

LAB FILE

Name: Boby

Semester/Year: 6th/3rd

University Roll no.: 191500223 Section/Serial No: L1/(12)

Subject: Digital Image Processing Lab (BCSE 0131)

**Submitted to:**

**Dr. Manmohan Singh**

# Assignment 1

## % Create command to familiarize with MATLAB & Create the matrices & perform the various operations on them.

a=[1,2,3,4,5;6,7,8,9,0,];

%size of matrics of 'a' in as row and column; [r,c]=size(a);

% zeros matrics a1=zeros(4,2);

% ones matrics a2=ones(3,4);

%initialize two matrics for performing arithmetic operations A

= [10 20 30; 11 12 13; 40 50 60];

B = [51 52 53; 21 23 21; 44 54 64];

%addition of two matrics addition = A+B;

%subtraction of two matrics subtraction = A-B;

%multiplication of two matrics multiplication = A\*B;

%division of two matrics division = A/B;

% Finding Length of a matrics l = length(a);

% If we want all the elements in the matrics are same. we can use 'ones' n=12\*ones(5,6);

% Find diagonal Elements in the matrics diagonal\_ele = diag(A);

%Find identity matrics identity\_mat = eye(3,3);

%find sin value of matrics sin\_value = sin(A);

% For Transpose of matrics T=transpose(A);

% Using Sum function

S = sum(T);

%Inverse of a matrics inverse\_mat = inv(A);

% For determinant of matrics detminant\_mat = det(A);

%Find rank of a matrics rank\_mat = rank(A);

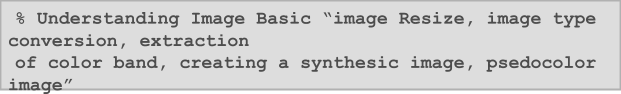
%Find eigon value and vector of a matrics [V,D] = eig(A);

%Find absolute value of matrics of complex elements C = [2+3i 1+9i; 6+7i 9+i];

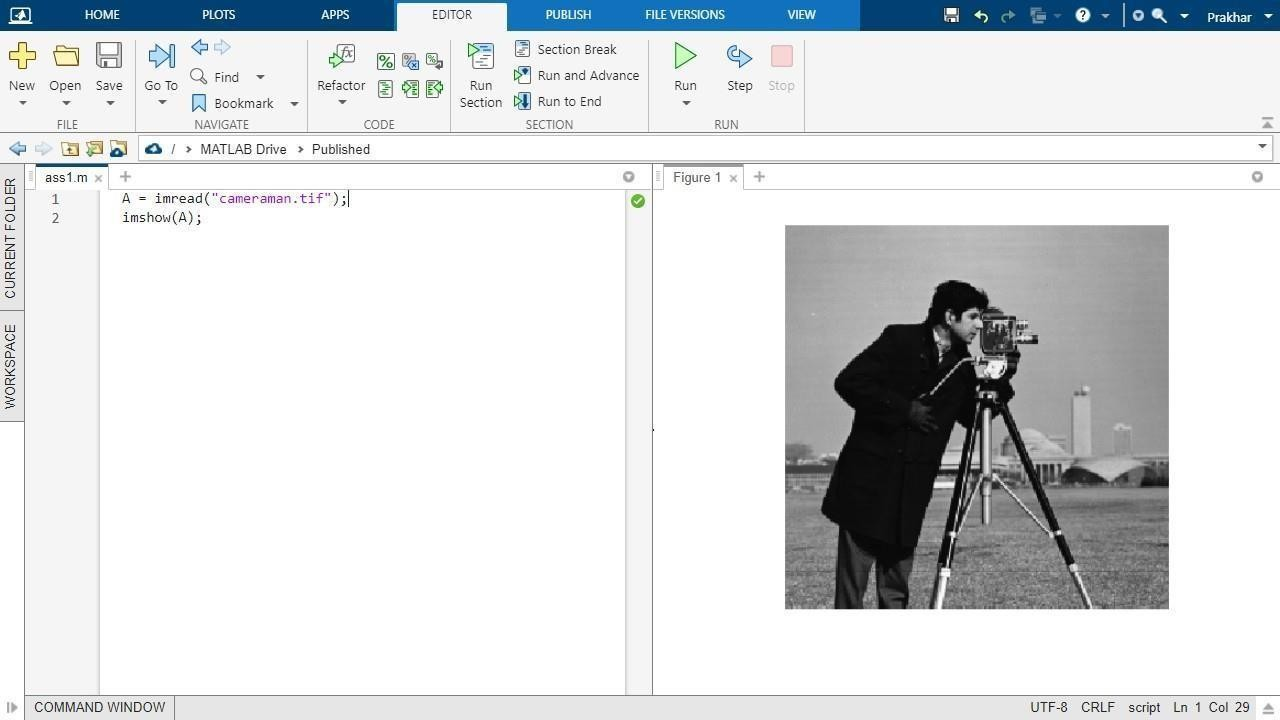
c1 = abs(C);

