1. Perform basicimage handling and processing operations on the image.Read on image in python and convert o]an image to grayscale?

Program:

import cv2

import numpy as np

kernel = np.ones((5,5),np.uint8) print(kernel)

path = r"C:\Users\DELL\Downloads\tom and jerry.jpg" img =cv2.imread(path)

imgGray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) cv2.imshow("GrayScale",imgGray)

cv2.waitKey(0)

INPUT:



OUTPUT:



1. Perform basic image handling and processing operations on the image.Read an image in python and convert an image to blurnusing Guassian blur?

Program:

import cv2

import numpy as np

kernel = np.ones((5,5),np.uint8) print(kernel)

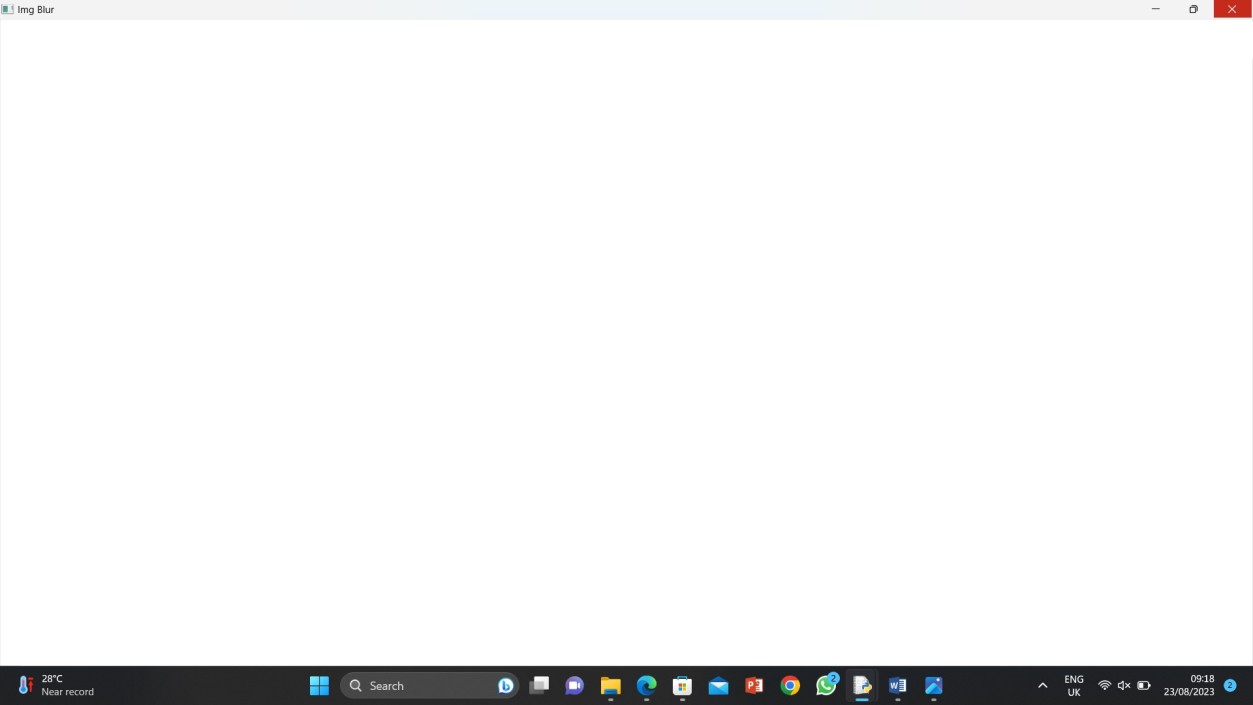
path = r"C:\Users\DELL\Downloads\hand-drawing-flower-vector.jpg" img =cv2.imread(path)

imgGray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) imgBlur = cv2.GaussianBlur(imgGray,(7,7),0) cv2.imshow("Img Blur",imgBlur)

cv2.waitKey(0)

INPUT:



OUTPUT:

1. Perform basic Image Handling and processing operations on the image• Read an image in python and Convert an Image to show outline using Canny function?

AIM: To Perform Basic Operations to Convert image to show outline Canny function in Python Program:

import cv2

import numpy as np

kernel = np.ones((5,5),np.uint8) print(kernel)

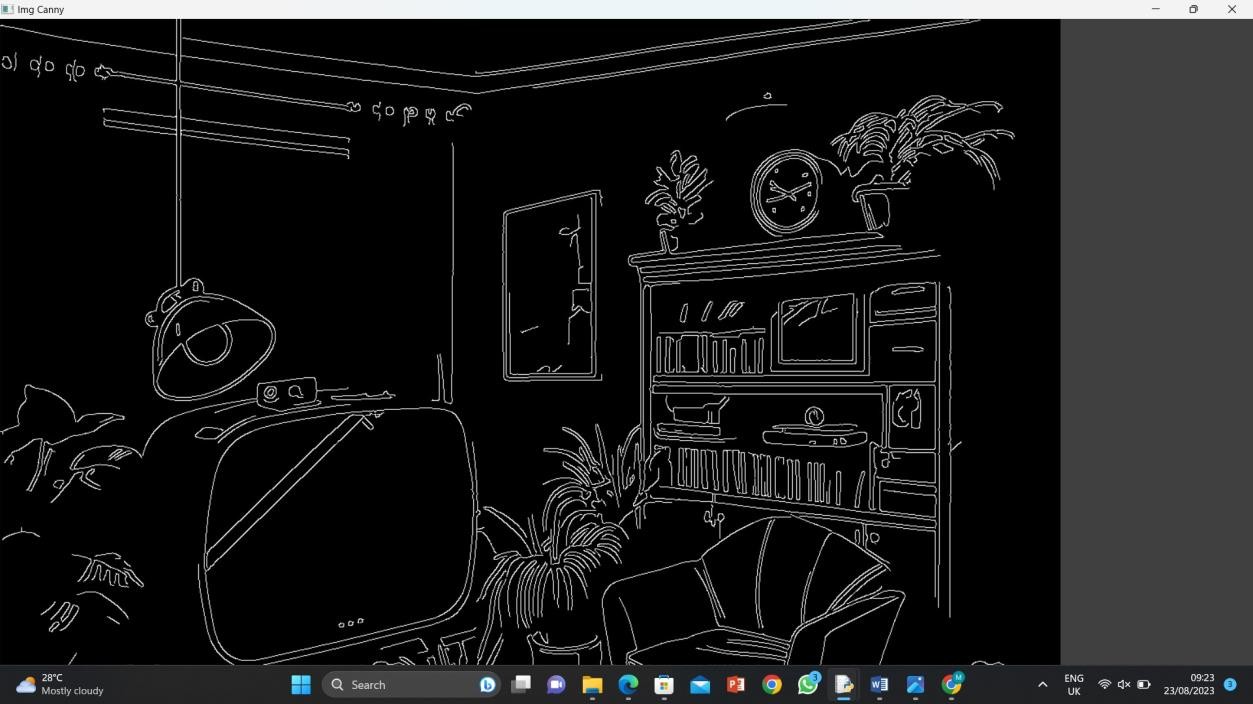
path = r"C:\Users\DELL\Downloads\cartoon home.jpg" img =cv2.imread(path)

imgGray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) imgBlur = cv2.GaussianBlur(imgGray,(7,7),0) imgCanny = cv2.Canny(imgBlur,100,200) cv2.imshow("Img Canny",imgCanny)

cv2.waitKey(0)

INPUT:



OUTPUT:

1. Perform basic Image Handling and processing operations on the image• Read an image in python and Dilate an Image using Dilate function?

AIM: To Perform Basic Operations to Read Image and Dilate an Image using Python Program:

import cv2

import numpy as np

kernel = np.ones((5,5),np.uint8) print(kernel)

path = r"C:\Users\DELL\Downloads\images food.png" img =cv2.imread(path)

imgGray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) imgBlur = cv2.GaussianBlur(imgGray,(7,7),0) imgCanny = cv2.Canny(imgBlur,100,200)

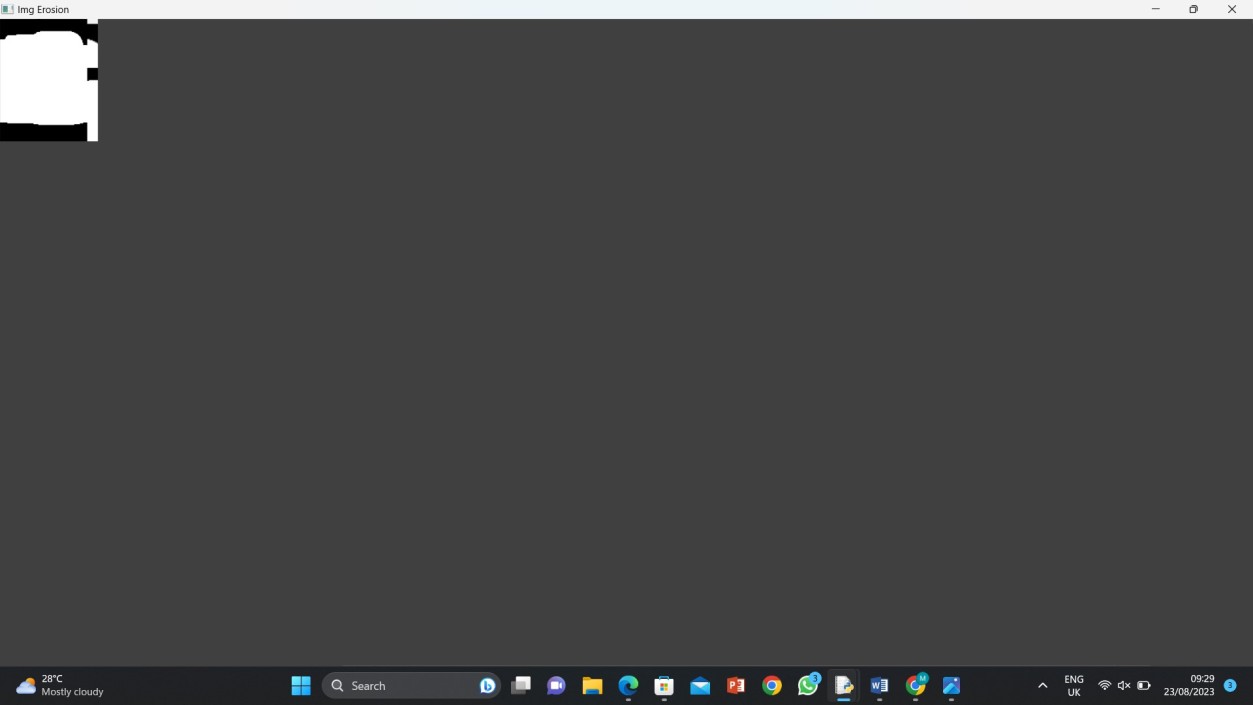
imgDilation = cv2.dilate(imgCanny,kernel , iterations = 10) imgEroded = cv2.erode(imgDilation,kernel,iterations=2) cv2.imshow("Img Erosion",imgEroded)

cv2.waitKey(0)

INPUT:



OUTPUT:



5..Perform basic Image Handling and processing operations on the image• Read an image in python and Erode an Image using erode function?

