FDVIP Practical 1 to 8

Name:- Krishna Mundada

Roll No:- 45

Batch:- E3

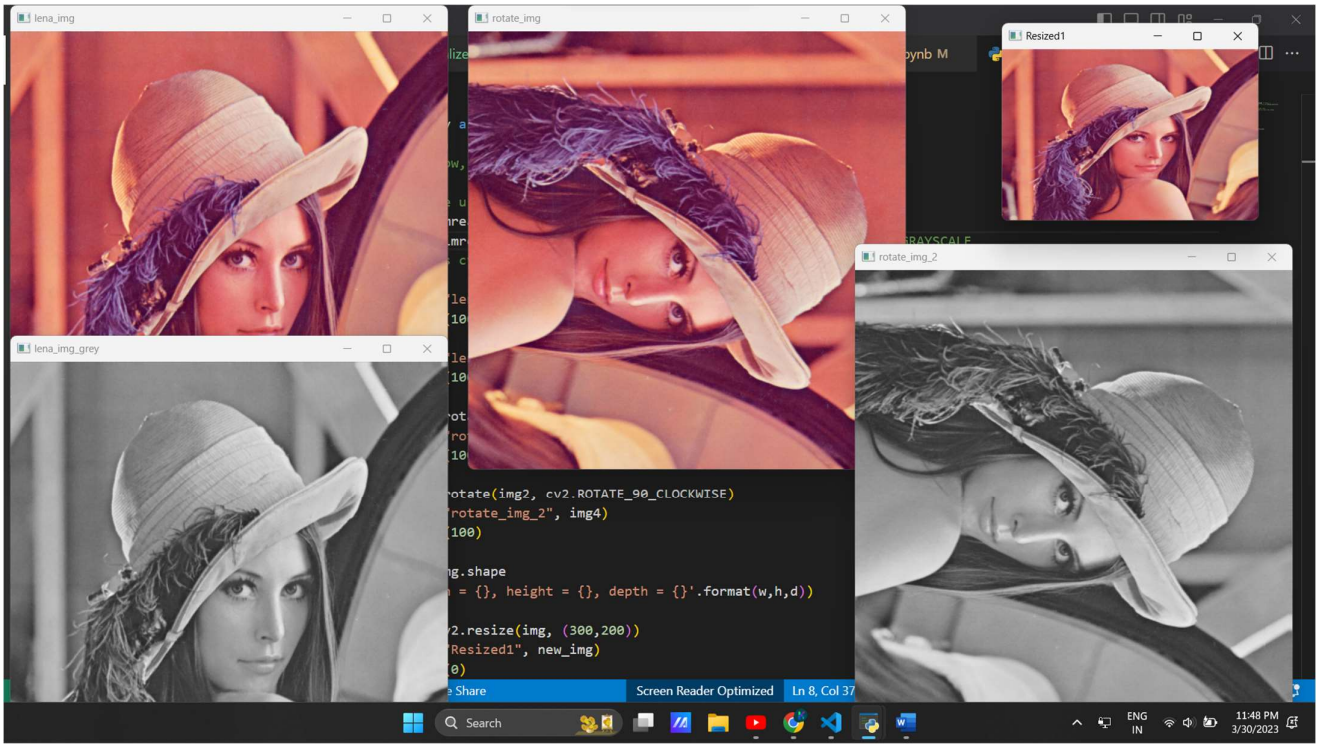
Branch:- AIML

Subject:- Fundamental of Video and Image Processing

Practical 1:-

```
import cv2
import numpy as np
## Read, Show, Greyscale, Rotate, Shape, Resize
# load image using imread
img = cv2.imread(r"assets\Lenna.png", cv2.IMREAD_COLOR)  # Flag = 1
img2 = cv2.imread(r"assets\Lenna.png", 0)   # Flag = 0 is cv2.IMREAD_GRAYSCALE
# Flag -1 is cv2.IMREAD_UNCHANGED
cv2.imshow("lena_img", img)   # lena_img is the window name
cv2.waitKey(100)     # Waits for a key to be pressed to close the window
cv2.imshow("lena_img_grey", img2)
cv2.waitKey(100)
img3 = cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)
cv2.imshow("rotate_img", img3)
cv2.waitKey(100)
img4 = cv2.rotate(img2, cv2.ROTATE_90_CLOCKWISE)
cv2.imshow("rotate_img_2", img4)
cv2.waitKey(100)
(h,w,d) = img.shape
print('width = {}, height = {}, depth = {}'.format(w,h,d))
new_img = cv2.resize(img, (300,200))
cv2.imshow("Resized1", new_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

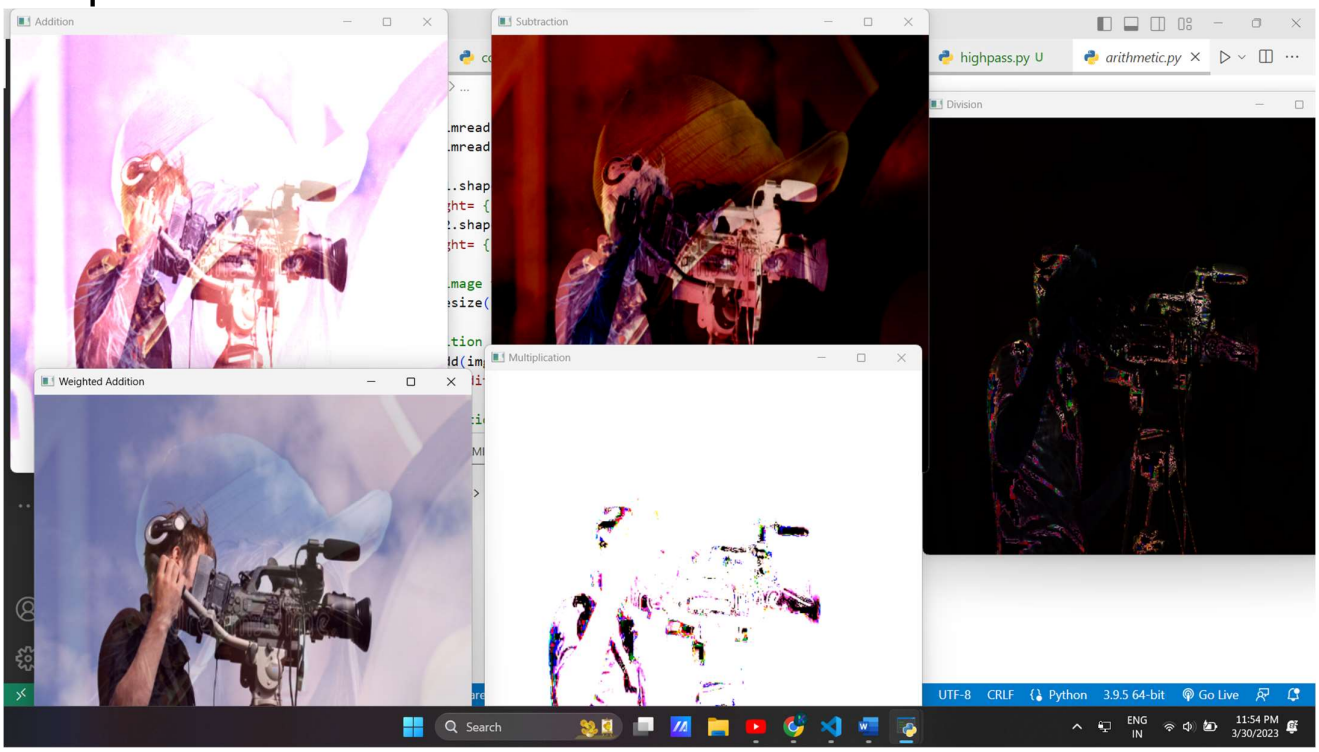
OutPut:-



Practical 2:-

```
import cv2
img1 = cv2.imread('./assets/Lenna.png', 1)
img2 = cv2.imread('./assets/camera_man.jpg', 1)
(h,w,d)=img1.shape
print(f"Height= {h}\nWidth= {w}\nDepth= {d}")
(h,w,d)=img2.shape
print(f"Height= {h}\nWidth= {w}\nDepth= {d}")
#resize of image to add them
new = cv2.resize(img2, (512, 512))
#normal addition
add = cv2.add(img1,new)
cv2.imshow('Addition', add)
#Weighted addition
addw = cv2.addWeighted(img1, 0.2, new, 0.8, 0)
cv2.imshow('Weighted Addition', addw)
#normal subtraction
sub = cv2.subtract(img1,new)
cv2.imshow('Subtraction',sub)
#multiplication
mul = cv2.multiply(img1, new)
cv2.imshow('Multiplication',mul)
#division
div = cv2.divide(img1, new)
cv2.imshow('Division',div)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:-