%Matrics of random elements R = rand(3,4

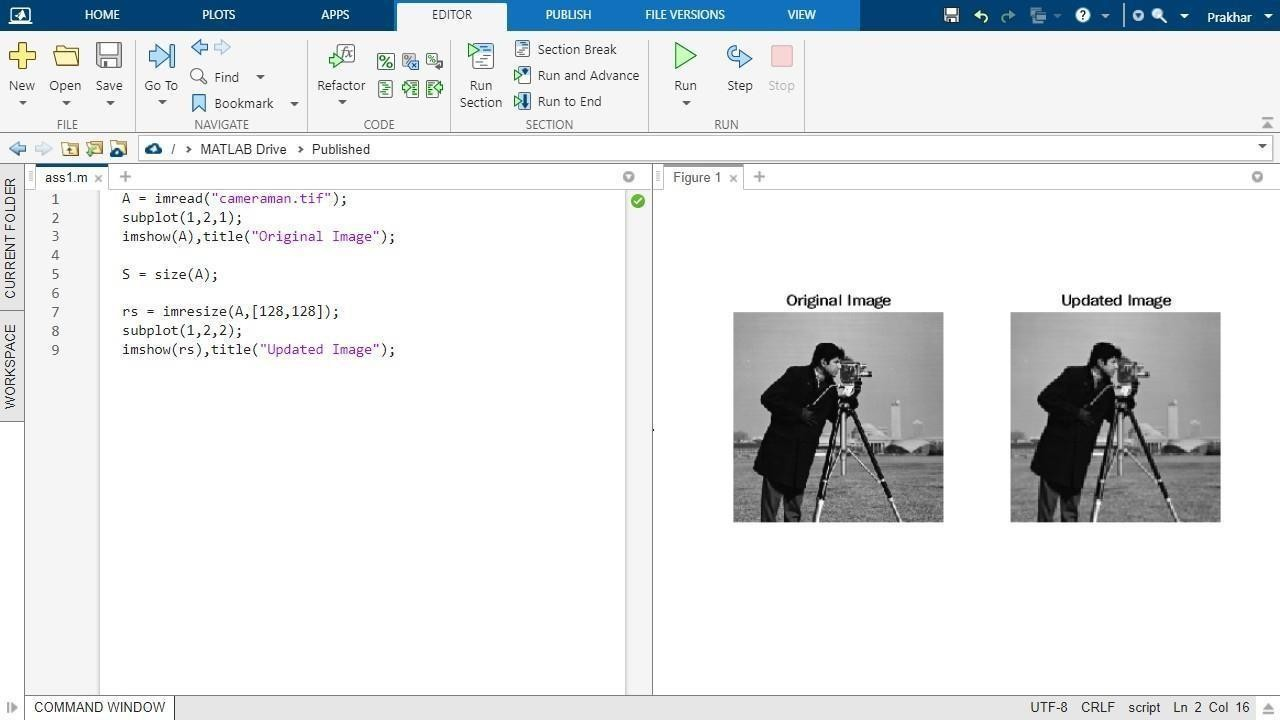
# Assignment 2



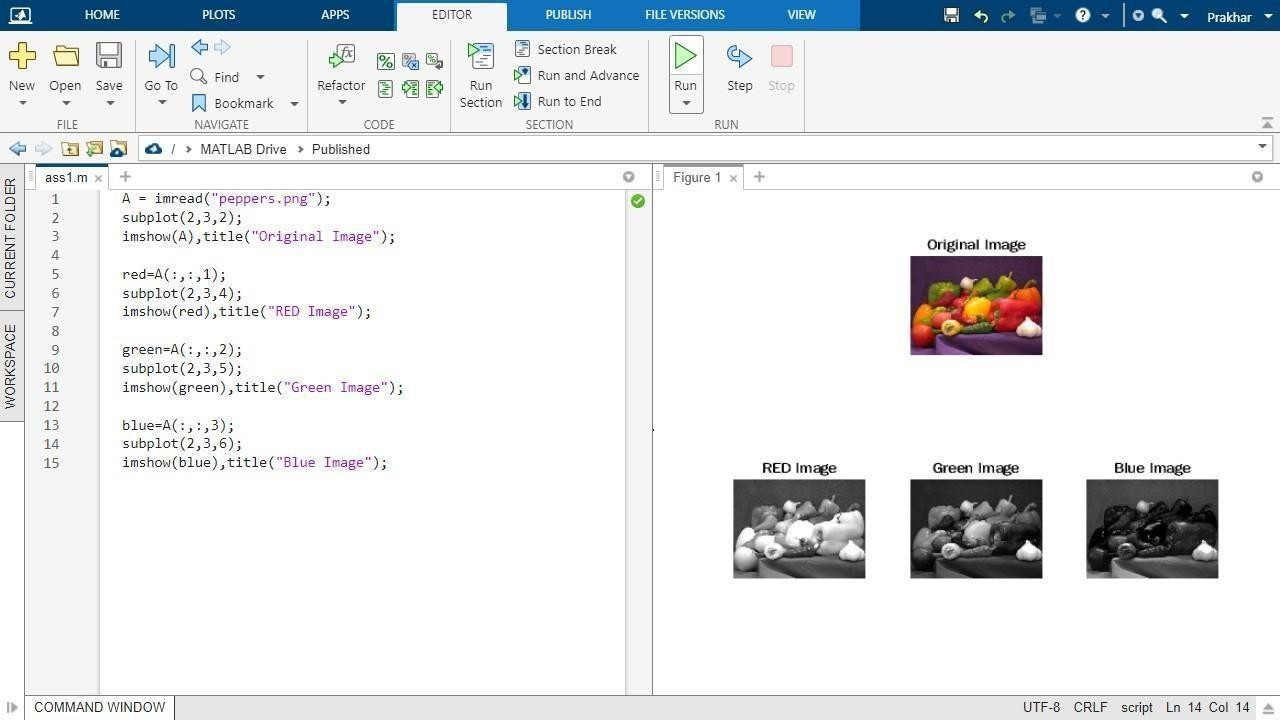
%Reading an image as input and show the output



