Image Processing Code Explanation

# Importing Libraries

import cv2  
import numpy as np  
from matplotlib import pyplot as plt

### Explanation:

These libraries are imported to work with images. 'cv2' is OpenCV used for computer vision tasks, 'numpy' is for array handling, and 'matplotlib.pyplot' is for displaying images.

# Reading and Saving an Image

img = cv2.imread(r'C:\Users\Administrator\OneDrive\Desktop\PAI TASKS\Syed Hassan.jpg')\ncv2.imwrite(r'C:\Users\Administrator\OneDrive\Desktop\PAI TASKS\saved\_image.jpg', img)\ncv2.imshow('Image', img)\ncv2.waitKey(0)\ncv2.destroyAllWindows()

### Explanation:

The code reads an image from the specified location, saves it under a new name, and then displays it. The window waits for a key press before closing.

# Resizing Image

resized = cv2.resize(img, (300, 200))  
cv2.imshow('Resized', resized)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

The image is resized to 300x200 pixels and displayed.

# Converting to Grayscale

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  
cv2.imshow('Gray', gray)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

This converts the original image to grayscale and displays it.

# Drawing Shapes

img2 = img.copy()  
cv2.rectangle(img2, (50, 50), (200, 200), (255, 0, 0), 3)  
cv2.circle(img2, (150, 150), 50, (0, 255, 0), -1)  
cv2.imshow('Shapes', img2)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

Draws a blue rectangle and a green filled circle on a copy of the image.

# Blurring Image

blur = cv2.GaussianBlur(img, (5, 5), 0)  
cv2.imshow('Blurred', blur)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

Applies Gaussian Blur to smooth the image and reduce noise.

# Thresholding

\_, thresh = cv2.threshold(gray, 127, 255, cv2.THRESH\_BINARY)  
cv2.imshow('Threshold', thresh)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

Converts the grayscale image to a binary (black and white) image using a threshold.

# Edge Detection

edges = cv2.Canny(img, 100, 200)  
cv2.imshow('Edges', edges)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

Detects edges in the image using the Canny algorithm.

# Finding and Drawing Contours

contours\_img = img.copy()  
contours, \_ = cv2.findContours(thresh, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
cv2.drawContours(contours\_img, contours, -1, (0, 255, 0), 2)  
cv2.imshow('Contours', contours\_img)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

Finds contours in the thresholded image and draws them on a copy of the original image.

# Face Detection

face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')  
face\_img = img.copy()  
faces = face\_cascade.detectMultiScale(gray, 1.1, 4)  
for (x, y, w, h) in faces:  
 cv2.rectangle(face\_img, (x, y), (x+w, y+h), (255, 0, 0), 2)  
cv2.imshow('Face Detection', face\_img)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

### Explanation:

Uses a pretrained Haar cascade to detect faces and draws rectangles around them.

**OUTPUT**