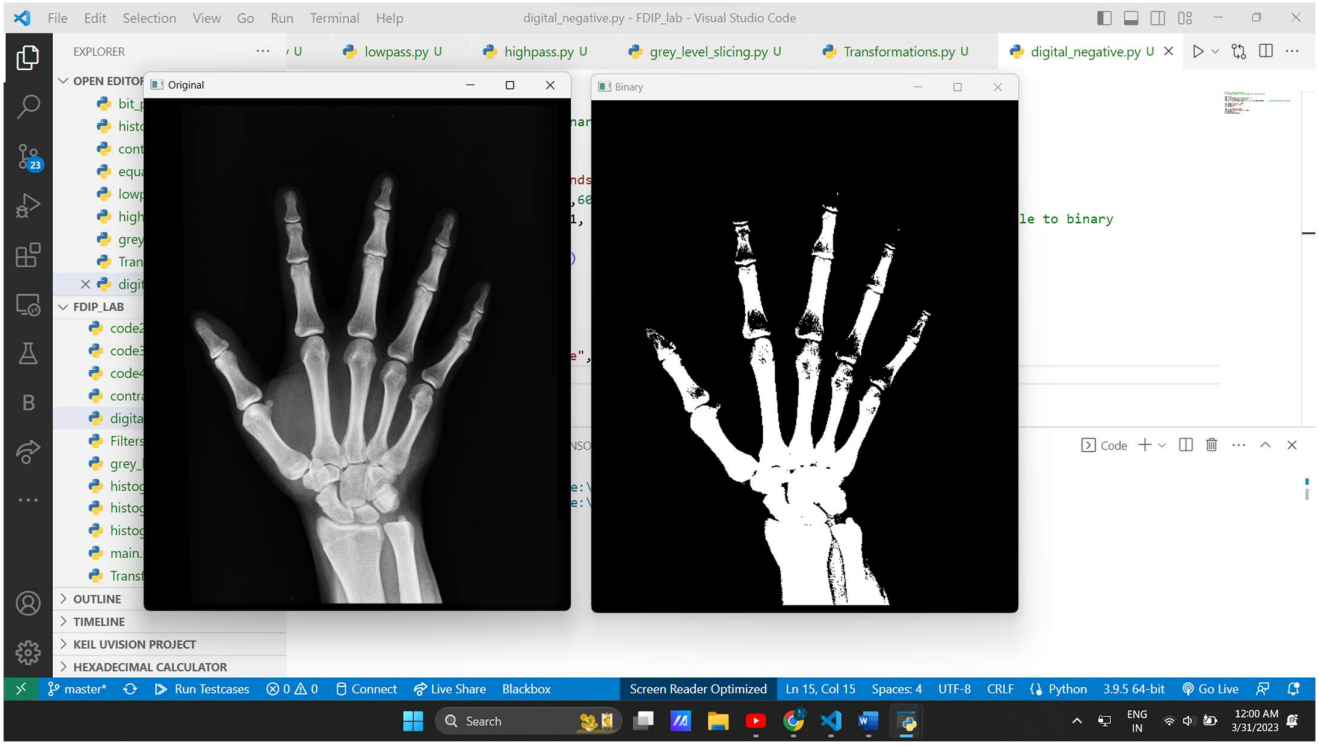


Practical 3:-

Digital Negative:-

```
# Point to point operations
# 1. Digital Negative of binary image (0->1 and vice versa)

import cv2
img = cv2.imread('assets\hands.jpeg', 1)
img1 = cv2.resize(img, (500,600))
ret, b1 = cv2.threshold(img1, 127, 255, cv2.THRESH_BINARY)      # converting greyscale to
binary

cv2.imshow('Original', img1)
cv2.imshow('Binary', b1)
cv2.waitKey(0)

img2 = cv2.bitwise_not(b1)
cv2.imshow("Digital Negative", img2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

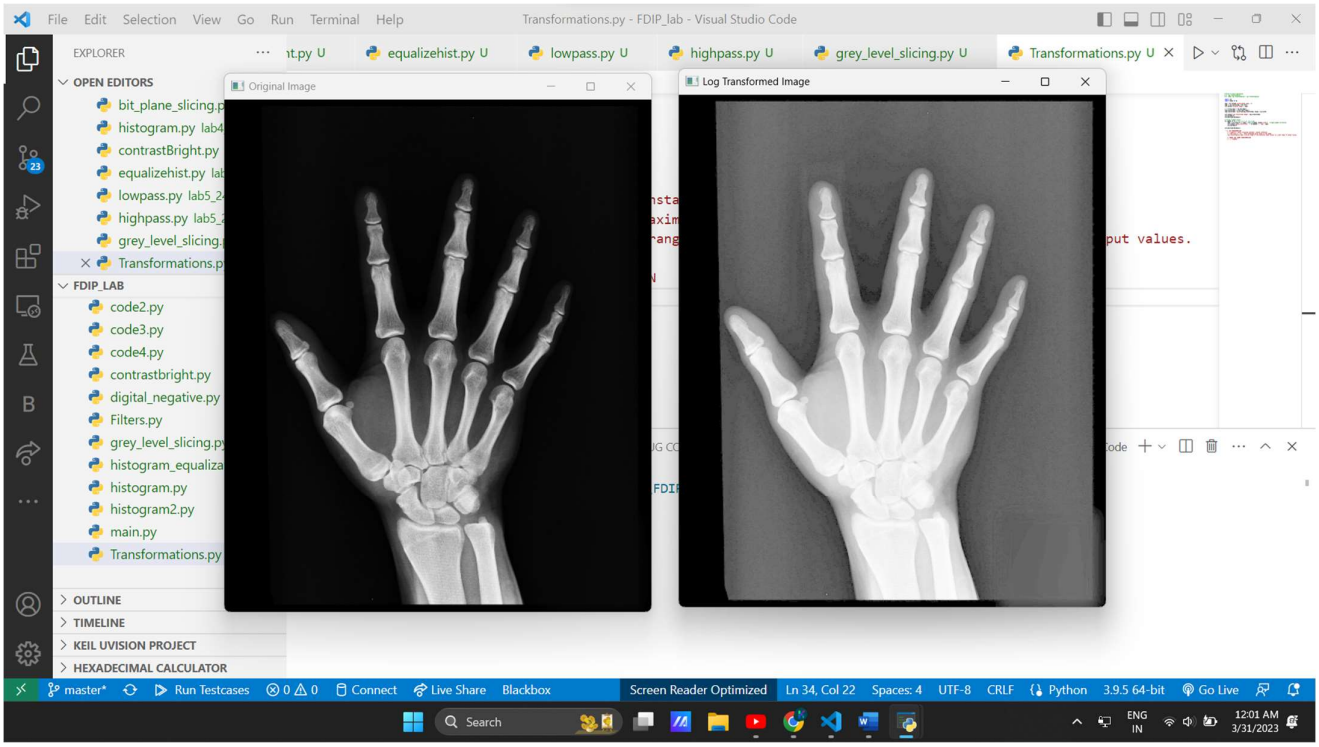
Output:-



Transformation:-

```
# Point to point operations
# 2. Log Transformation
# 3. Power Law Transformation / Law Transformation

import cv2
import numpy as np

img = cv2.imread('assets\hands.jpeg', 1)
img1 = cv2.resize(img, (500,600))
cv2.imshow("Original Image", img1)

c = 255/np.log(1 + np.max(img1))
log_transformed = c*(np.log(1+img1))
log_transformed = np.array(log_transformed, dtype = np.uint8)

cv2.imshow("Log Transformed Image", log_transformed)
cv2.waitKey(0)
cv2.destroyAllWindows()

# Trying 4 gamma values.
for gamma in [0.1, 0.5, 1.5, 2.5, 3.5, 5.5]:
    img2 = np.array(255 * (img1 / 255) ** gamma, dtype='uint8')   # Apply gamma
correction.
    cv2.imshow('gamma_transformed ' + str(gamma) + '.jpg', img2)
    cv2.waitKey(0)