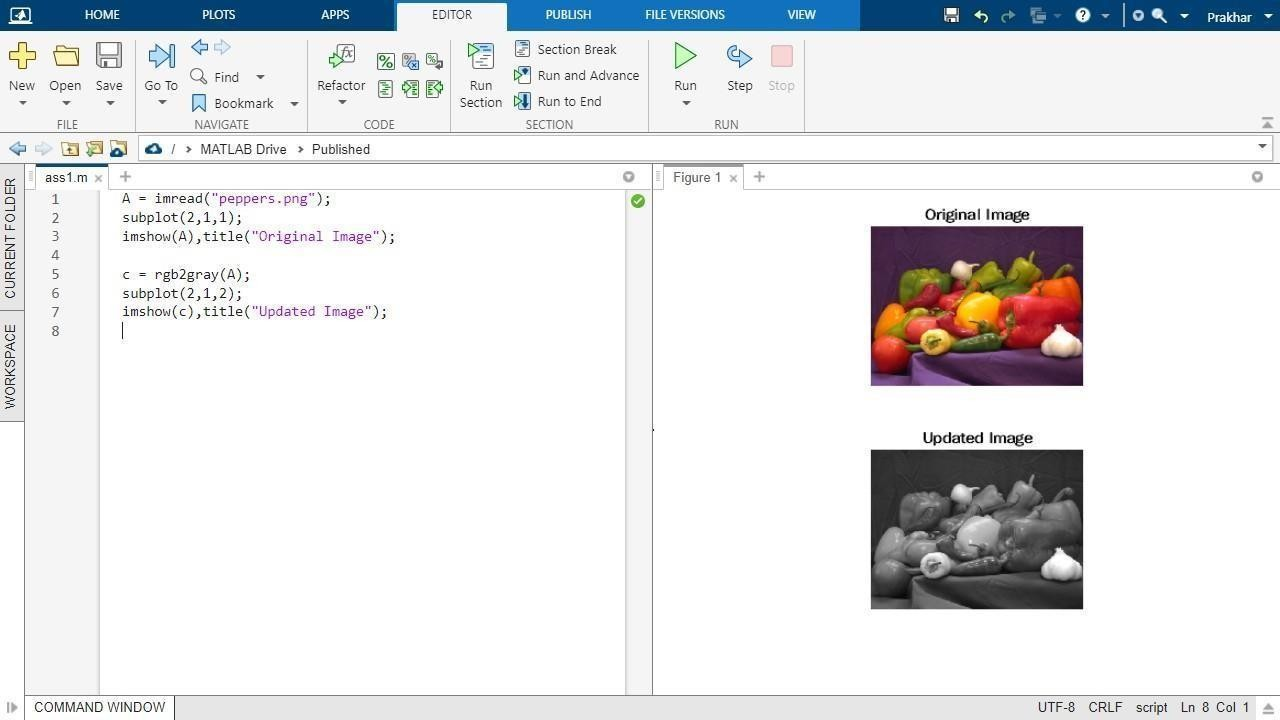
%Resize an image



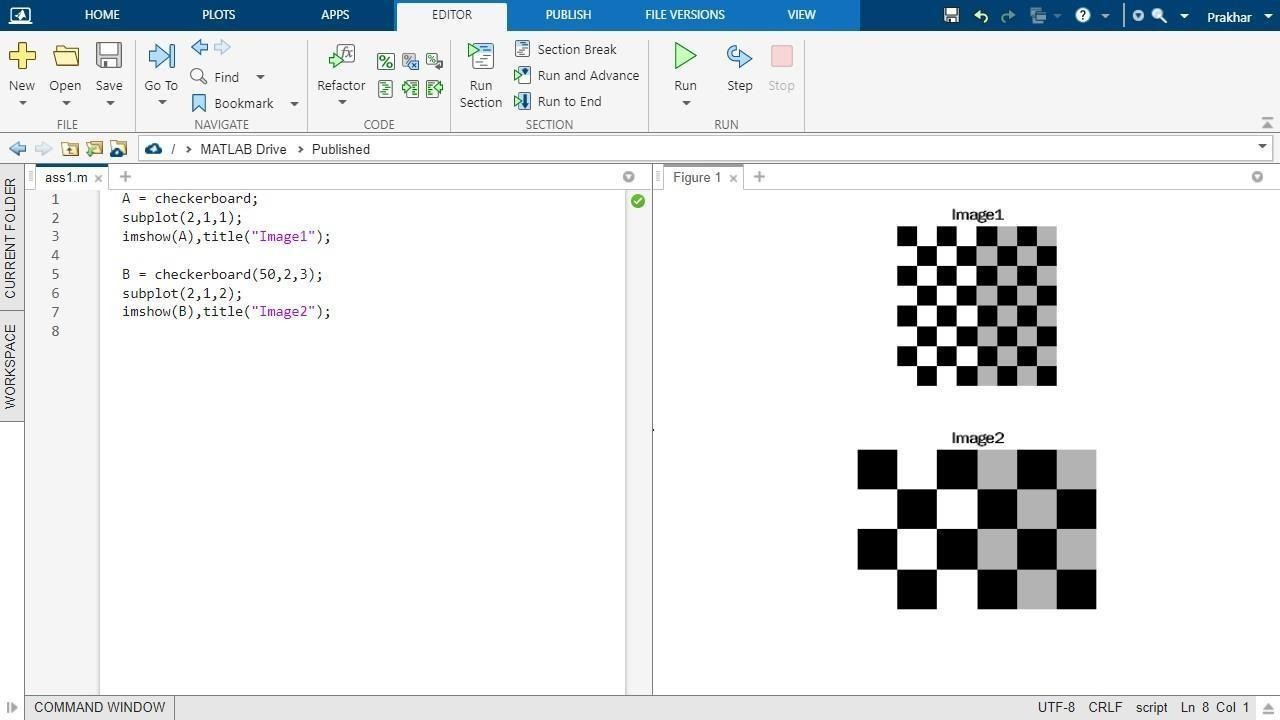
%Extraction of color band of an image.