AIM: The Aim of the experiment is to Read an image in python and Erode an Image using erode function

Program:

import cv2

import numpy as np

kernel = np.ones((5,5),np.uint8) print(kernel)

path = r"C:\Users\DELL\Downloads\images food.png" img =cv2.imread(path)

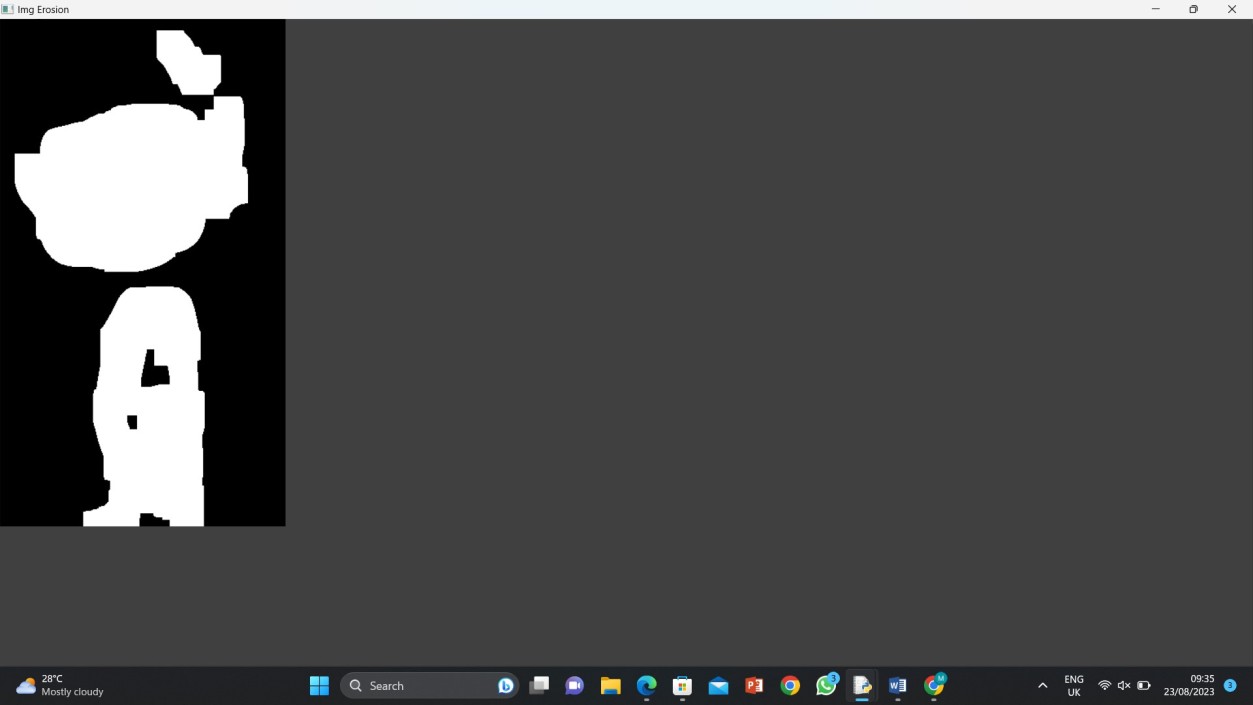
imgGray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) imgBlur = cv2.GaussianBlur(imgGray,(7,7),0) imgCanny = cv2.Canny(imgBlur,100,200)

imgDilation = cv2.dilate(imgCanny,kernel , iterations = 10) imgEroded = cv2.erode(imgDilation,kernel,iterations=2) cv2.imshow("Img Erosion",imgEroded)

cv2.waitKey(0)

INPUT:



OUTPUT:

1. Perform basic video processing operations on the captured video• Read captured video in python and display the video, in slow motion and in fast motion?

AIM: The Aim of the Experiment is to Read captured video in python and display the video, in slow motion and in fast motion

Program:

import cv2

image\_path = r"C:\Users\DELL\Downloads\flower.jpg" cap = cv2.VideoCapture(image\_path)

if not cap.isOpened():

print("Error opening image file")

while cap.isOpened(): ret, frame = cap.read() if ret:

cv2.imshow('Frame', frame)

if cv2.waitKey(250) & 0xFF == ord('q'): break

else:

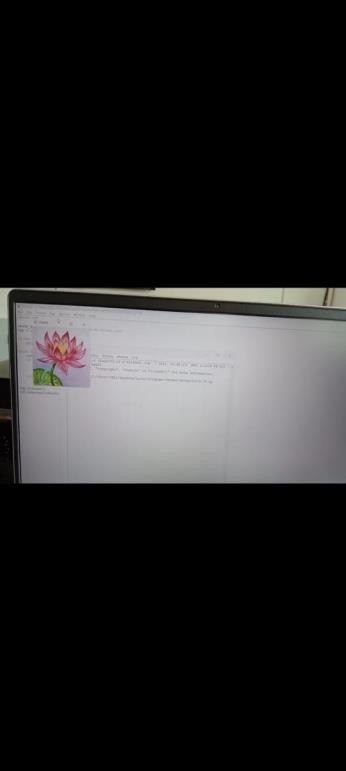
break

cap.release() cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Capture video from web Camera and Display the video, in slow motion and in fast motion operations on the captured video?

AIM: The Aim is to Capture video from web Camera and Display the video, in slow motion and in fast motion operations on the captured video

Program:

import cv2

cap = cv2.VideoCapture(0)

height = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT)) width = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH)) fps = cap.get(cv2.CAP\_PROP\_FPS)

path = "output.mp4"

fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')

output = cv2.VideoWriter(path, fourcc, fps, (width, height))

while True:

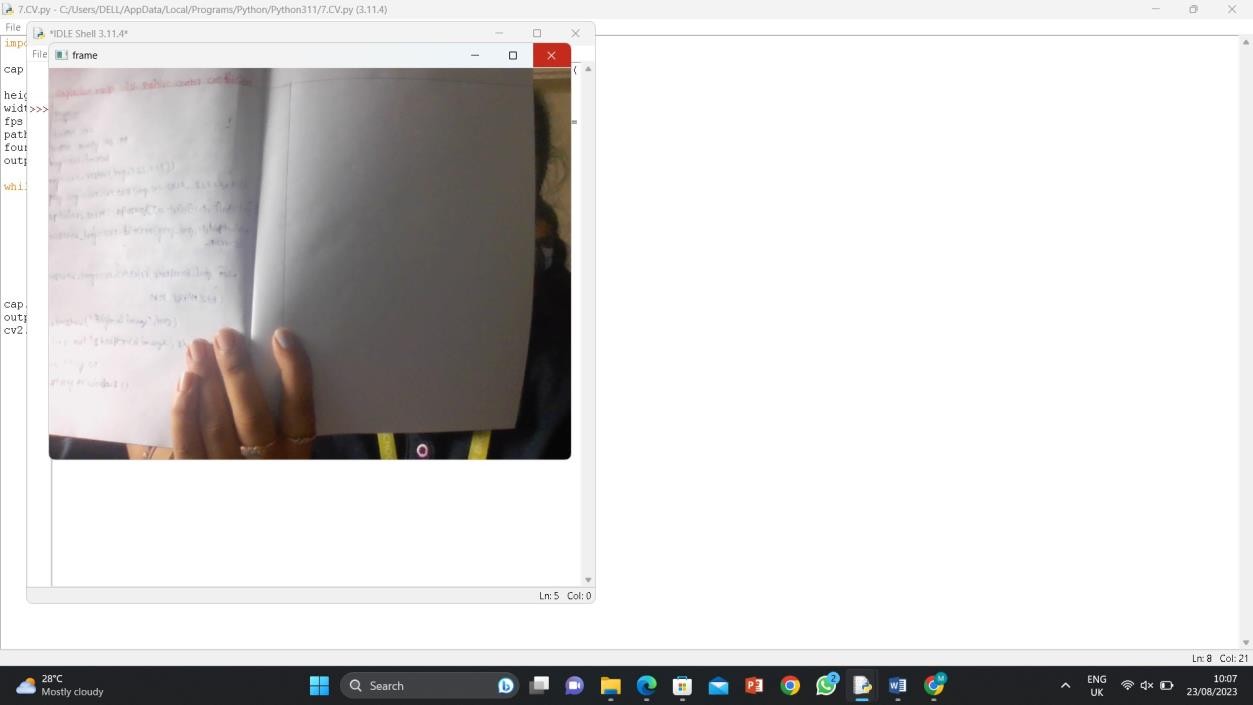
ret, frame = cap.read() cv2.imshow("frame", frame) output.write(frame)

k = cv2.waitKey(24) if k == ord("q"):

break

cap.release() output.release() cv2.destroyAllWindows()

OUTPUT:



1. Scaling an image to its Bigger and Smaller sizes?

AIM: The Aim is resize the image from bigger to smaller size Program:

import cv2

import numpy as np

kernel = np.ones((5,5),np.uint8)

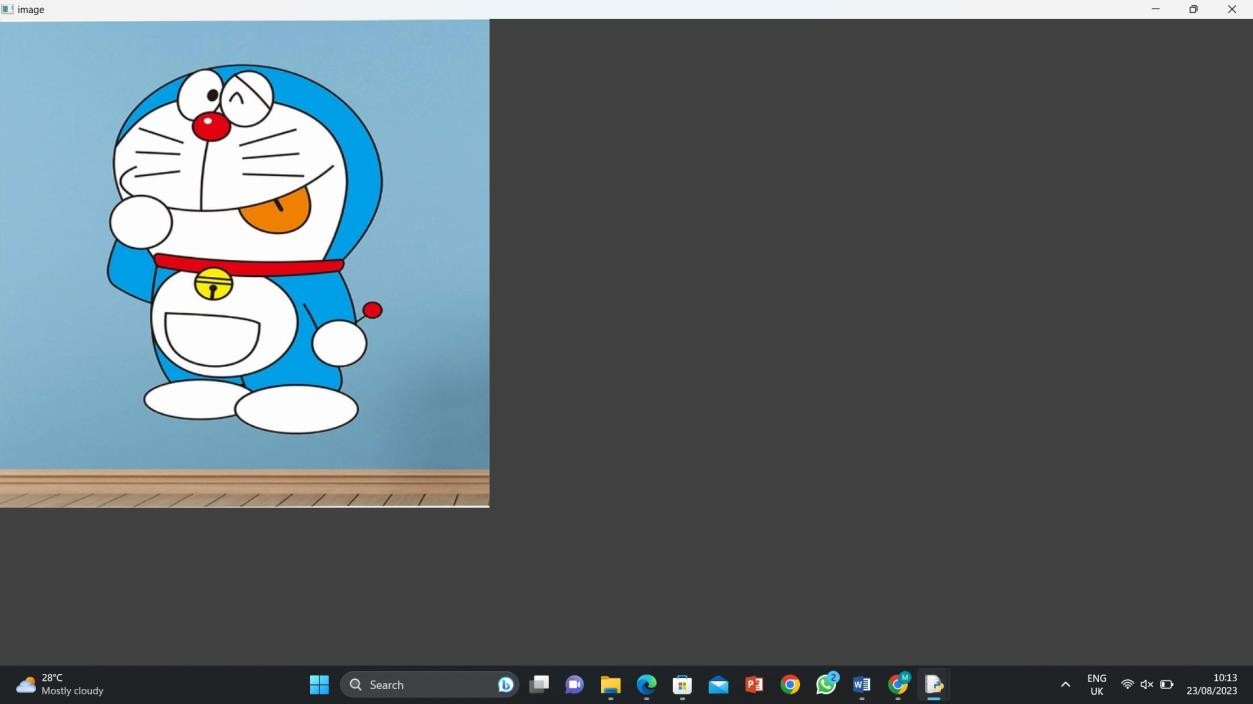
img = cv2.imread(r"C:\Users\DELL\Downloads\doremon.jpg",cv2.IMREAD\_COLOR) img = cv2.resize(img,(600,600))

cv2.imshow("image",img) cv2.waitKey(0)

INPUT:



OUTPUT:



ROTATION 90 ALONG DEGREE:

AIM The Aim of the Experiment is to perform Rotation of an image along 90 degree Program:

import cv2

path = r"C:\Users\DELL\Downloads\meghana saved pictures.webp" src = cv2.imread(path)

window\_name = 'Image'

if src is not None:

image = cv2.rotate(src, cv2.ROTATE\_180) cv2.imshow(window\_name, image) cv2.waitKey(0)

