DIP LAB FILE

prince kumar section :-3z(36) univ roll:- 2215001319

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Experment no** | **Experment Name** | **Date** |
| **Experment no:- 1** | write a program to import digital images and perform logical OR,AND, NOT opration . |  |
| **Experment no:- 2** | Write a program to import Digital Images and reduce the size of the image (50%). |  |
| **Experment no:- 3** | To Select a Digital Image and perform Basic Intensity Transformation "Image Negative” |  |
| **Experment no:- 4** | To Select a Digital Image and perform Basic Intensity Transformation "Logarithmic Transformation". |  |
| **Experment no:- 5** | To import a Digital Image and apply an image averaging filter,apply a median filter and apply a min and max filter. |  |
| **Experment no:- 6** | To perform edge detection. |  |
| **Experment no:- 7** | Coswave and sinwave |  |
| **Experment no:- 8** | To import an image and use Blaring effect . |  |
| **Experment no:- 9** |  |  |
| **Experment no:- 10** |  |  |
| **Experment no:- 11** |  |  |

**Experment no:- 1**

**Object:-** write a program to import digital images and perform logical OR,AND, NOT opration .

**Theory:-**

**OR:-** A **Logical OR Gate** is a fundamental digital logic gate that operates based on the principle of "OR" in Boolean logic. It has two or more input signals and produces an output signal based on the following rule:

* The output is **HIGH (1)** if **any** one or more of the inputs are HIGH (1).
* The output is **LOW (0)** only if **all** of the inputs are LOW (0).

### Truth Table for a 2-input OR Gate:

| Input A | Input B | Output (A OR B) |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

### Example:

* **A = 0, B = 1 → Output = 1**
* **A = 1, B = 0 → Output = 1**
* **A = 1, B = 1 → Output = 1**

**AND:-** An AND Gate is a fundamental digital logic gate that operates based on the principle of logical conjunction (AND) in Boolean algebra. It takes two or more input signals and produces an output signal according to the following rule:

* The output is HIGH (1) only if all of the inputs are HIGH (1).
* The output is LOW (0) if any of the inputs is LOW (0).

**Truth Table for a 2-input AND Gate:**

| Input A | Input B | Output (A AND B) |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**Example:**

* A = 0, B = 0 → Output = 0
* A = 1, B = 0 → Output = 0
* A = 1, B = 1 → Output = 1

**NOT:-**

A **NOT Gate**, also known as an **inverter**, is a basic digital logic gate that outputs the **inverse** (opposite) of the input signal. It has only one input and one output. The primary function of a NOT gate is to **flip** or **invert** the state of its input:

* If the input is **HIGH (1),** the output will be **LOW (0)**.
* If the input is **LOW (0),** the output will be **HIGH (1).**

### Truth Table for a NOT Gate:

| Input | Output (NOT A) | |
| --- | --- | --- |
| 0 | 1 |
| 1 | 0 |

### Example:

* **Input = 0 → Output = 1**
* **Input = 1 → Output = 0**

**CODE:-**

**AND :-** a = imread('cameraman.tif');

b = imread('hel.jpg');

a1 = im2bw(a);

subplot(1, 3, 1);

imshow(a1);

title('Binary Image a');

b1 = im2bw(b);

subplot(1, 3, 2);

imshow(b1);

title('Binary Image b');

C = and(a1, b1);

subplot(1, 3, 3);

imshow(C);

title('AND Operation Result');

**OR:-**

a = imread('cameraman.tif');

b = imread('hel.jpg');

a1 = im2bw(a); %

subplot(1, 3, 1);

imshow(a1);

title('Binary Image a');

b1 = im2bw(b);

subplot(1, 3, 2);

imshow(b1);

title('Binary Image b');

C = or(a1, b1);

subplot(1, 3, 3);

imshow(C);

title('OR Operation Result');

**NOT:-**

a = imread('cameraman.tif');

a1 = im2bw(a);

subplot(1, 2, 1);

imshow(a1);

title('Binary Image a');

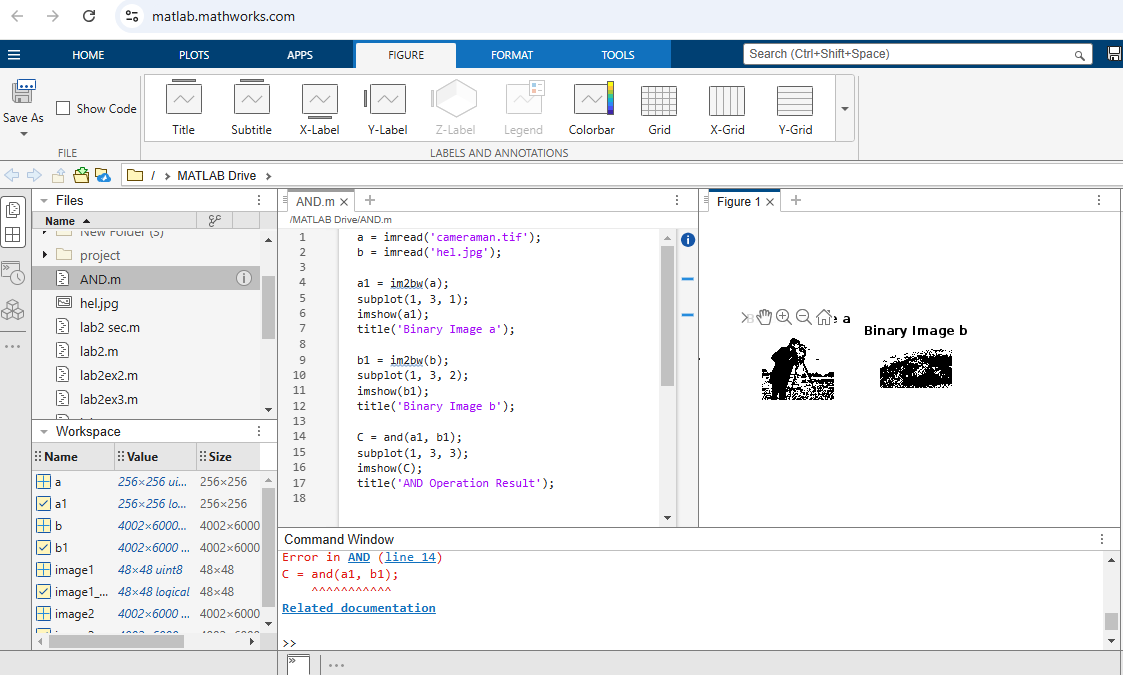
C = not(a1);

subplot(1, 2, 2);

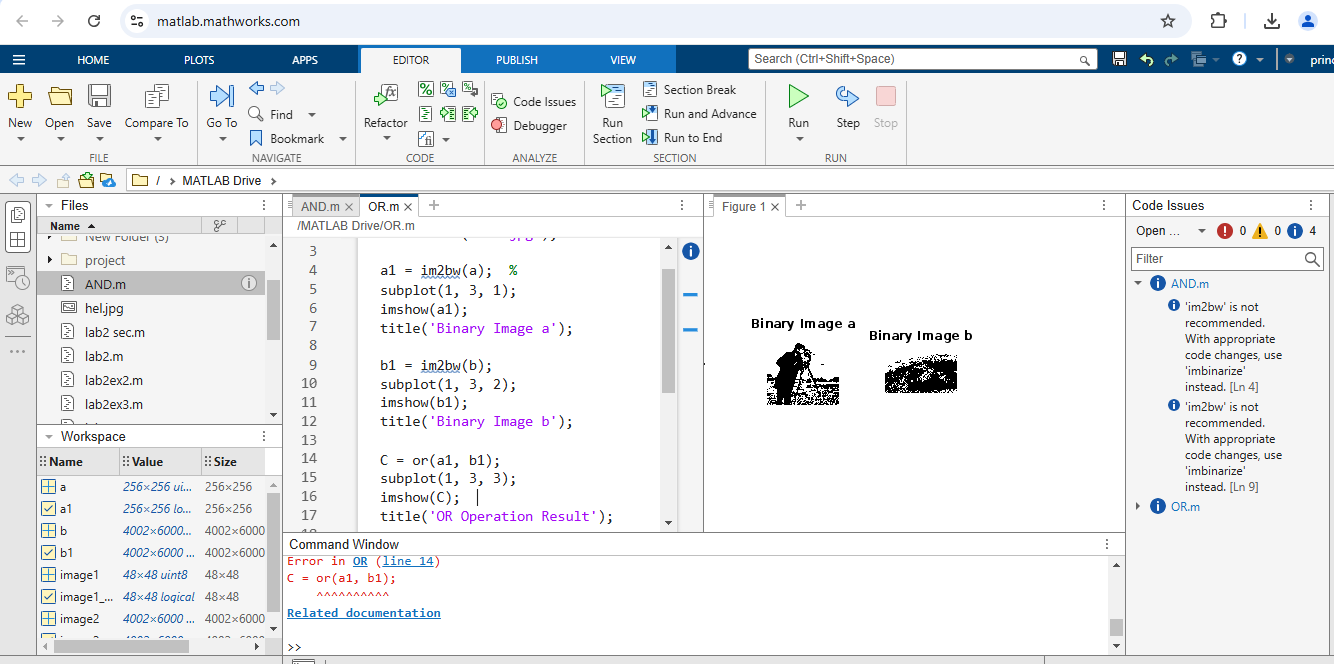
imshow(C);

title('NOT Image');

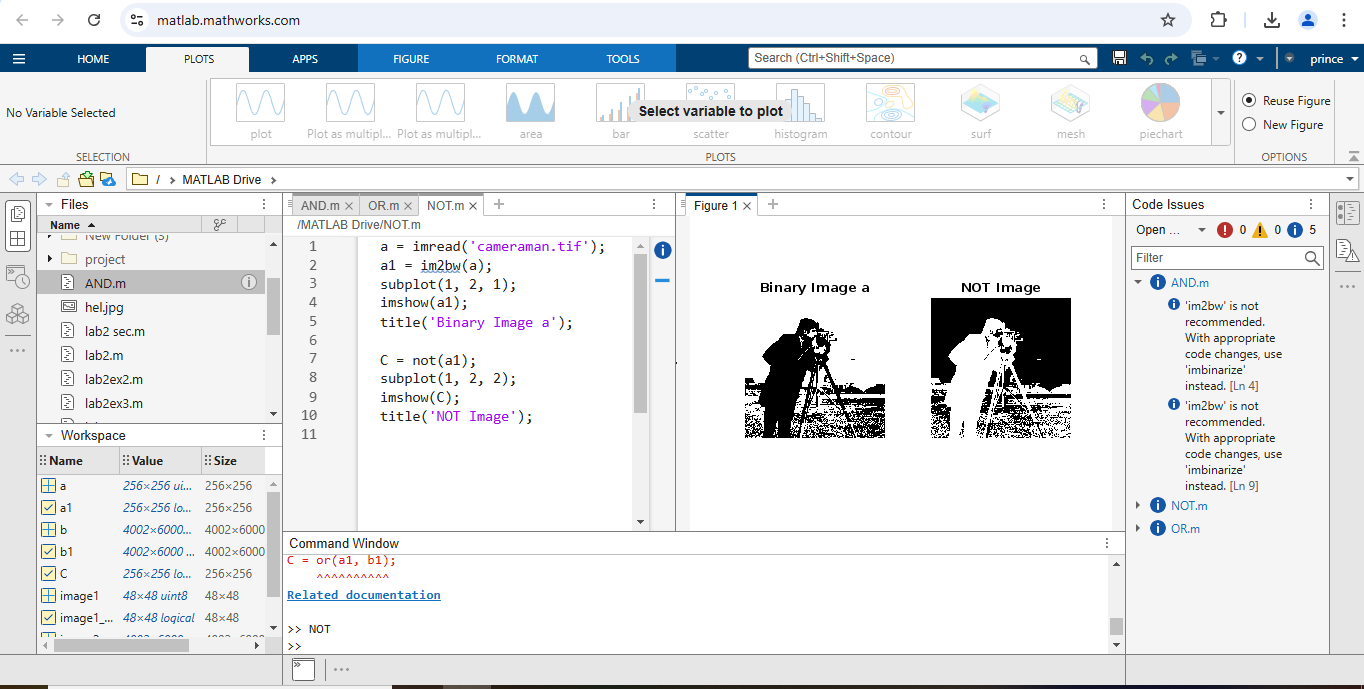
**image:-**

**AND:- **

**OR:-**

****

**NOT:-**

****

**Experment no:- 2**

**Object:-** Write a program to import Digital Images and reduce the size of the image (50%).

**CODE:-**

x = imread('hel.jpg');

y = imresize(x, 0.5);

subplot(1, 2, 1);

imshow(x);

title('Original Image');

subplot(1, 2, 2);

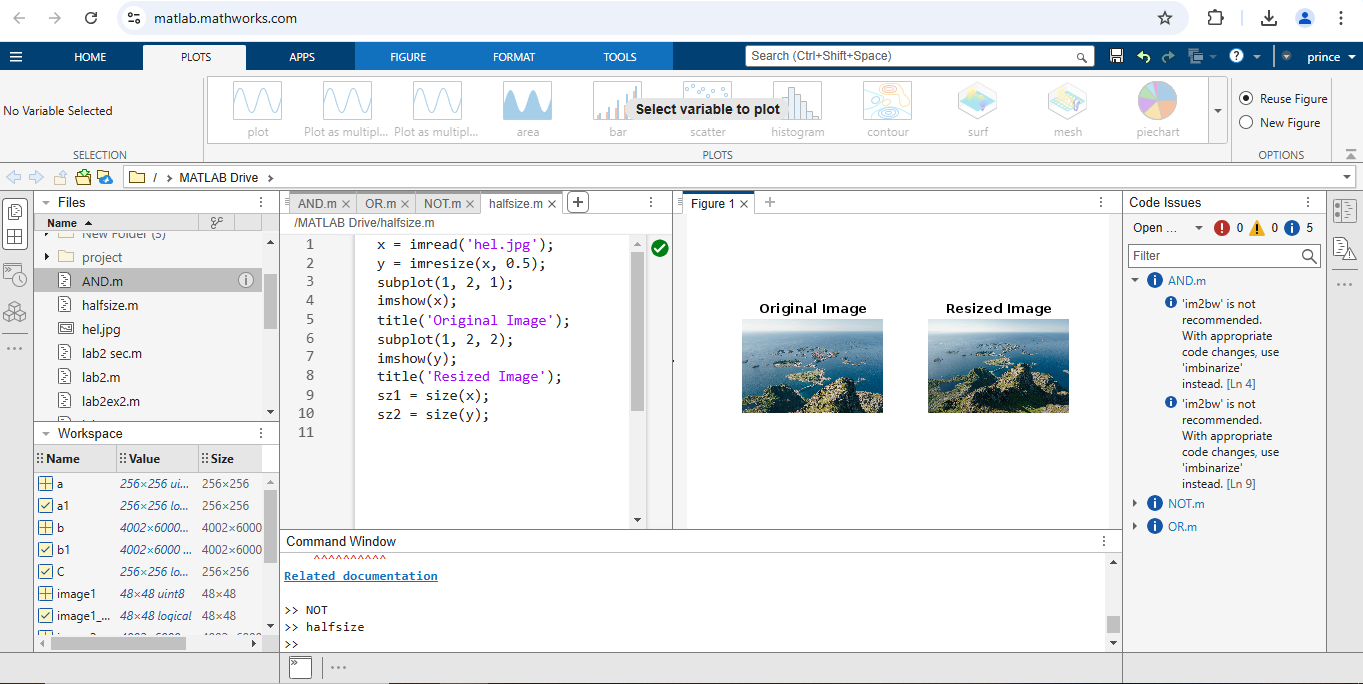
imshow(y);

title('Resized Image');

sz1 = size(x);

sz2 = size(y);

**image:-**

****

**Experiment:-3**

**Object:-** To Select a Digital Image and perform Basic Intensity Transformation "Image Negative”.

**Code:-**

y = imread('hel.jpg');

subplot(1, 2, 1),

imshow(y);

title('Original image');

x = 2^8;

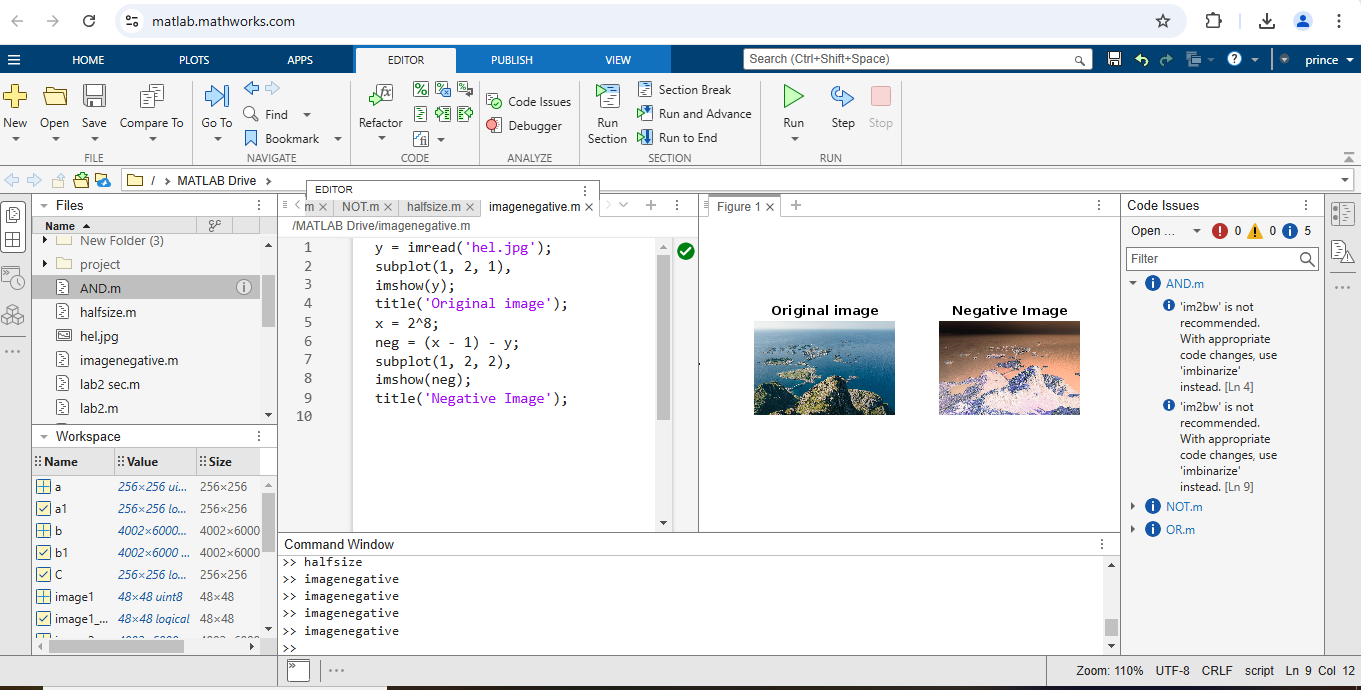
neg = (x - 1) - y;

subplot(1, 2, 2),

imshow(neg);

title('Negative Image');

**image:-**

****

**Experiment no:-4**

**Object :-** To Select a Digital Image and perform Basic Intensity Transformation "Logarithmic Transformation".

**CODE:-**

img = imread('hel.jpg');

x = double(img);

y = 1;

z = y \* log(1 + x);

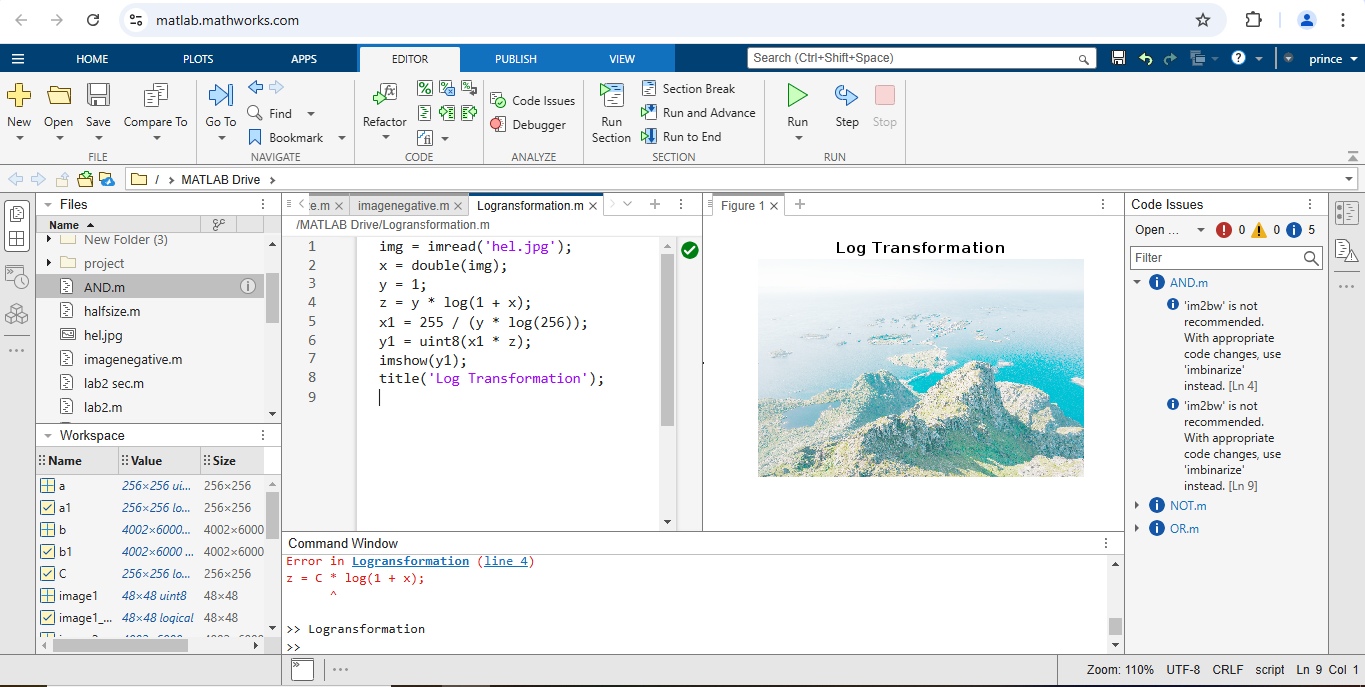
x1 = 255 / (y \* log(256));

y1 = uint8(x1 \* z);

imshow(y1);

title('Log Transformation');

**Image:-**

****

**Experment:-5**

**Object:-** To import a Digital Image and apply an image averaging filter,apply a median filter and apply a min and max filter.

**CODE:-**

**1.A Digital Image and apply an image averaging filter**

a = imread('hel.jpg');

subplot(1, 2, 1);

imshow(a);

title('Original Image');

x = fspecial('average', [4, 4]);

y = imfilter(a, x);

subplot(1, 2, 2);

imshow(y);

title('4x4 Average Filter');

**2.A Digital Image and apply a median filter**

x = imread('hel.jpg');

subplot(1, 3, 1);

imshow(x);

title('Original Image');

y = imnoise(x, 'salt & pepper', 0.02);

subplot(1, 3, 2);

imshow(y);

title('Noise Image');

z = medfilt2(y);

subplot(1, 3, 3);

imshow(z);

title('Filtered Image');

**3.A Digital Image and apply a min and max filter.**

Original = imread('hel.jpg');

BW = im2bw(Original, 0.6);

minf = @(x) min(x(:));

maxf = @(x) max(x(:));

min\_Image = nlfilter(BW, [3 3], minf);

max\_Image = nlfilter(BW, [3 3], maxf);

subplot(2, 2, 1);

imshow(BW);

title('Original');

subplot(2, 2, 2);

imshow(min\_Image);

title('Min');

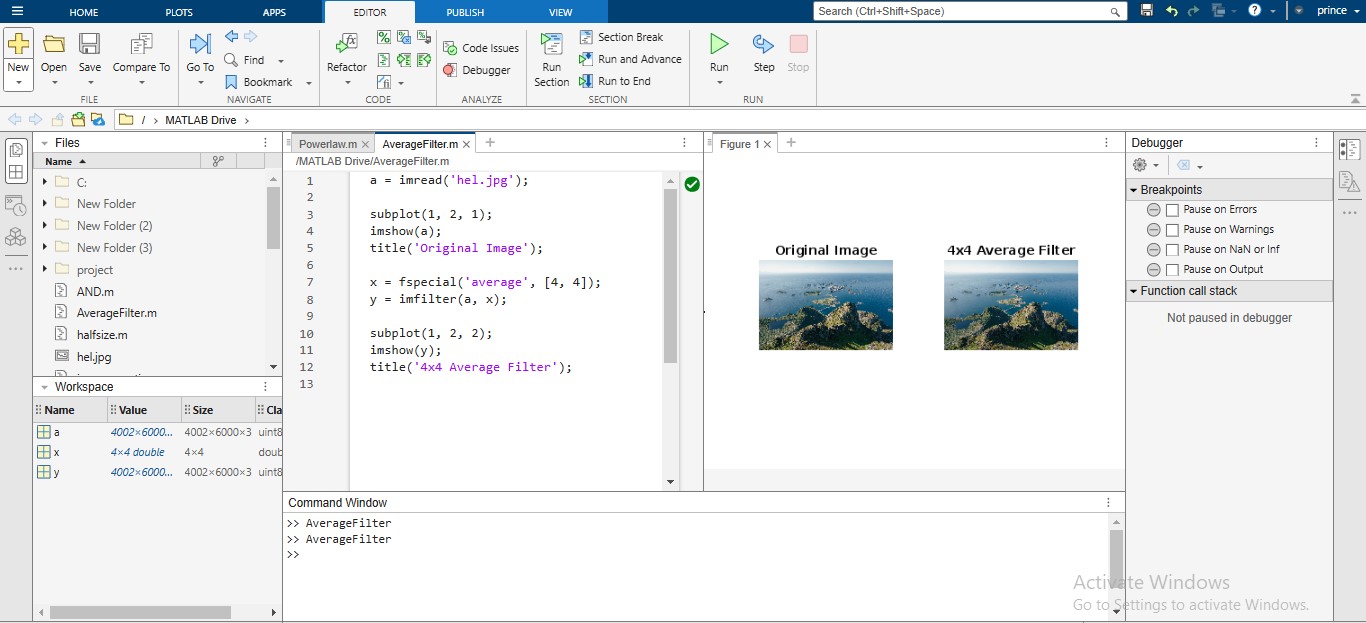
subplot(2, 2, 3);

imshow(max\_Image);

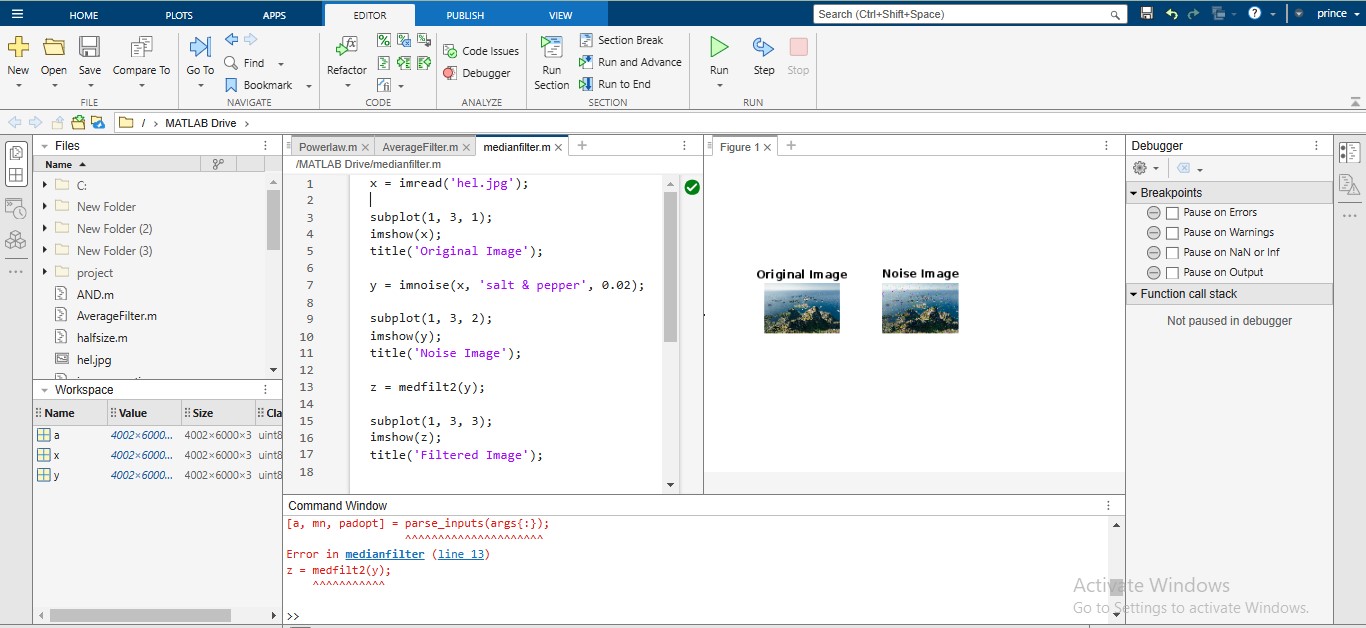
title('Max');

**Image:-**

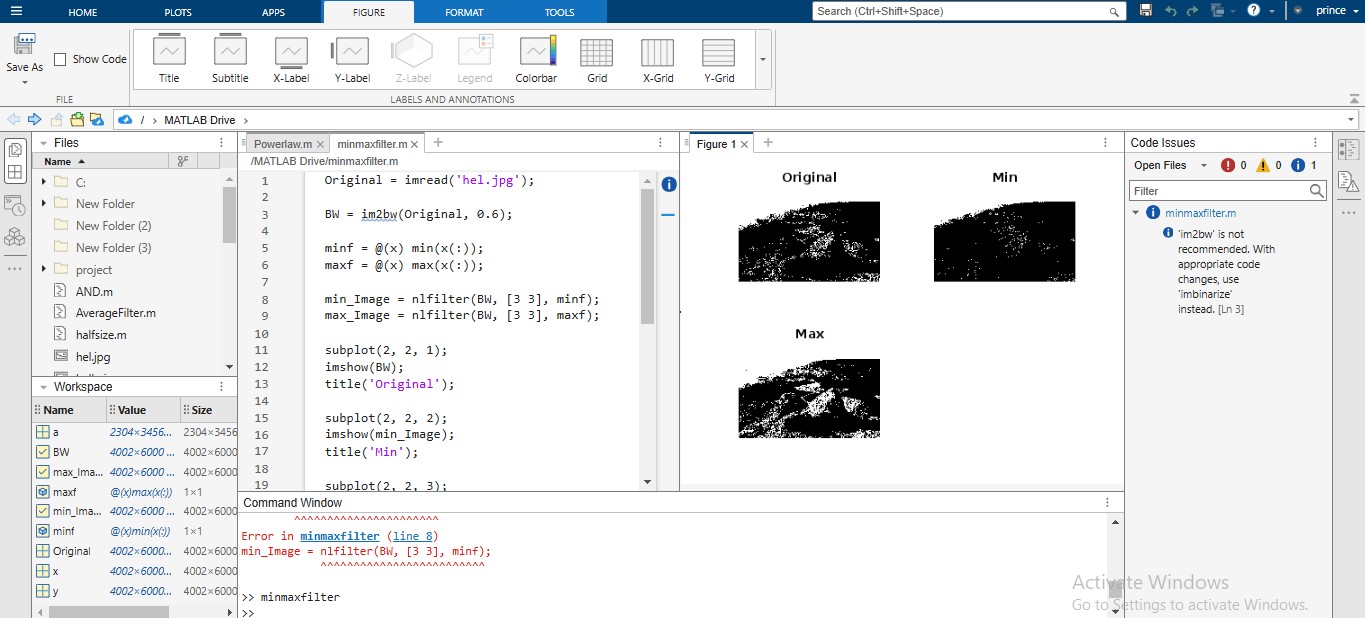
**1.A Digital Image and apply an image averaging filter**

****

**2.A Digital Image and apply a median filter**



**3.A Digital Image and apply a min and max filter.**

****

**eExperment:-6**

**Object:-** To perform edge detection.

**Code:-**

% to perform edge dectection

a = imread("hel.jpg");

a1 = rgb2gray(a);

subplot(2,5,1);

imshow(a1);

b=edge(a1,'sobel');

subplot(2,4,2);

imshow(b);

c=edge(a1,'prewitt');

subplot(2,4,3);

imshow(c);

d=edge(a1,'roberts');

subplot(2,4,4);

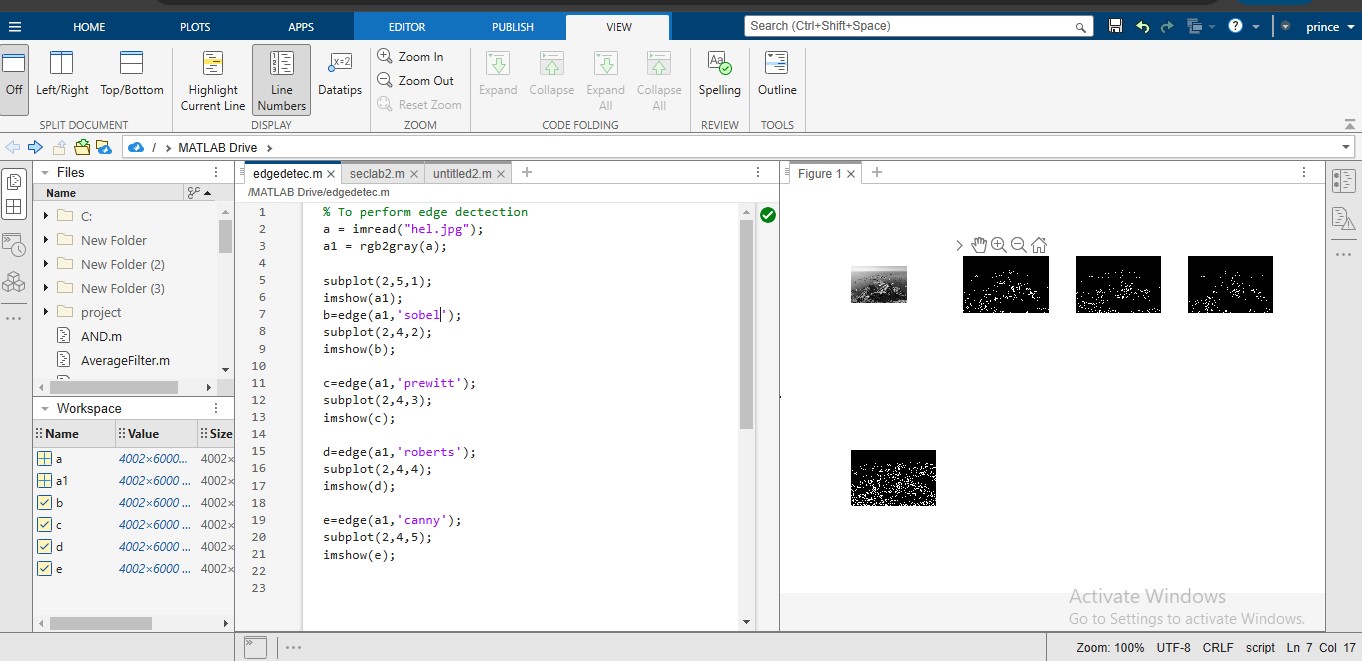
imshow(d);

e=edge(a1,'canny');

subplot(2,4,5);

imshow(e);

**Image:-**



**eExperment:-7**

**Object:-** coswave and sinwave

**CODE:-**

clc;

clear all;

close all;

a=5;

x=[1 2 5 1];

y=[1 2 3; 4 5 6 ; 7 8 9];

z=y';

b=zeros(6);

c=ones(3);

d=rand(6);

t=1:10;

t1=1:2:10;

k=2:-0.5:-1;

e=[1:5; 8:12];

a1 =[3 5 3; 6 8 2; 2 7 3];

a1(3,1);

a1(2:3,2);

c1=[1:2:100];

t2=[0:pi/50:6\*pi];

y2=sin(t2);

plot(t2,y2);

t3=[0:pi/50:6\*pi];

y3=cos(t3);

plot(t3,y3);

t4=[0:pi/25:4\*pi];

y4=sin(t4);

plot(t4,y4);

title('sinewave');

xlabel('time');

ylabel('amplitude');

t5=[0:pi/25:4\*pi];

y5=cos(t5);

plot(t5,y5);

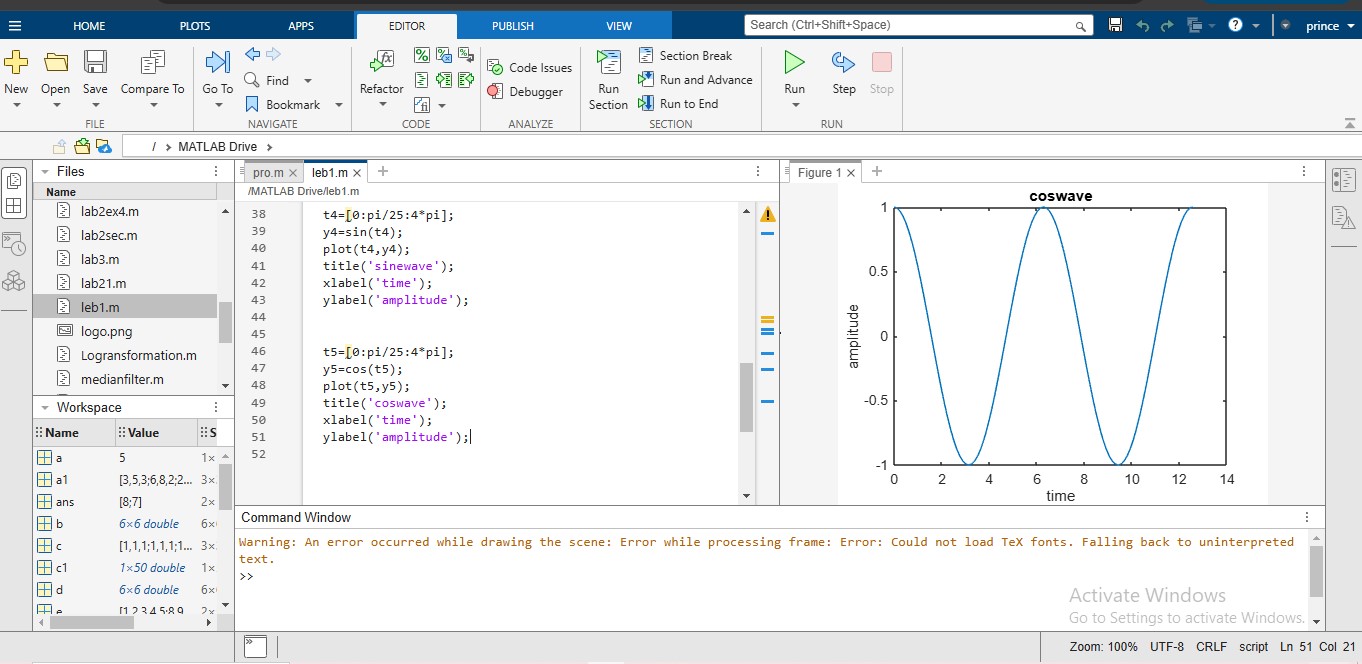
title('coswave');

xlabel('time');

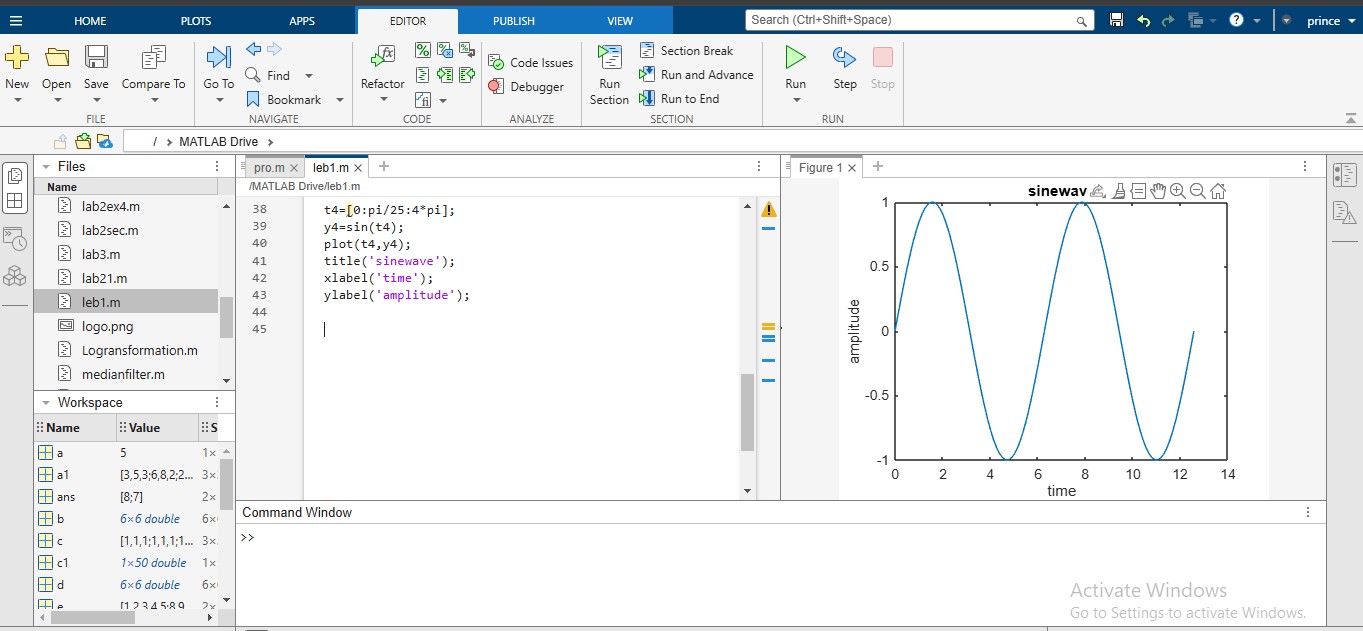
ylabel('amplitude');

**image:-**

**# coswave**



**#Sinwave**

****

**eExperment:-8**

**object:-** To import an image and use Blaring effect **.**

**Code:-**

j=imread("logo.png");

j1=rgb2gray(j);

n=25\*randn(size(j1));

j2=n+double(j1);

imtool(j,[]);

imtool(j1,[]);

imtool(j2,[]);

Lap=[0 -1 0; -1 4 -1; 0 -1 0];

j3=conv2(j2, Lap,'same');

imtool(j3,[]);

Lap1=[1 1 1; 1 -8 1; 1 1 1];

j4=conv2(j2, Lap1,'same');

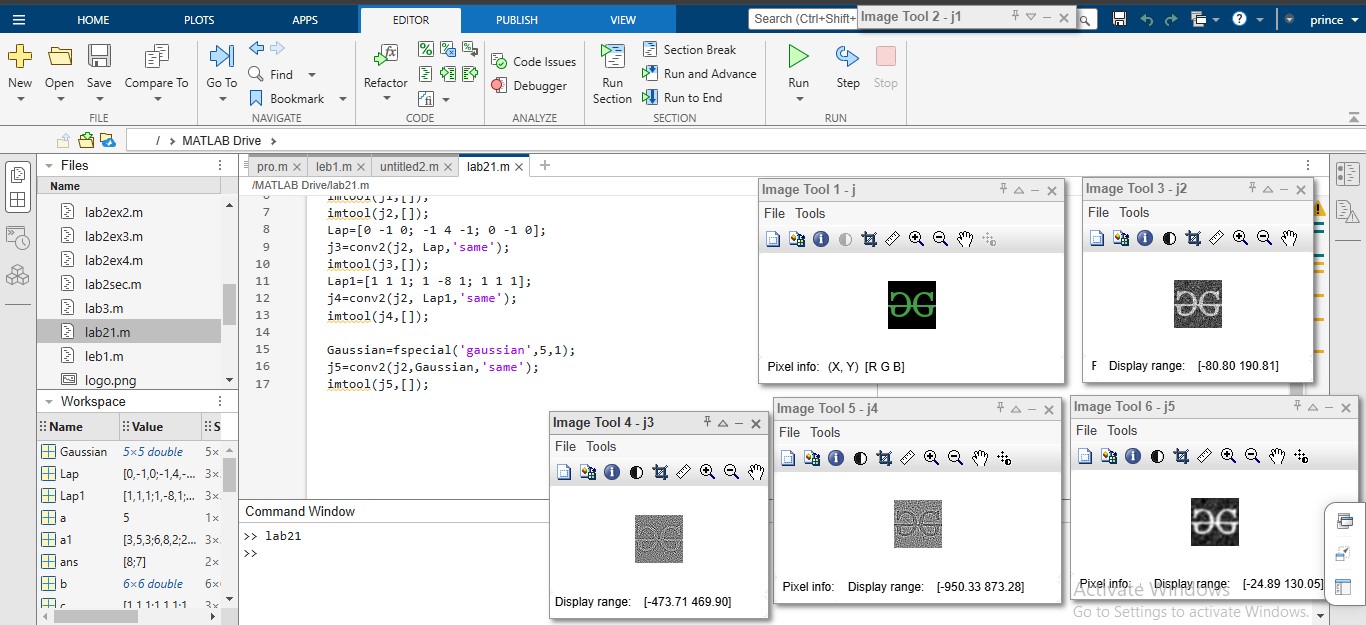
imtool(j4,[]);

Gaussian=fspecial('gaussian',5,1);

j5=conv2(j2,Gaussian,'same');

imtool(j5,[]);

**Image:-**

****