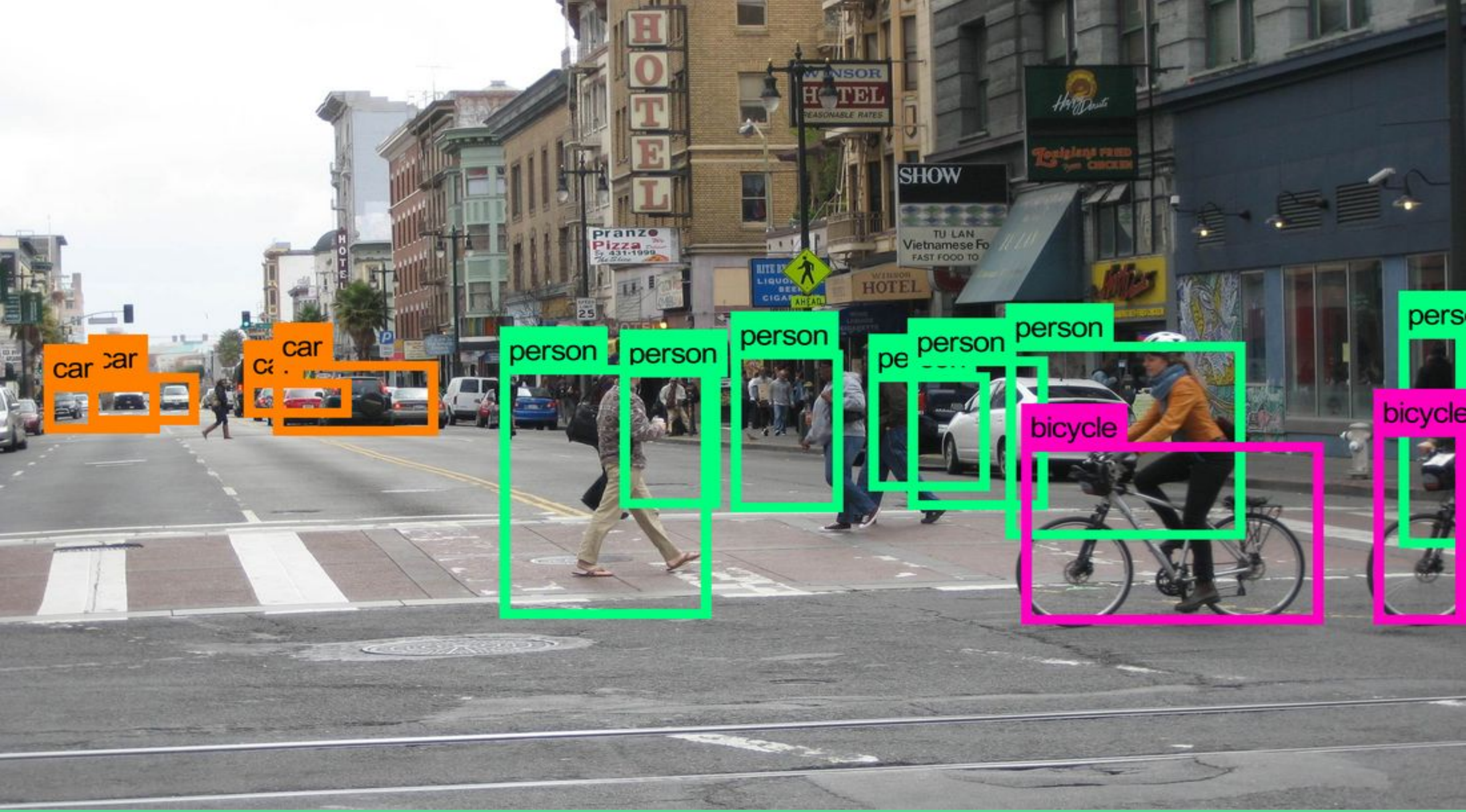
# COMPUTER VISION


car
car
car
ca
car
person
person
person
pe
person
person
perso
bicycle
bicycle
HOTEL
WINSOR HOTEL
REASONABLE RATES
SHOW
TU LAN
Vietnamese Fo
FAST FOOD TO
Pranzo Pizza
431-1999
WINSOR HOTEL
Louisiana FRIED CHICKEN
AHEAD
25

## Definition

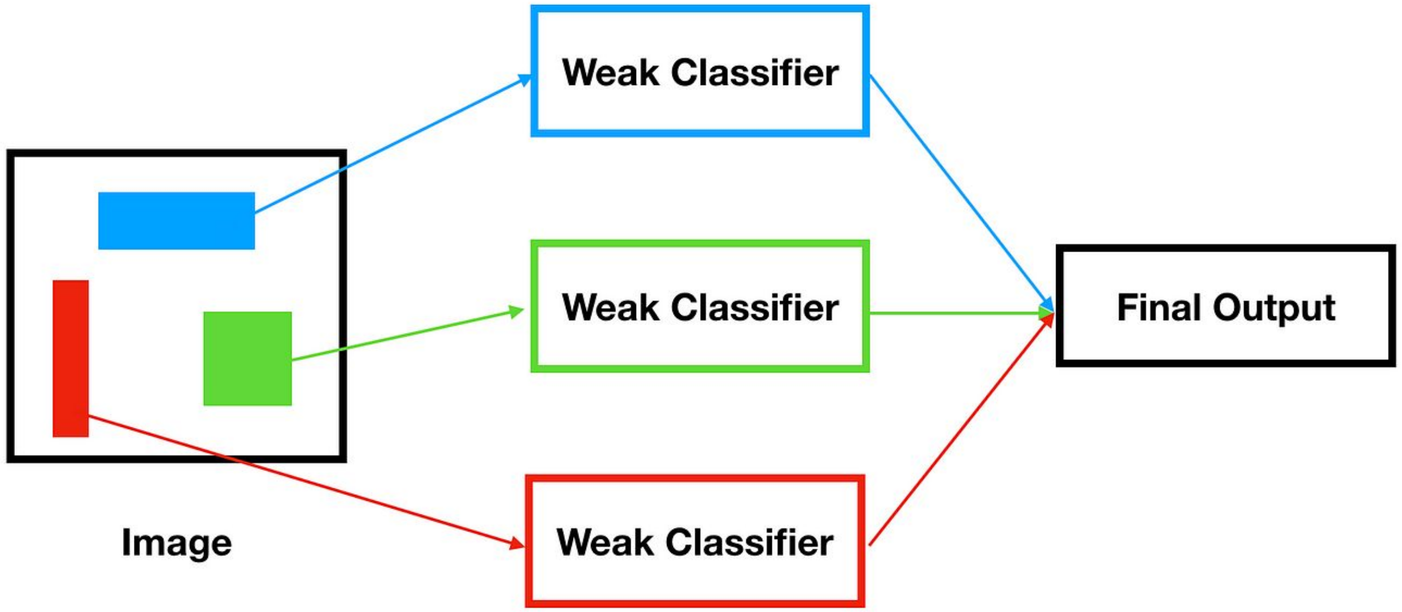
- acquiring, processing, understanding and analyzing digital images
- scientific discipline concerned with the theory behind artificial systems that extract information from images

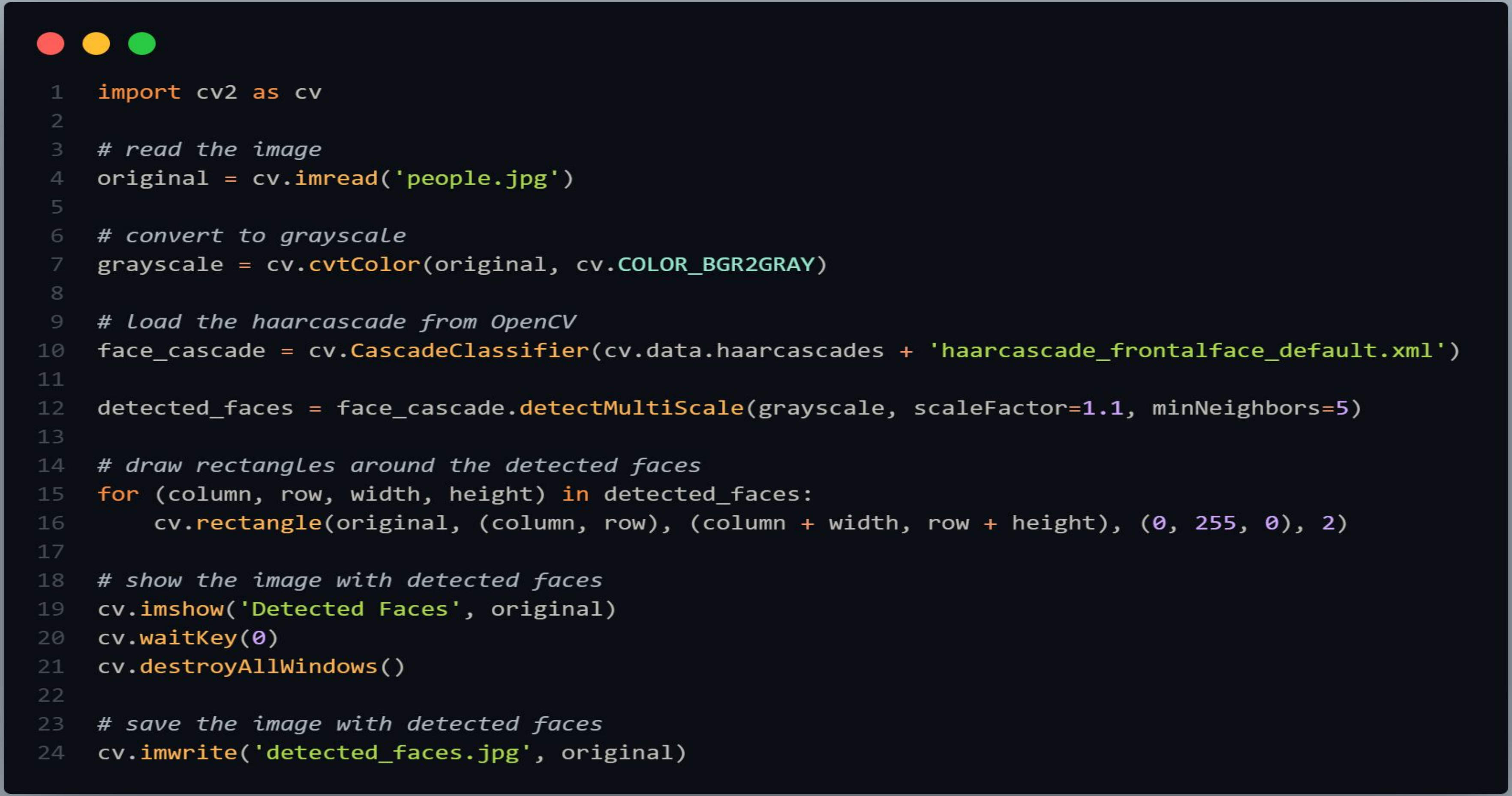
## Goals and real-world implementation

- solid-state physics
- neurobiology
- signal processing
- robotic navigation
- visual computing
- generally doing anything the human visual system can do

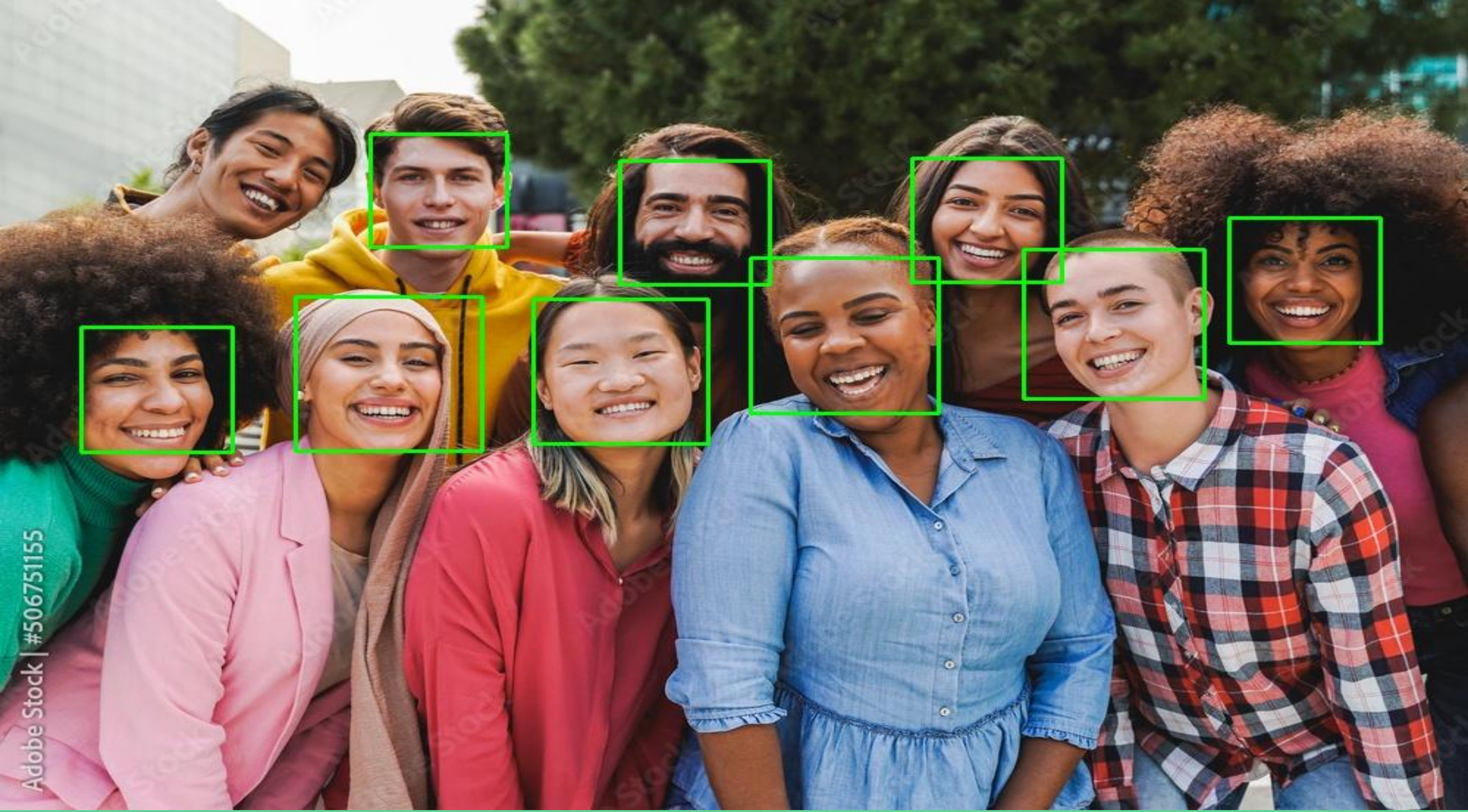
## Python implementation

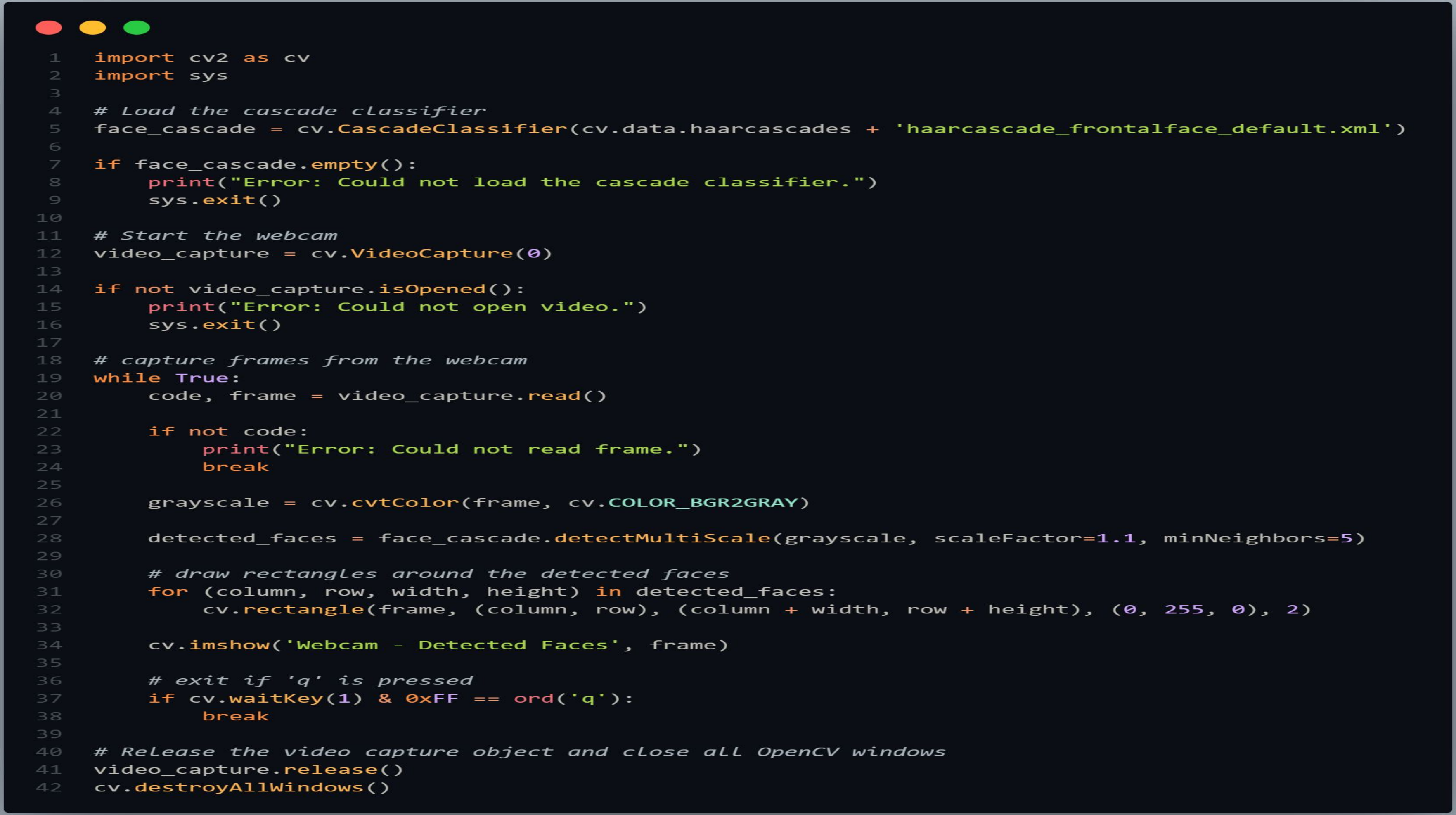
## Viola-Jones object-detection framework





```python
import cv2 as cv

# read the image
original = cv.imread('people.jpg')

# convert to grayscale
grayscale = cv.cvtColor(original, cv.COLOR_BGR2GRAY)

# Load the haarcascade from OpenCV
face_cascade = cv.CascadeClassifier(cv.data.haarcascades + 'haarcascade_frontalface_default.xml')

detected_faces = face_cascade.detectMultiScale(grayscale, scaleFactor=1.1, minNeighbors=5)

# draw rectangles around the detected faces
for (column, row, width, height) in detected_faces:
    cv.rectangle(original, (column, row), (column + width, row + height), (0, 255, 0), 2)

# show the image with detected faces
cv.imshow('Detected Faces', original)
cv.waitKey(0)
cv.destroyAllWindows()

# save the image with detected faces
cv.imwrite('detected_faces.jpg', original)
```


Adobe Stock | #506751155



```python
import cv2 as cv
import sys

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

if face_cascade.empty():
    print("Error: Could not load the cascade classifier.")
    sys.exit()

# Start the webcam
video_capture = cv.VideoCapture(0)

if not video_capture.isOpened():
    print("Error: Could not open video.")
    sys.exit()

# capture frames from the webcam
while True:
    code, frame = video_capture.read()

    if not code:
        print("Error: Could not read frame.")
        break

    grayscale = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)

    detected_faces = face_cascade.detectMultiScale(grayscale, scaleFactor=1.1, minNeighbors=5)

    # draw rectangles around the detected faces
    for (column, row, width, height) in detected_faces:
        cv.rectangle(frame, (column, row), (column + width, row + height), (0, 255, 0), 2)

    cv.imshow('Webcam - Detected Faces', frame)

    # exit if 'q' is pressed
    if cv.waitKey(1) & 0xFF == ord('q'):
        break

# Release the video capture object and close all OpenCV windows
video_capture.release()
cv.destroyAllWindows()
```

## Thanks for your attention!

Github repository of the project:

https://github.com/vect000r/computervision