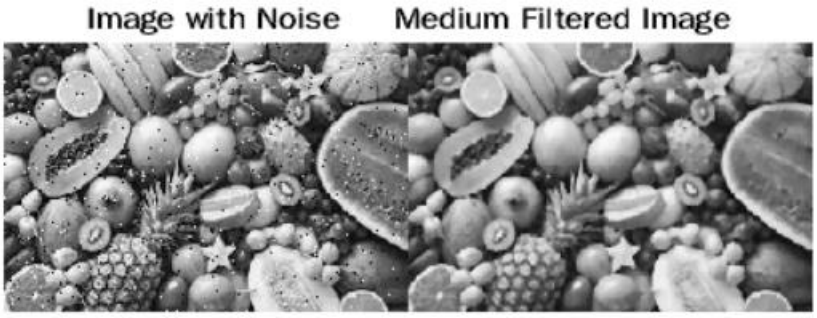


Code:

(Non Linear Filtering)

```
% Non Linear Filtering (Medium Filter)
a = imread('fruits.png');
a = rgb2gray(a);
a = im2double(a);
a = imnoise(a, 'salt & pepper', 0.02);
I = medfilt2(a); %Apply Medium filter of the image in 2-D
montage({a,I});
title('Image with Noise     Medium Filtered Image');
```

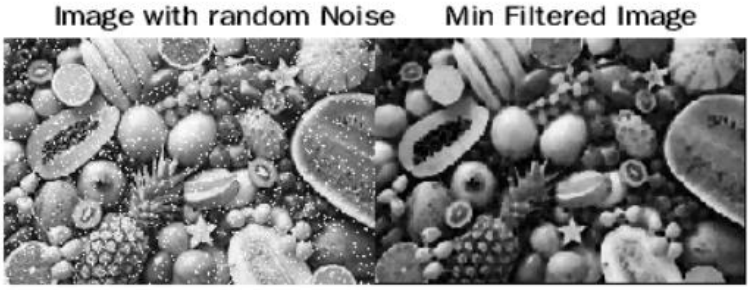
Output:



Code:

```
% Non Linear Filtering (Min Filter)
a = imread('fruits.png');
a = rgb2gray(a);
a = im2double(a);

x = rand(size(a));%Create matrix with random size of (a) image
%Rand generated random numbers between 0 to 1
a(x(:)>0.95)=255; % Assigning white dots to image
min_Img = ordfilt2(a, 1, ones(3,3)); %Ordred filter 2
montage({a,min_Img});
title('Image with random Noise     Min Filtered Image');
```

Output:



Code:

```
% Non Linear Filtering (Max Filter)
a = imread('fruits.png');
a = rgb2gray(a);
a = im2double(a);
x = rand(size(a)); %Random value between 0 to 1

a(x(:)<0.05)=0; %Add black dots to image
```

```
max_Img = ordfilt2(a, 9, ones(3,3)); %Order filtered size 2 find max value
montage({a,max_Img});
title('Image with random Noise      Max Filtered Image');
```

Output:

Code:

```
clc
clear all

k=imread('fruits.png');
figure;
imshow(k);

title('Input Image');
x=rand(size(k));
x
k(x(:)>0.95)=255;

figure;
imshow(k);

sto=[];
[a b]=size(k);

output=zeros(a,b);
for i=2:a-1
   for j=2:b-1
      sto=[k(i-1,j-1),k(i-1,j),k(i-1,j+1),k(i,j-1),k(i,j),k(i,j+1),k(i+1,j-1),k(i+1,j),k(i+1,j+1)];
      es=median(sto);
         output(i,j)=es;
      sto=[];

   end
end

figure;
imshow(uint8(output));
```

Output:

Input Image

Aim: Write a MATLAB Script for Fourier Transform

Code:

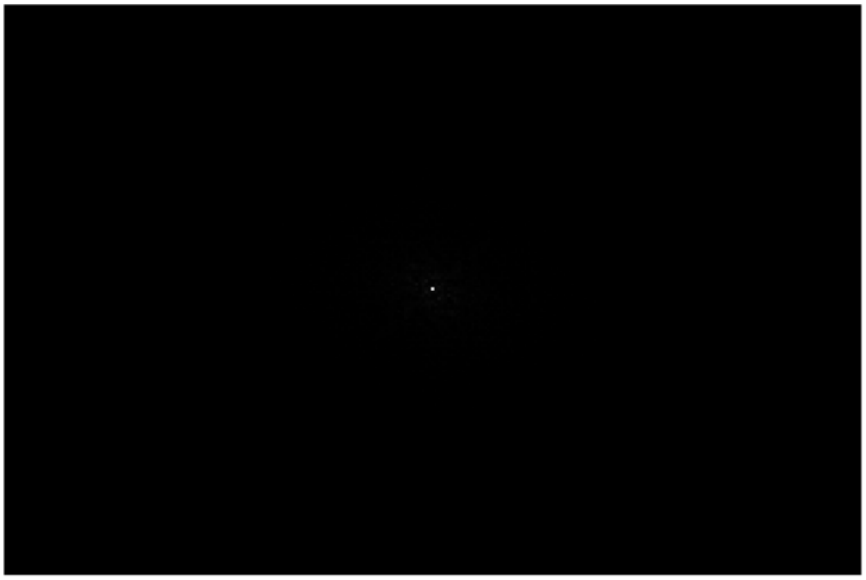
```
imdata = imread('fruits.png');
figure(1);imshow(imdata);title('Original Image');

imdata = rgb2gray(imdata);
figure(2);imshow(imdata);title('Gray Image');

%Get Fourier Transform of any Image
F = fft2(imdata);
%Fourier Transform of Image
S = abs(F);
figure(3);imshow(S,[]);title('Fourier Transform of Image');
%Get the Centered Specturm
Fsh = fftshift(F);
figure(4);imshow(abs(Fsh),[]);title('Centered Specturm of Image');
%Apply Log Transform
S2 = log(1+abs(Fsh));
figure(5);imshow(S2,[]),title('Log Transform of Image');
%Reconstruct Image
F = ifftshift(Fsh);
f = ifft2(F);
figure(6);imshow(f,[]),title('Reconstruct of Image');
```

Output:

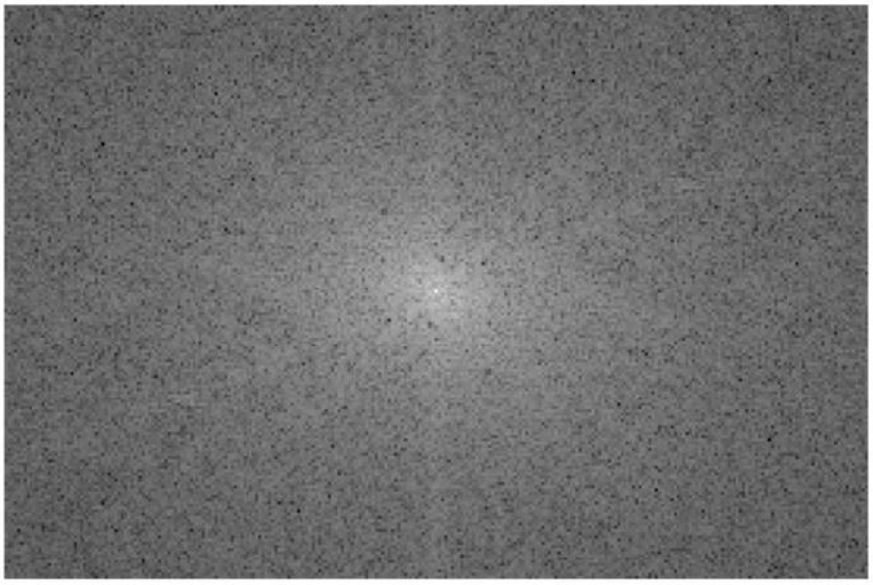
Gray Image

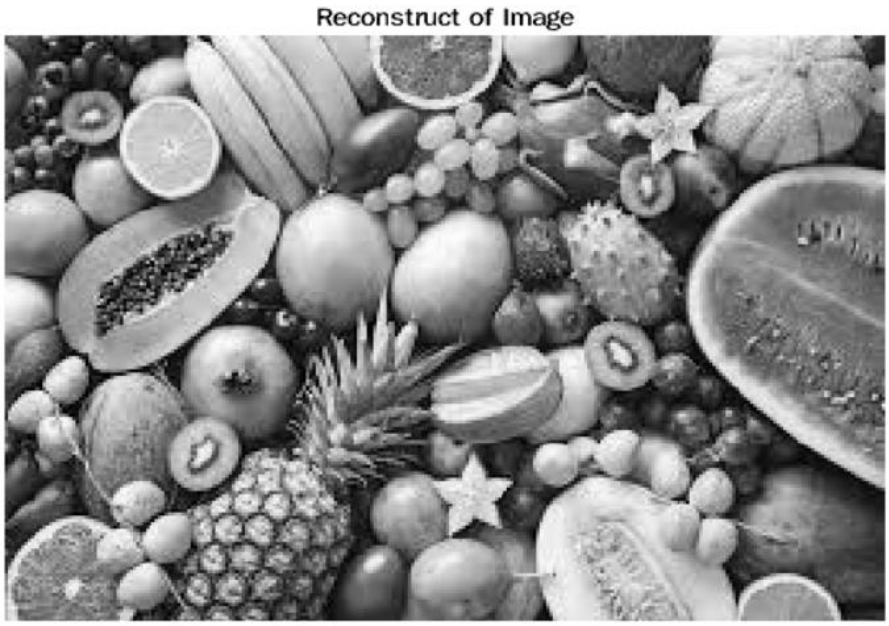
Fourier Transform of Image

Centered Specturm of Image



Log Transform of Image





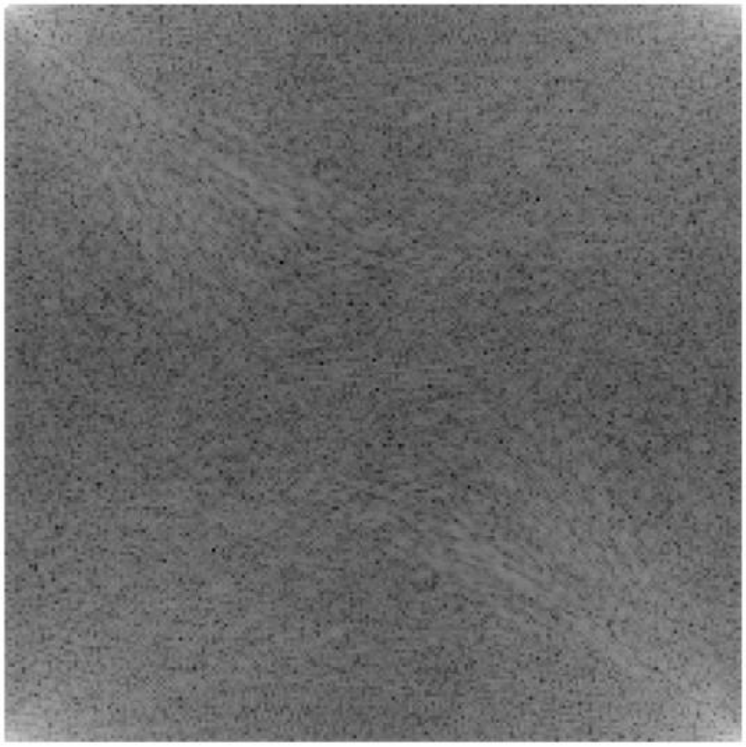
Code:

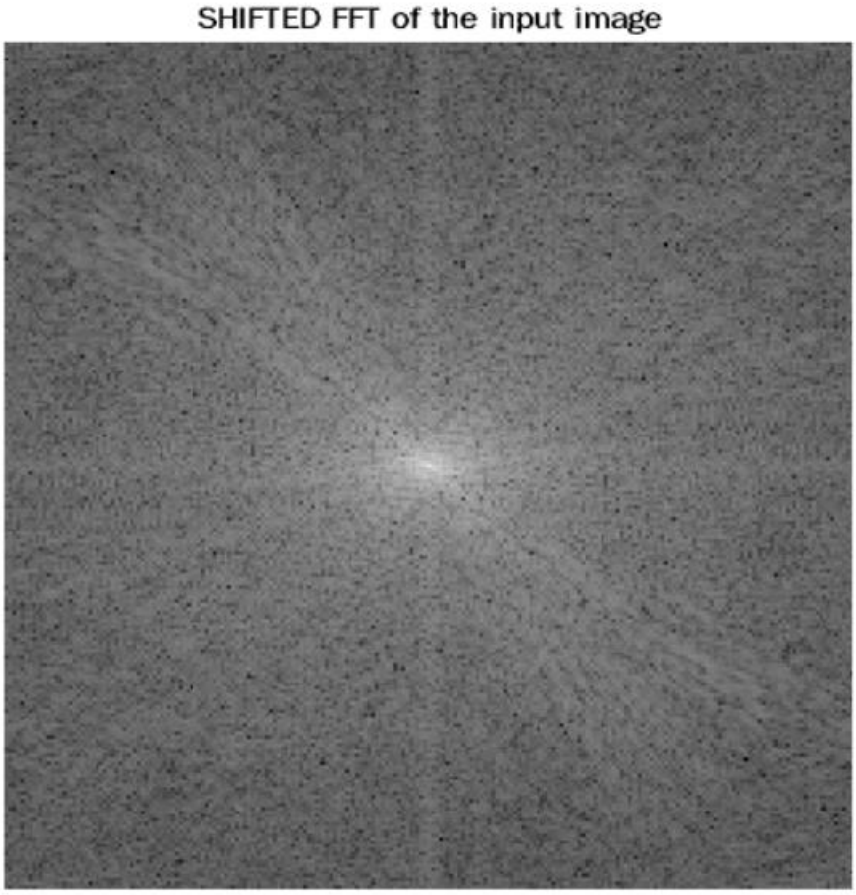
```
%Fourier Transform
a1 = imread('lena.bmp');
a = rgb2gray(a1);
figure(1);
imshow(a);
title('Original Image');

af = fft2(a);
figure(2);
imshow(mat2gray(log(1+abs(af))));
title('FFT of the input image');
impixelinfo

af = fft2(a);
af1 = fftshift(af);
figure(3);
imshow(mat2gray(log(1+abs(af1))));
title('SHIFTED FFT of the input image');
impixelinfo
```

Output:

Original Image

FFT of the input image





Aim: Write a MATLAB Script for Frequency Domain Filters.

Code:

```
I=double(imread('cameraman.tif'));
figure; imshow(I,[]);
[r, c]=size(I);
z=zeros(r,c);
C=20;
mr=r/2;
mc=c/2;
z(mr-C:mr+C,mc-C:mc+C)=255;

Da=fft2(I);
figure, imshow(log(abs(Da)),[]);
Da=fftshift(Da);
figure; imshow(log(abs(Da)),[]);
figure; imshow(z);

% low pass filtering
```

```
Db=Da .*z;
figure; imshow(log(abs(Db)),[]);
Dc=fftshift(Db);
b=real(ifft2(Dc));
figure; imshow(b,[]);

% High pass filter
Z=255-z;
Db=Da .*z;
figure; imshow(log(abs(Db)),[]);
Dc=fftshift(Db);
b=real(ifft2(Dc));
figure; imshow(b, []);

subplot()

%Band Pass Filtering
Z(mr-40:mr+40,mc-40:mc-30)=255;
Z(mr-40:mr+40,mc+30:mc+40)=255;
Z(mr-40:mr-30,mc-40:mc+40)=255;
Z(mr+30:mr+40,mc-40:mc+40)=255;
figure; imshow(Z);
Db = Da .*Z;
figure; imshow(log(abs(Db)),[]);
Dc = fftshift(Db);
b = real(ifft2(Dc));
figure; imshow(b,[]);
Dc=fftshift(Db);
b=real(ifft2(Dc));
figure; imshow(b,[]);
I=double(imread('cameraman.tif'));
figure; imshow(I,[]);
[r, c]=size(I);
z=zeros(r,c);
C=20;
mr=r/2;
mc=c/2;
z(mr-C:mr+C,mc-C:mc+C)=255;

Da=fft2(I);
figure, imshow(log(abs(Da)),[]);
Da=fftshift(Da);
figure; imshow(log(abs(Da)),[]);
figure; imshow(z);

% low pass filtering
Db=Da .*z;
figure; imshow(log(abs(Db)),[]);
Dc=fftshift(Db);
b=real(ifft2(Dc));
```

```
figure; imshow(b,[]);

% High pass filter
Z=255-z;
Db=Da .*z;
figure; imshow(log(abs(Db)),[]);
Dc=fftshift(Db);
b=real(ifft2(Dc));
figure; imshow(b, []);
subplot()
%Band Pass Filtering
Z(mr-40:mr+40,mc-40:mc-30)=255;
Z(mr-40:mr+40,mc+30:mc+40)=255;
Z(mr-40:mr-30,mc-40:mc+40)=255;
Z(mr+30:mr+40,mc-40:mc+40)=255;
figure; imshow(Z);
Db = Da .*Z;
figure; imshow(log(abs(Db)),[]);
Dc = fftshift(Db);
b = real(ifft2(Dc));
figure; imshow(b,[]);
Dc=fftshift(Db);
b=real(ifft2(Dc));
figure; imshow(b,[]);

%Band Reject Filtering
Z(mr-40:mr+40,mc-40:mc-30)=255;
Z(mr-40:mr+40,mc+30:mc+40)=255;
Z(mr-40:mr-30,mc-40:mc+40)=255;
Z(mr+30:mr+40,mc-40:mc+40)=255;
Z=255-Z;
figure; imshow(Z);
Db = Da .*Z;
figure; imshow(log(abs(Db)),[]);

Dc=fftshift(Db);
b=real(ifft2(Dc));
figure; imshow(b,[]);

%Band Reject Filtering
Z(mr-40:mr+40,mc-40:mc-30)=255;
Z(mr-40:mr+40,mc+30:mc+40)=255;
Z(mr-40:mr-30,mc-40:mc+40)=255;
Z(mr+30:mr+40,mc-40:mc+40)=255;
Z=255-Z;
figure; imshow(Z);
Db = Da .*Z;
figure; imshow(log(abs(Db)),[]);

Dc=fftshift(Db);
```

```
b=real(ifft2(Dc));
figure; imshow(b,[]);
```

Output:

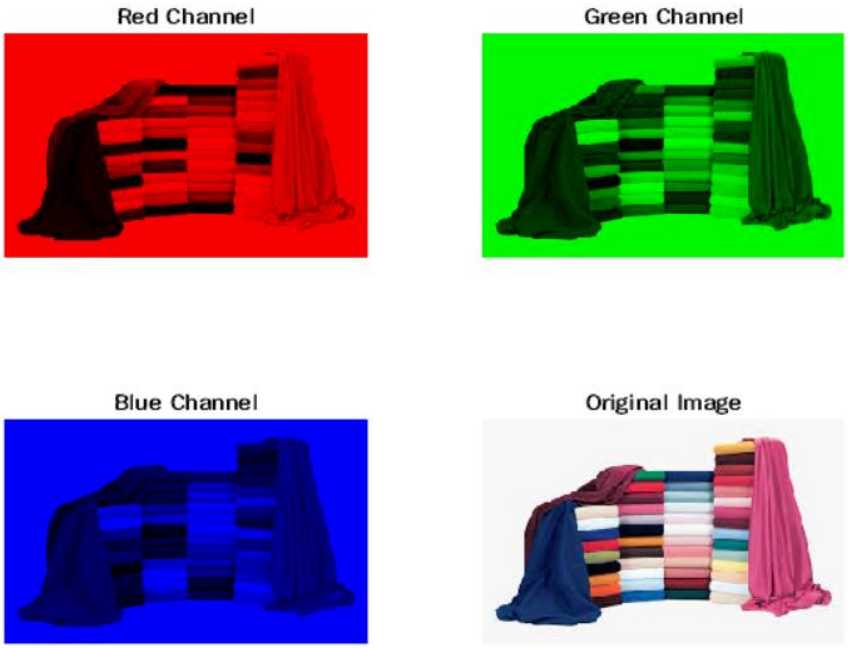
Aim: Write a MATLAB Script for illustrating Color Image Processing.

Code:

```
RGB=imread('fabric.png');
imshow(RGB)
title('Original RGB Image')
R=RGB(:,:,1);
G=RGB(:,:,2);
B=RGB(:,:,3);
RR=RGB;
RR(:,:,2)=0;
RR(:,:,3)=0;
GG=RGB;
GG(:,:,1)=0;
GG(:,:,3)=0;
BB=RGB;
BB(:,:,1)=0;
BB(:,:,2)=0;

subplot(2,2,1)
imshow(R)
title('Red Channel')

subplot(2,2,2)
imshow(G)
title('Green Channel')

subplot(2,2,3)
imshow(B)
title('Blue Channel')
```

```
subplot(2,2,4)
imshow(RGB)
title('Original Image')

subplot(2,2,1)
imshow(RR)
title('Red Channel')

subplot(2,2,2)
imshow(GG)
title('Green Channel')

subplot(2,2,3)
imshow(BB)
title('Blue Channel')

subplot(2,2,4)
imshow(RGB)
title('Original Image')

%Recombine separate color channels into a single, true color RGB image.
rgbImage = cat(3,R,G,B); %cat is function to concate arrays.
figure; imshow(rgbImage)
```

Output:

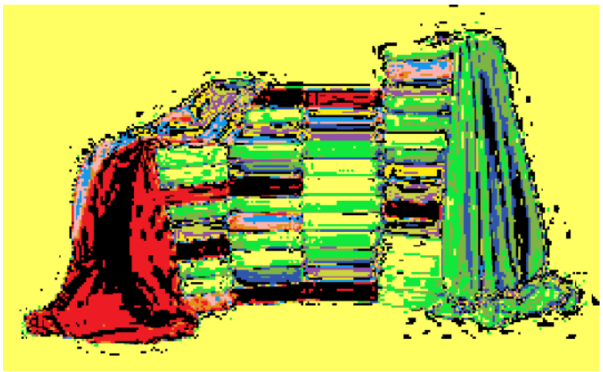


Code:

```
y=imread('fabric.png');
z=rgb2gray(y);
[p,q,r]=size(y);
for i=1:1:p
   for j=1:1:q
      if(y(i,j)>=0) && (y(i,j)<18)
         x(i,j,1)=0;
         x(i,j,2)=0;
         x(i,j,3)=0;
      elseif(y(i,j)>=18) && (y(i,j)<36)
         x(i,j,1)=237;
         x(i,j,2)=27;
         x(i,j,3)=36;
      elseif(y(i,j)>=36) && (y(i,j)<54)
         x(i,j,1)=228;
         x(i,j,2)=142;
         x(i,j,3)=31;
      elseif(y(i,j)>=54) && (y(i,j)<72)
         x(i,j,1)=251;
         x(i,j,2)=179;
         x(i,j,3)=180;
       elseif(y(i,j)>=72) && (y(i,j)<90)
         x(i,j,1)=21;
         x(i,j,2)=154;
         x(i,j,3)=233;
       elseif(y(i,j)>=108) && (y(i,j)<126)
         x(i,j,1)=252;
         x(i,j,2)=234;
         x(i,j,3)=12;
       elseif(y(i,j)>=126) && (y(i,j)<144)
         x(i,j,1)=146;
         x(i,j,2)=80;
         x(i,j,3)=167;
       elseif(y(i,j)>=144) && (y(i,j)<162)
         x(i,j,1)=203;
         x(i,j,2)=213;
         x(i,j,3)=62;
       elseif(y(i,j)>=180) && (y(i,j)<198)
         x(i,j,1)=48;
         x(i,j,2)=85;
         x(i,j,3)=173;
       elseif(y(i,j)>=198) && (y(i,j)<216)
         x(i,j,1)=126;
         x(i,j,2)=180;
         x(i,j,3)=67;
       elseif(y(i,j)>=216) && (y(i,j)<232)
         x(i,j,1)=16;
         x(i,j,2)=233;
```

```
            x(i,j,3)=59;
          elseif(y(i,j)>=232) && (y(i,j)<255)
            x(i,j,1)=255;
            x(i,j,2)=255;
            x(i,j,3)=100;
        end
    end
end
%subplot(1,2,1);
figure;
imshow(y);
figure;
imshow(z);
%subplot(1,2,2);
x=x/255;
figure
%image(x);
imshow(x);
```