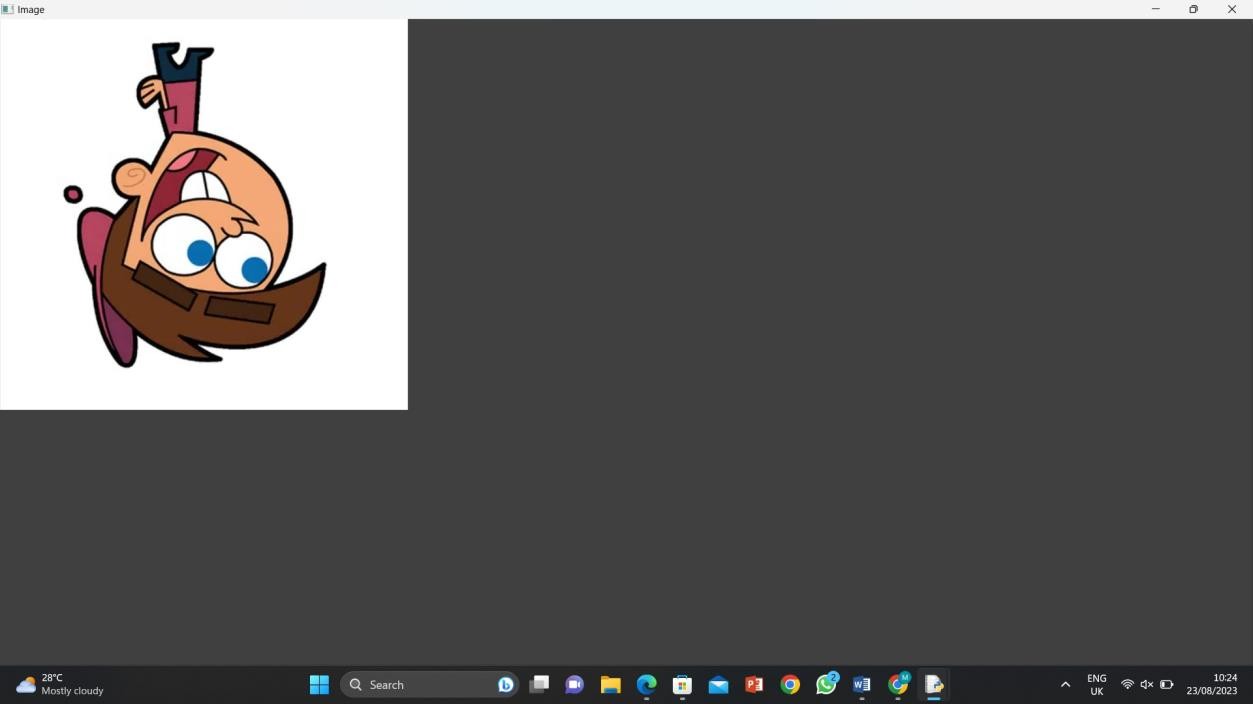
cv2.destroyAllWindows() else:

print("Error: Unable to load the image")

INPUT:



OUTPUT:



ROTATION ALONG 180 DEGREE

AIM The Aim of the Experiment is to perform Rotation of an image along 180 degree Program:

import cv2

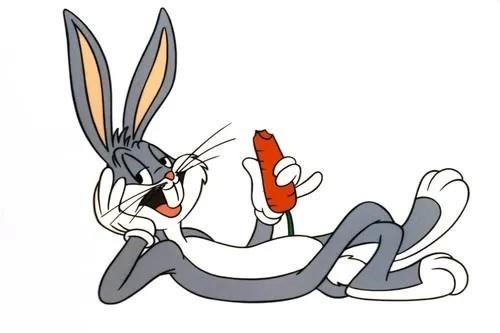
path = r"C:\Users\DELL\Downloads\RABBIT.webp" src = cv2.imread(path)

window\_name = 'Image'

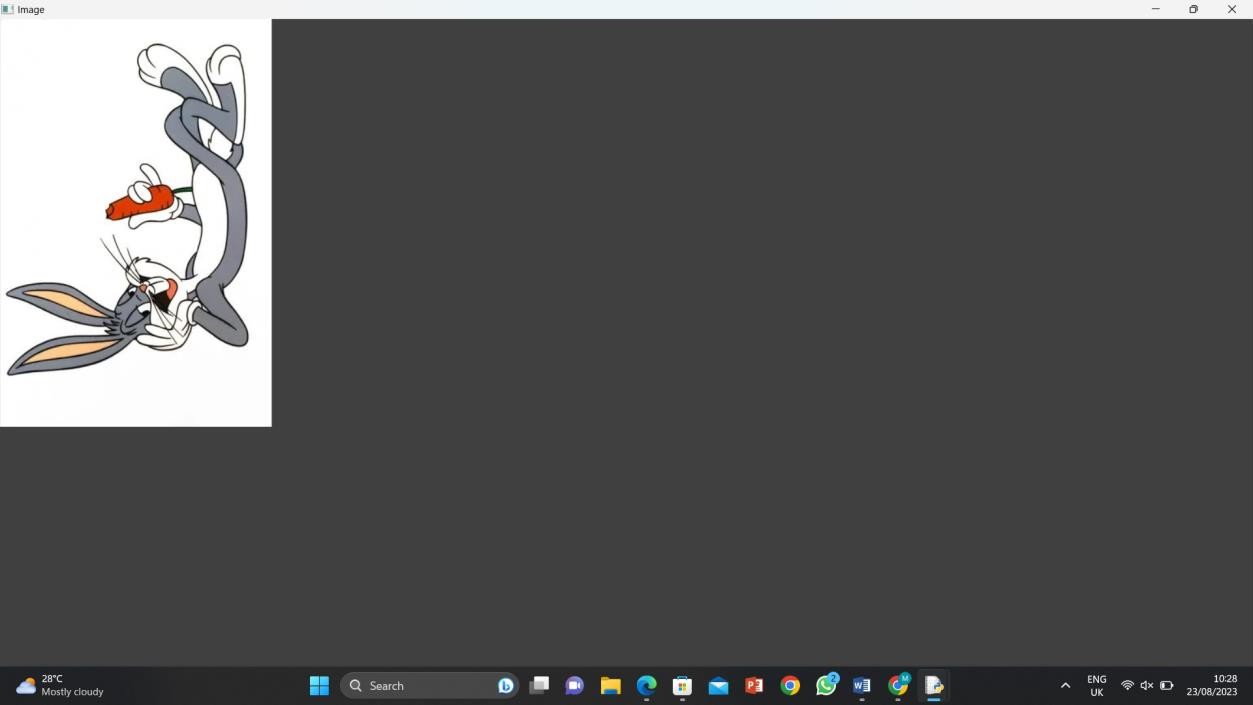
image = cv2.rotate(src, cv2.ROTATE\_90\_COUNTERCLOCKWISE) # Displaying the image

cv2.imshow(window\_name, image) cv2.waitKey(0)

INPUT:



OUTPUT:



1. ROTATION ALONG 270 DEGREE

AIM The Aim of the Experiment is to perform Rotation of an image along 270 degree Program:

import cv2

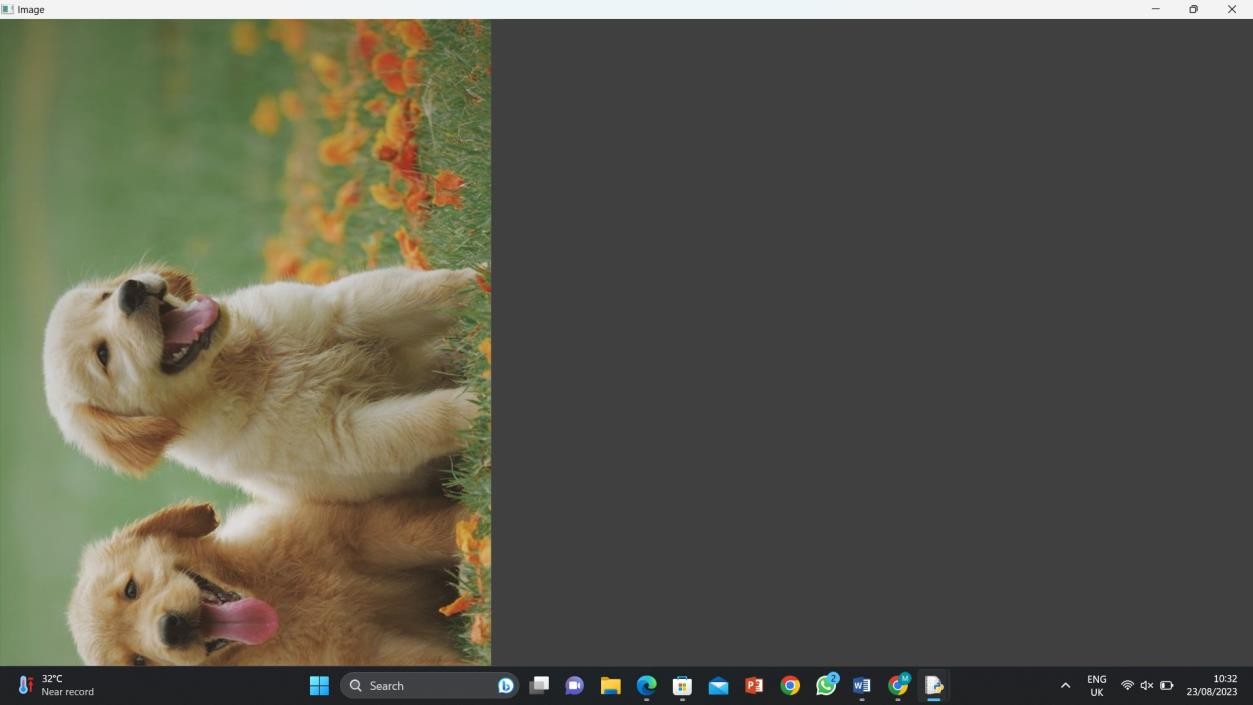
path = r"C:\Users\DELL\Downloads\meghana fav.jpg" src = cv2.imread(path)

window\_name = 'Image'

image = cv2.rotate(src, cv2.ROTATE\_90\_COUNTERCLOCKWISE) cv2.imshow(window\_name, image)

cv2.waitKey(0) INPUT:

OUTPUT:



1. Perform Affine Transformation on the image?

AIM:To perform affine transformation on the image using IDLE python Program:

import cv2

import numpy as np

# read the input image

img = cv2.imread(r"C:\Users\DELL\Downloads\download folder meghana.jpg") # access the image height and width

rows,cols,\_ = img.shape

# define at three point on input image

pts1 = np.float32([[50,50],[200,50],[50,200]])

# define three points corresponding location to output image pts2 = np.float32([[10,100],[200,50],[100,250]])

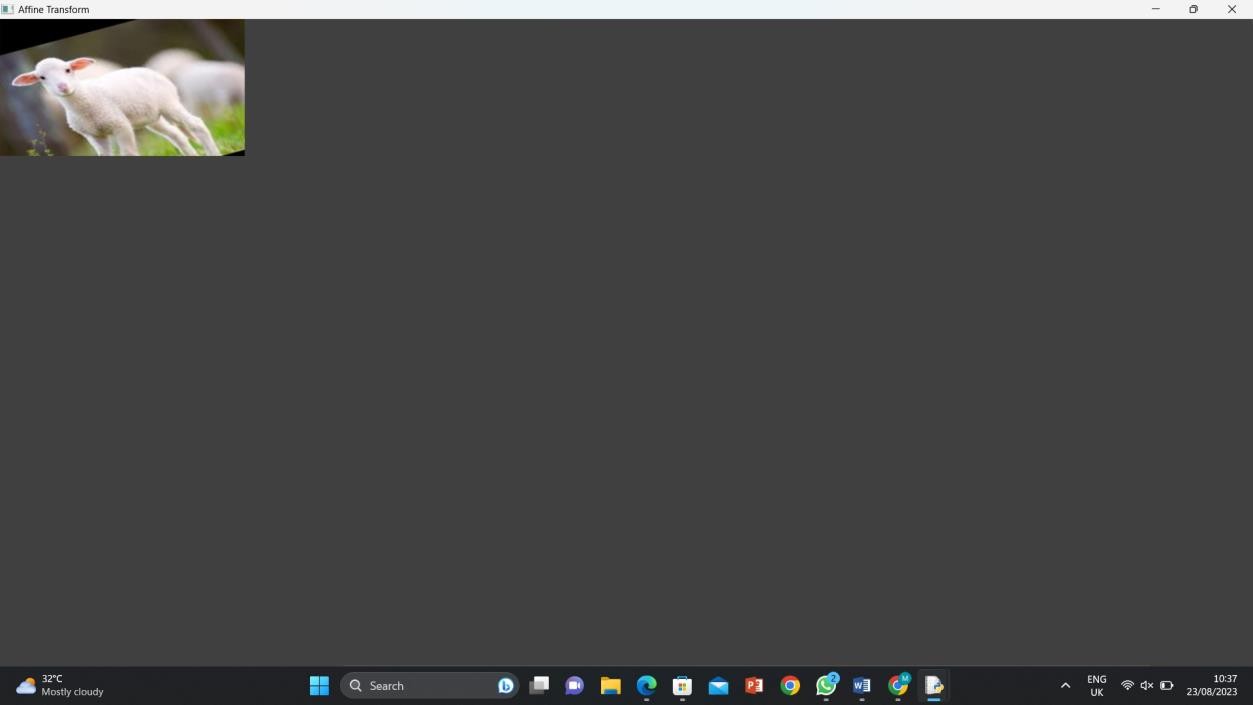
# get the affine transformation Matrix M = cv2.getAffineTransform(pts1,pts2)

# apply affine transformation on the input image dst = cv2.warpAffine(img,M,(cols,rows)) cv2.imshow("Affine Transform", dst) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Perform Perspective Transformation on the image?

AIM:To perform perspective transformation on the image using IDLE python Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\Tiger-1.jpg") rows,cols,ch = img.shape

pts1 = np.float32([[56,65],[368,52],[28,387],[389,390]])

pts2 = np.float32([[100,50],[300,0],[0,300],[300,300]])

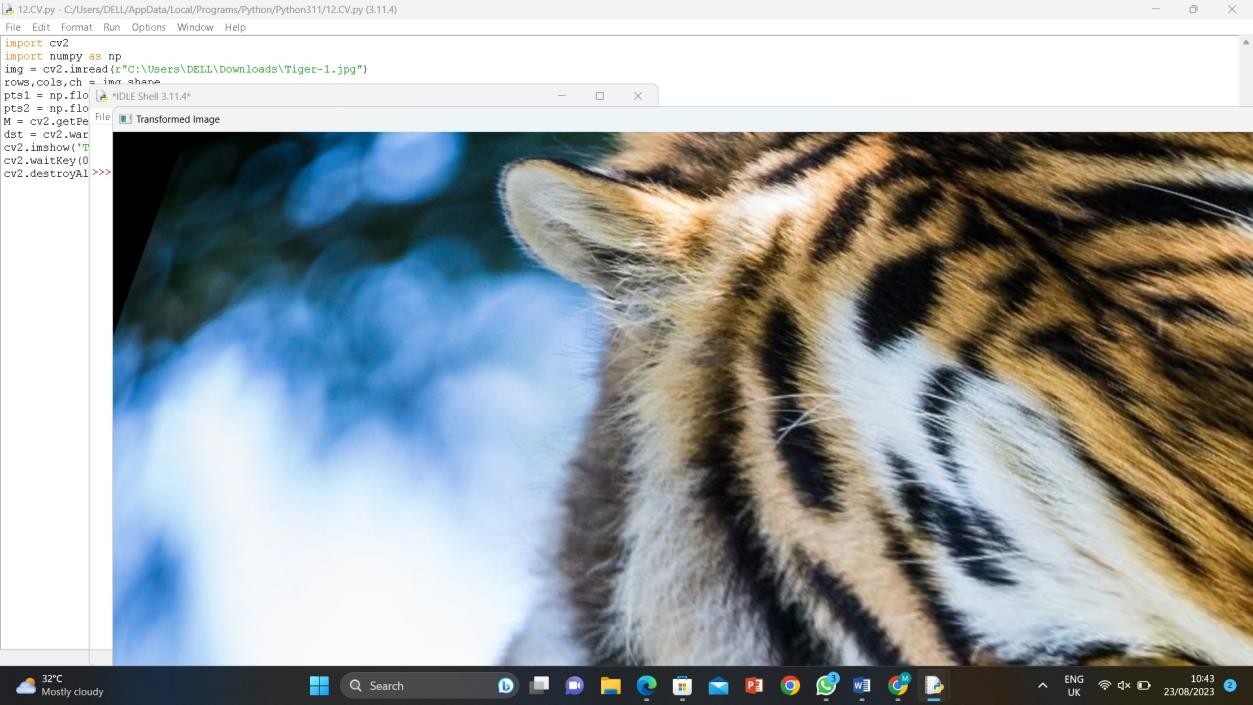
M = cv2.getPerspectiveTransform(pts1,pts2) dst = cv2.warpPerspective(img,M,(cols, rows)) cv2.imshow('Transformed Image', dst) cv2.waitKey(0)

cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Perform Perspective Transformation on the Video?

AIM:To perform the perspective transformation on the video using IDLE python Program:

import cv2

import numpy as np

cap = cv2.VideoCapture(r"C:\Users\DELL\Downloads\GettyImages-1145794687.jpg")

while True:

ret, frame = cap.read()

if not ret:

break

pts1 = np.float32([[200, 300], [5, 2], [0, 4], [6, 0]])

pts2 = np.float32([[0, 0], [4, 0], [0, 1], [4, 6]])

matrix = cv2.getPerspectiveTransform(pts1, pts2)

result = cv2.warpPerspective(frame, matrix, (frame.shape[1], frame.shape[0]))

cv2.imshow('frame', frame) cv2.imshow('frame1', result)

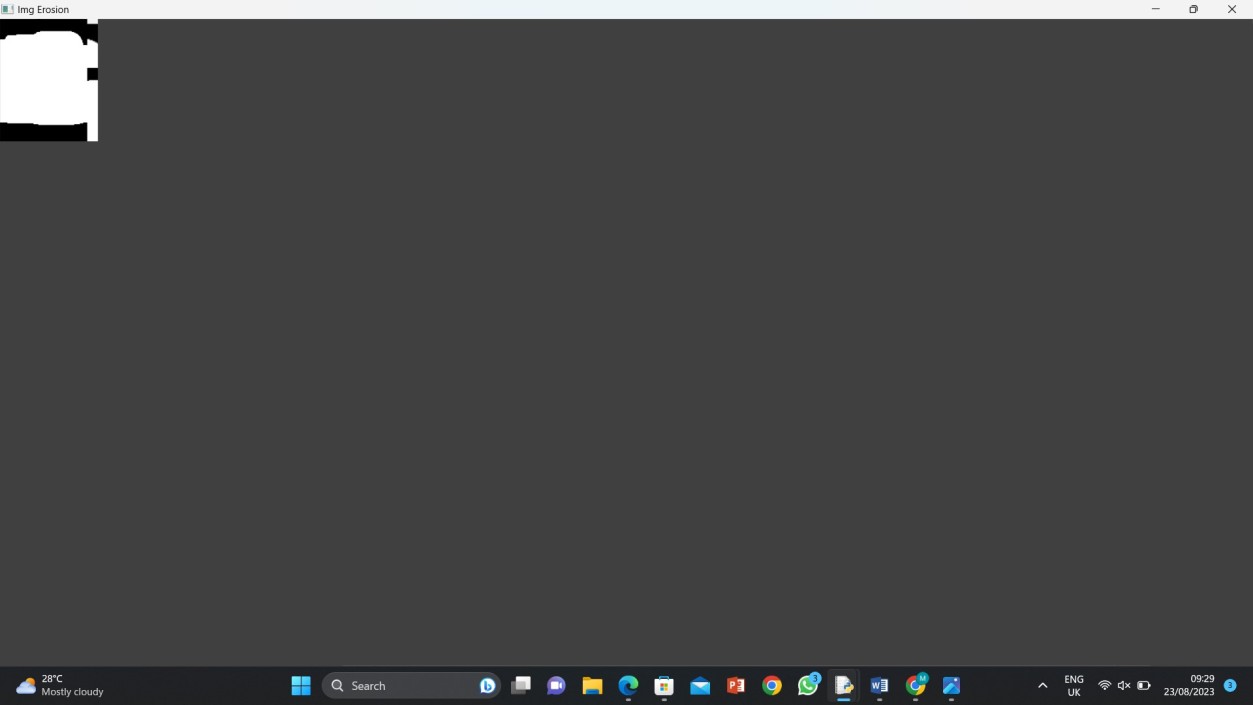
if cv2.waitKey(24) == 27: break

cap.release() cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Perform transformation using Homography matrix?

AIM:To perform transformation using Homography matric by using IDLE python Program:

import cv2

import numpy as np

im\_src = cv2.imread(r"C:\Users\DELL\Downloads\virat kohli.webp") pts\_src = np.array([[141, 131], [480, 159], [493, 630],[64, 601]])

im\_dst = cv2.imread(r"C:\Users\DELL\Downloads\virat kohli.webp") pts\_dst = np.array([[318, 256],[534, 372],[316, 670],[73, 473]])

h, status = cv2.findHomography(pts\_src, pts\_dst)

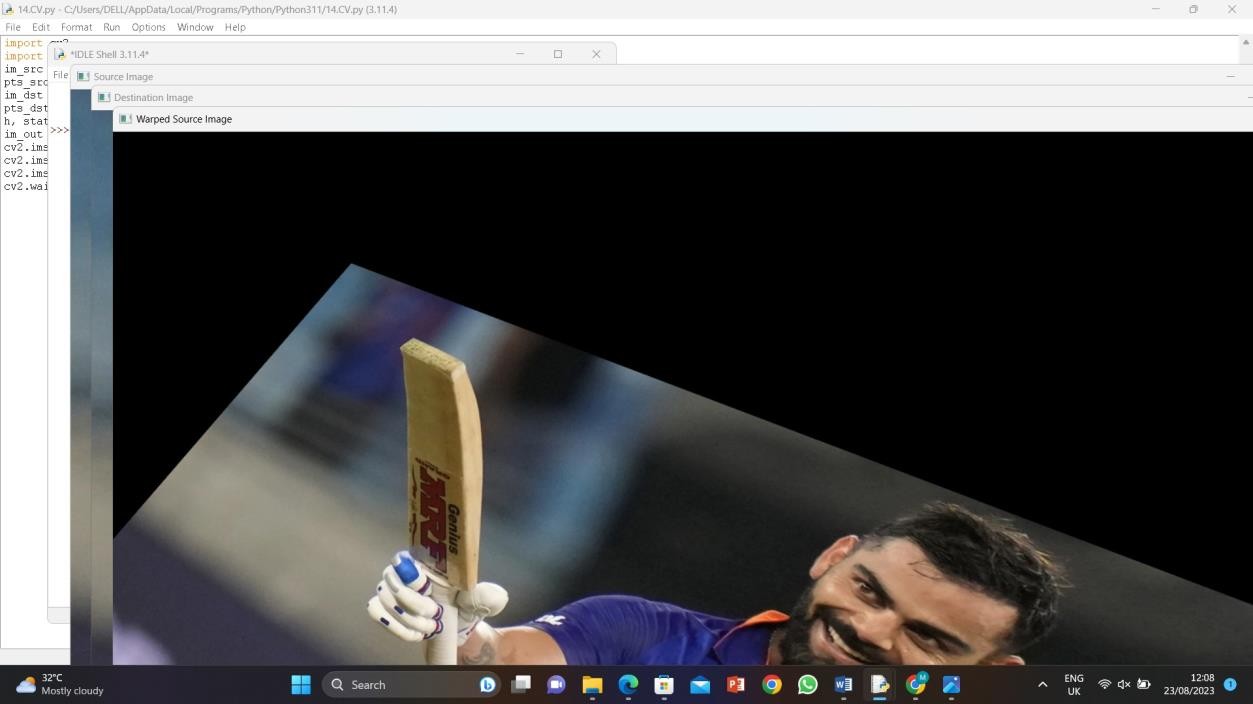
im\_out = cv2.warpPerspective(im\_src, h, (im\_dst.shape[1],im\_dst.shape[0])) cv2.imshow("Source Image", im\_src)

cv2.imshow("Destination Image", im\_dst) cv2.imshow("Warped Source Image", im\_out) cv2.waitKey(0)

INPUT:



OUTPUT:



1. Perform transformation using Direct Linear Transformation?

AIM:To perform transformation using the direct linear transformation in python Program:

import cv2

import numpy as np # Load images

img1 = cv2.imread(r"C:\Users\DELL\Downloads\game.jpg") img2 = cv2.imread(r"C:\Users\DELL\Downloads\game.jpg") # Define corresponding points

pts1 = np.array([[50, 50], [200, 50], [50, 200], [200, 200]])

pts2 = np.array([[100, 100], [300, 100], [100, 300], [300, 300]])

# Estimate projective transformation matrix using DLT H, \_ = cv2.findHomography(pts1, pts2)

# Apply projective transformation to img1

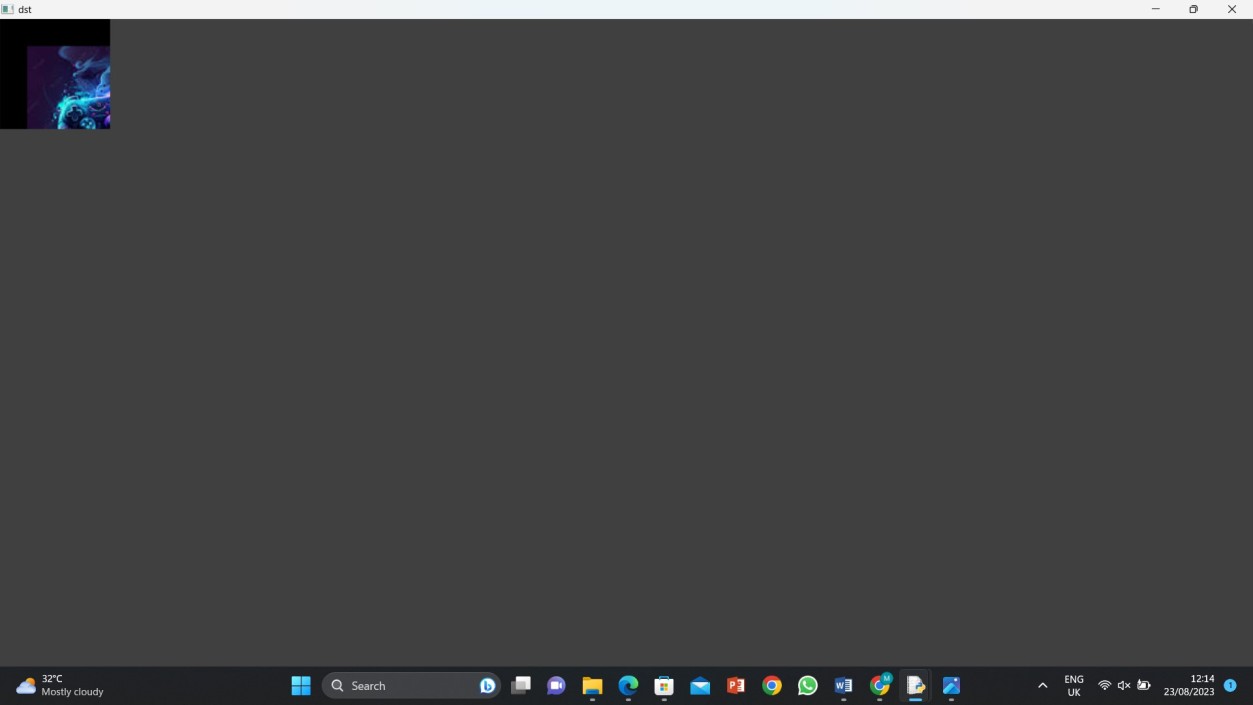
dst = cv2.warpPerspective(img1, H, (img2.shape[1], img2.shape[0])) # Display images

cv2.imshow('img1', img1) cv2.imshow('img2', img2) cv2.imshow('dst', dst) cv2.waitKey(0) cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Perform Edge detection using canny method?

AIM:To perform edge detection using the canny method in python programming Program:

import cv2

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana saved.jpg") cv2.imshow('Original',img)

cv2.waitKey(0)

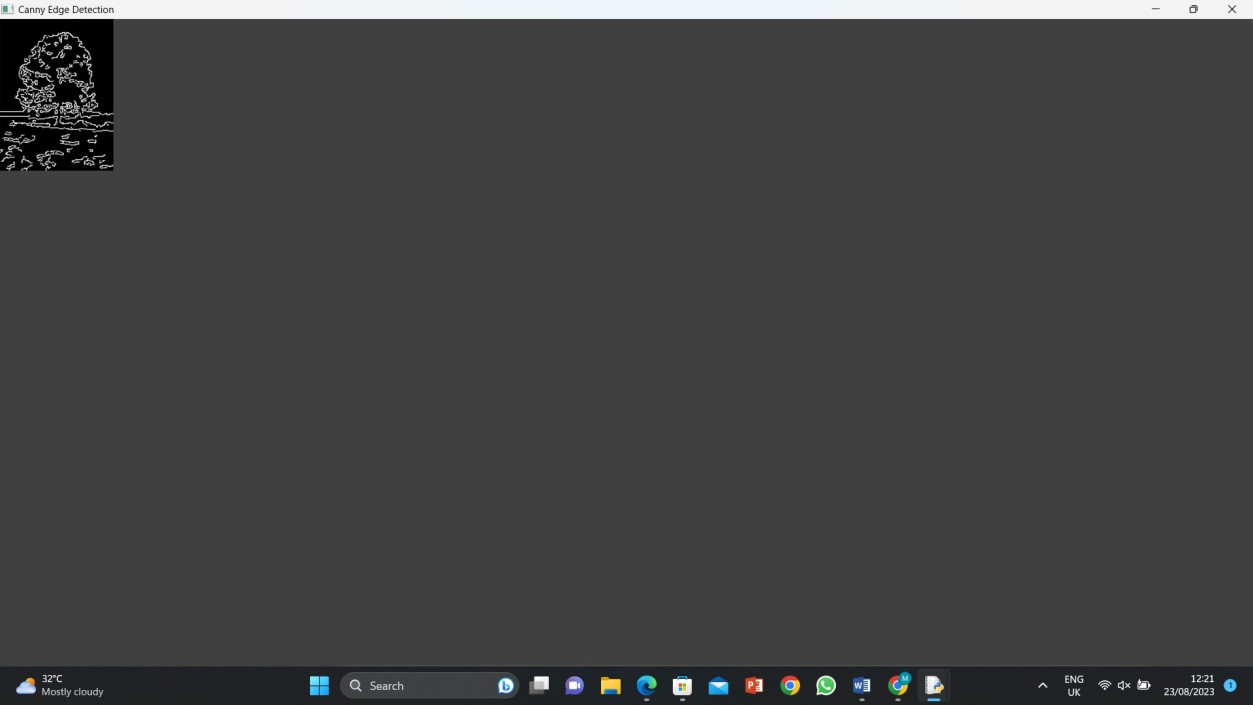
img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) img\_blur = cv2.GaussianBlur(img\_gray, (3,3), 0)

edges = cv2.Canny(image=img\_blur, threshold1=100, threshold2=200) # Canny Edge Detection cv2.imshow('Canny Edge Detection', edges)

cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Perform Edge detection using Sobel Matrix along X axis?

AIM:To perform edge detection using the sobel matric along in X-axis using python Program:

import cv2

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana saved pic.jpg") cv2.imshow('Original', img)

cv2.waitKey(0)

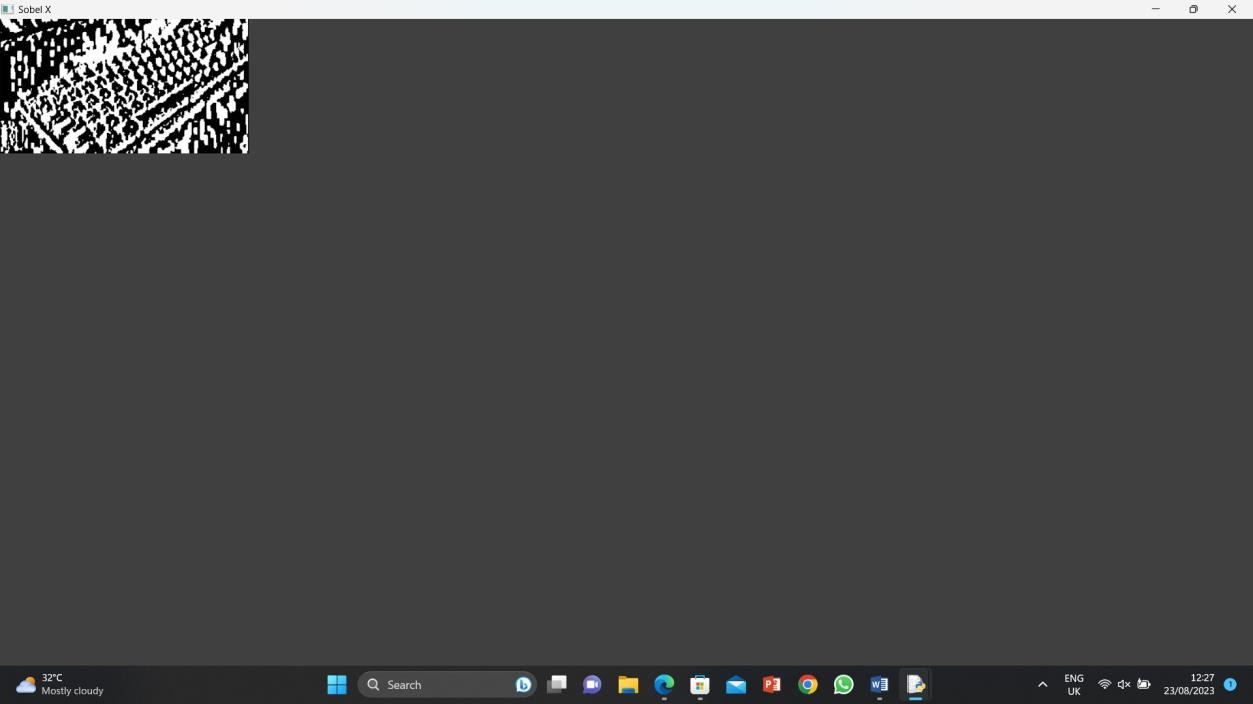
img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) # Blur the image for better edge detection

img\_blur = cv2.GaussianBlur(img\_gray, (3,3), 0)

sobelx = cv2.Sobel(src=img\_blur, ddepth=cv2.CV\_64F, dx=1, dy=0, ksize=5) cv2.imshow('Sobel X', sobelx)

cv2.waitKey(0) INPUT:



OUTPUT:

1. Perform Edge detection using Sobel Matrix along Y axis

AIM:To perform edge detection using sobel matrix along Y- axis using python Program:

import cv2

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana saved pic.jpg") cv2.imshow('Original',img)

cv2.waitKey(0)

img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) img\_blur = cv2.GaussianBlur(img\_gray, (3,3), 0)

sobely = cv2.Sobel(src=img\_blur, ddepth=cv2.CV\_64F, dx=0, dy=1, ksize=5) cv2.imshow('Sobel Y', sobely)

cv2.waitKey(0) INPUT:



OUTPUT:



1. Perform Edge detection using Sobel Matrix along XY axis?

AIM:To perform edge detection using sobel matrix along with XY-axis using python Program:

import cv2

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana saved pic.jpg") # Display original image

cv2.imshow('Original', img) cv2.waitKey(0)

img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) # Blur the image for better edge detection

img\_blur = cv2.GaussianBlur(img\_gray, (3,3), 0)

sobelxy = cv2.Sobel(src=img\_blur, ddepth=cv2.CV\_64F, dx=1, dy=1, ksize=5) cv2.imshow('Sobel X Y using Sobel() function', sobelxy)

cv2.waitKey(0) INPUT:



OUTPUT:



1. Perform Sharpening of Image using Laplacian mask with negative center coefficient?

AIM:To perform sharpening of image using Laplacian mask with negative center coefficient using IDLE python

Program:

import cv2

import numpy as np

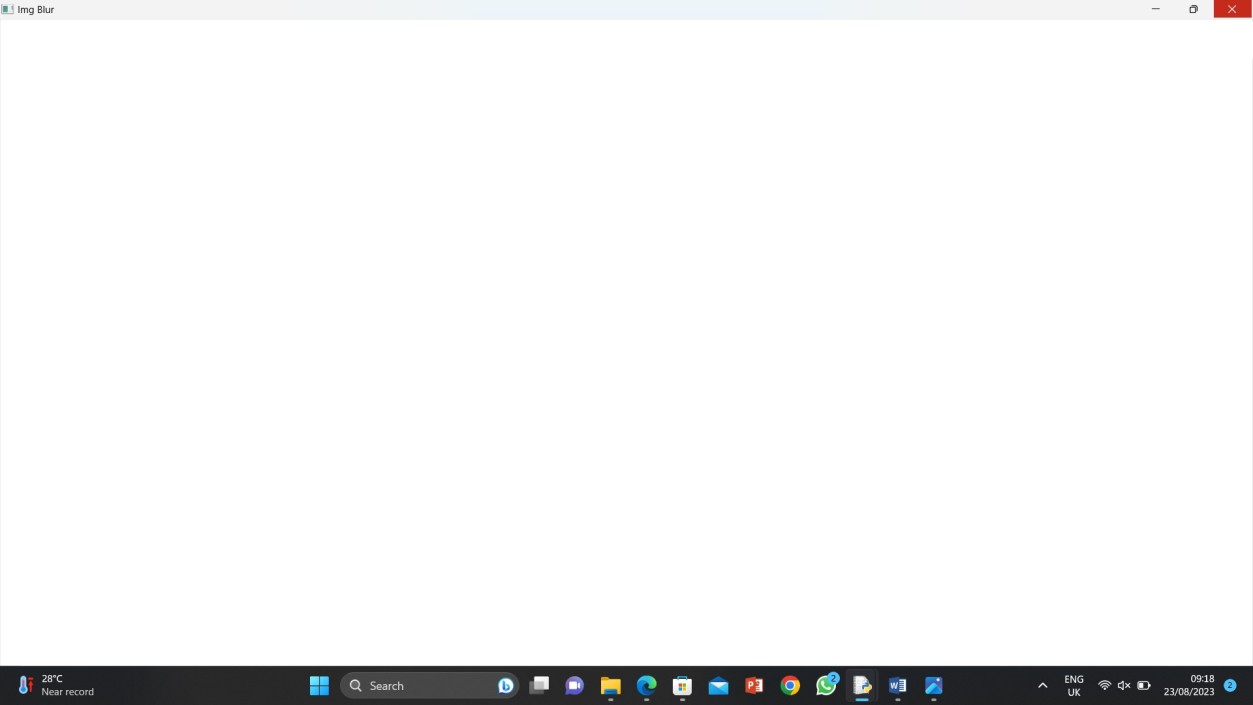
img = cv2.imread(r"C:\Users\DELL\Downloads\meghana.jpg") gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

kernel = np.array([[0,1,0], [1,-8,1], [0,1,0]]) sharpened = cv2.filter2D(gray, -1, kernel) cv2.imshow('Original', gray) cv2.imshow('Sharpened', sharpened) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Perform Sharpening of Image using Laplacian mask implemented with an extension of diagonal neighbors

AIM:To perform sharpening of image using Laolacian mask implemented with an extension of diagonal neighbors using python

Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\maggi.jpg") gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

kernel = np.array([[0,1,0], [1,-4,1], [0,1,0]]) sharpened = cv2.filter2D(gray, -1, kernel) cv2.imshow('Original', gray) cv2.imshow('Sharpened', sharpened) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Perform Sharpening of Image using Laplacian mask with positive center coefficient?

AIM:To Perform Sharpening of Image using Laplacian mask with positive center coefficient using IDLE python

Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\maggi paintings.jpg") img = cv2.resize(img,(255, 255))

gray\_img = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

# Apply the Laplacian filter with a positive center coefficient laplacian\_kernel = np.array([[0, -1, 0], [-1, 5, -1], [0, -1, 0]]) sharpened\_img = cv2.filter2D(gray\_img, -1, laplacian\_kernel) sharpened\_img = cv2.cvtColor(sharpened\_img, cv2.COLOR\_GRAY2BGR) cv2.imshow('Original Image', img)

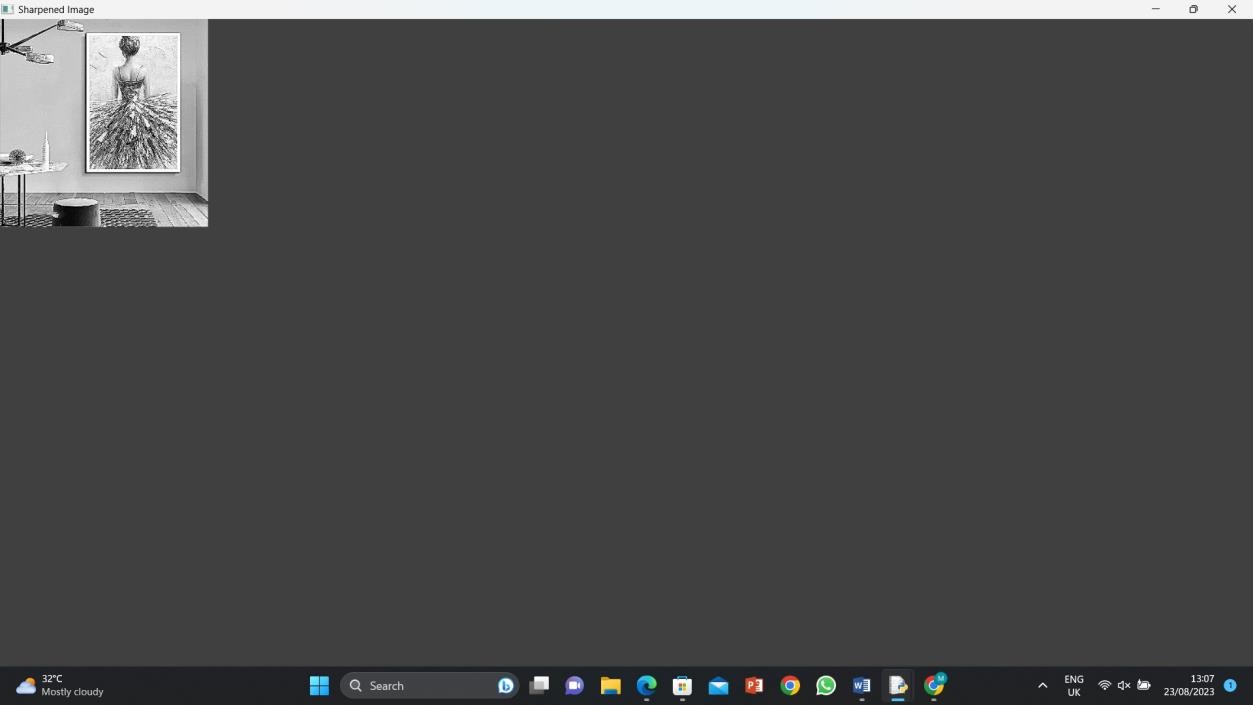
cv2.imshow('Sharpened Image', sharpened\_img) cv2.waitKey(0)

cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Perform Sharpening of Image using unsharp masking?

AIM:To perform sharpening of image using unsharp masking using IDLE python Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana saved pictures.jpg") gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

laplacian\_kernel = np.array([[0, 1, 0],

[1, -4, 1],

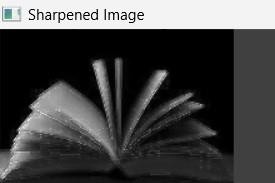
[0, 1, 0]])

laplacian = cv2.filter2D(gray, -1, laplacian\_kernel) sharpened = cv2.add(gray, laplacian) cv2.imshow('Original Image',gray) cv2.imshow('Sharpened Image',sharpened) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Perform Sharpening of Image using High-Boost Masks?

AIM:To perform the sharpening of an image using High-Boost masks in python language Program:

import cv2

import numpy as np

image\_path = r"C:\Users\DELL\Downloads\meghana saved pictures.jpg" original\_image = cv2.imread(image\_path)

kernel = np.array([[-1, -1, -1],

[-1, 9, -1],

[-1, -1, -1]])

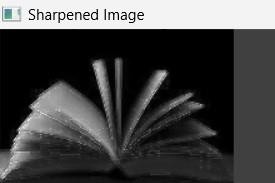
sharpened\_image = cv2.filter2D(original\_image, -1, kernel)

cv2.imshow('Original Image', original\_image) cv2.imshow('Sharpened Image', sharpened\_image)

cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Perform Sharpening of Image using Gradient masking?

AIM:To perform sharpening of image using gradient masking in python Program:

import cv2

import numpy as np

import matplotlib.pyplot as plt

def sharpen\_image\_with\_gradient(image\_path, alpha=1.5):

original\_image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE) gradient\_x = cv2.Sobel(original\_image, cv2.CV\_64F, 1, 0, ksize=3) gradient\_y = cv2.Sobel(original\_image, cv2.CV\_64F, 0, 1, ksize=3) gradient\_magnitude = np.sqrt(gradient\_x\*2 + gradient\_y\*2)

gradient\_magnitude = cv2.normalize(gradient\_magnitude, None, 0, 255, cv2.NORM\_MINMAX, cv2.CV\_8U)

sharpened\_image = cv2.addWeighted(original\_image, 1 + alpha, gradient\_magnitude, -alpha, 0) return original\_image, sharpened\_image

image\_path = r"C:\Users\DELL\Downloads\meghana saved pictures.jpg "

original\_image, sharpened\_image = sharpen\_image\_with\_gradient(image\_path, alpha=1.5) plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1) plt.imshow(original\_image, cmap='gray') plt.title('Original Image')

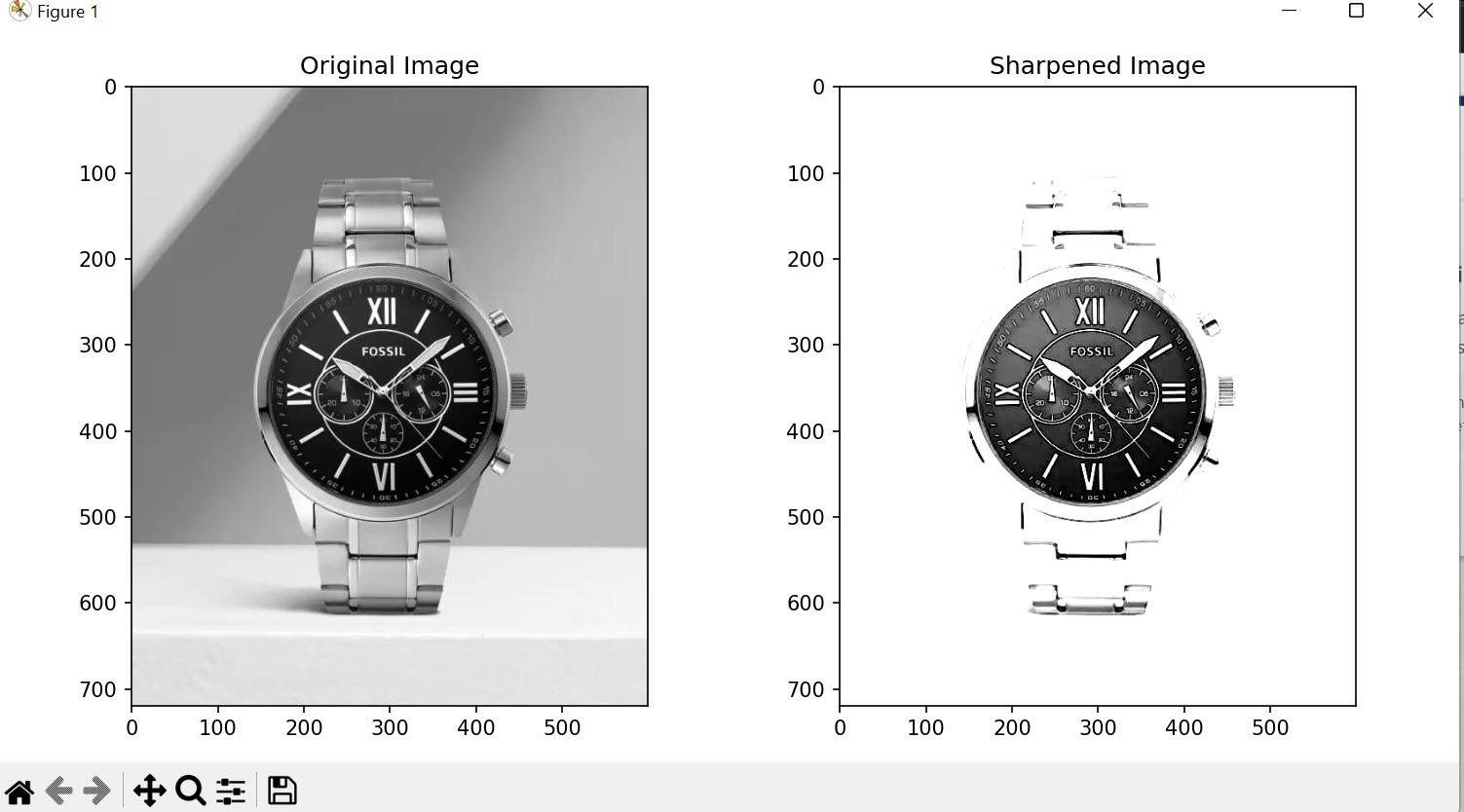
plt.subplot(1, 2, 2) plt.imshow(sharpened\_image, cmap='gray') plt.title('Sharpened Image') plt.tight\_layout()

plt.show()

INPUT:



OUTPUT:



1. Insert water marking to the image using OpenCV?

AIM:To insert water marking to the image using openCV using python Program:

import cv2

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana download.jpg") wm = cv2.imread(r"C:\Users\DELL\Downloads\meghana download.jpg") h\_wm, w\_wm = wm.shape[:2]

h\_img, w\_img = img.shape[:2] center\_x = int(w\_img/2) center\_y = int(h\_img/2)

top\_y = center\_y - int(h\_wm/2) left\_x = center\_x - int(w\_wm/2) bottom\_y = top\_y + h\_wm right\_x = left\_x + w\_wm

roi = img[top\_y:bottom\_y, left\_x:right\_x] result = cv2.addWeighted(roi, 1, wm, 0.3, 0) img[top\_y:bottom\_y, left\_x:right\_x] = result cv2.imshow("Watermarked Image", img) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Do Cropping, Copying and pasting image inside another image using OpenCV?

AIM:To do cropping,copying and pasting images inside another image using OpenCV in IDLE python Program:

import cv2

import numpy as np

image = cv2.imread(r"C:\Users\DELL\Downloads\maggii.jpg") img2 = cv2.imread(r'C:\Users\DELL\Downloads\maggii.jpg') print(image.shape) # Print image shape cv2.imshow("original", image)

imageCopy = image.copy()

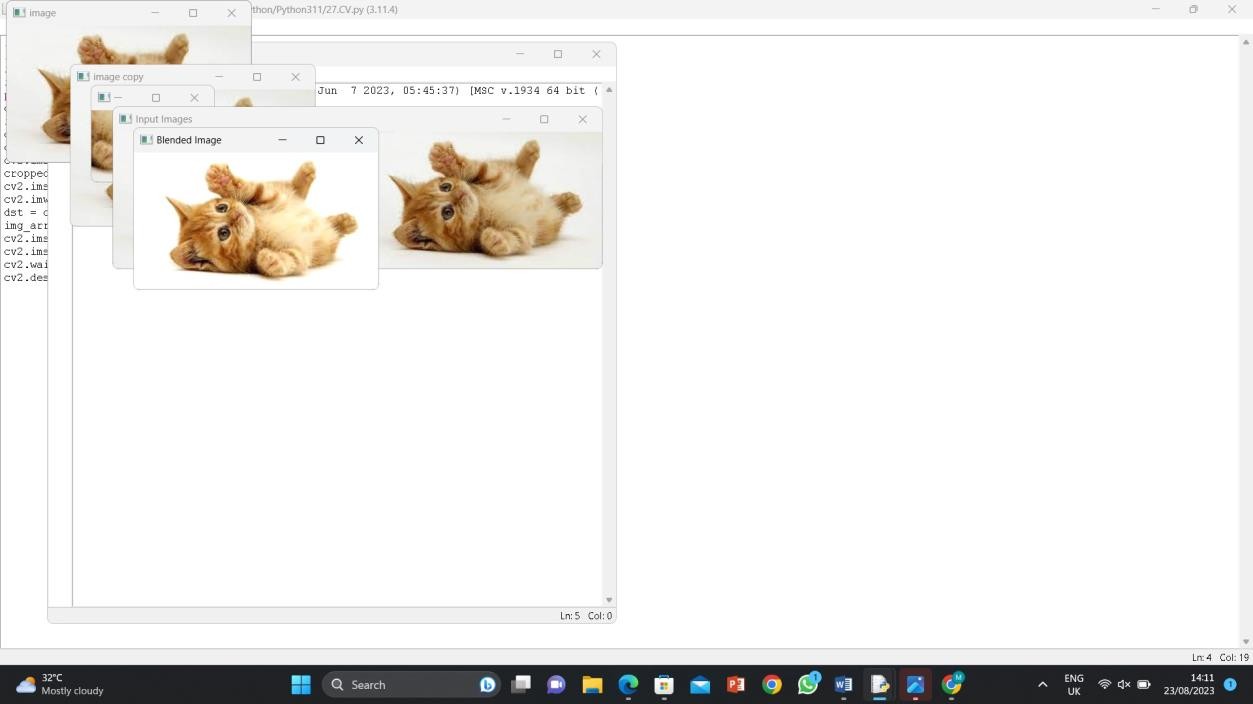
cv2.circle(imageCopy, (100, 100), 30, (255, 0, 0), -1) cv2.imshow('image', image)

cv2.imshow('image copy', imageCopy) cropped\_image = image[80:280, 150:330] cv2.imshow("cropped", cropped\_image) cv2.imwrite("Cropped Image.jpg", cropped\_image) dst = cv2.addWeighted(image, 0.5, img2, 0.7, 0) img\_arr = np.hstack((image, img2)) cv2.imshow('Input Images',img\_arr) cv2.imshow('Blended Image',dst)

cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Find the boundary of the image using Convolution kernel for the given image?

AIM;To find the boundary of the image using Convolution kernel for the given image using IDLE python

Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\meghanaa.jpg", cv2.IMREAD\_GRAYSCALE)

dx = cv2.Sobel(img, cv2.CV\_64F, 1, 0)

dy = cv2.Sobel(img, cv2.CV\_64F, 0, 1) edges = cv2.magnitude(dx, dy) thresh = 100

edges[edges < thresh] = 0 edges[edges >= thresh] = 255 cv2.imshow("Edges", edges) cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Erosion technique?

AIM:To do the Morphological operations based on OpenCV using Erosion technique in python Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\images.jpg", cv2.IMREAD\_GRAYSCALE) kernel = np.ones((5,5), np.uint8)

erosion = cv2.erode(img, kernel, iterations=1) cv2.imshow("Original", img) cv2.imshow("Erosion", erosion) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Dilation technique?

AIM:To do the Morphological operations based on OpenCV using Dilation technique using python program

Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\images.jpg", cv2.IMREAD\_GRAYSCALE) kernel = np.ones((5,5), np.uint8)

dilation = cv2.dilate(img, kernel, iterations=1) cv2.imshow("Original", img) cv2.imshow("Dilation", dilation) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Opening technique?

AIM:To do the Morphological operations based on OpenCV using Opening technique using python Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\images.jpg", cv2.IMREAD\_GRAYSCALE) kernel = np.ones((5,5), np.uint8)

opening = cv2.morphologyEx(img, cv2.MORPH\_OPEN, kernel) cv2.imshow("Original", img)

cv2.imshow("opening", opening) cv2.waitKey(0) cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Closing technique?

AIM:To do the Morphological operations based on OpenCV using Closing technique using python Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\images.jpg", cv2.IMREAD\_GRAYSCALE) kernel = np.ones((5,5), np.uint8)

closing = cv2.morphologyEx(img, cv2.MORPH\_CLOSE, kernel) cv2.imshow("Original", img)

cv2.imshow("Closing", closing) cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Morphological Gradient technique?

AIM:To do the Morphological operations based on OpenCV using Morphological Gradient technique using python

Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\images.jpg", cv2.IMREAD\_GRAYSCALE) kernel = np.ones((5,5), np.uint8)

grad = cv2.morphologyEx(img, cv2.MORPH\_GRADIENT, kernel) cv2.imshow("Original", img)

cv2.imshow("Gradient", grad) cv2.waitKey

INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Top hat technique?

AIM:To do the Morphological operations based on OpenCV using Top hat technique using python Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana22.jpg", cv2.IMREAD\_GRAYSCALE)

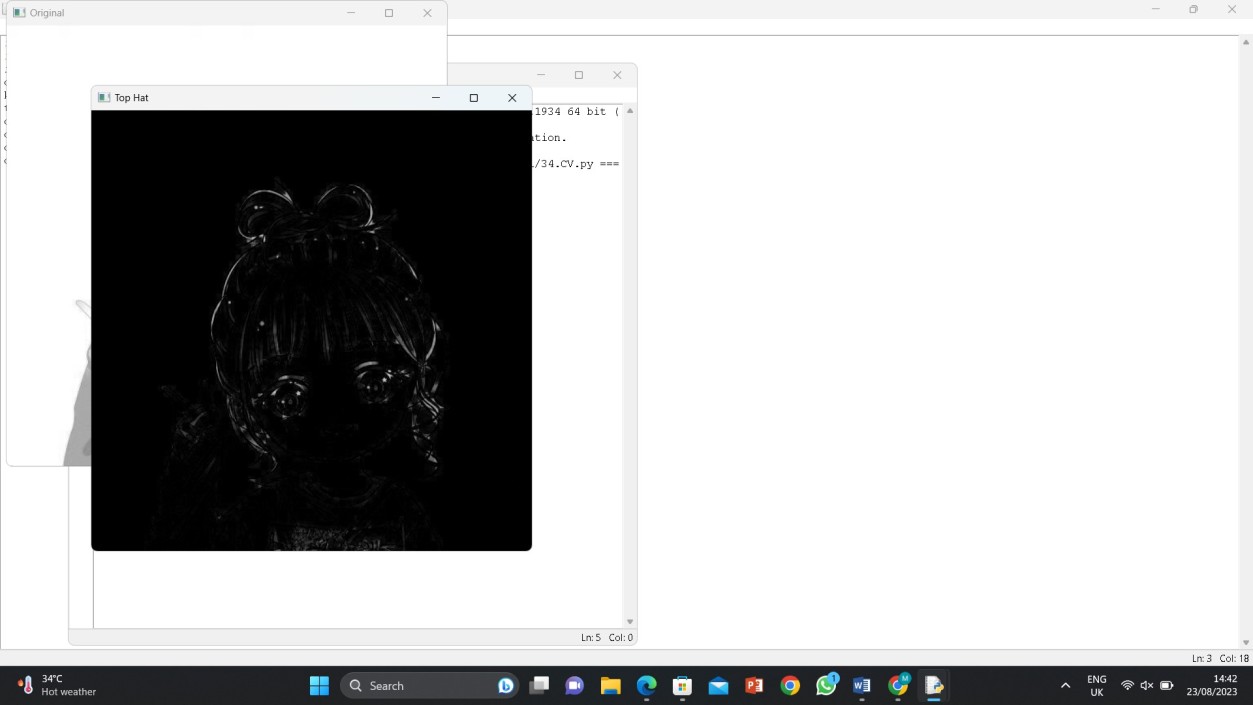
kernel = np.ones((5,5), np.uint8)

tophat = cv2.morphologyEx(img, cv2.MORPH\_TOPHAT, kernel) cv2.imshow("Original", img)

cv2.imshow("Top Hat", tophat) cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Morphological operations based on OpenCV using Black hat technique?

AIM:To do the Morphological operations based on OpenCV using Black hat technique using python language

Program:

import cv2

import numpy as np

img = cv2.imread(r"C:\Users\DELL\Downloads\meghana22.jpg", cv2.IMREAD\_GRAYSCALE)

kernel = np.ones((5,5), np.uint8)

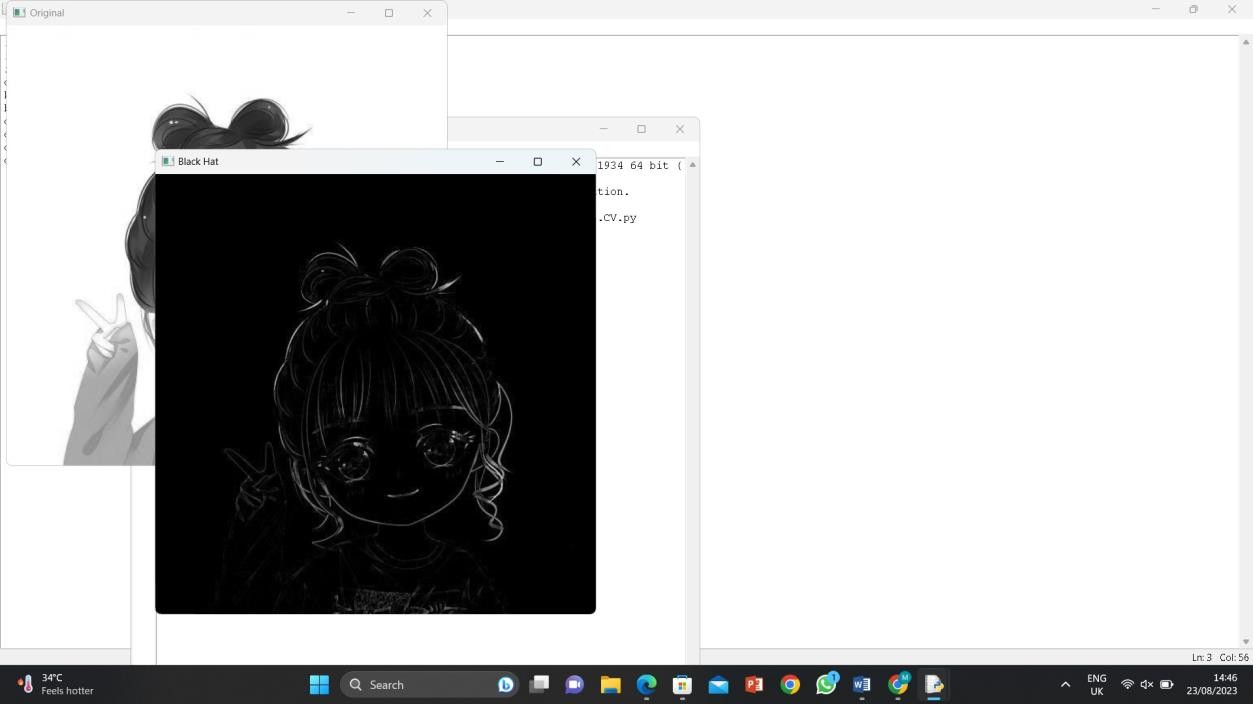
blackhat = cv2.morphologyEx(img, cv2.MORPH\_BLACKHAT, kernel) cv2.imshow("Original", img)

cv2.imshow("Black Hat", blackhat) cv2.waitKey(0) cv2.destroyAllWindows()

INPUT:



OUTPUT:



1. Recognise watch from the given image by general Object recognition using OpenCV?

AIM:To Recognise watch from the given image by general Object recognition using OpenCV using python

Program:

import cv2

watch\_cascade = cv2.CascadeClassifier(r" C:\Users\DELL\Downloads\meghana saved.webp") img = cv2.imread(r"C:\Users\DELL\Downloads\meghana saved.webp")

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

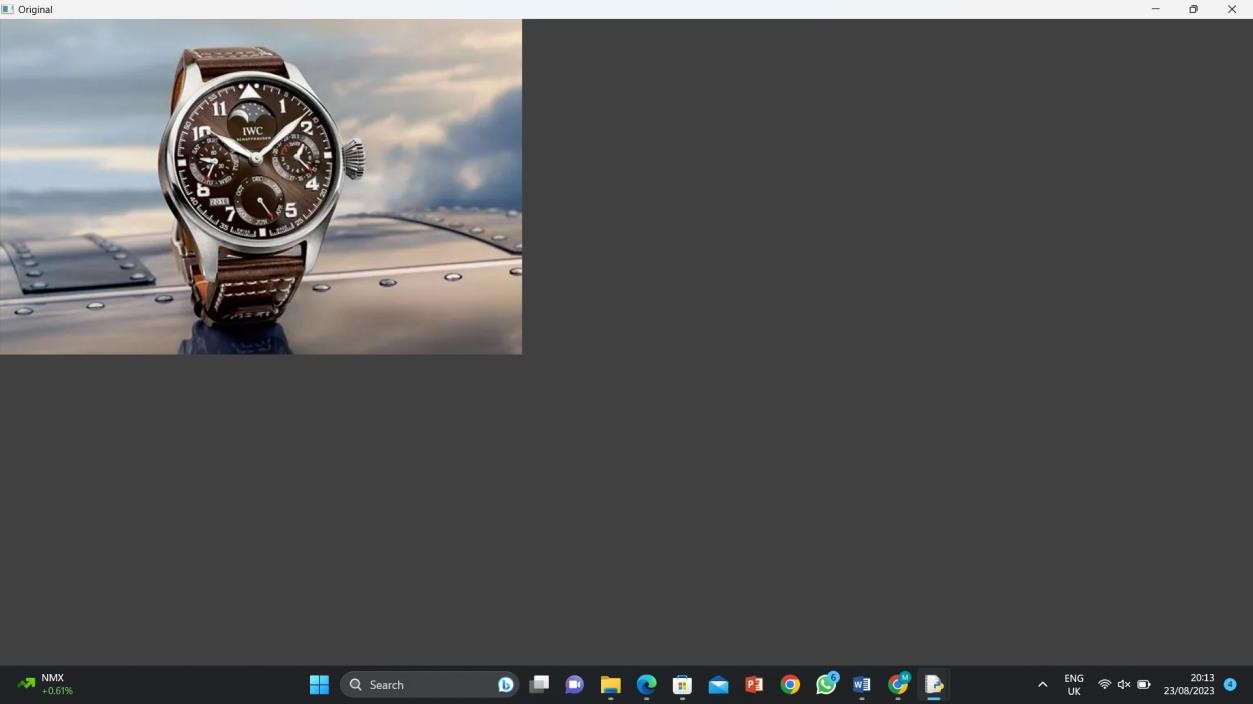
watches = watch\_cascade.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5) for (x, y, w, h) in watches:

cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) cv2.imshow('Watches Detected', img) cv2.waitKey(0)

cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Using Opencv play Video in Reverse model?

AIM:By using opencv play video in reverse model by using IDLE python Program:

import cv2

cap = cv2.VideoCapture(r"C:\Users\DELL\Downloads\meghana pic.jpg") total\_frames = cap.get(cv2.CAP\_PROP\_FRAME\_COUNT)

current\_frame = total\_frames - 1

while current\_frame >= 0: cap.set(cv2.CAP\_PROP\_POS\_FRAMES, current\_frame) ret, frame = cap.read()

if not ret:

break

cv2.imshow('Video in Reverse', frame)

if cv2.waitKey(25) & 0xFF == ord('q'): break

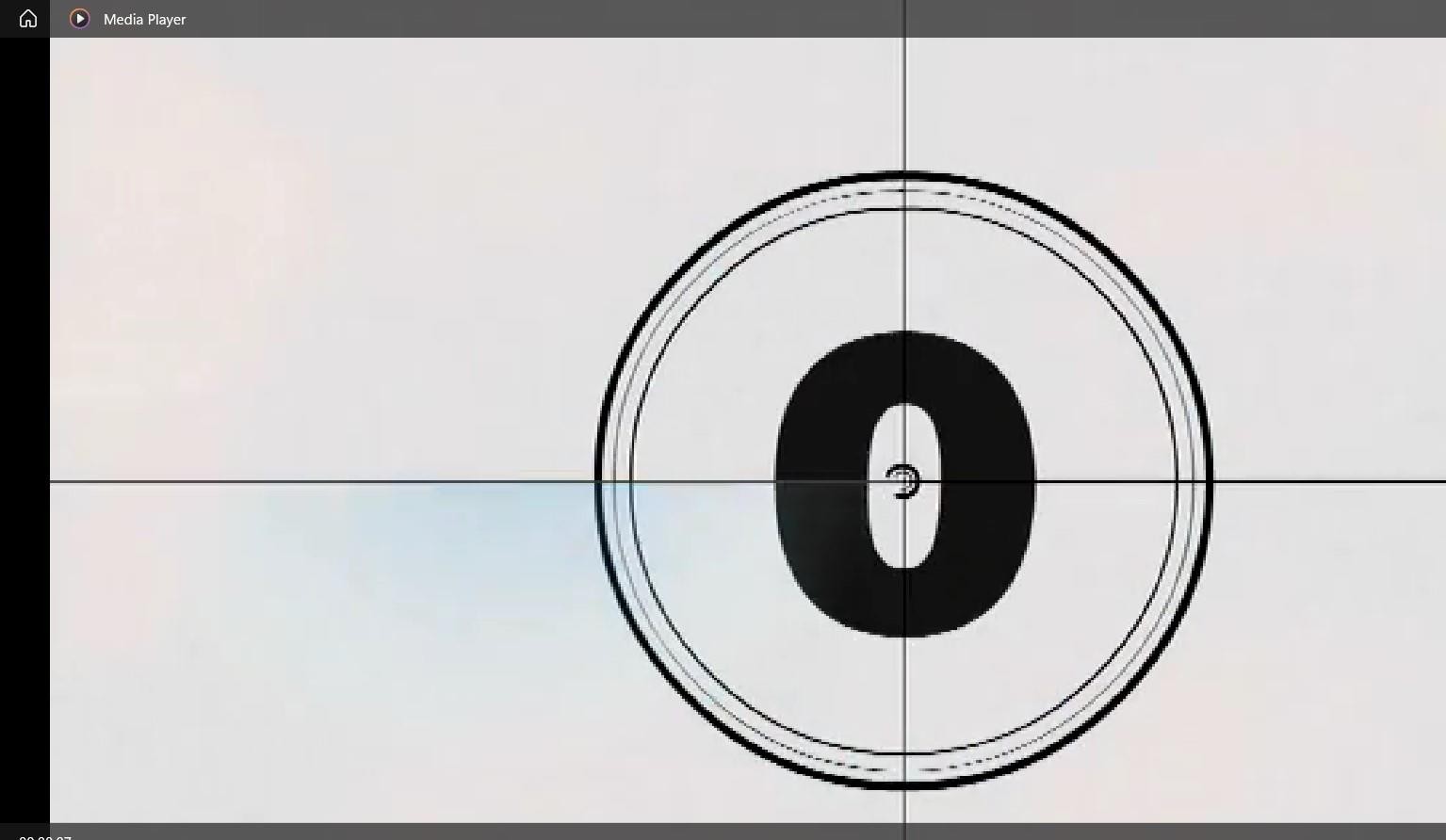
current\_frame -= 1

cap.release() cv2.destroyAllWindows() INPUT:



countdown\_-\_2637 (360p).mp4

OUTPUT:



1. Face Detection using Opencv?

AIM:Face detection using opencv by using IDLE python Program:

import cv2

img = cv2.imread(r"C:\Users\DELL\Downloads\WhatsApp Image 2023-01-24 at 2.27.51 PM (1).jpeg") gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

face\_cascade = cv2.CascadeClassifier(r"C:\Users\DELL\Downloads\WhatsApp Image 2023-01-24 at

2.27.51 PM (1).jpeg")

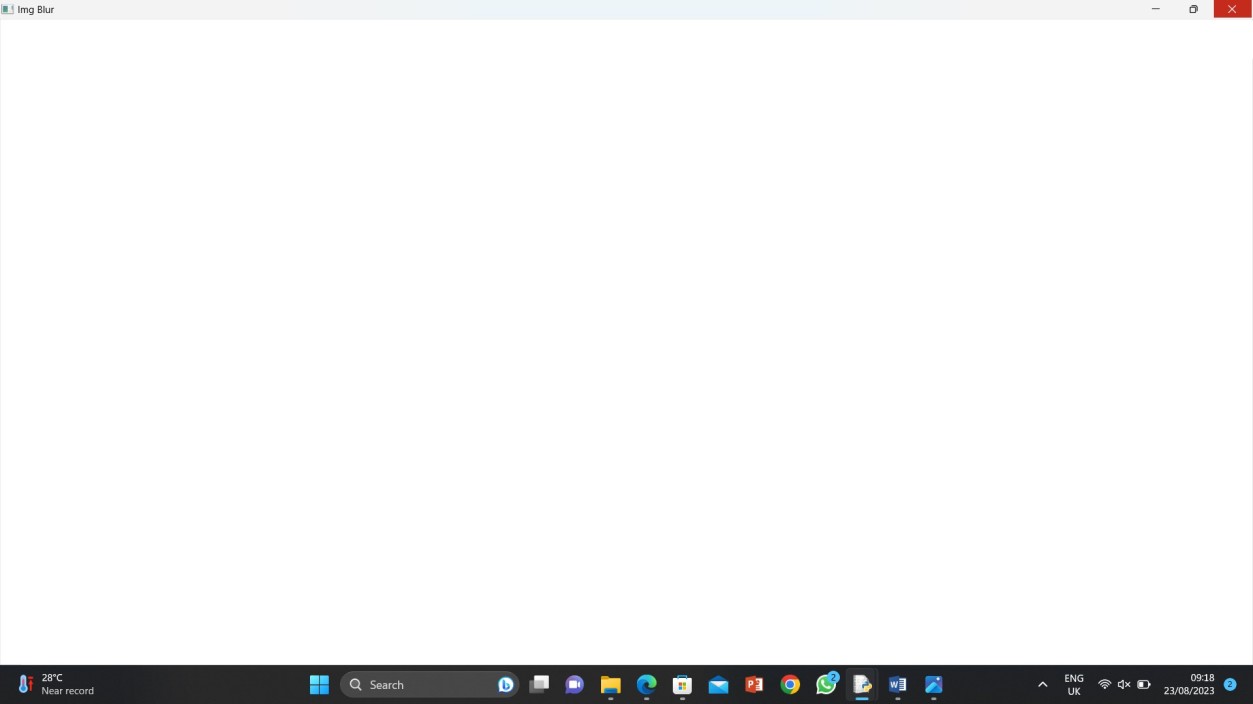
faces = face\_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5) for (x, y, w, h) in faces:

cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2) cv2.imshow('Faces Detected', img)

cv2.waitKey(0) cv2.destroyAllWindows() INPUT:



OUTPUT:



1. Vehicle Detection in a Video frame using OpenCV?

AIM:For vehicle detection in a video frame using OpenCV by using the IDLE python Program: