Basic Python Programming

[Session 3] Lab session

ITinerary X University of Ghana

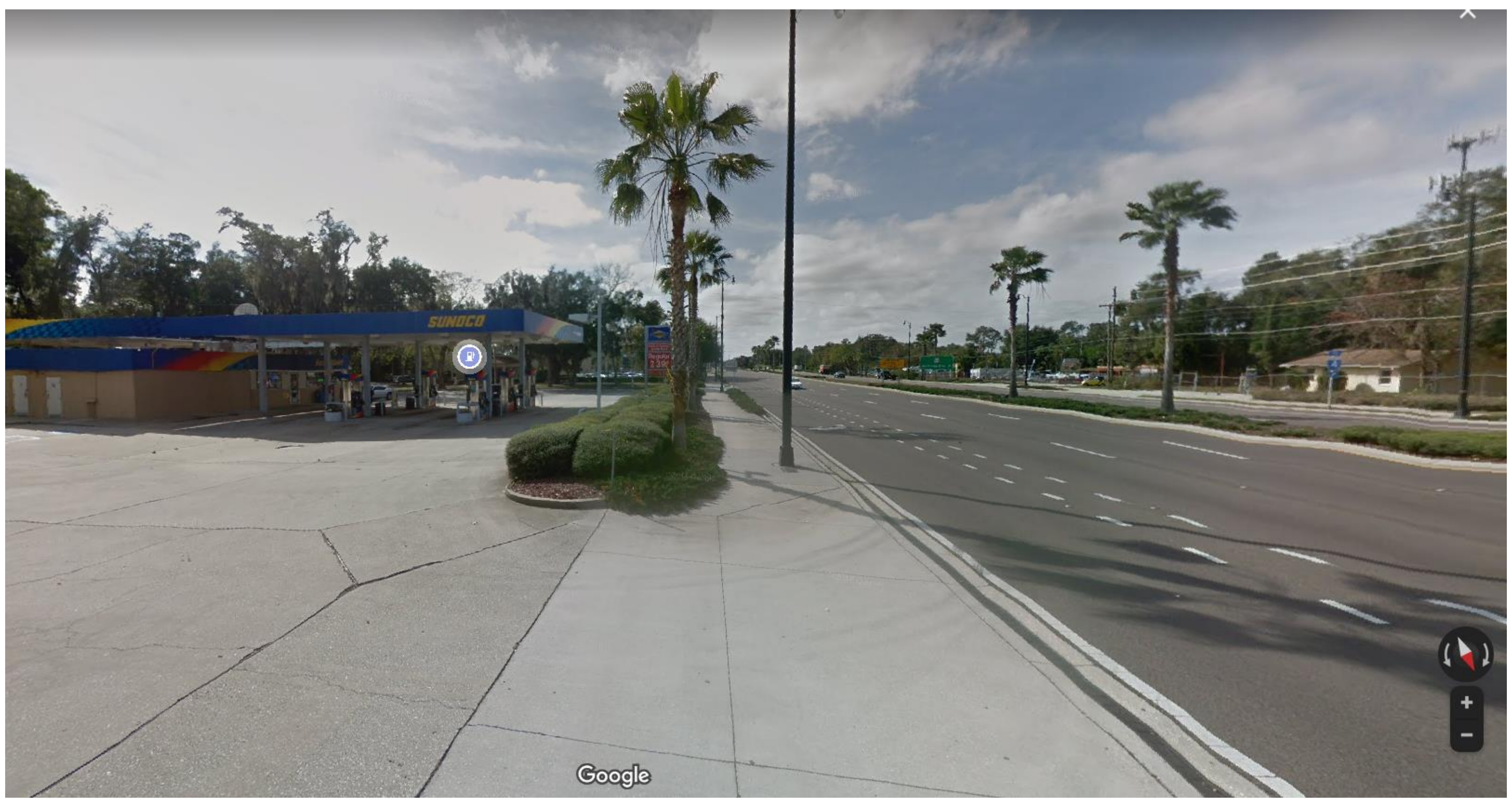
Contents

- **Facial recognition: Pixelation**

Facial Recognition

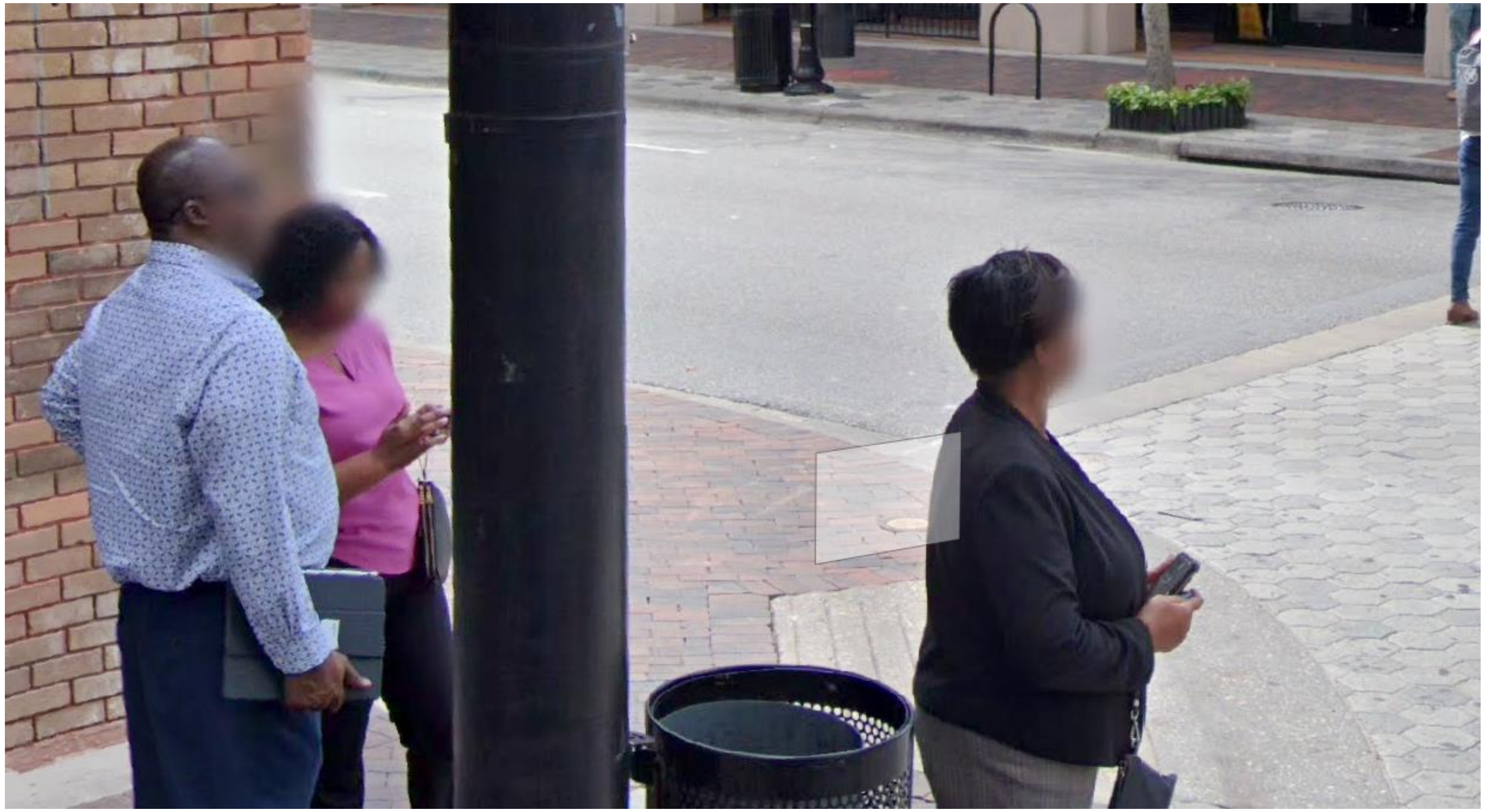
Piexlation [1]

- Google map provides “Street view” service



Piexlation [2]

- **You can see that the face is blured (for privacy)**

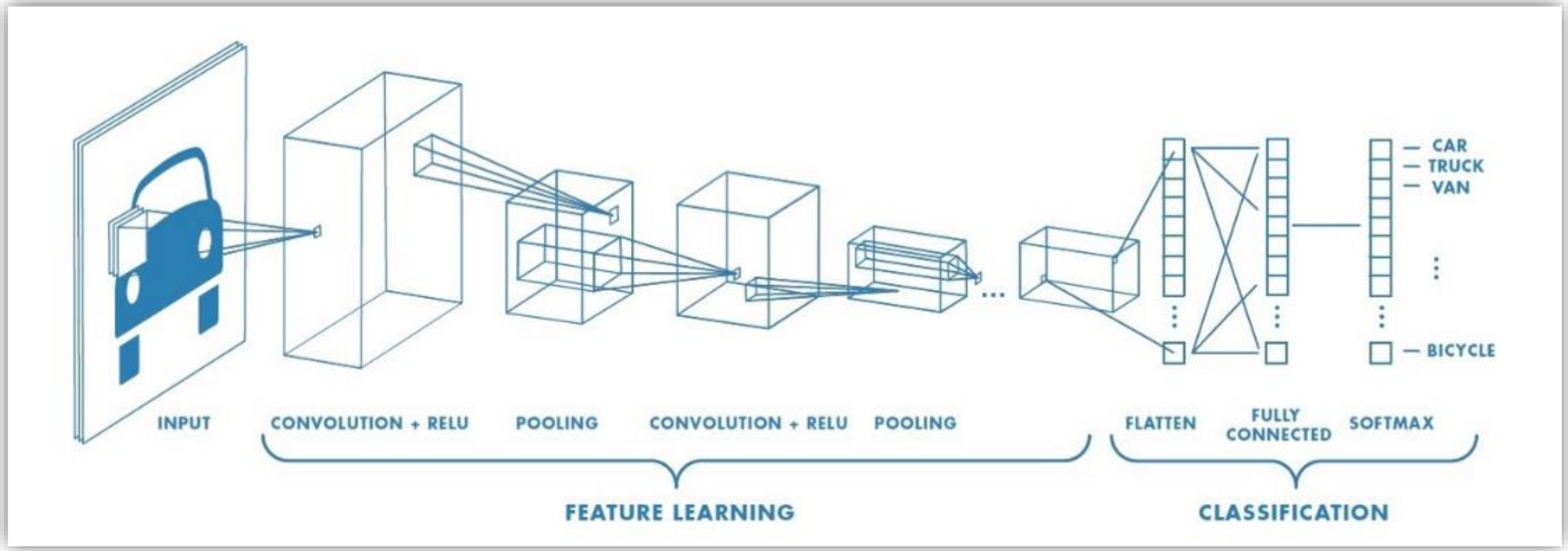


We need...

- We need two things for that:
  - Detect face from the given image
  - and blur out

OpenCV Facial Recognition [1]

- **Haar Cascade is a ML-based algorithm for object detection**
  - **This works like CNN(Convolutional Neural Network)**



OpenCV Facial Recognition [2]

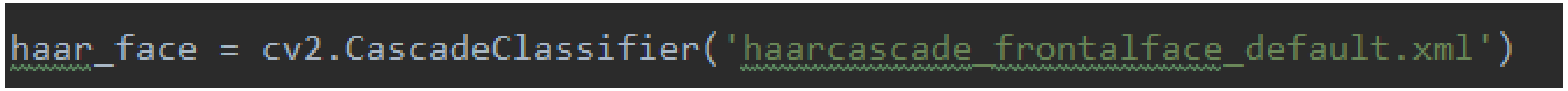
- The steps for Haar cascade is…
  1. Prepare lots of data to learn
  2. Extract Haar-feature
  3. Compute the feature
  4. Apply the haar classifier to the given image / frame

- …But you don’t need to learn about this
  - because it contains too annoying math

OpenCV Facial Recognition [3]

- We will not conduct learning directly
  - OpenCV provides pre-learning data
  - https://github.com/opencv/opencv/tree/master/data/haarcascades

  - You can load this and use without anything more



```
haar_face = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

- We uploaded this to the page
  - with the skeleton code

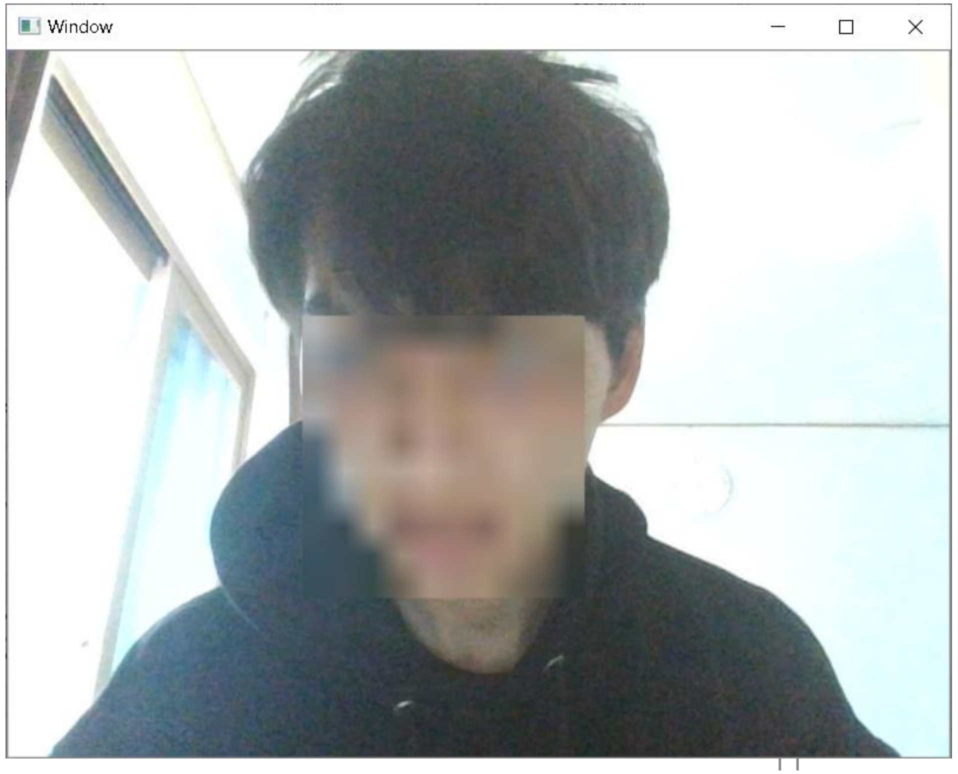
Pixelation

- The method is simple:
  - (intensely) Zoom in, and zoom out

- The function for this is provided in the skeleton

Objectives

1. Use your webcam in OpenCV
2. Detect your face
3. After that, pixelate the face area
4. Show the video in real-time



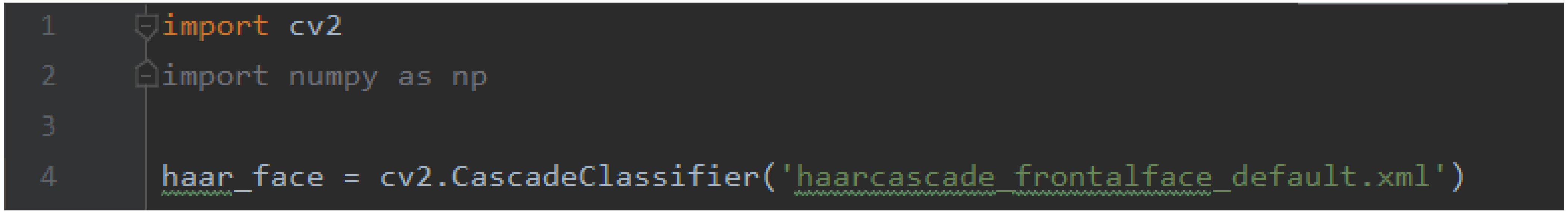
Notes

- Be careful about the “light”
  - If the image contains backlight, detection sometimes doesn’t work

Code Explanation

Haar Classifier

- It is a dataset for facial recognition



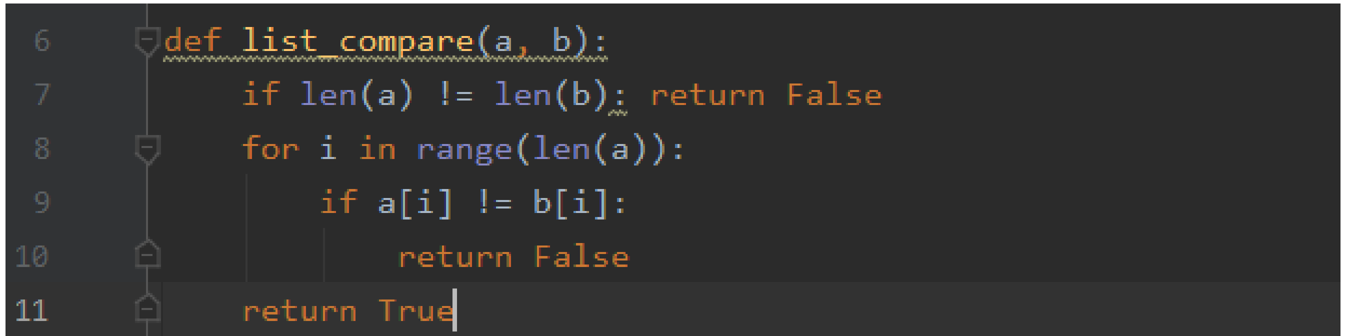
```python
import cv2
import numpy as np

haar_face = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

- Make sure that the .xml file should be in the same directory with your .py file!!

List Comparison

- Compares the contents of two lists and return…
  - True if all of them are same
  - False if at least one of them are different

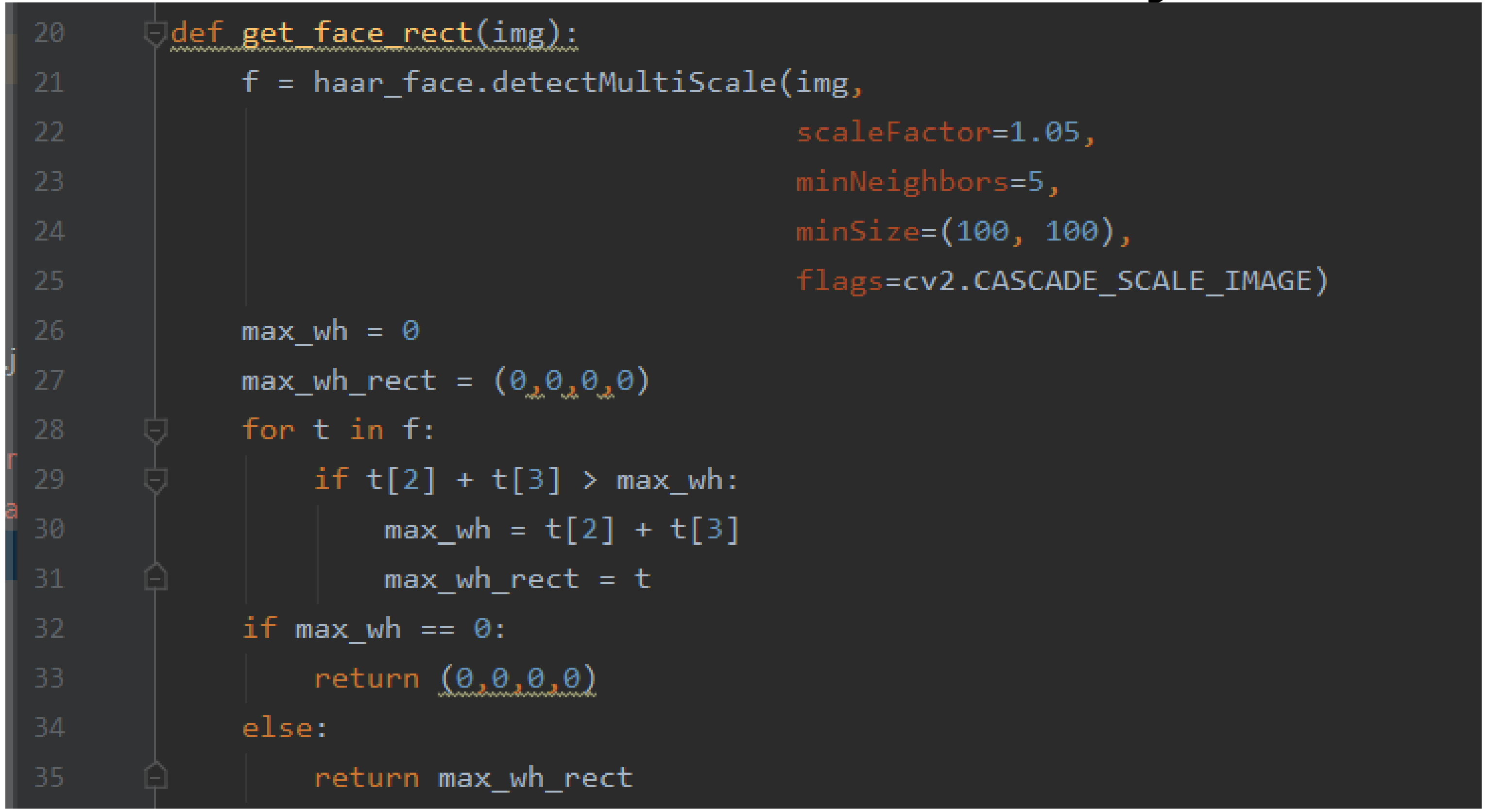
- Example:
  - list_compare([1, 2, 3], (1, 2, 3)] => True
  - list_compare([1, "b", 3], [1, "b", 3] => True
  - list_compare([1, 2, 3], (1, 2)] => False
  - list_compare([3, 2, 1], [1, 2, 3]) => False



```
def list_compare(a, b):
    if len(a) != len(b): return False
    for i in range(len(a)):
        if a[i] != b[i]:
            return False
    return True
```

Face Detection

- Parameter, img: the image you want to detect
- Return value: the area of face (x, y, w, h)

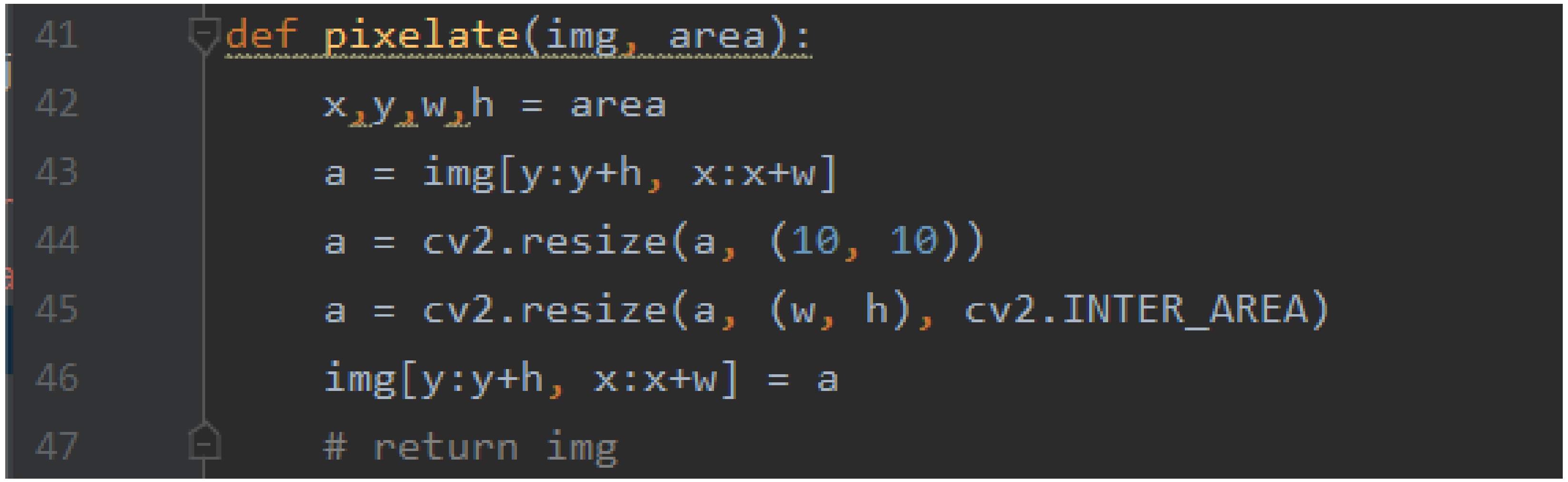


```python
def get_face_rect(img):
    f = haar_face.detectMultiScale(img,
                                   scaleFactor=1.05,
                                   minNeighbors=5,
                                   minSize=(100, 100),
                                   flags=cv2.CASCADE_SCALE_IMAGE)
    max_wh = 0
    max_wh_rect = (0,0,0,0)
    for t in f:
        if t[2] + t[3] > max_wh:
            max_wh = t[2] + t[3]
            max_wh_rect = t
    if max_wh == 0:
        return (0,0,0,0)
    else:
        return max_wh_rect
```

Pixelation

- Parameter, img: the image you want to pixelate
- Parameter, area: the area you want to pixelate

- There's no return value, you just call, then your image gets modified



```
def pixelate(img, area):
    x,y,w,h = area
    a = img[y:y+h, x:x+w]
    a = cv2.resize(a, (10, 10))
    a = cv2.resize(a, (w, h), cv2.INTER_AREA)
    img[y:y+h, x:x+w] = a
    # return img
```

Notes

- The pre-class video shows how to use your webcam and show the video capture
    - It will be helpful for this lab session

Let’s start!