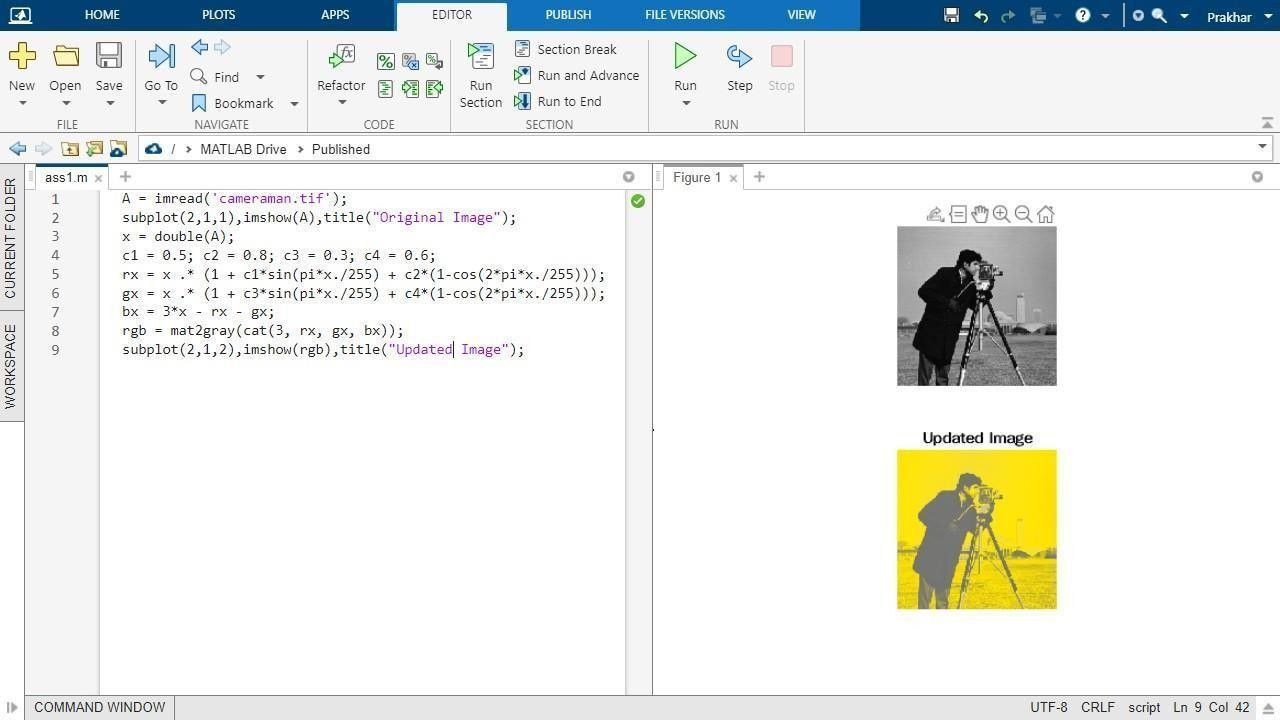
%Image type conversion



%Creating a synthetic image



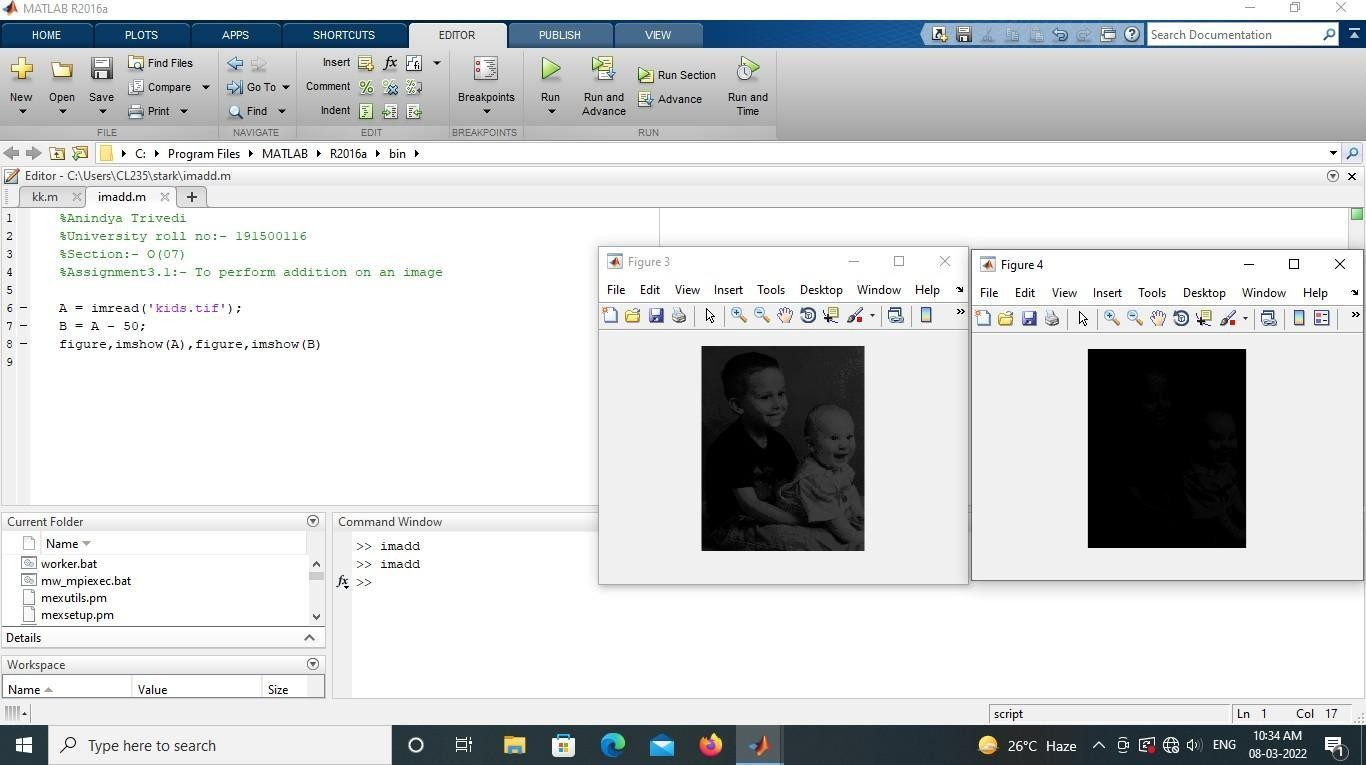
%Psuedocolor Image



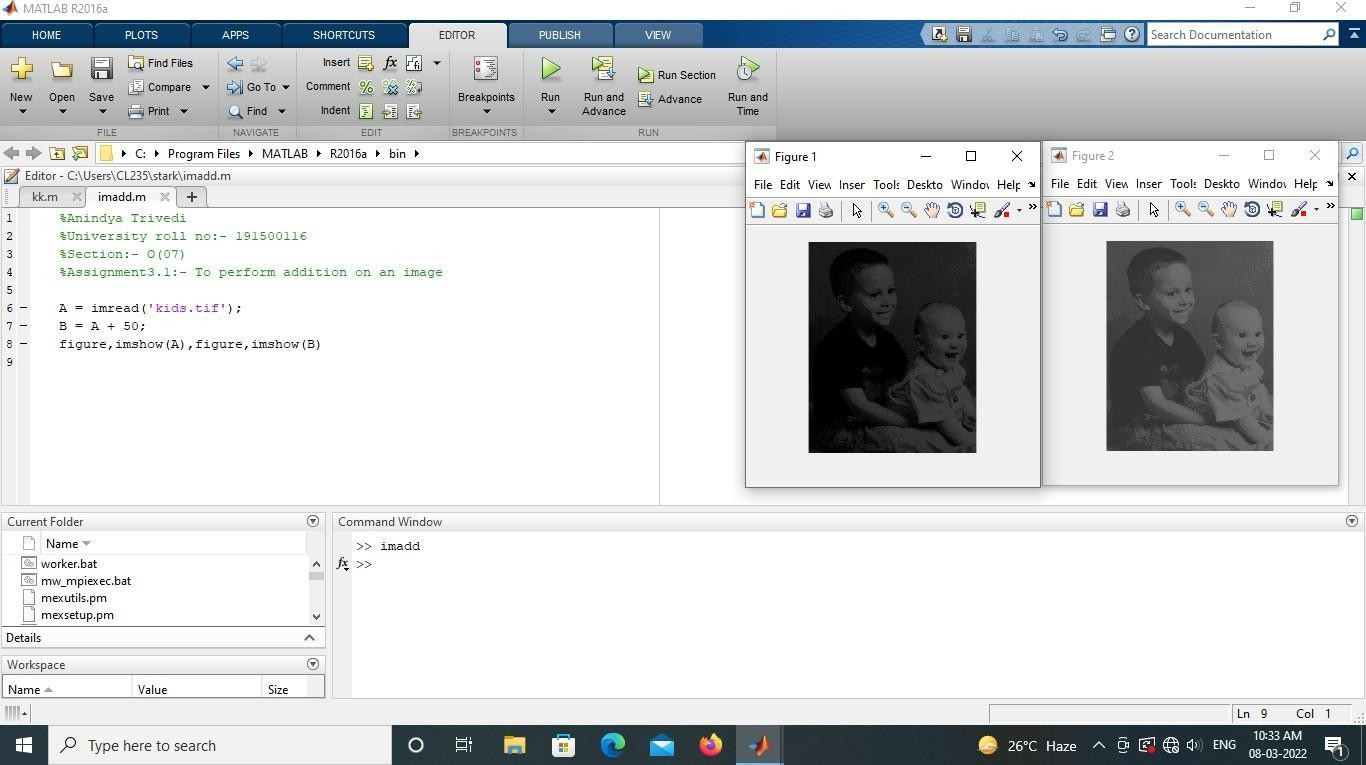
# Assignment 3

## Perform various arithmetic and logical operations on image.

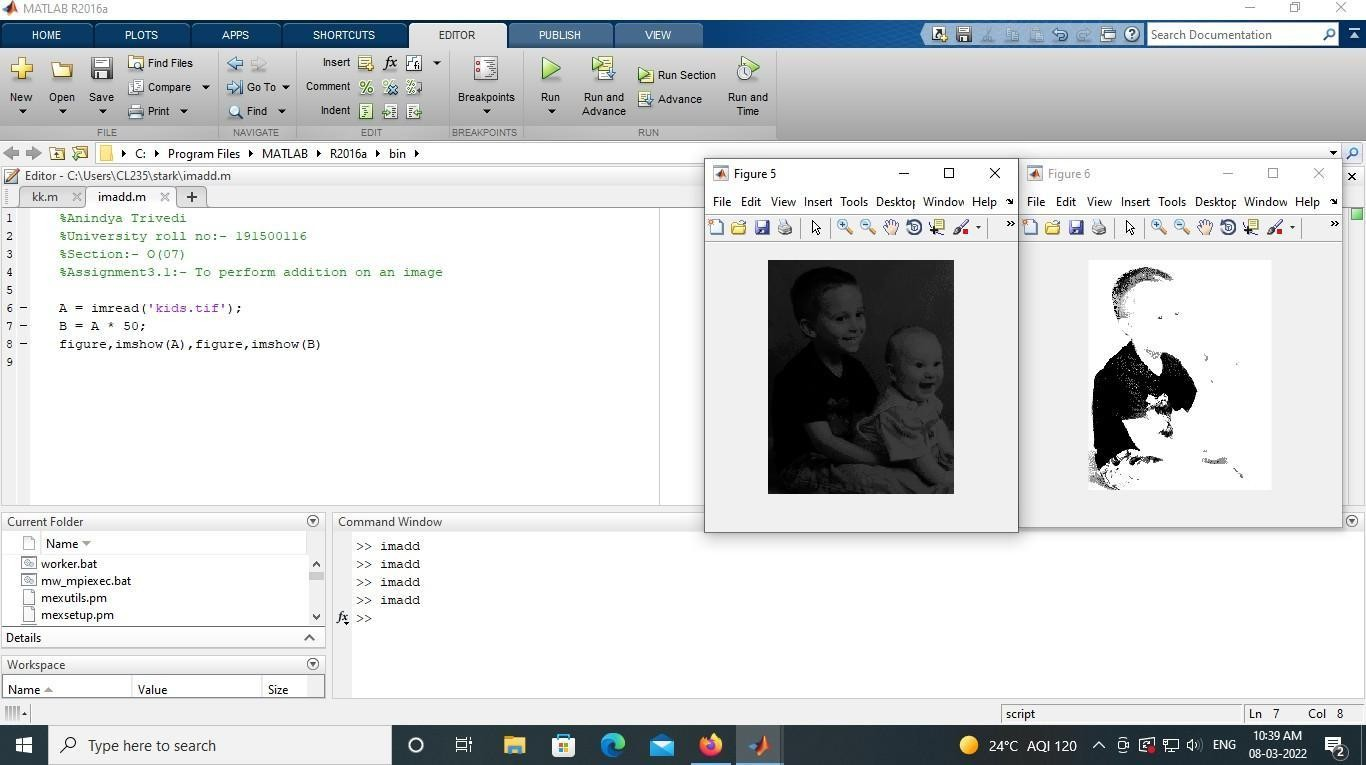
### % 3.1



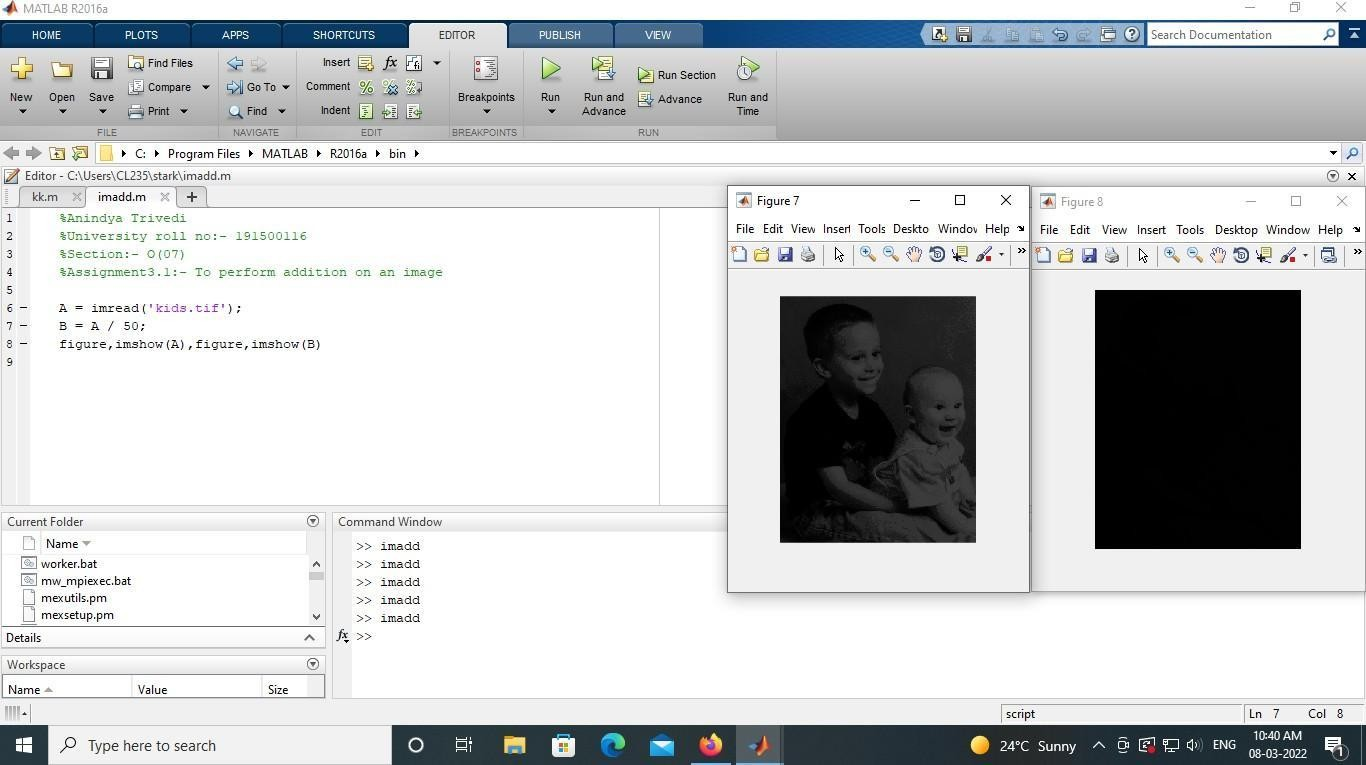
**% 3.2**



### % 3.3



**%3.4**



# Assignment 4

## % Perform the various image enhancement operations:

**% Image negative function,**

## % logarithmic transformation,

**% power law transformation ,**

## % histogramequalization contract stretching ,

**% plot histogram without using imhist function.**

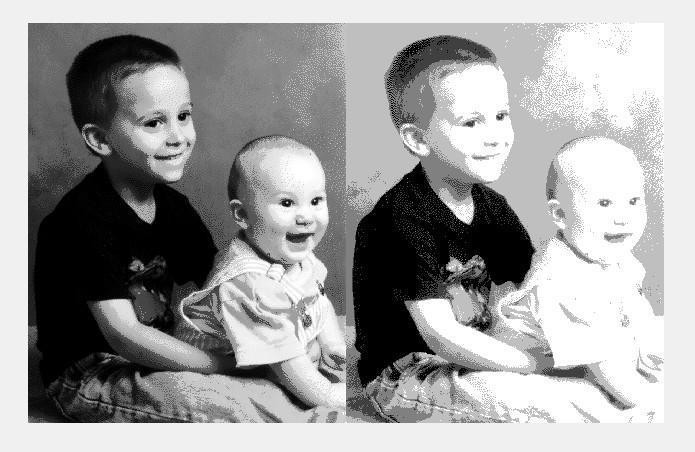
A=imread('kids.tif'); B=A; [M,N]=size(B);

c=input('Enter the constant value, c='); B=double(B); for i=1:M

for j=1:N B(i,j)=c.\*log(1+B(i,j)); end

end B=uint8(B);

imshowpair(A,B ,'montage');