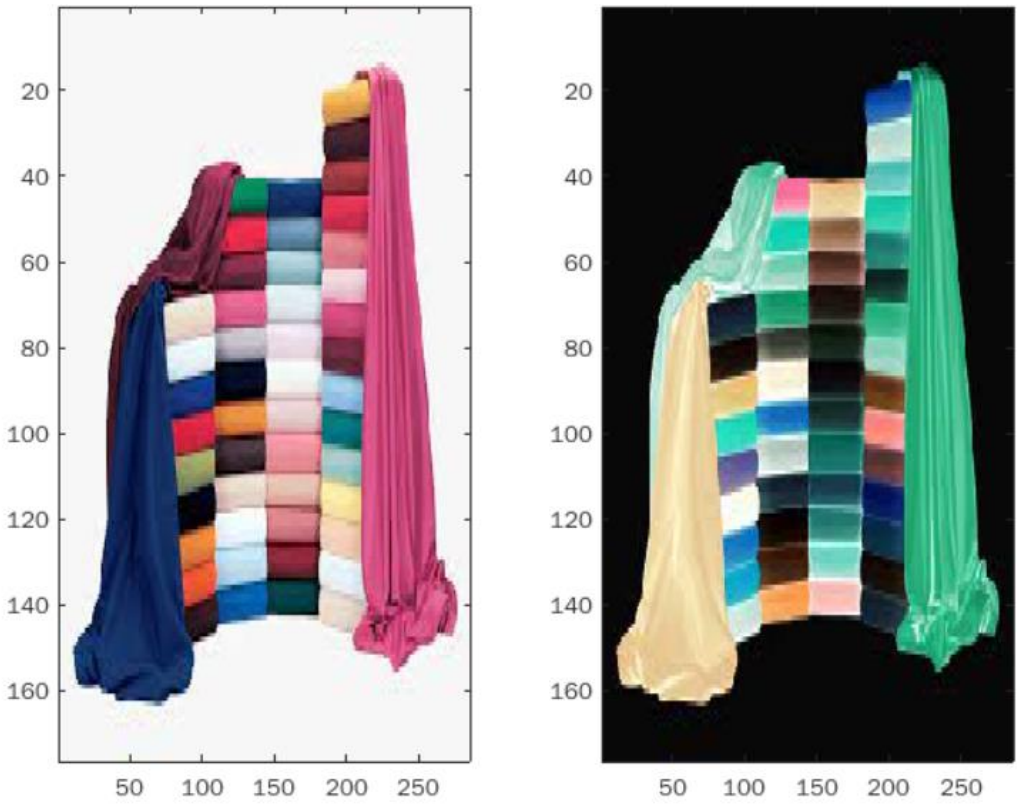
Output:



Code:

```
% RGB to CMY (Cyan Magenta Yellow)
F=imread('fabric.png');
F=im2double(F);
r=F(:,:,1);
g=F(:,:,2);
b=F(:,:,3);
c=1-r;
m=1-g;
y=1-b;
CMY=cat(3,c,m,y);
subplot(121),image(F);
subplot(122),image(CMY);
```

Output:

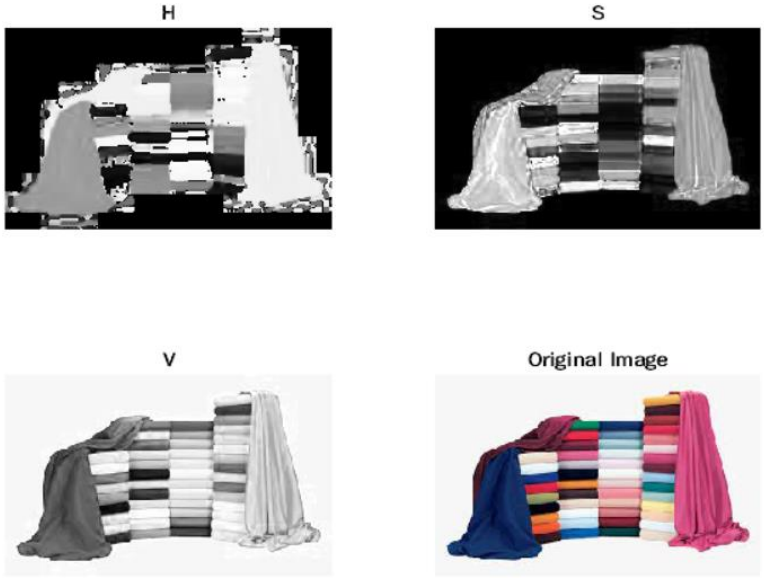


Code:

```
%Convert RGB image to HSV
RGB=imread('fabric.png');
hsvImage=rgb2hsv(RGB);
%Extract out the H, S, and V images individually
hImage=hsvImage(:,:,1);
sImage=hsvImage(:,:,2);
vImage=hsvImage(:,:,3);
```

```
subplot(2,2,1)
imshow(hImage)
title('H')

subplot(2,2,2)
imshow(sImage)
title('S')

subplot(2,2,3)
imshow(vImage)
title('V')

subplot(2,2,4)
imshow(RGB)
title('Original Image')
```

Output:

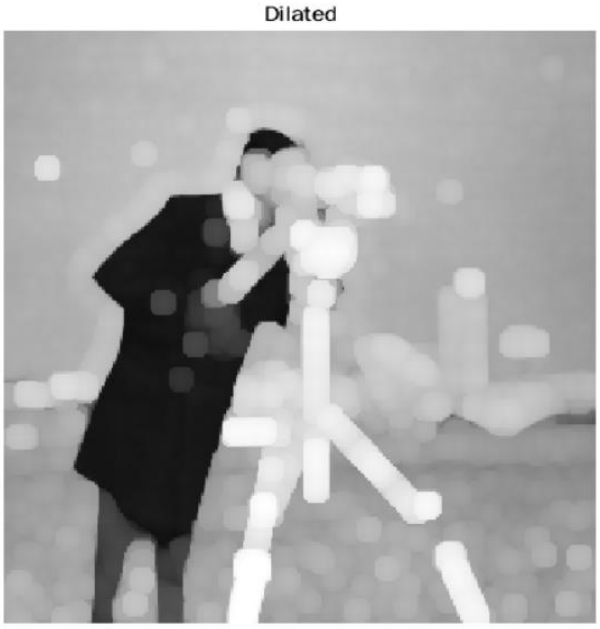


Aim: Write a MATLAB Script for Morphological Operations.

Code:

```
close all;
I=imread('cameraman.tif');
se=strel('ball',5,5);
I2=imdilate(I,se);
imshow(I), title('Original')
figure, imshow(I2), title('Dilated');
```

Output:



Code:

```
I=imread('cameraman.tif');
se=strel('ball',5,5);
I2=imerode(I,se);
imshow(I), title('Original')
figure, imshow(I2), title('Eroded');
```

Output:

Aim: Write a MATLAB Script for Image Segmentation

Code:

```
k=input('Enter the file name: ','s');
im=imread(k);
im1=rgb2gray(im);
im1=medfilt2(im1,[3 3]);
BW = edge(im1,'sobel');
[imx,imy]=size(BW);
msk=[0 0 0 0 0;
    0 1 1 1 0;
    0 1 1 1 0;
    0 1 1 1 0;
    0 0 0 0 0;];
B=conv2(double(BW),double(msk));
L=bwlabel(B,8);
mx=max(max(L))
[r,c]=find(L==17);
rc=[r c];
[sx sy]=size(rc);
n1=zeros(imx,imy);
for i=1:sx
   x1=rc(i,1);
   y1=rc(i,2);
   n1(x1,y1)=255;
end
```

```
figure;
subplot(2,2,1);imshow(im);
subplot(2,2,2);imshow(im1);
subplot(2,2,3);imshow(B);
subplot(2,2,4);imshow(n1,[]);
```

Output: