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

Output: (1a)

```
--> exec('C:\Users\ayush\OneDrive\Desktop\TCSC\SEM6\Digital Image

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

  "Linear 2D convolution y="
```

```
clc ;
x = input("Enter the values of x(n)");
h = input("Enter the values of h(n)");
X = fft2(x);
H = fft2(h);
Y = X.*H;
y = ifft(Y);
disp (y, 'Circular Convolution Result y = ');
```

Output: (1b)

```
Enter the values of x(n)[1,2;3,4]

Enter the values of h(n)[5,6;7,8]


 Circular Convolution Result y =

    70.    68.
    62.    60.
```

```
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 plus alias = ');
```

Output: (2)

```
Linear Convolution Result y =

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

Circular Convolution Expressed as Linear Convolution plus alias =

   70.   68.
   62.   60.
```

```
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: (3a)

```
 Linear Cross Correlation Result y =

    9.     9.     2.
    21.    24.    9.
    10.    22.    4.
```

```
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: (3b)

```
 Circular Correlation Result y =

    37.    23.
    35.    25.
```

```
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 (3c)

```
 Linear auto Correlation Result x =

    1.    2.    1.
    2.    4.    2.
    1.    2.    1.
```

```
clc ;
f = [1 ,1 ,1 ,1;1 ,1 ,1 ,1;1 ,1 ,1 ,1;1 ,1 ,1 ,1];
t = fft2(f);
disp(t, "2D DFT of given 2D image = ");
```

Output: DFT of 4x4 gray scale image (4)

```
 2D DFT of given 2D image =

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

```
f=[2 4 4 2;4 6 8 3;2 8 10 4;3 8 6 2];
[M N]=size(f);
const=sqrt(2/N);
for k=0:1:N-1
for l=0:1:N-1
if k==0
c(k+1,l+1)=1/sqrt(N);
else
a=2*l;
c(k+1,l+1)=const*cos((%pi*k*(a+1))/(2*N));
end end
end
f1=c*f;
disp(f1);
F=c*f*c';
disp(c,'Discrete Cosine Transfor');
disp(F,'Discrete Cosine Transform of f(m,n) is');
subplot(221);
imshow(f);
title('Image in spatial domain')
subplot(223);
```

```
imshow(F);
title('Image in frequency domain')
```

output: (5 a)

```
--> exec('C:\Users\ayush\OneDrive\Desktop\TCSC\SEM6\Digital :

   5.5          13.          14.          5.
  -0.1120854   -3.154322    -1.8477591   -0.2705981
  -0.5         -1.          -4.          -1.5
  -1.577161     0.2241708    0.7653669    0.6532815

   0.5          0.5          0.5          0.5
   0.6532815    0.2705981   -0.2705981   -0.6532815
   0.5         -0.5         -0.5          0.5
   0.2705981   -0.6532815    0.6532815   -0.2705981

  "Discrete Cosine Transfor"

   19.         -0.2705981   -8.          0.6532815
  -2.6923823   -0.25         2.3096988    0.8964466
  -3.5          1.4650756    1.5         -1.6892464
   0.032829    -1.6035534   -0.9567086   -0.25

  "Discrete Cosine Transform of f(m,n) is"
```

Image in spatial domain

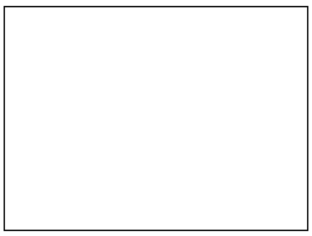
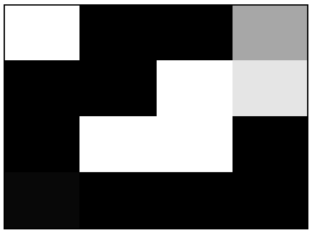


Image in frequency domain



```
clc ;
//X = [4 ,3 ,5 ,6;4 ,2 ,7 ,7;5 ,5 ,6 ,7];
X=input('Enter the matrix');
[m , n ]= size (X) ;
A = 0;
E = 0;
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 = ' )
//KL transform
for i = 1: n
Y (: , i )= T * X (: , i ) ;
end
```

```
disp (Y , 'KL transformation of the input matrix Y = ' )
// Reconstruction
for i = 1: n
x (: , i )= T'* Y (: , i ) ;
end
disp (x , ' Reconstruct matrix of the given sample matrix X = ' )
```

output: 5b

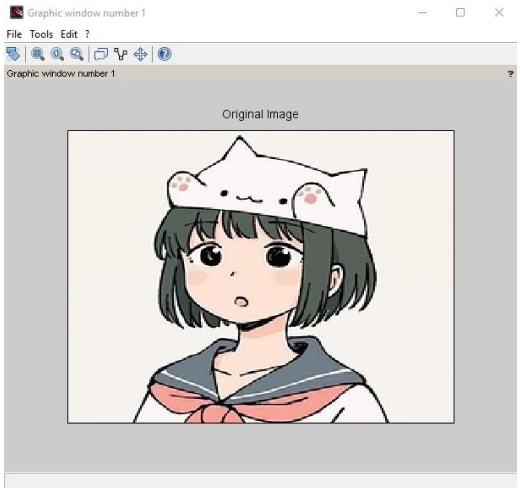
```
 Enter the matrix [4 3 5 6;4 2 7 7;5 5 6 7]


  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
  -0.780262    0.1929481   0.5949473

  The KL tranform basis is =

   0.4384533   0.8471005   0.3002988
   0.4460381  -0.4951684   0.7455591
  -0.780262    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.
```

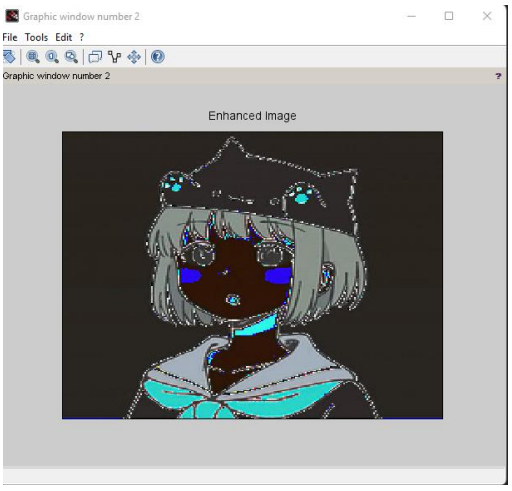


```
clc;
a=imread("E:\hello\dodoj.jpg");
b = double (a) +50;
b = uint8 (b);
figure (1)
imshow (uint8(a));
title ('Original Image')
figure (2)
imshow (uint8(b));
```

title ('Enhanced Image')

output: Brightness enhancement of an image (6a)






```
clc ;
close ;
a = imread ("E:\hello\dodoj.jpg");
a = rgb2gray (a);
b = double (a) *0.5;
b = uint8 (b);
c = double (b) *2.5;
c = uint8 (c);
figure (1)
imshow(uint8(a));
02
title ( 'Original Image' );
figure (2)
imshow(b);
title ( 'Decrease in Contrast' );
figure (3)
imshow(c);
title ( 'Increase in Contrast' );
```

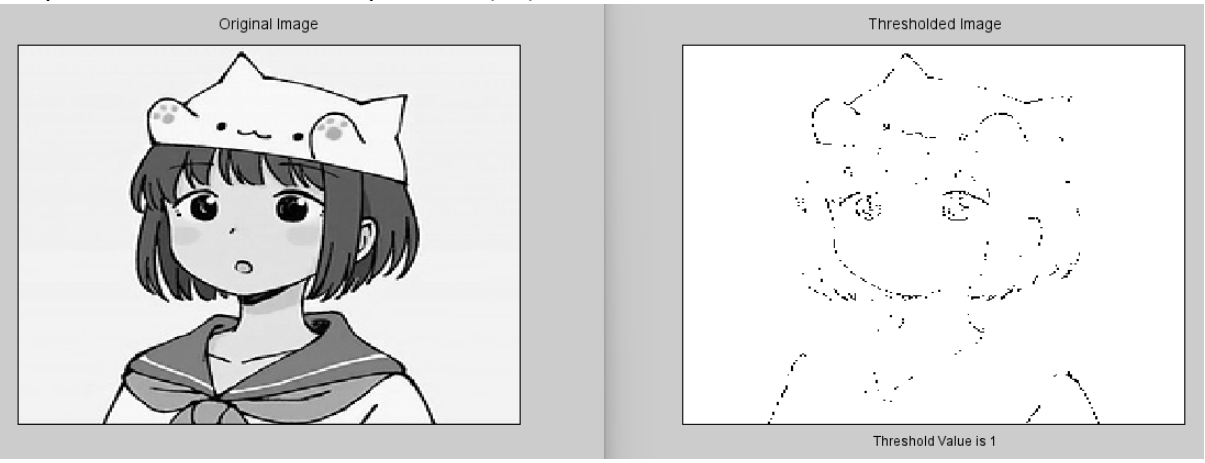
output: Contrast Manipulation (6b)

```
clc ;
close ;
a = imread ("E:\hello\dodoj.jpg");
k = 255 - double (a);
k = uint8 (k);
figure(1);
imshow (uint8(a));
title ( 'Original Image' );
figure(2);
imshow (k);
title ( 'Negative of Original Image' );
```

output: image negative (6c)


```
clc;
a=imread("E:\hello\dodoj.jpg");
a=rgb2gray(a);
[m n]=size(a);
t = input("Enter the 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(uint8(a));
title('Original Image ')
figure(2)
imshow(uint8(b));
title('Thresholded Image')
xlabel(sprintf('Threshold Value is %g',t))
```

output: Perform threshold operation (7a)

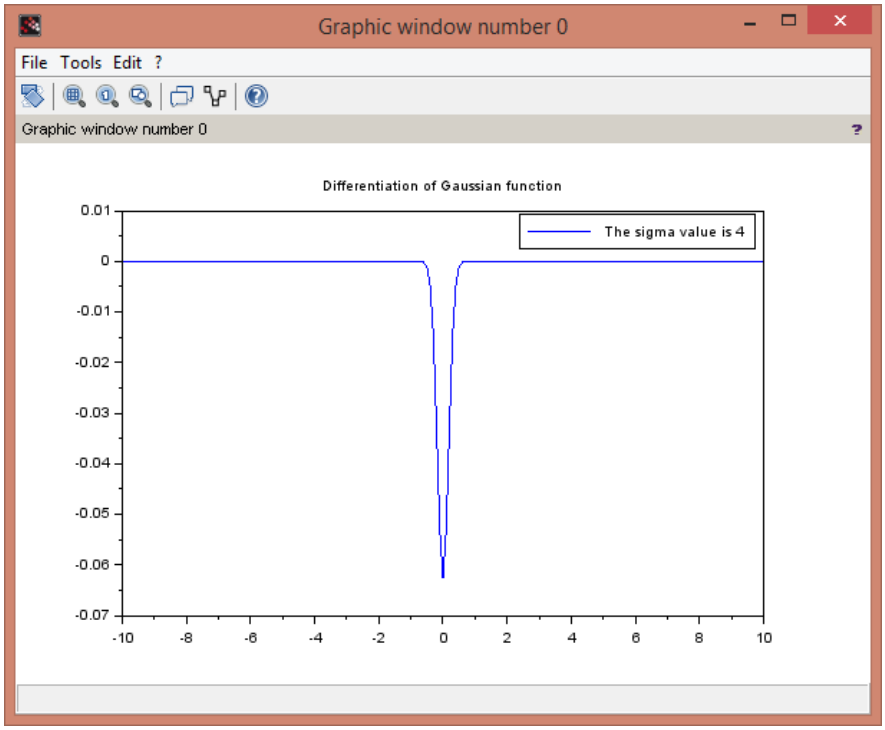



```
clc;
x=imread("E:\hello\dodoj.jpg");
x=rgb2gray(x);
y=double(x);
[m n]=size(y);
L=max(x);
a=round(L/2);
b=L;
for i =1: m
   for j =1: n
      if(y(i,j) >=a & y(i,j) <=b)
         z(i,j) = L;
      else
         z(i,j)=0;
      end
   end
end
z=uint8(z);
figure(1)
imshow(x)
title('Orginal Image')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

output: 7b


```
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 ')
```

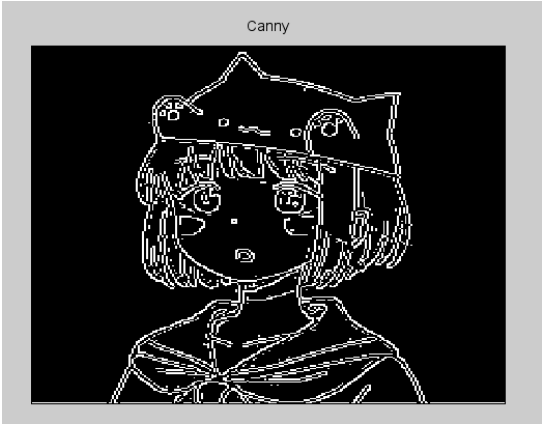
output: 8a




```
clc;
img = imread("E:\hello\dodoj.jpg");
img=rgb2gray(img);
c=edge(img,'sobel',0.5)
```

```
e=edge(img,'canny')
figure(1)
imshow(img)
title('Original Image')
figure(2)
imshow(c)
title('Sobel')
figure(3)
imshow(e)
title('Canny')
figure(5)
```

output: 8b



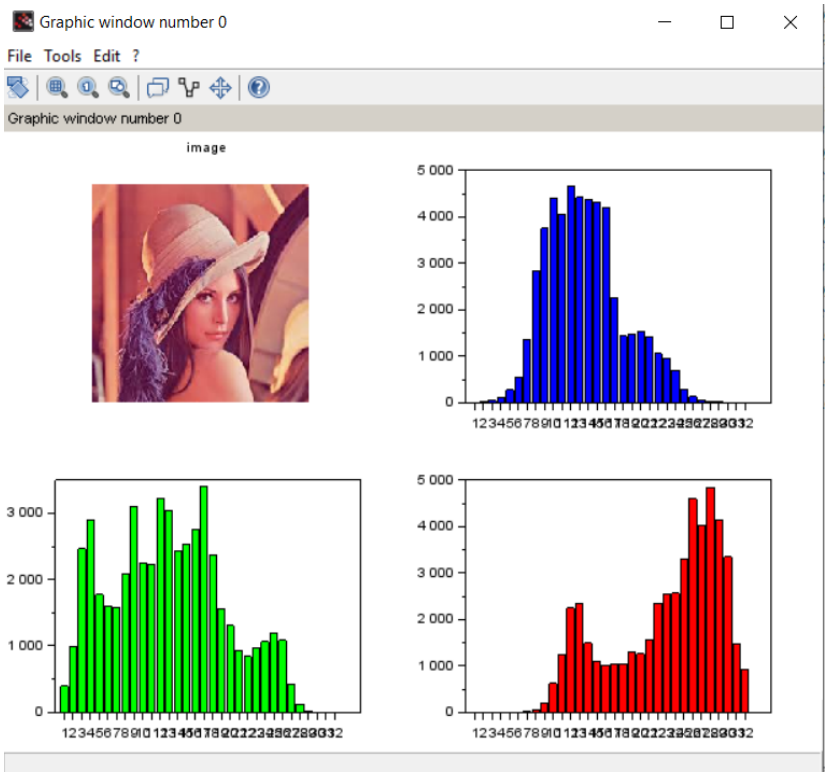
```
clc;
a=[0 1 1 0 1 0 1 1 1 1 1 0 0 0 0 0 0 0 1 1 0 1];
n=2
for k=1:length(a)-1
   if a(k)~=a(k+1)
      n=n+1;
end
end
rle=ones(1,n);
rle(1)=a(1);
m=2;
for i=1:k
   if a(i)==a(i+1)
      rle(m)=rle(m)+1;
    else
       m=m+1;
     end
end
disp('RLE Compression' );
disp(rle);
```

output:

```
RLE Compression

   0.    1.    2.    1.    1.    1.    5.    7.    2.    1.    1.

-->
```

==========

```
scicv_Init();

img = imread(getSampleImage("lena.jpg"));
subplot(2, 2, 1);
matplot(img);
title("image");
//Histogram of the three RGB channels taken separately
//Note: OpenCV color channel order is reversed (BGR)
histB = calcHist(img, 0, [], 1, 32, [0 256]);
subplot(2, 2, 2);
bar(histB(:), 'blue');
histG = calcHist(img, 1, [], 1, 32, [0 256]);
subplot(2, 2, 3);
bar(histG(:), 'green');
histR = calcHist(img, 2, [], 1, 32, [0 256]);
subplot(2, 2, 4);
bar(histR(:), 'red');
delete_Mat(img);
delete_Mat(histB);
delete_Mat(histG);
delete_Mat(histR);

```



```
scicv_Init();
img = imread(getSampleImage("noise.png"));
[res, img_bw] = threshold(img, 127, 255, THRESH_BINARY_INV);
element = getStructuringElement(MORPH_RECT, [5,5])
img_dilate = dilate(img_bw, element);

img_erode = erode(img_bw, element);
img_open = morphologyEx(img_bw, MORPH_OPEN, element);
img_close = morphologyEx(img_bw, MORPH_CLOSE, element);
subplot(221);
img_dilate_reverse = bitwise_not(img_dilate);
matplot(img_dilate_reverse);
title("dilate");
subplot(222);
img_erode_reverse = bitwise_not(img_erode);
matplot(img_erode_reverse);
title("erode");
subplot(223);
img_open_reverse = bitwise_not(img_open);
matplot(img_open_reverse);
title("open");
subplot(224);
img_close_reverse = bitwise_not(img_close);
matplot(img_close_reverse);
title("Close");
delete_Mat(img);
delete_Mat(img_bw);
delete_Mat(element);
delete_Mat(img_dilate);
delete_Mat(img_dilate_reverse);
delete_Mat(img_erode);
delete_Mat(img_erode_reverse);
delete_Mat(img_open);
delete_Mat(img_open_reverse);
delete_Mat(img_close);
delete_Mat(img_close_reverse);
```