# Assignment 5

## % Perform smoothing using linear(average filter) and order statistics

**% filter(min, max & median) of varying sizes**

## % 5.1 -> Perform smoothing using linear and order statics filter of varying sizes

A = imread('peppers.png');

%imshow(A);

B = rgb2gray(A);

%imshow(B); [m,n] = size(B); B1 = double(B);

S = input('size of filter(odd number):'); f = ones(S); C = (S+1)/2;

for i= C:m-C+1 for j=C:n-C+1 sum = 0; for k=1:S

for l=1:S

sum=sum+B1(i-C+k, j-C+l)\*f(k,l); end

end

B1(i,j)=sum/(S^2); end

end

figure(1),subplot(1,2,1), imshow(B), title 'Original', subplot(1,2,2), imshow(uint8(B1)), title 'Smooth';



## % 5.2 -> Sharpning an image using Laplacian filter L\_4 & L\_8

A=imread('moon.tif'); A=double(A); [m,n]=size(A); f=[0 1 0;1 -4 1;0 1 0] s=A;

for i=2: m-1 for j=2: n-1

sum=0; for k=1:3 for l=1:3

sum=sum+A(i-2+k,j-2+l)\*f(k,l); end

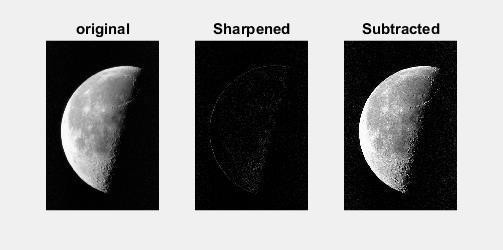
end

end s(i,j)=sum;

end

sm=A-s;

figure(1),subplot(1,3,1), imshow(uint8(A)), title('original'),subplot(1,3,2), imshow(uint8(s)), title('Sharpened'),subplot(1,3,3), imshow(uint8(sm)), title('Subtracted')



# Assignment 6

## %Write a program to create a chess board

A=ones(50,50);

B=zeros(50,50); chess=zeros(400,400); block=A; for i=1:50:351 if(block(1,1)==1)

block=B; else block=A;

end

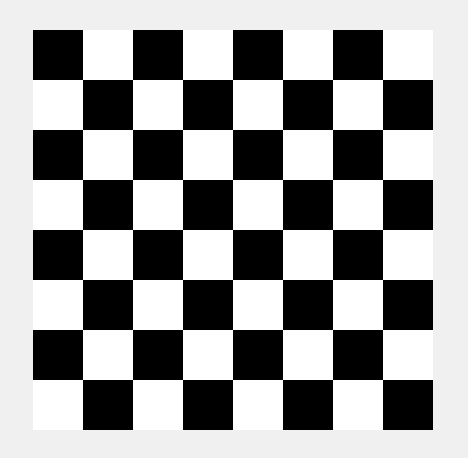
for j=1:50:351 chess(i:i+49,j:j+49)=block; if(block(1,1)==1) block=B; else

block=A;

end end

end

imshow(chess)



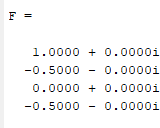
# Assignment 6

## % 6.1 -> Find the DFT of [0 1 2 1]

clear all f(1:4)=[0 1 2 1];

F=zeros(1:4); for u = 1:4 for x = 1:4

F(u)=F(u)+f(x)\*(cos(2\*pi\*(u-1)\*(x-1)/4)-sin(2\*pi\*(u-1)\*(x-1)/4)\*1i); end

end F=1/4.\*F

## % 6.2 -> Find the DFT of [0 1 2 1] using Twiddle Matrix.

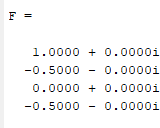
f=[0 1 2 1];

for u= 0:3 for x = 0:3

val = exp(-i\*2\*pi\*u\*x/4); t(u+1, x+1) = val; end

end

F = 1/4.\*t\*f'



## % 6.3 -> Apply 2D Fourier Transform using 1D transformation on an actual square image.

p=imread('peppers.png'); p=rgb2gray(p); [m, n]=size(p);

p=imresize(p,[m, m]); p=double(p); t=zeros(m,m);

%twiddle matrix for u=0:m-1 for x=0:m-1

aa=exp(-i\*2\*pi\*u\*x/m); t(u+1,x+1)=aa; end

end

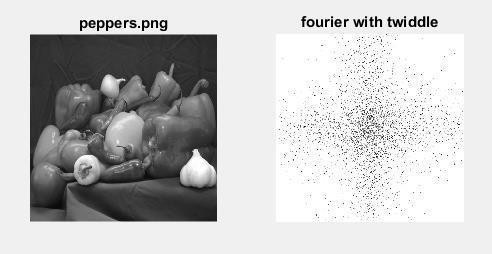
%rowwise FR=zeros(m,m); for x=1:m FR(x,:)=(t\*p(x,:)')';

end

%coumn wise F=zeros(m,m); for y=1:m F(:,y)=t\*FR(:,y);

end F=abs(F);

figure, subplot(1,2,1), imshow(uint8(p)), title 'peppers.png', subplot(1,2,2), imshow(uint8(F)), title 'fourier with twiddle';



# Assignment 7

## % Perform various morphological operation on their applications.

**% 7.1 -> Perform dilation operation**

## % 7.2 -> Perform Erode Operation

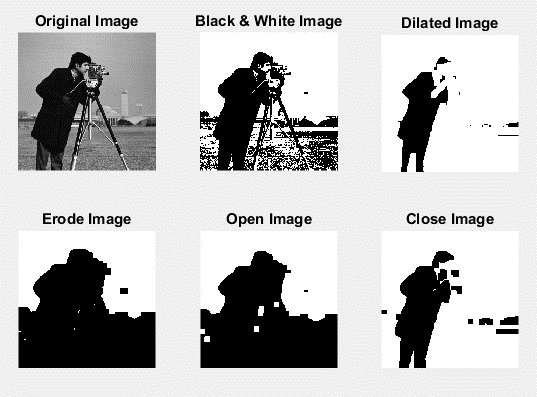
**% 7.3 -> Perform Open Operation**

## % 7.4 -> Perform Close Operation

I=imread('cameraman.tif'); A=im2bw(I);

imshow(A); se=strel('square',9); dilate = imdilate(A, se); erode = imerode(A, se); open = imopen(A, se); close = imclose(A, se);

figure(1),subplot(2,3,1), imshow(I), title 'Original Image', subplot(2,3,2), imshow(A), title 'Black & White Image', subplot(2,3,3), imshow(dilate), title 'Dilated Image', subplot(2,3,4), imshow(erode), title 'Erode Image',subplot(2,3,5), imshow(open), title 'Open Image', subplot(2,3,6), imshow(close), title 'Close Image';



A=imread('cameraman.tif'); B=im2bw(A); se=strel('square',4); C=imerode(B,se); D=imdilate(B,se);

E = D-C;

F=bwmorph(B,'thin'); G=bwmorph(B,'thicken'); H= bwmorph(B, 'skel', 9);

%without using direct function

figure(1),subplot(2,3,1), imshow(A), title 'Original Image',subplot(2,3,2), imshow(B),title 'Black &white Image';subplot(2,3,3), imshow(E) ,title 'Boundary extracted Image',subplot(2,3,4), imshow(F), title 'Thin Image', subplot(2,3,5), imshow(G), title 'Thicken Image',subplot(2,3,6), imshow(H), title 'Skel Image';



# Assignment 8

### % 8.1 :-> To understand segmentation and implement the following tasks :

**% 1. Simple Thresholding**

### % 2. Multiple Thresholding

**% 3. Adaptive Thresholding**

### % 4. Optimal Thresholding

A = imread('coins.png'); level = 0.3; level1 = 0.6;

C = im2bw(A,level); D = im2bw(A,level1);

subplot(3,3,1), imshow(A), title 'Original Image'; subplot(3,3,2), imshow(C),title 'Level 0.3'; subplot(3,3,3), imshow(D),title 'Level 0.6';

temp = A;

[m,n]= find(A<26); for j=1:length(m) tmp (m(j), n(j))=0;

end

[m,n]=find(A>26 & A<=230);

for j=1: length(m) tmp (m(j), n(j))=0.8; end

[m,n]= find(A>230); for j=1:length(m) tmp (m(j), n(j))=0;

end E=im2bw(tmp,0);

subplot(3,3,4), imshow(E),title 'Multiple Thresholding'; level2=graythresh(A); F=im2bw(A,level2);

subplot(3,3,5), imshow(F),title 'Otus-Optimal Thresholding';

%figure(1),subplot(3,3,1), imshow(A), title 'Original Image',subplot(3,3,2), imshow(C),title 'Level 0.3',subplot(3,3,3), imshow(D),title 'Level 0.6', subplot(3,3,4), imshow(E),title 'Multiple Thresholding';



**% 8.2 -> Read the image coins.png prewitt,sobel ansd Robert operation using function**

I = imread('coins.png'); subplot(2, 2, 1), imshow(I); title('Original Image');

% Robert Edge Detection L = edge(I, 'Roberts'); subplot(2, 2, 2), imshow(L); title('Robert');

% Prewitt Edge detection K = edge(I, 'Prewitt'); subplot(2, 2, 3), imshow(K); title('Prewitt');

% Sobel Edge Detection J = edge(I, 'Sobel');

subplot(2, 2, 4), imshow(J); title('Sobel');



# Assignment 9

**% Read the image coins.png**

**% prewitt,sobel and Robert operation without using function**

clc I=imread('coins.png'); X=I;

I=uint8(I);

I=double(I);

F=zeros(size(I));

### %ROBERTS OPERATOR

Mx = [1 0; 0 -1];

My = [0 1; -1 0];

for i = 1:size(I,1) -1

for j = 1:size(I,2) -1

Gx = sum(sum(Mx.\*I(i:i+1, j:j+1)));

Gy = sum(sum(My.\*I(i:i+1, j:j+1)));

F(i, j) = sqrt(Gx.^2 +Gy.^2);

end

end

F=uint8(F); thresholdValue

=50; O=max(F,

thresholdValue);

O(O == round(thresholdValue))=50; O=im2bw(O);

**%PREWITT OPERATOR**

Mx1=[-1 0 1; -1 0 1; -1 0 1];

My1=[-1 -1 -1; 0 0 0; 1 1 1];

for i =1:size(I,1) -2

for j =1:size(I,2) -2

Gx =sum(sum(Mx1.\*I(i:i+2, j:j+2)));

Gy =sum(sum(My1.\*I(i:i+2, j:j+2))); F(i+1, j+1)= sqrt(Gx.^2 +Gy.^2);

end

end

F=uint8(F); thresholdValue =50;

O2=max(F, thresholdValue);

O2(O2 == round(thresholdValue))=50; O2=im2bw(O2);

**%SOBEL OPERATOR**

Mx=[-1 0 1; -2 0 2; -1 0 1];

My=[-1 -2 -1; 0 0 0; 1 2 1];

for i =1:size(I,1) -2

for j =1:size(I,2) -2

Gx =sum(sum(Mx.\*I(i:i+2, j:j+2)));

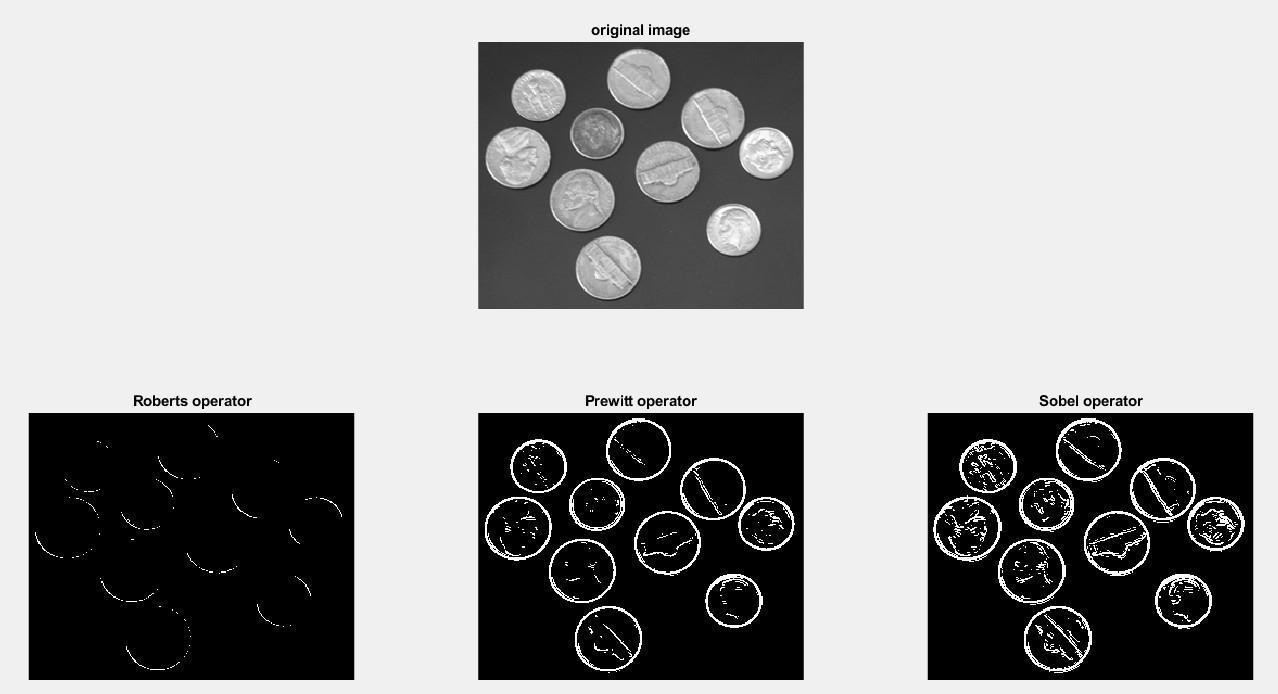
Gy =sum(sum(My.\*I(i:i+2, j:j+2))); F(i+1, j+1)= sqrt(Gx.^2 +Gy.^2); end

end

F=uint8(F); thresholdValue =50; O3=max(F, thresholdValue);

O3(O3 == round(thresholdValue))=50; O3=im2bw(O3);

subplot(2,3,2), imshow (X), title('original image'), subplot(2,3,4), imshow (O), title('Roberts operator'), subplot(2,3,5), imshow (O2), title('Prewitt operator'), subplot(2,3,6), imshow (O3), title('Sobel operator');



# Assignment 10

## % Write a program to find the no. of coin in given image

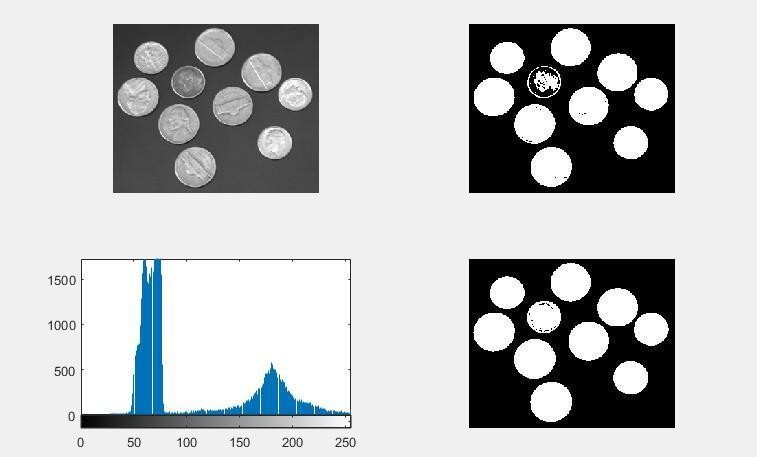
**% Write a program to find canny edge detection**

function ret = CountCoins(img) img=imread('coins.png'); subplot(2,2,1);

imshow(img); subplot(2,2,2); imgBW=im2bw(img); imshow(imgBW) subplot(2,2,3); imhist(img); subplot(2,2,4); imgZ=zeros(size(img)); imgZ(img>100)=1; imshow(imgZ);

ret = round(sum(imgBW(:)) /2100); imgConn=bwconncomp(imgZ);

ret = imgConn.NumObjects; end



**% Write a program to find canny edge detection**

I = imread('coins.png'); subplot(1, 2, 1), imshow(I); title('Original Image');

% Canny Edge Detection N = edge(I, 'Canny');

subplot(1, 2, 2), imshow(N); title('Canny');