cv2.destroyAllWindows()
```

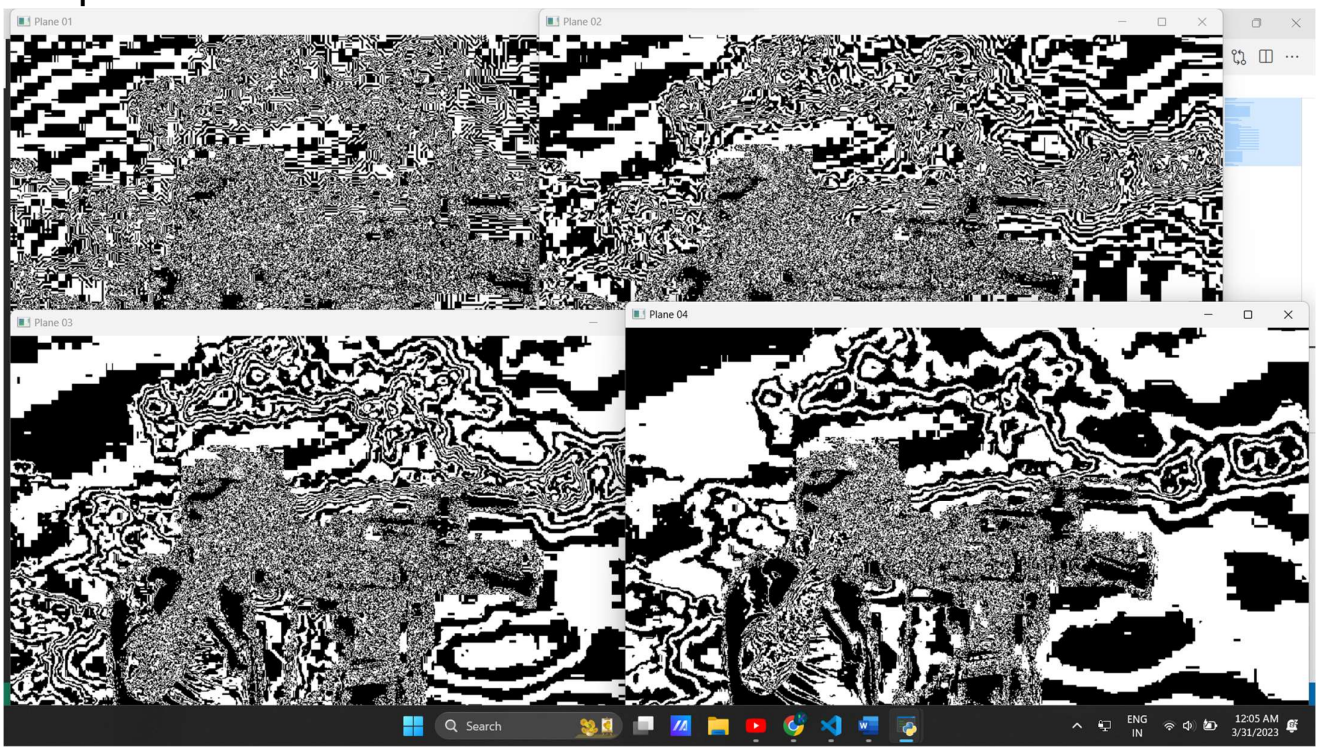
Output:-

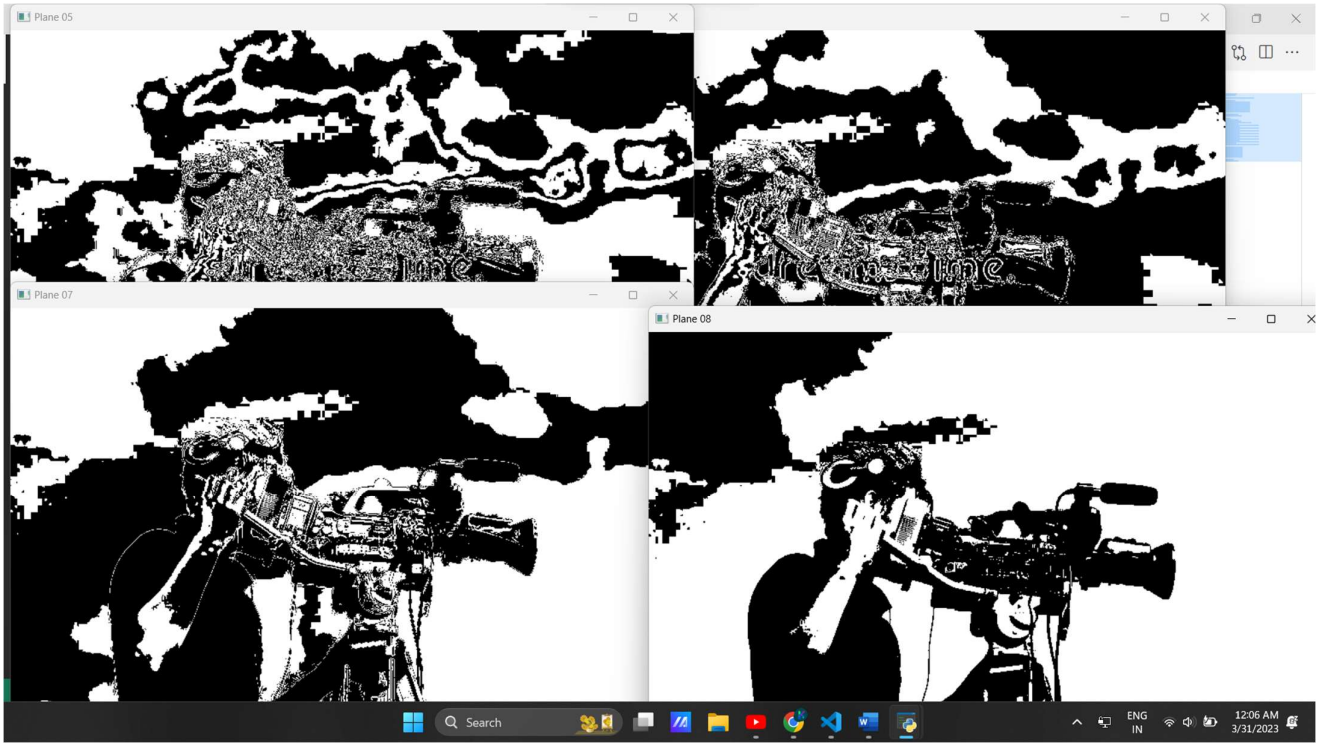


Bit Plane Slicing:-

```
import numpy as np
import cv2 as cv

img=cv.imread("./assets/camera_man1.png",0)

[w,h]=img.shape
img1=np.zeros([w,h,3],dtype=np.uint8)
img2=np.zeros([w,h,3],dtype=np.uint8)
img3=np.zeros([w,h,3],dtype=np.uint8)
img4=np.zeros([w,h,3],dtype=np.uint8)
img5=np.zeros([w,h,3],dtype=np.uint8)
img6=np.zeros([w,h,3],dtype=np.uint8)
img7=np.zeros([w,h,3],dtype=np.uint8)
img8=np.zeros([w,h,3],dtype=np.uint8)

def bitget(nbr,pos):
    return (nbr>>pos) & 1

planes=8
for i in range(0,w):
    for j in range(0,h):
        for p in range(planes-1,-1,-1):
            if p==7:
                img1[i][j]=255*bitget(img[i][j],p)
            elif p==6:
                img2[i][j]=255*bitget(img[i][j],p)
            elif p==5:
                img3[i][j]=255*bitget(img[i][j],p)
            elif p==4:
                img4[i][j]=255*bitget(img[i][j],p)
            elif p==3:
                img5[i][j]=255*bitget(img[i][j],p)
            elif p==2:
                img6[i][j]=255*bitget(img[i][j],p)
            elif p==1:
                img7[i][j]=255*bitget(img[i][j],p)
            else:
                img8[i][j]=255*bitget(img[i][j],p)

cv.imshow('Plane 08',img1)
cv.imshow('Plane 07',img2)
cv.imshow('Plane 06',img3)
cv.imshow('Plane 05',img4)
cv.imshow('Plane 04',img5)
cv.imshow('Plane 03',img6)
cv.imshow('Plane 02',img7)
cv.imshow('Plane 01',img8)

cv.waitKey(0)
cv.destroyAllWindows
```

Output:-

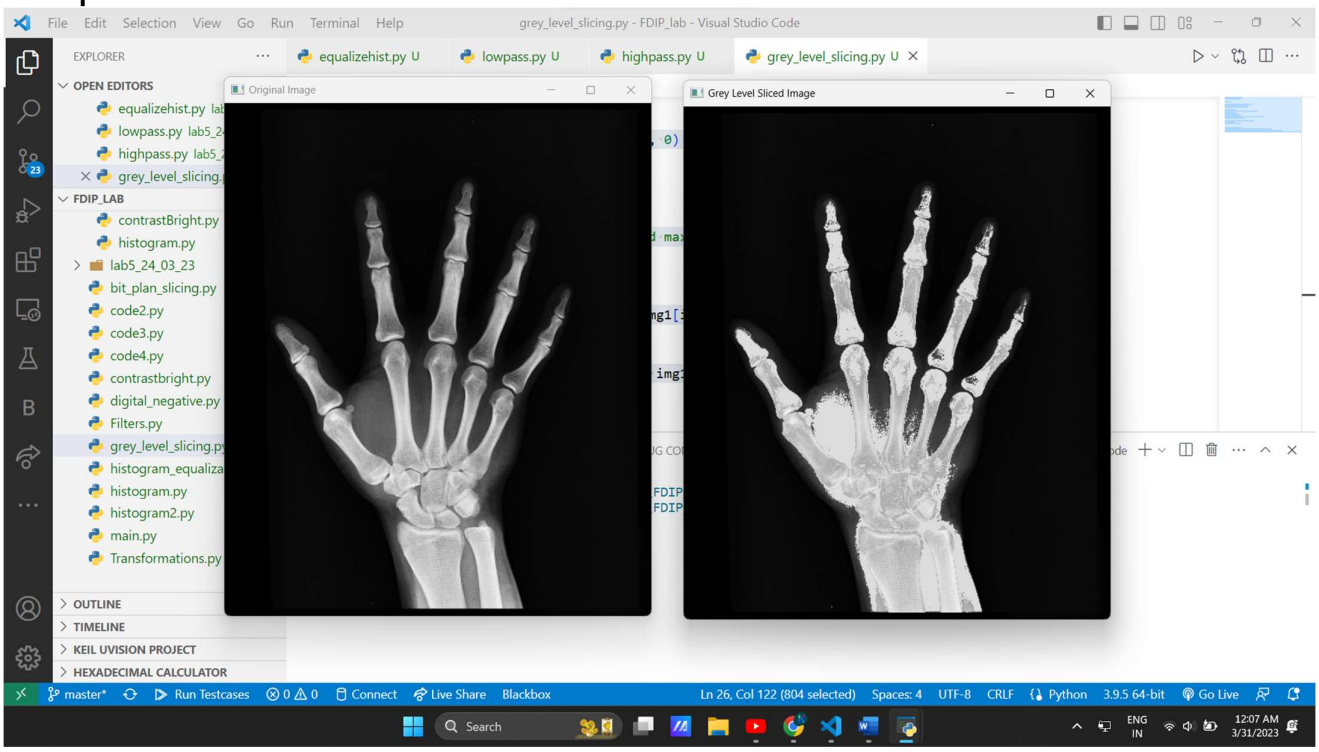

Plane 01
Plane 02
Plane 03
Plane 04
Search
ENG
IN
12:05 AM
3/31/2023


Plane 05
Plane 07
Plane 08
Search
ENG
IN
12:06 AM
3/31/2023

Grey Level Slicing:-

```
# Point to point operations
# 4. Grey Level Slicing

import cv2

img = cv2.imread(r"assets\hands.jpeg", 0)
img1 = cv2.resize(img, (500, 600))
cv2.imshow('Original Image', img1)

w,h = img1.shape
min_range = 80   # Specify the min and max range
max_range = 160
for i in range(0,w):
    for j in range(0,h):
        if img1[i][j]>min_range and img1[i][j]<max_range:
            img1[i][j] = 220

cv2.imshow('Grey Level Sliced Image', img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:-

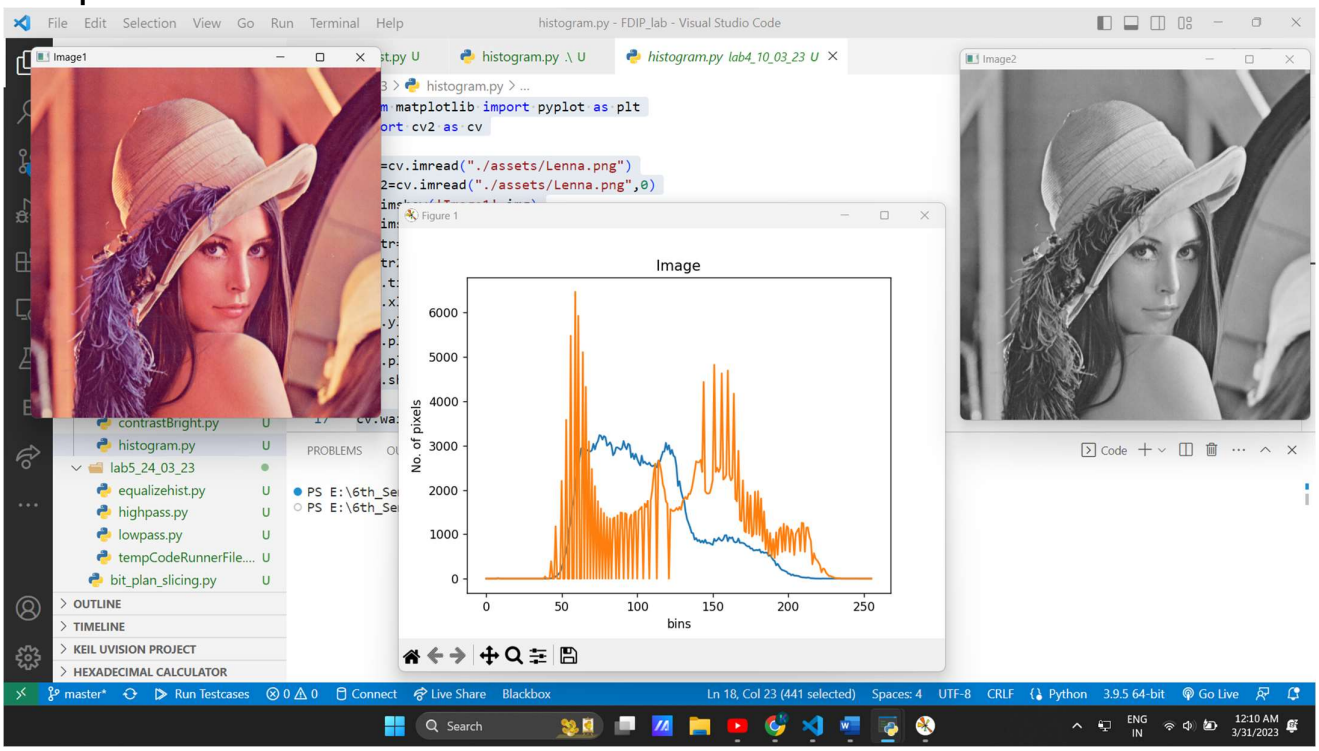


Practical 4:-

Histogram:-

```
from matplotlib import pyplot as plt
import cv2 as cv

img=cv.imread("./assets/Lenna.png")
img2=cv.imread("./assets/Lenna.png",0)
cv.imshow('Image1',img)
cv.imshow('Image2',img2)
histr=cv.calcHist([img],[0],None,[256],[0,256])
histr2=cv.calcHist([img2],[0],None,[256],[0,256])
plt.title("Image")
plt.xlabel('bins')
plt.ylabel('No. of pixels')
plt.plot(histr)
plt.plot(histr2)
plt.show()

cv.waitKey(0)
cv.destroyAllWindows()
```

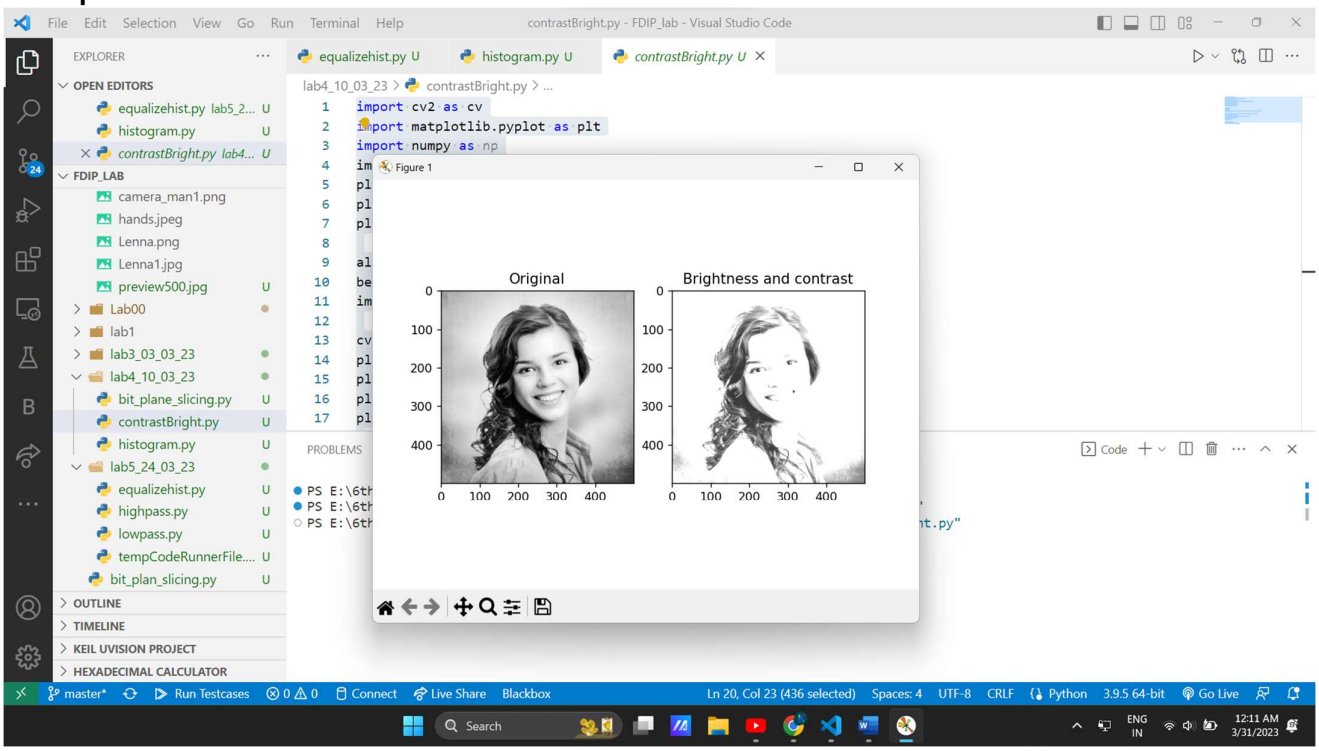
Output:-



Contrast Bright:-

```
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np
image=cv.imread("./assets/preview500.jpg")
plt.subplot(1,2,1)
plt.title("Original")
plt.imshow(image)

alpha=1.5
beta=50
image2=cv.convertScaleAbs(image,alpha=alpha,beta=beta)

cv.imwrite("Brightness and contrast.jpg",image2)
plt.subplot(1,2,2)
plt.title("Brightness and contrast")
plt.imshow(image2)
plt.show()

cv.waitKey(0)
cv.destroyAllWindows()
```

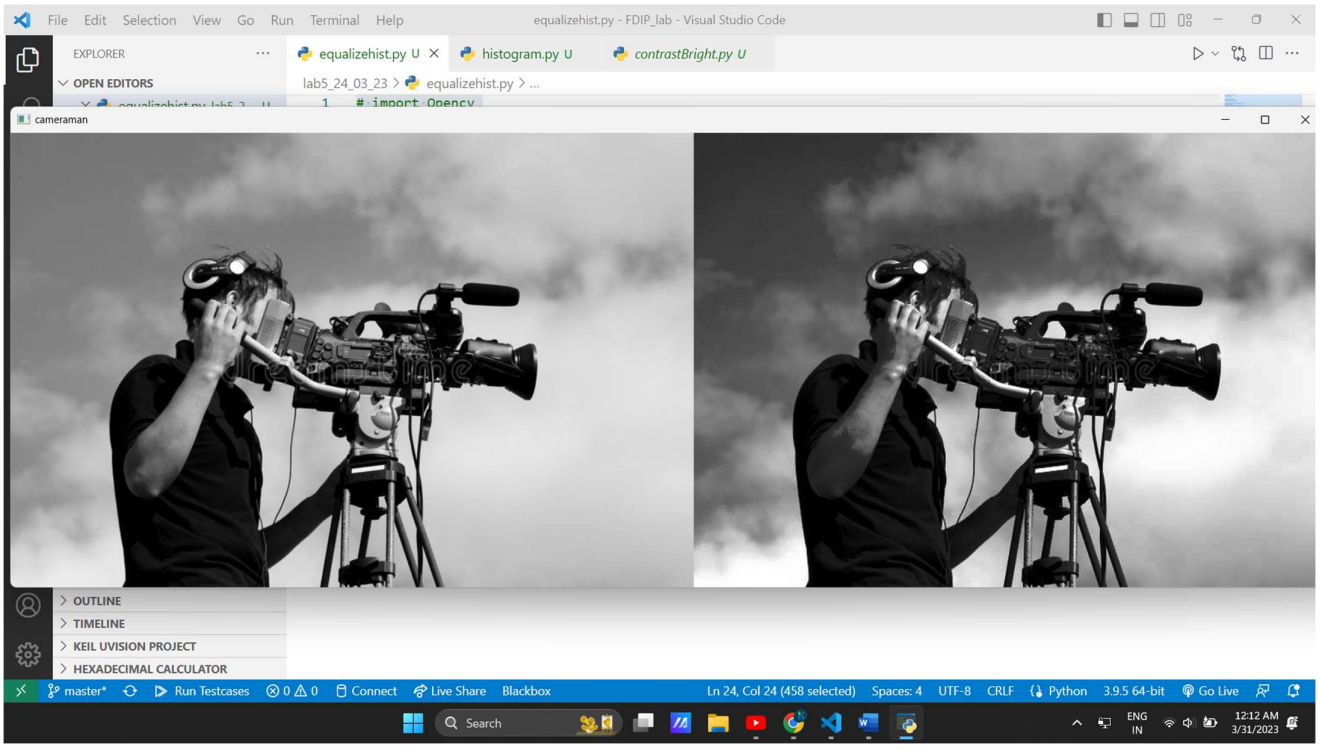
Output:-



Histogram Eqalization:-

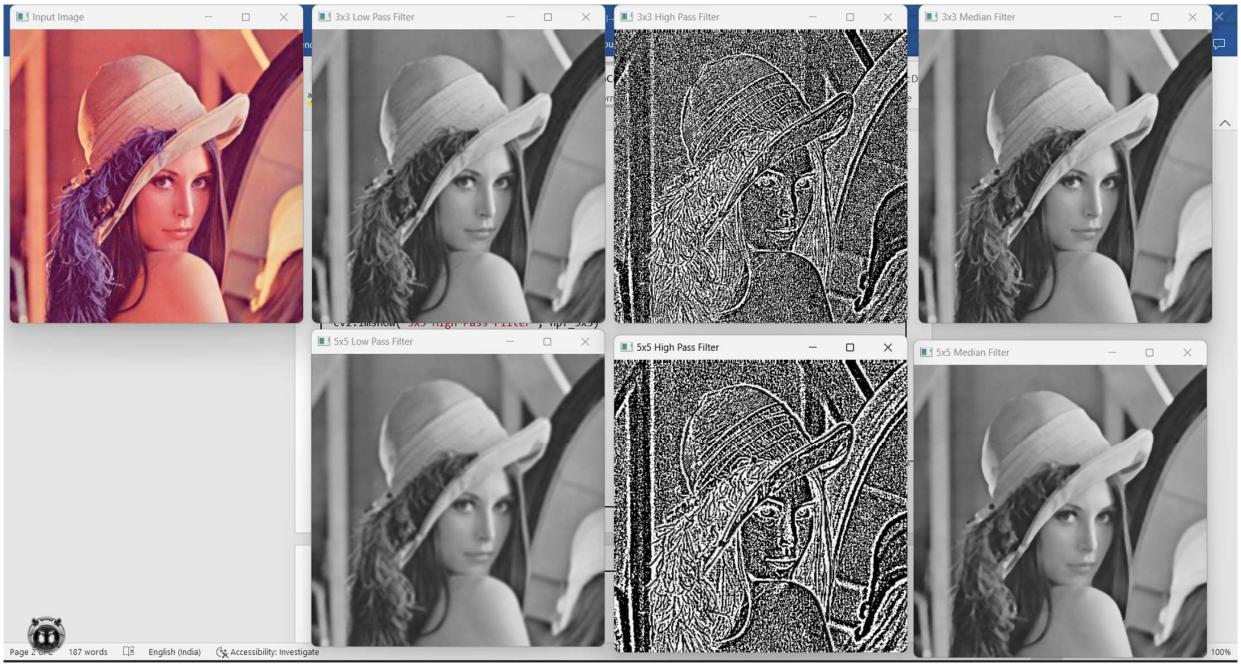
```
# import Opencv
import cv2

# import Numpy
import numpy as np

# read a image using imread
# read image
path = "assets\camera_man.jpg"
flag = 0
img = cv2.imread(path, flag)

# creating a Histograms Equalization
# of a image using cv2.equalizeHist()
equ = cv2.equalizeHist(img)

# stacking images side-by-side
res = np.hstack((img, equ))

# show image input vs output
cv2.imshow('cameraman', res)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

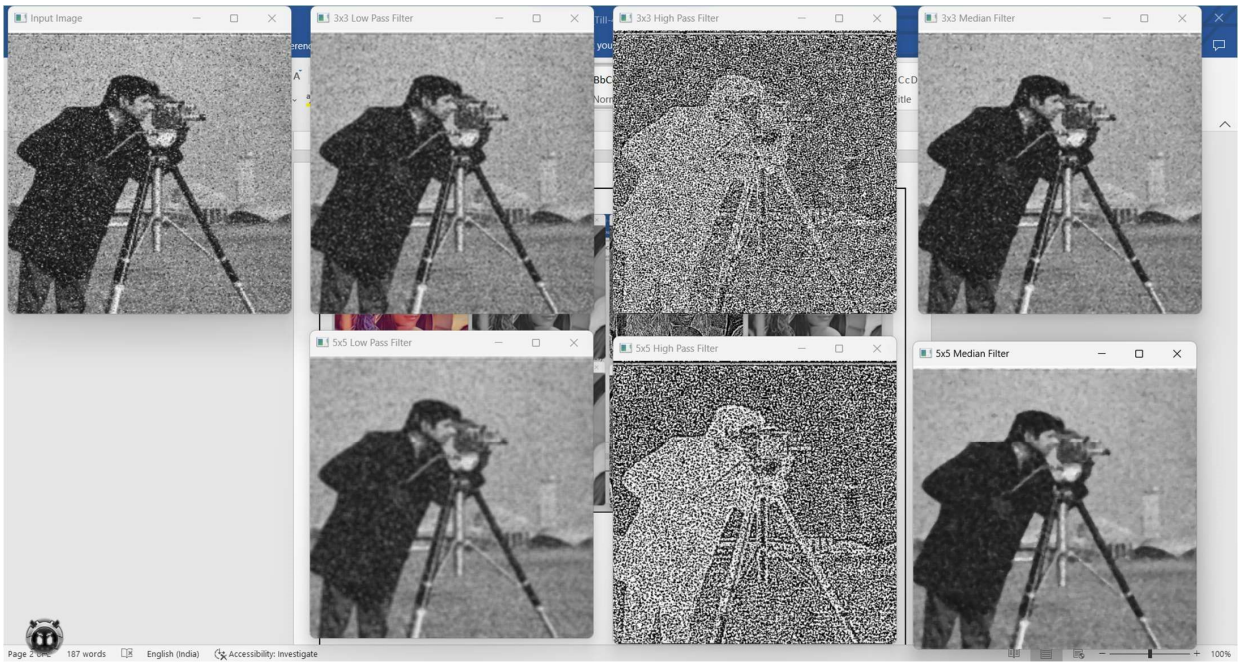
Output:-



Practical 5:-

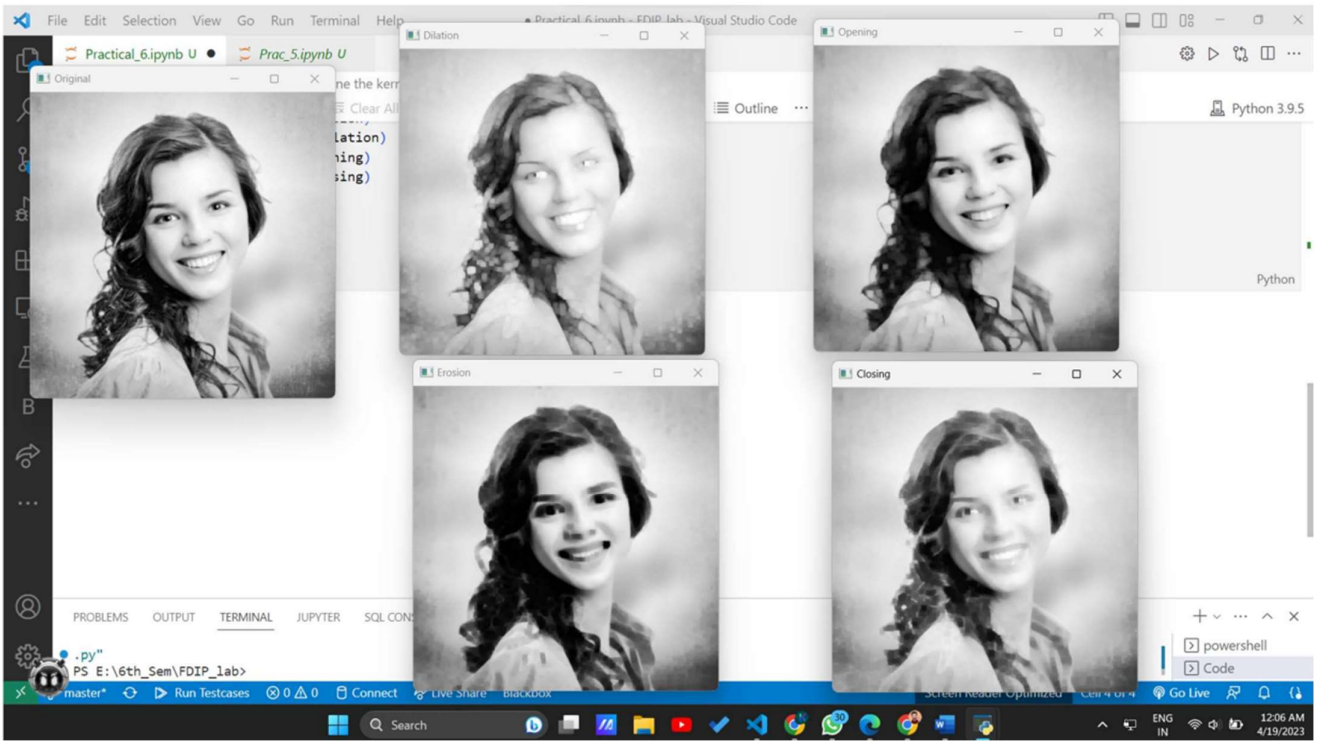
```
import cv2
import numpy as np # Load an image
img = cv2.imread("../assets/Lenna.png")
# Get the current size of the image
height, width, _ = img.shape

# Scale down the image to half its original size
img = cv2.resize(img, (int(width/1.4), int(height/1.4)))
# Convert image to grayscale
= cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Define 3x3 and 5x5 kernels for filtering
kernel_3x3 = np.ones((3,3),np.float32)/9
kernel_5x5 = np.ones((5,5),np.float32)/25

# Apply low pass filter using 3x3 kernel
lpf_3x3 = cv2.filter2D(gray, -1, kernel_3x3)

# Apply low pass filter using 5x5 kernel
lpf_5x5 = cv2.filter2D(gray, -1, kernel_5x5)

# Apply high pass filter using 3x3 kernel
hpf_3x3 = gray - lpf_3x3

# Apply high pass filter using 5x5 kernel
hpf_5x5 = gray - lpf_5x5

# Apply median filter using 3x3 kernel
mf_3x3 = cv2.medianBlur(gray, 3)

# Apply median filter using 5x5 kernel
 mf_5x5 = cv2.medianBlur(gray, 5)

# Display the filtered images
cv2.imshow("Input Image", img)
cv2.imshow("3x3 Low Pass Filter", lpf_3x3)
 cv2.imshow("5x5 Low Pass Filter", lpf_5x5)
cv2.imshow("3x3 High Pass Filter", hpf_3x3)
cv2.imshow("5x5 High Pass Filter", hpf_5x5)
cv2.imshow("3x3 Median Filter", mf_3x3)
cv2.imshow("5x5 Median Filter", mf_5x5)

# Wait for a key press and then exit
cv2.waitKey(0)
cv2.destroyAllWindows()
```

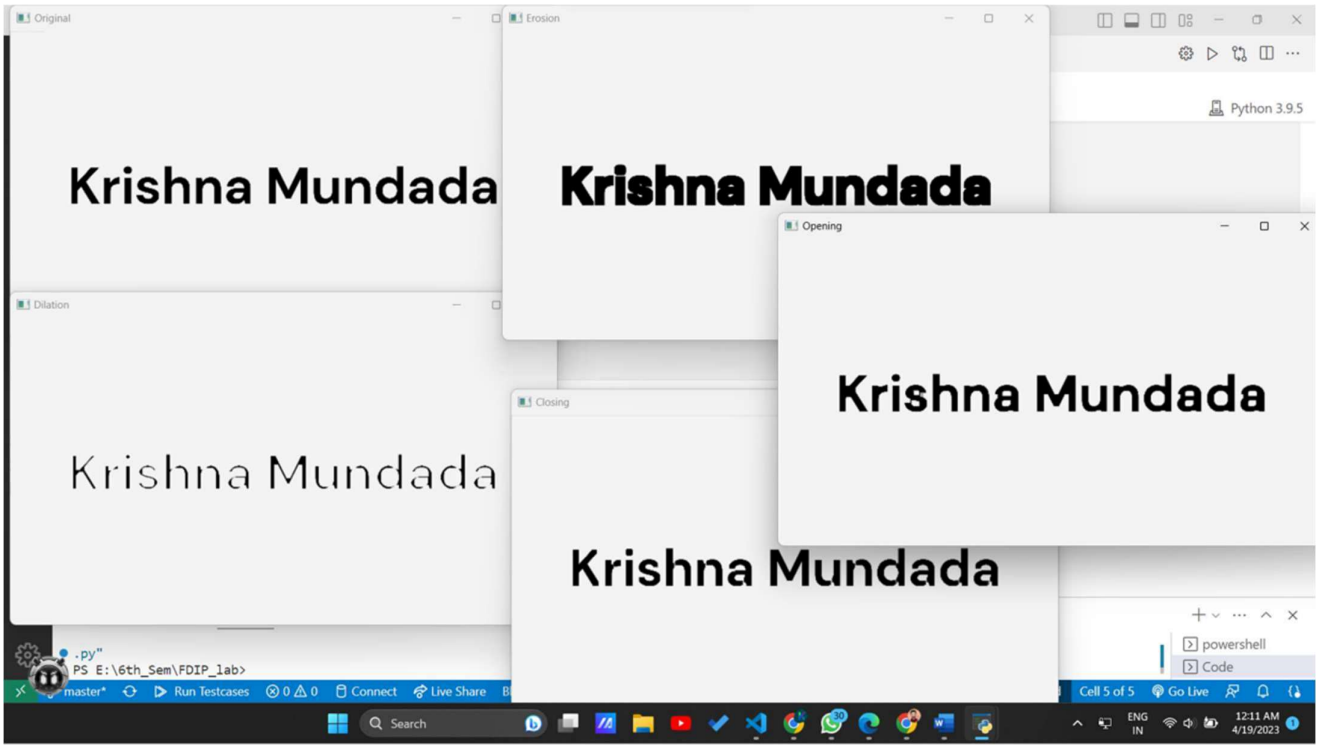

Input Image
3x3 Low Pass Filter
3x3 High Pass Filter
3x3 Median Filter
5x5 Low Pass Filter
5x5 High Pass Filter
5x5 Median Filter


Input Image
3x3 Low Pass Filter
3x3 High Pass Filter
3x3 Median Filter
5x5 Low Pass Filter
5x5 High Pass Filter
5x5 Median Filter

Practical 6:-

```
import cv2 import numpy as np
# Load the image
img = cv2.imread('../assets/preview500.jpg')
# Get the current size of the image
height, width, _ = img.shape

# Scale down the image to half its original size
img = cv2.resize(img, (int(width/1.4), int(height/1.4)))
# Define the kernel
kernel = np.ones((5,5), np.uint8)

# Erosion
erosion = cv2.erode(img, kernel, iterations = 1)

# Dilation
dilation = cv2.dilate(img, kernel, iterations = 1)

# Opening
opening = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)
# Closing
closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)
# Display the images
cv2.imshow('Original', img)
cv2.imshow('Erosion', erosion)
 cv2.imshow('Dilation', dilation)
cv2.imshow('Opening', opening)
cv2.imshow('Closing', closing)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

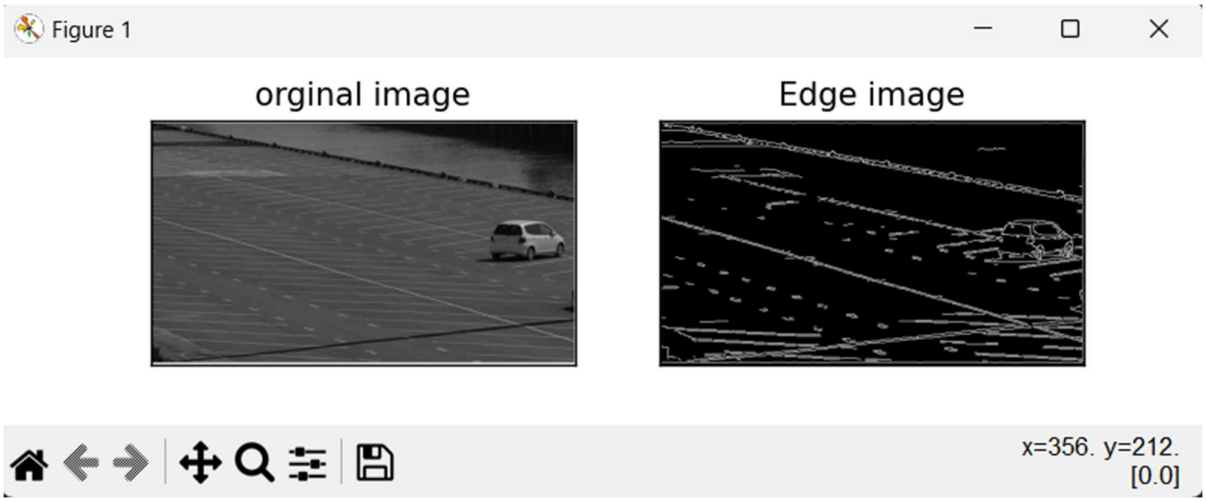

Original
Dilation
Opening
Erosion
Closing
Practical_6.ipynb U
Prac_5.ipynb U
Python 3.9.5
PROBLEMS
OUTPUT
TERMINAL
JUPYTER
PS E:\6th_Sem\FDIP_lab>


Original
Erosion
Dilation
Closing
Opening
Krishna Mundada
Python 3.9.5
PS E:\6th_Sem\FDIP_lab>

Practical 7:-

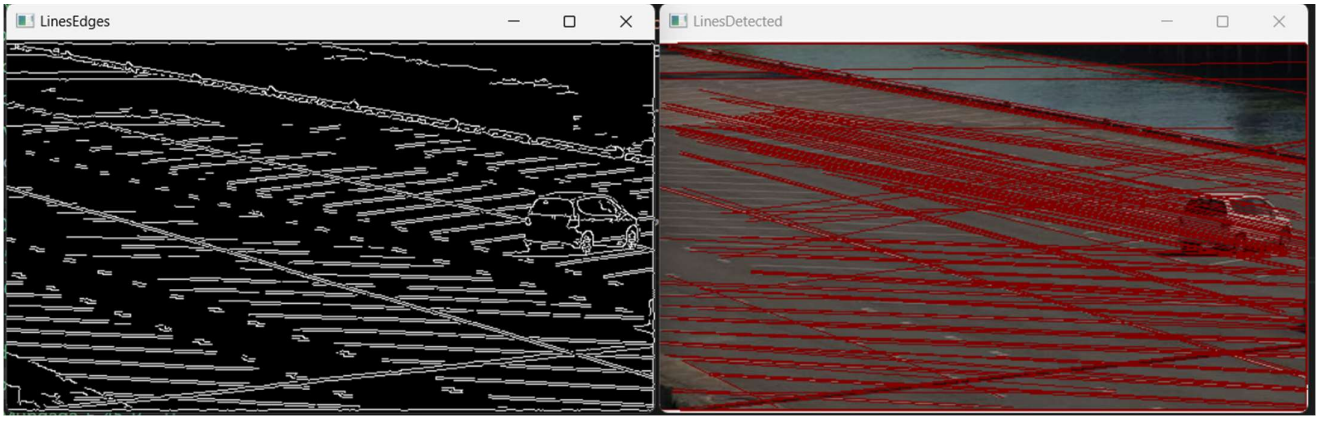
Canny Edge

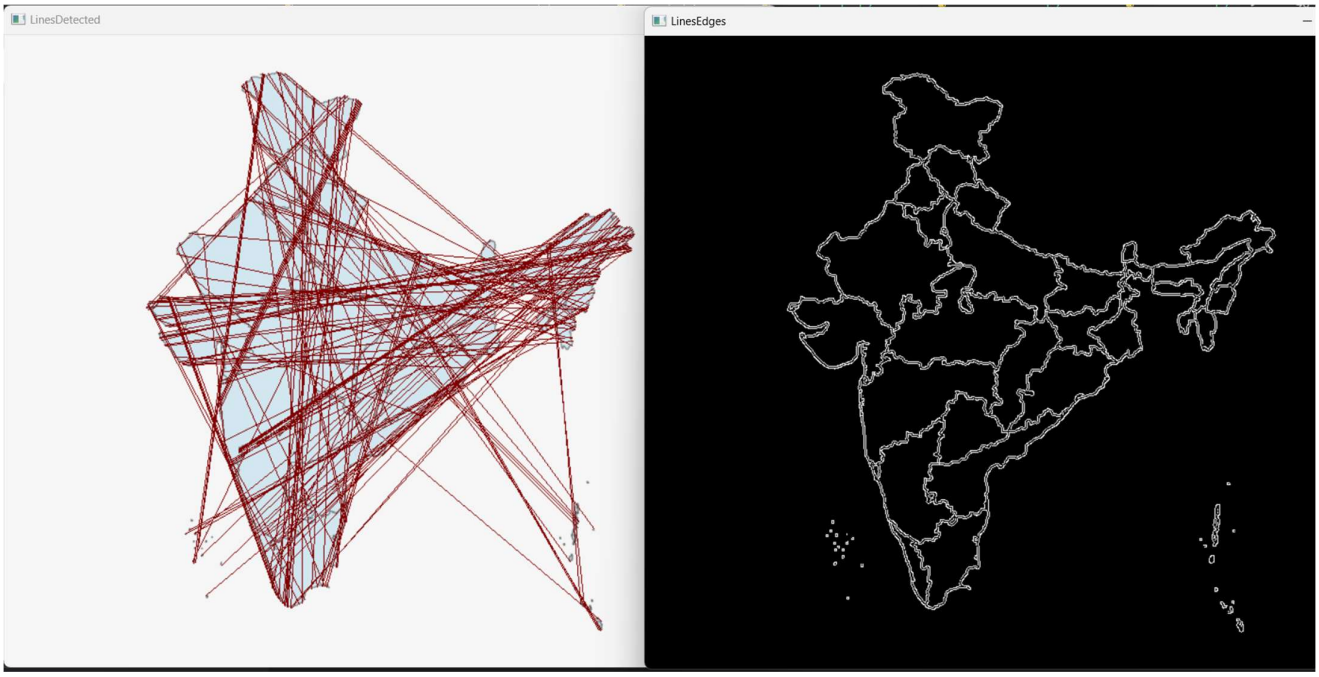
```
import cv2
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
img=cv2.imread("assets/tiger.jpg",cv2.IMREAD_GRAYSCALE)
#cv2.imshow('original',img)
# assert img is not None: "file could not read"
edges=cv2.Canny(img,100,200)
#print(img.shape)
plt.subplot(121), plt.imshow(img,cmap='gray')
plt.title('orginal image'),plt.xticks([]),plt.yticks([])
plt.subplot(122),plt.imshow(edges,cmap='gray')
plt.title('Edge image'),plt.xticks([]),plt.yticks([])
plt.show()
```



Hough Filter:-

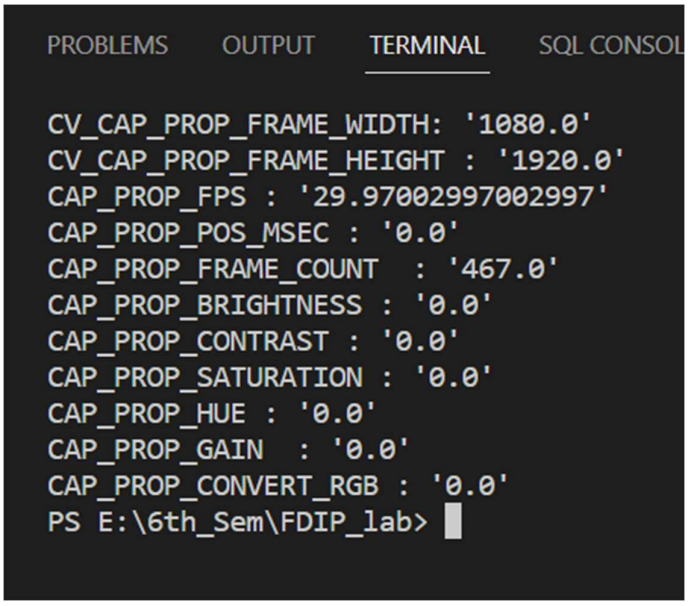
```
import cv2
import numpy as np
img=cv2.imread('assets/parking_lot.jpg')
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
edges=cv2.Canny(gray,75,150)
lines=cv2.HoughLinesP(edges,1,np.pi/180,30,maxLineGap=250)

for line in lines:
    x1,y1,x2,y2=line[0]
    cv2.line(img,(x1,y1),(x2,y2),(0,0,128,1))
cv2.imshow("LinesEdges ",edges)
cv2.imshow("LinesDetected ",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```





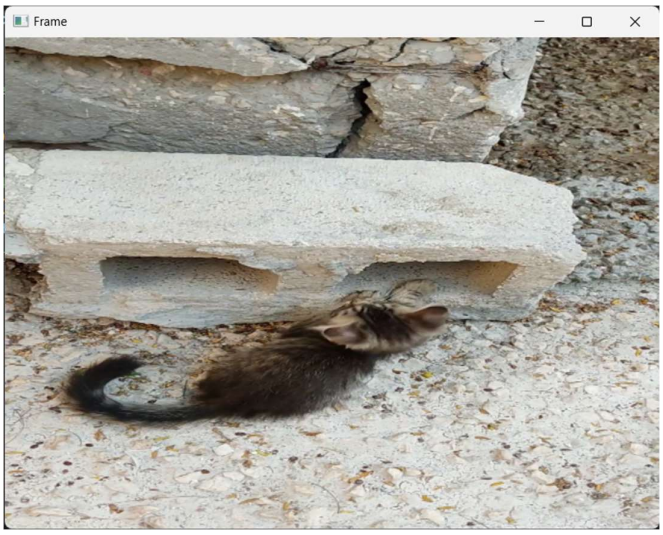
Practical 8
Video Properties:-

```
# importing cv2
import cv2

# For Video File
capture=cv2.VideoCapture(r"assets\video.mp4")

# For webcam
#capture = cv2.VideoCapture(0)

# showing values of the properties
print("CV_CAP_PROP_FRAME_WIDTH: '{}'".format(capture.get(cv2.CAP_PROP_FRAME_WIDTH)))
print("CV_CAP_PROP_FRAME_HEIGHT : '{}'".format(capture.get(cv2.CAP_PROP_FRAME_HEIGHT)))
print("CAP_PROP_FPS : '{}'".format(capture.get(cv2.CAP_PROP_FPS)))
print("CAP_PROP_POS_MSEC : '{}'".format(capture.get(cv2.CAP_PROP_POS_MSEC)))
print("CAP_PROP_FRAME_COUNT  : '{}'".format(capture.get(cv2.CAP_PROP_FRAME_COUNT)))
print("CAP_PROP_BRIGHTNESS : '{}'".format(capture.get(cv2.CAP_PROP_BRIGHTNESS)))
print("CAP_PROP_CONTRAST : '{}'".format(capture.get(cv2.CAP_PROP_CONTRAST)))
print("CAP_PROP_SATURATION : '{}'".format(capture.get(cv2.CAP_PROP_SATURATION)))
print("CAP_PROP_HUE : '{}'".format(capture.get(cv2.CAP_PROP_HUE)))
print("CAP_PROP_GAIN  : '{}'".format(capture.get(cv2.CAP_PROP_GAIN)))
print("CAP_PROP_CONVERT_RGB : '{}'".format(capture.get(cv2.CAP_PROP_CONVERT_RGB)))

# release window
capture.release()
cv2.destroyAllWindows()
```



VideoProcess:-

```
import cv2
# Create a video capture object, in this case we are reading the video from a file
vid_capture = cv2.VideoCapture(r"assets\video.mp4")
if (vid_capture.isOpened() == False):
    print("Error opening the video file")
# Read fps and frame count
else:
    # Get frame rate information
    # You can replace 5 with CAP_PROP_FPS as well, they are enumerations
    fps = vid_capture.get(5)
    print('Frames per second : ', fps, 'FPS')
    # Get frame count
    # You can replace 7 with CAP_PROP_FRAME_COUNT as well, they are enumerations
    frame_count = vid_capture.get(7)
    print('Frame count : ', frame_count)
while (vid_capture.isOpened()):
    # vid_capture.read() methods returns a tuple, first element is a bool
    # and the second is frame
    ret, frame = vid_capture.read()
    if ret == True:
        # Resize the frame to 640x480
        resized_frame = cv2.resize(frame, (640, 480))
        cv2.imshow('Frame', resized_frame)
        # 20 is in milliseconds, try to increase the value, say 50 and observe
        key = cv2.waitKey(20)
        if key == ord('q'):
            break
    else:
        break
# Release the video capture object
vid_capture.release()
cv2.destroyAllWindows()
```




```
PROBLEMS  OUTPUT  TERMINAL  SQL CONSOLE  DEBUG CONSOLE  COMMENTS

PS E:\6th_Sem\FDIP_lab> python -u "e:\6th_Sem\FDIP_lab\Practical_8\videoProcess.py"
Frames per second :  29.97002997002997 FPS
Frame count :  467.0
PS E:\6th_Sem\FDIP_lab>
```

Video Save:-

```
# Python program to save a video using OpenCV
import cv2
# Create an object to read from camera
video = cv2.VideoCapture(r"assets\video.mp4")

# We need to check if camera is opened previously or not
if (video.isOpened() == False):
    print("Error reading video file")

# We need to set resolutions. so, convert them from float to integer.
frame_width = int(video.get(3))
frame_height = int(video.get(4))

size = (frame_width, frame_height)

# Below VideoWriter object will create a frame of above defined The output
# is stored in 'filename.avi' file.
result = cv2.VideoWriter('filename.avi',cv2.VideoWriter_fourcc(*'MJPG'),10, size)

while (True):
    ret, frame = video.read()
    if ret == True:
        # Write the frame into the
        # file 'filename.avi'
        result.write(frame)
        # Display the frame saved in the file
        frame = cv2.resize(frame, (640, 480))
        cv2.imshow('Frame', frame)
        # Press S on keyboard to stop the process
        if cv2.waitKey(1) & 0xFF == ord('s'):
            break
    # Break the loop
    else:
        break
# When everything done, release the video capture and video write objects
video.release()
result.release()
# Closes all the frames
cv2.destroyAllWindows()

print("The video was successfully saved")
```

```
PS E:\6th_Sem\FDIP_lab> python -u "e:\6th_Sem\FDIP_lab\Practical_8\videoSave.py"
The video was successfully saved
PS E:\6th_Sem\FDIP_lab>
```