FACE DETECTION IN THE STATIC IMAGE

```
import cv2

# Load the Haar cascade for face detection

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')


# Read the input image (update the path to your image)

image_path = r"C:\Users\admin\Desktop\abdul kalam.jpg"  # Replace with your actual image path


# Load the image

image = cv2.imread(image_path)


# Check if the image was loaded successfully

if image is None:

    print("Error: Could not read the image. Please check the file path or format.")

else:

    # Convert the image to grayscale

    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)


    # Detect faces in the image

    faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5)


    # Draw rectangles around detected faces

    for (x, y, w, h) in faces:

        cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)


    # Display the output image with detected faces

    cv2.imshow('Detected Faces', image)

    # Wait for a key press and close the displayed image

    cv2.waitKey(0)

    cv2.destroyAllWindows()
```

FACE DETECTION IN THE REAL TIME VIDEO FRAME

```
import cv2

# Load the Haar cascade
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

# Start capturing video from webcam
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()  # Read a frame
    if not ret:
        break

    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)  # Convert to grayscale
    faces = face_cascade.detectMultiScale(gray_frame, scaleFactor=1.1, minNeighbors=2)

    # Draw rectangles around detected faces
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)

    cv2.imshow('Detected Faces', frame)

    # Exit loop if 'q' is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```