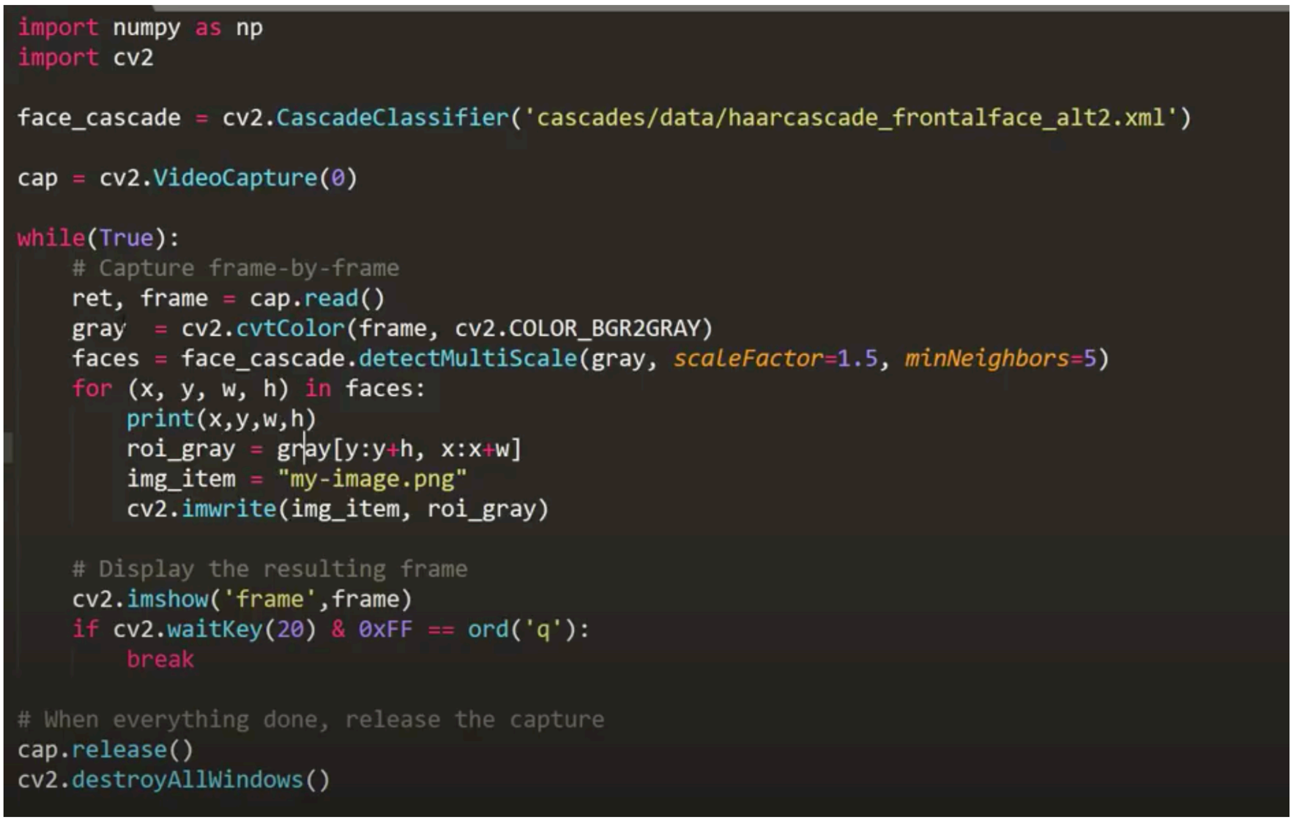
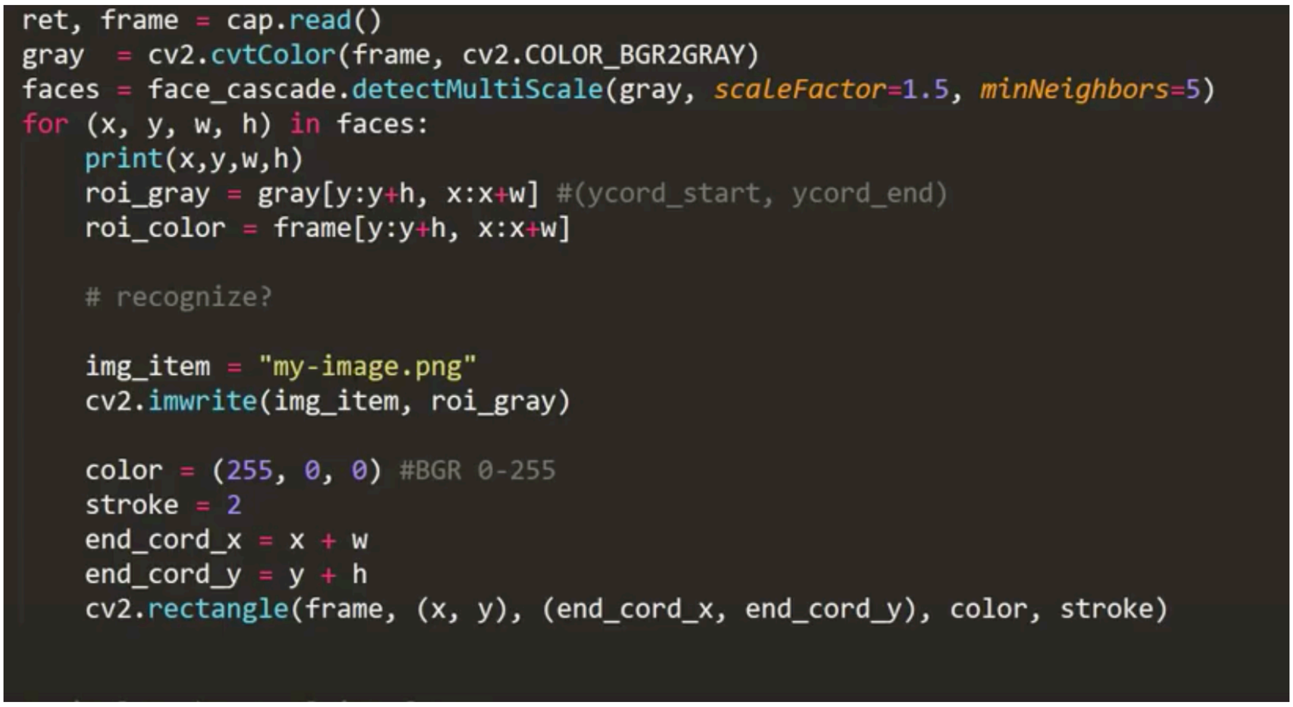


```python
import numpy as np
import cv2

face_cascade = cv2.CascadeClassifier('cascades/data/haarcascade_frontalface_alt2.xml')

cap = cv2.VideoCapture(0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()
    gray  = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.5, minNeighbors=5)
    for (x, y, w, h) in faces:
        print(x,y,w,h)
        roi_gray = gray[y:y+h, x:x+w]
        img_item = "my-image.png"
        cv2.imwrite(img_item, roi_gray)

    # Display the resulting frame
    cv2.imshow('frame',frame)
    if cv2.waitKey(20) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

We read the face. We convert the frame in color grey. detectMultiScale to find all the faces in the frame. Then we print all the values and we save the image in gray.gray[,] are the coordinates of the face. (ycord_start,ycord_end) taking into account the height and the width.



```python
ret, frame = cap.read()
gray  = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.5, minNeighbors=5)
for (x, y, w, h) in faces:
    print(x,y,w,h)
    roi_gray = gray[y:y+h, x:x+w] #(ycord_start, ycord_end)
    roi_color = frame[y:y+h, x:x+w]

    # recognize?

    img_item = "my-image.png"
    cv2.imwrite(img_item, roi_gray)

    color = (255, 0, 0) #BGR 0-255
    stroke = 2
    end_cord_x = x + w
    end_cord_y = y + h
    cv2.rectangle(frame, (x, y), (end_cord_x, end_cord_y), color, stroke)
```

Draws a rectangle in the coordinate cv2.rectangle(, , , ,)