# CPE018 Midterm Exam (1st Sem, A.Y. 2023-2024)

Student Submission Details:

- Name: Christian Ed B. Efa
- Section:BSCPE31S2
- Schedule: Tuesday 1:30-4:30pm
- Instructor: Dr. Jonathan V. Taylar / Engr. Verlyn V. Nojor / Engr. Roman M. Richard

## Intended Learning Outcomes

By the end of this activity, the student should be able to:

- ILO1: Demonstrate different methods for feature matching and detection learned in class and indepdentently from new sources.
- ILO2: Evaluate the accuracy of different feature matching and detection methods and scrutinize its applicability in solving a given real-life problem.

---

## Tasks

For this examination, you must create a **mood detection** program with an object-oriented programming approach (same as project CAMEO), it must detect mood changes through the use of algorithms/techniques/schemes learned in class, and from external sources.

In this file, you have to include for each section of your solution your completion of the following:

- Part 1: **Face Detection**: Once your face is detected using any algorithm, it must draw an ROI. The color for the ROI is your choice; however, it must detect for all faces in the frame and draw a corresponding ROI.
- Part 2: **Face Recognition**: The detected face must then be recognized, using any of the provided tools in class, the ROIs must indicate whether it is your face or someone it doesn't recognize.
- Part 3: **Mood Detection**: Use three different feature detection and matching techniques to determine three emotion: happy, sad and neutral. Two of the techniques must be learned from class, and 1 must be one you independently learned.

Properly show through your notebook the output for each part of the exam.

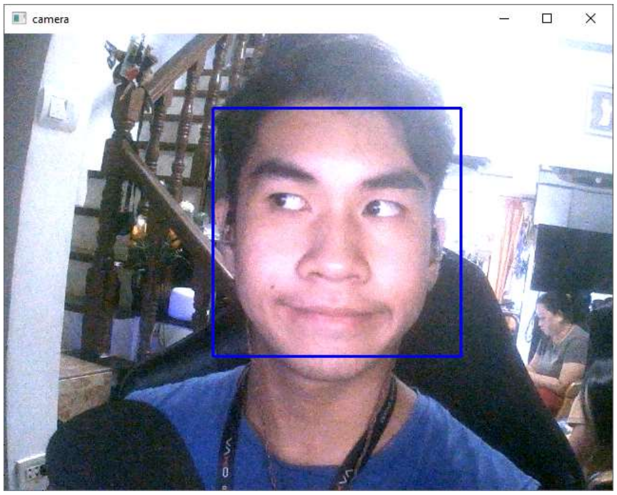
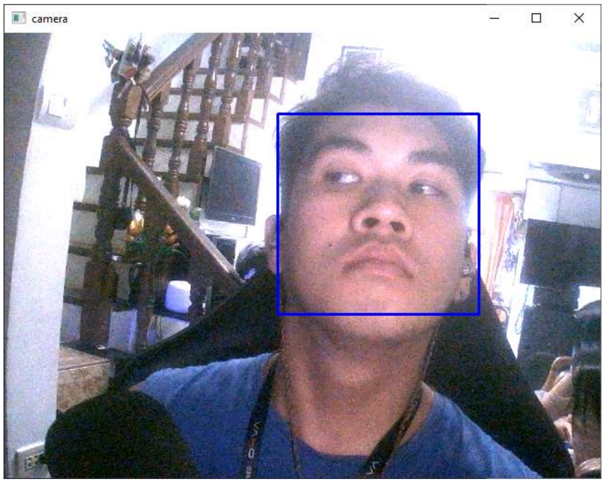
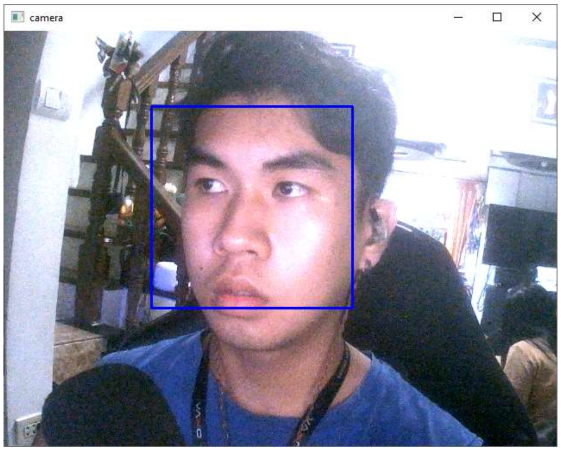
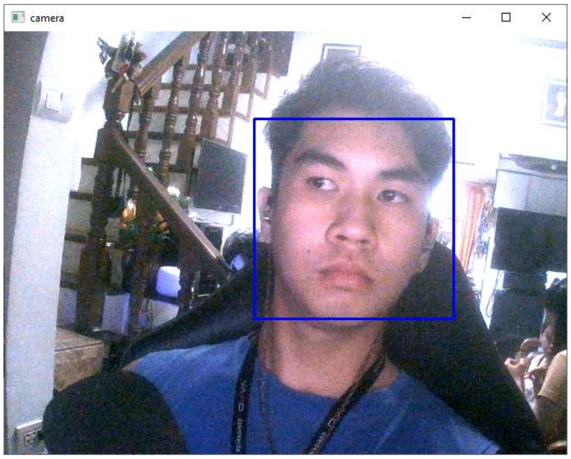
---

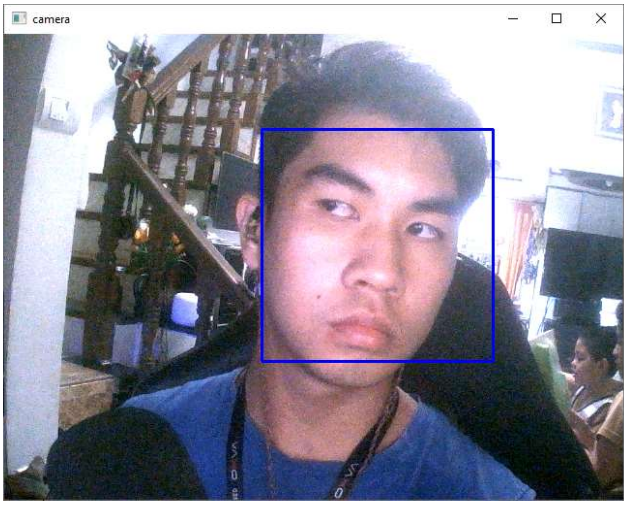
## Procedure and Outputs

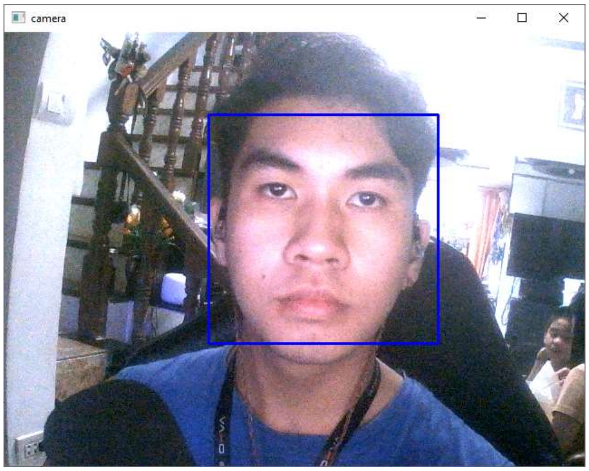
Notes:

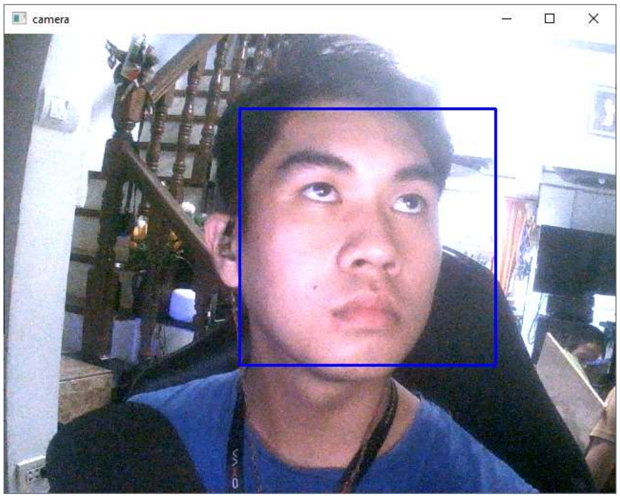
- This is the section where you have to include all your answers to the items provided in the tasks section.
- Tasks 1 and 2 contribute directly to ILO1: Demonstrate different methods for feature matching and detection learned in class and indepdentently from new sources.
- Task 3 contributes directly to ILO2: Evaluate the accuracy of different feature matching and detection methods and scrutinize its applicability in solving a given real-life problem.

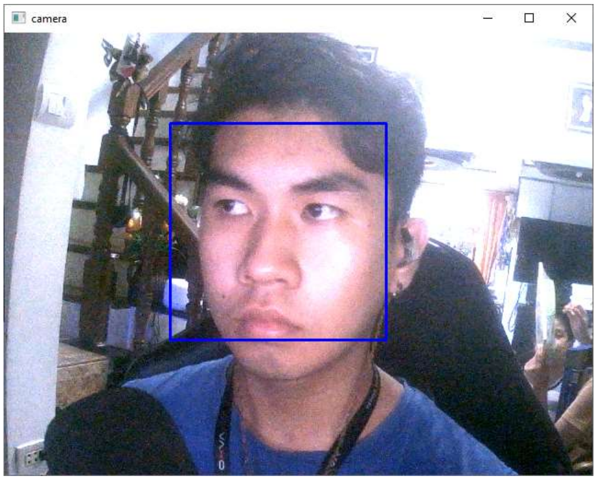
## Task 1: Face Detection

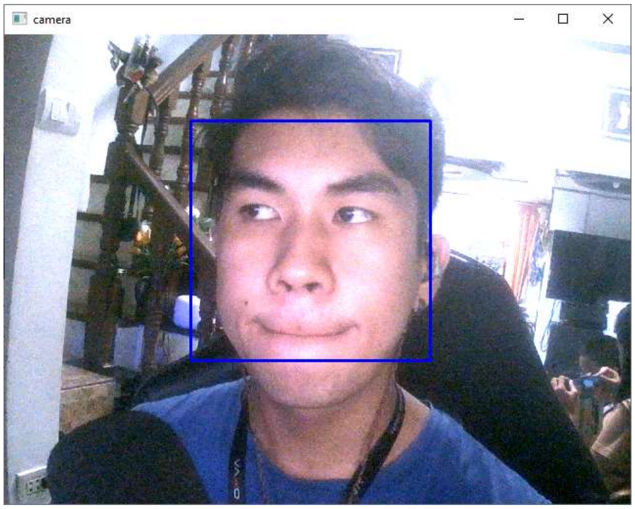
| Photos |
|---|
|  <br> <br> <br> |

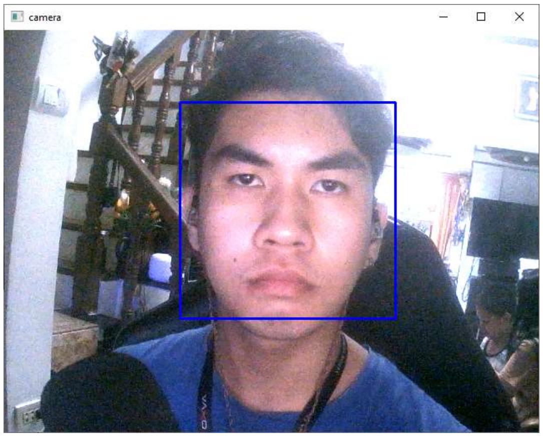




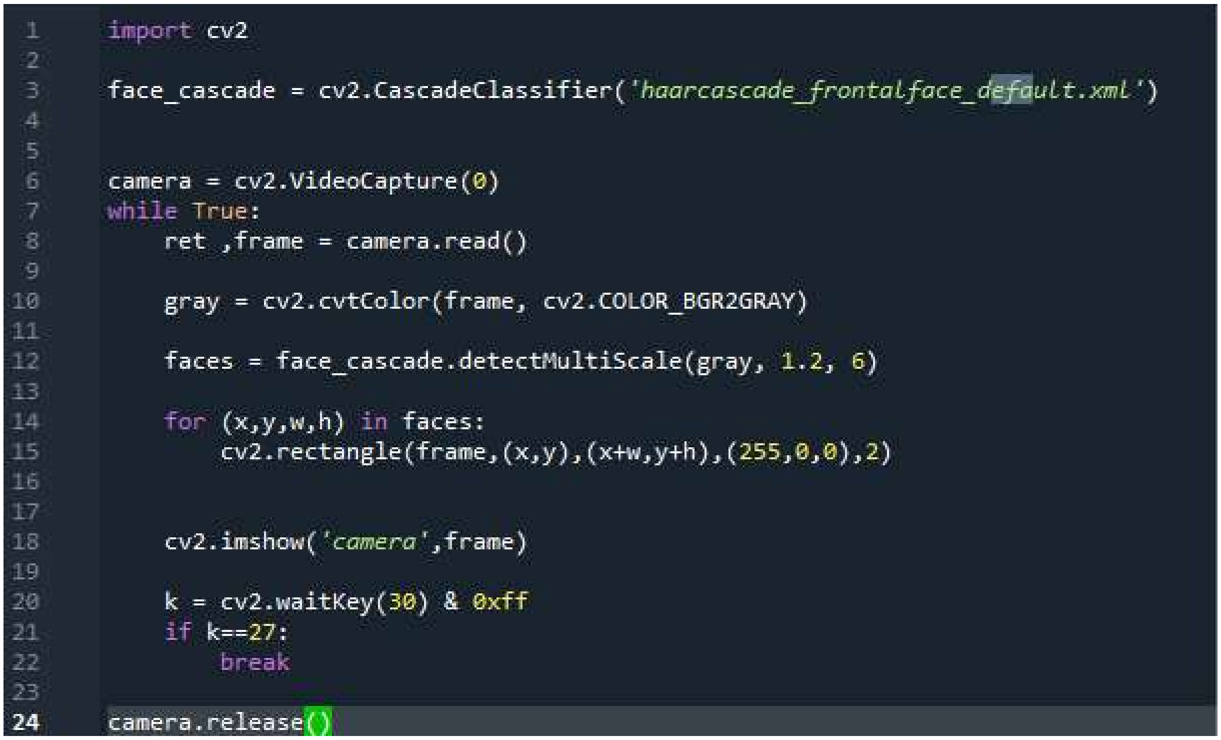






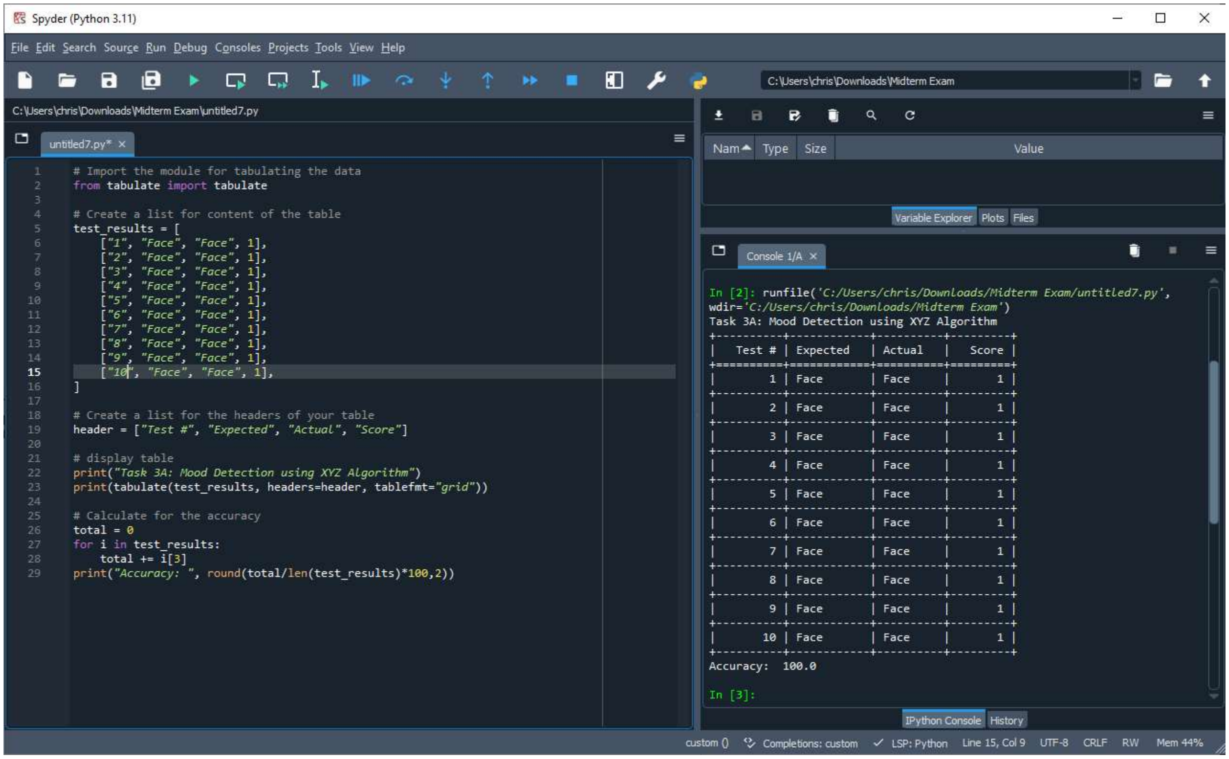


Code



```python
import cv2

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')


camera = cv2.VideoCapture(0)
while True:
    ret ,frame = camera.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(gray, 1.2, 6)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)


    cv2.imshow('camera',frame)

    k = cv2.waitKey(30) & 0xff
    if k==27:
        break

camera.release()
```

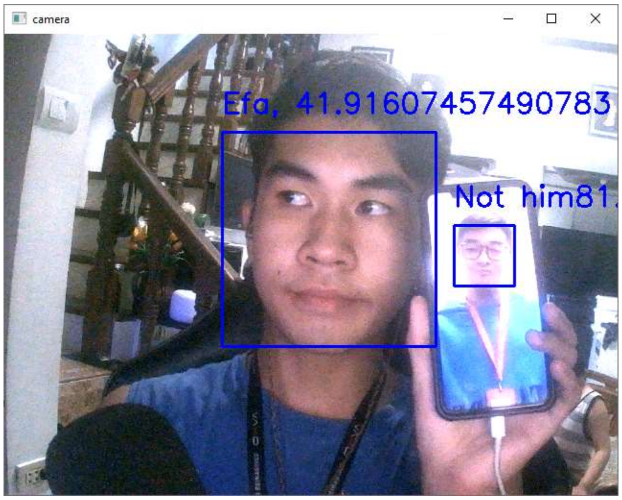
Analysis: After implementing this code and executing it, it manage to capture my face and detect it precisely

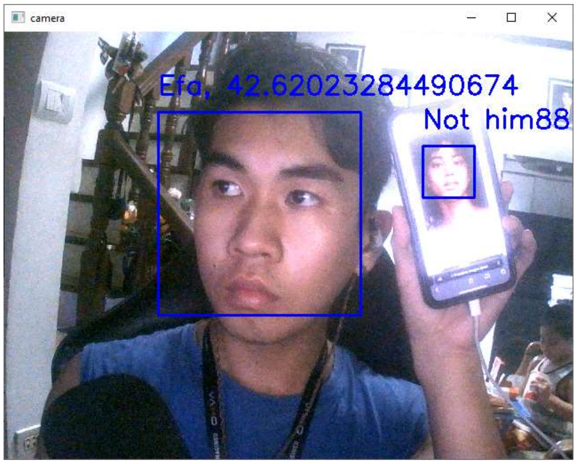

Spyder (Python 3.11)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\chris\Downloads\Midterm Exam
C:\Users\chris\Downloads\Midterm Exam\untitled7.py
untitled7.py*
Name Type Size Value
Variable Explorer Plots Files
Console 1/A
IPython Console History
custom () Completions: custom LSP: Python Line 15, Col 9 UTF-8 CRLF RW Mem 44%

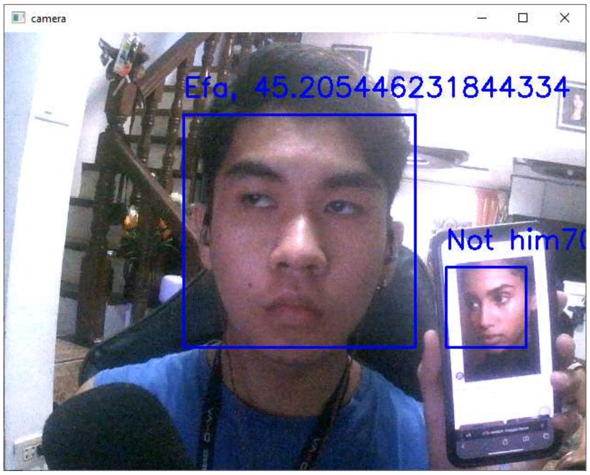
```python
# Import the module for tabulating the data
from tabulate import tabulate

# Create a list for content of the table
test_results = [
    ["1", "Face", "Face", 1],
    ["2", "Face", "Face", 1],
    ["3", "Face", "Face", 1],
    ["4", "Face", "Face", 1],
    ["5", "Face", "Face", 1],
    ["6", "Face", "Face", 1],
    ["7", "Face", "Face", 1],
    ["8", "Face", "Face", 1],
    ["9", "Face", "Face", 1],
    ["10", "Face", "Face", 1],
]

# Create a list for the headers of your table
header = ["Test #", "Expected", "Actual", "Score"]

# display table
print("Task 3A: Mood Detection using XYZ Algorithm")
print(tabulate(test_results, headers=header, tablefmt="grid"))

# Calculate for the accuracy
total = 0
for i in test_results:
    total += i[3]
print("Accuracy: ", round(total/len(test_results)*100,2))
```

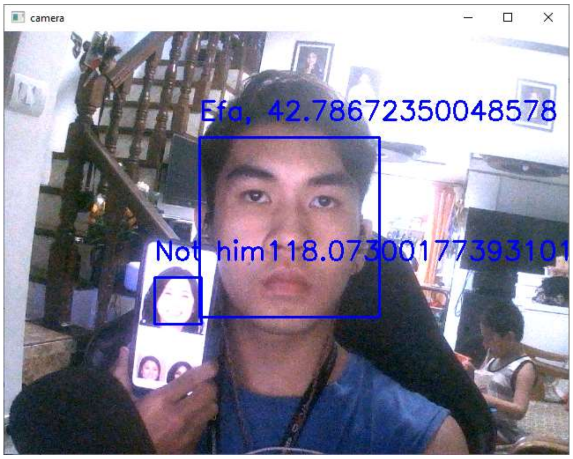
```
In [2]: runfile('C:/Users/chris/Downloads/Midterm Exam/untitled7.py',
wdir='C:/Users/chris/Downloads/Midterm Exam')
Task 3A: Mood Detection using XYZ Algorithm
+----------+------------+----------+---------+
|   Test # | Expected   | Actual   |   Score |
+==========+============+==========+=========+
|        1 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        2 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        3 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        4 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        5 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        6 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        7 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        8 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|        9 | Face       | Face     |       1 |
+----------+------------+----------+---------+
|       10 | Face       | Face     |       1 |
+----------+------------+----------+---------+
Accuracy:  100.0

In [3]:
```

## Task 2: Face Recognition

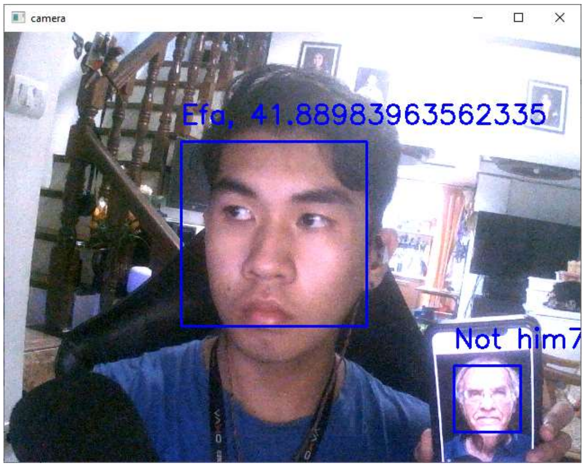
<table>
<tr><th>Images</th></tr>
<tr><td>


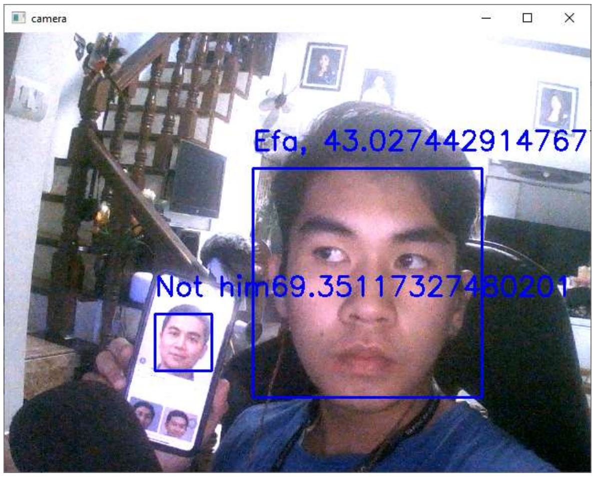


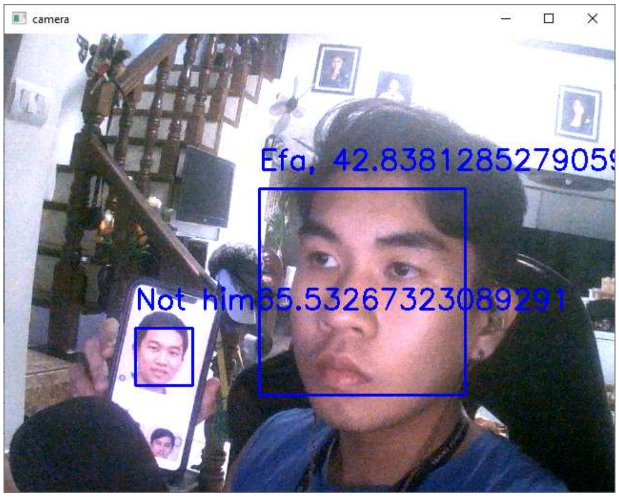


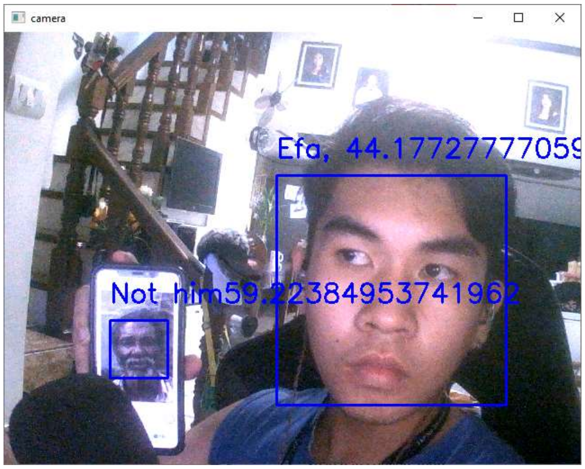


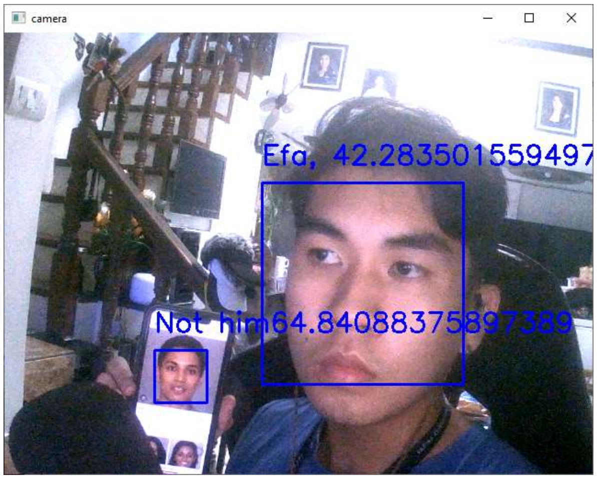
</td></tr>
</table>

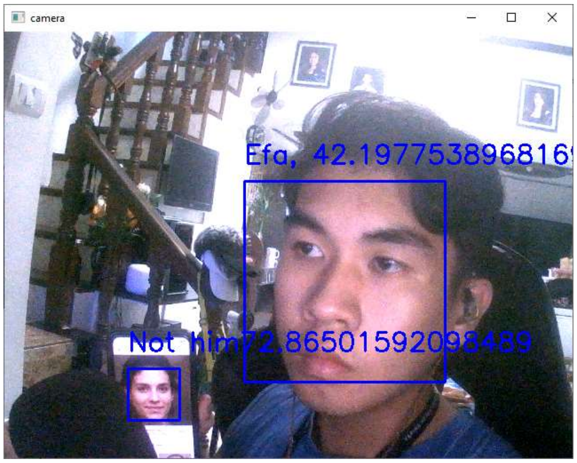

camera
Efa, 41.88983963562335
Not him7


camera
Efa, 43.027442914767
Not him69.35117327480201

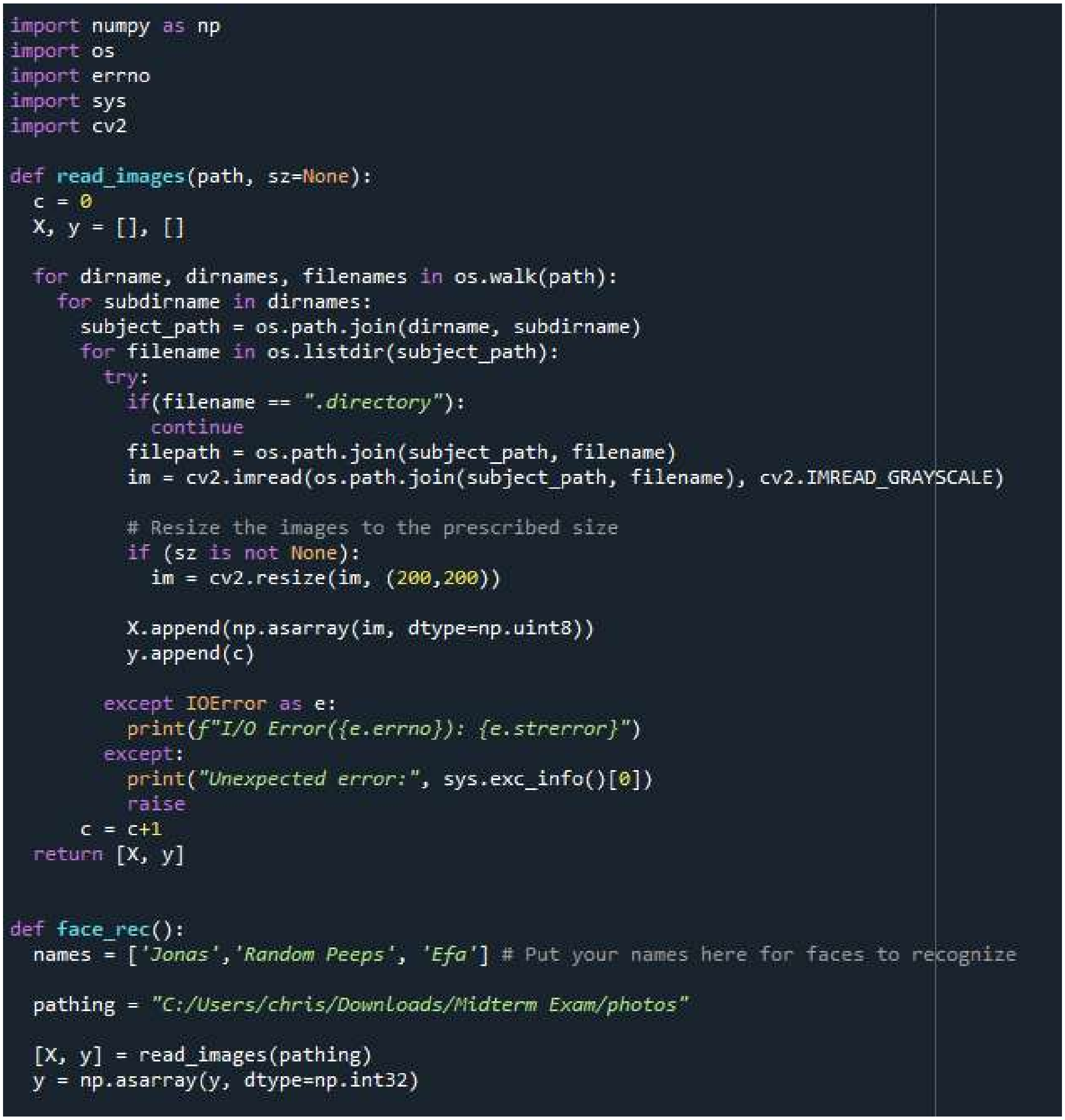

camera
Efa, 42.838128527905
Not him65.53267323089291


camera
Efa, 44.1772777059
Not him59.22384953741962

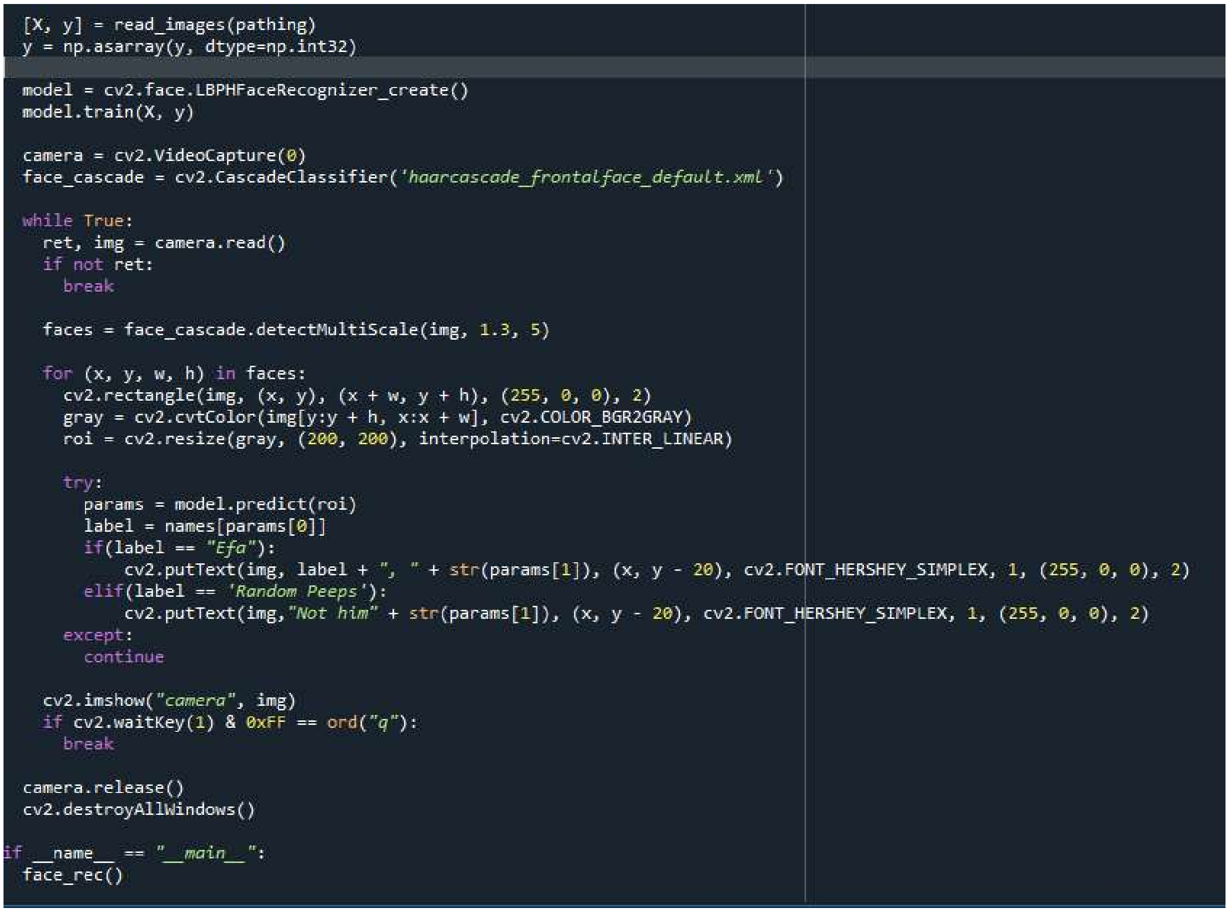

camera
Efa, 42.283501559497
Not him64.84088375897389


camera
Efa, 42.197753896816
Not him72.86501592098489

Codes



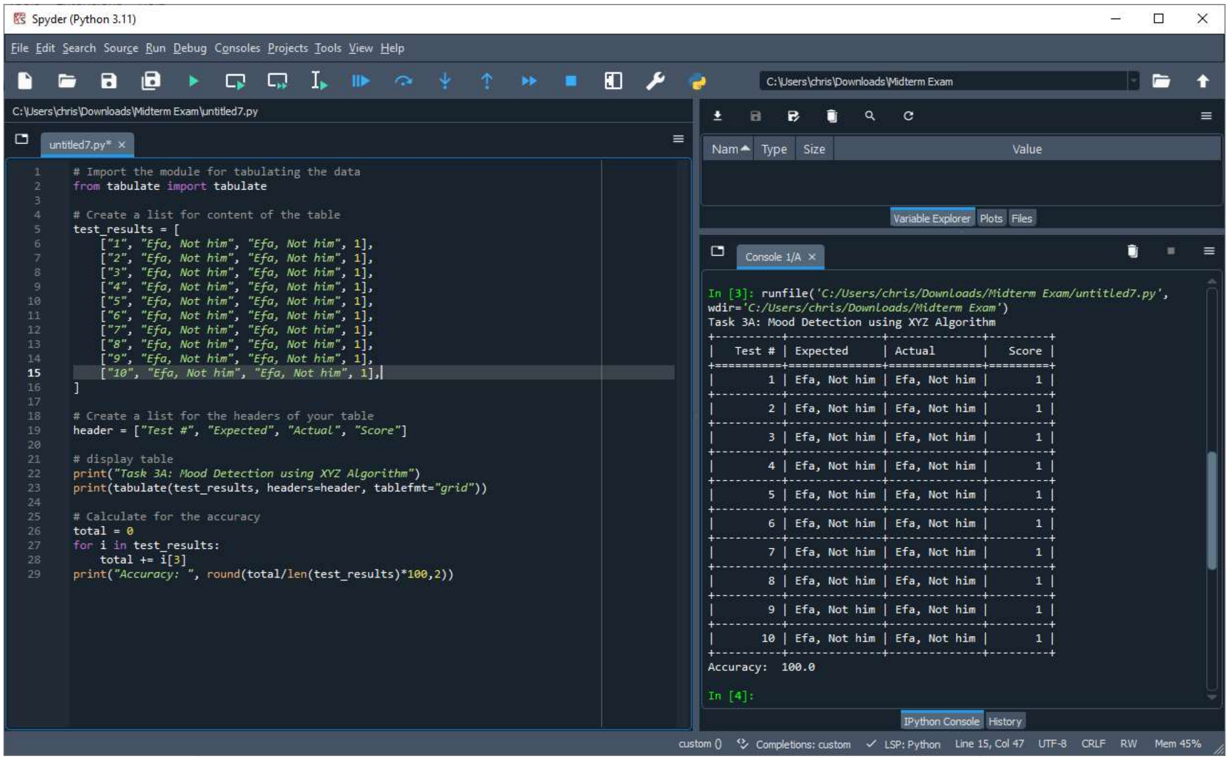
```python
import numpy as np
import os
import errno
import sys
import cv2

def read_images(path, sz=None):
  c = 0
  X, y = [], []

  for dirname, dirnames, filenames in os.walk(path):
    for subdirname in dirnames:
      subject_path = os.path.join(dirname, subdirname)
      for filename in os.listdir(subject_path):
        try:
          if(filename == ".directory"):
            continue
          filepath = os.path.join(subject_path, filename)
          im = cv2.imread(os.path.join(subject_path, filename), cv2.IMREAD_GRAYSCALE)

          # Resize the images to the prescribed size
          if (sz is not None):
            im = cv2.resize(im, (200,200))

          X.append(np.asarray(im, dtype=np.uint8))
          y.append(c)

        except IOError as e:
          print(f"I/O Error({e.errno}): {e.strerror}")
        except:
          print("Unexpected error:", sys.exc_info()[0])
          raise
      c = c+1
  return [X, y]


def face_rec():
  names = ['Jonas','Random Peeps', 'Efa'] # Put your names here for faces to recognize

  pathing = "C:/Users/chris/Downloads/Midterm Exam/photos"

  [X, y] = read_images(pathing)
  y = np.asarray(y, dtype=np.int32)
```



```
  [X, y] = read_images(pathing)
  y = np.asarray(y, dtype=np.int32)

  model = cv2.face.LBPHFaceRecognizer_create()
  model.train(X, y)

  camera = cv2.VideoCapture(0)
  face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

  while True:
    ret, img = camera.read()
    if not ret:
      break

    faces = face_cascade.detectMultiScale(img, 1.3, 5)

    for (x, y, w, h) in faces:
      cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
      gray = cv2.cvtColor(img[y:y + h, x:x + w], cv2.COLOR_BGR2GRAY)
      roi = cv2.resize(gray, (200, 200), interpolation=cv2.INTER_LINEAR)

      try:
        params = model.predict(roi)
        label = names[params[0]]
        if(label == "Efa"):
            cv2.putText(img, label + ", " + str(params[1]), (x, y - 20), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)
        elif(label == 'Random Peeps'):
            cv2.putText(img,"Not him" + str(params[1]), (x, y - 20), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)
      except:
        continue

    cv2.imshow("camera", img)
    if cv2.waitKey(1) & 0xFF == ord("q"):
      break

  camera.release()
  cv2.destroyAllWindows()

if __name__ == "__main__":
  face_rec()
```

Analysis :

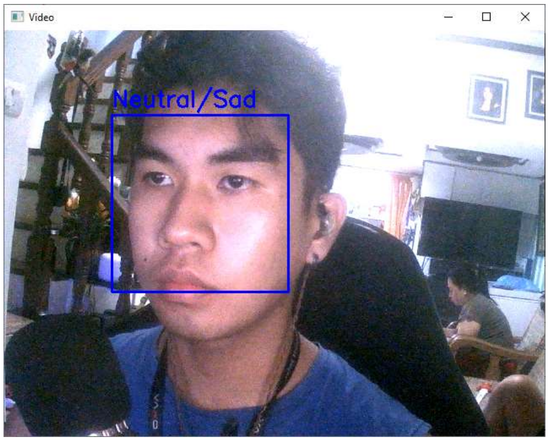
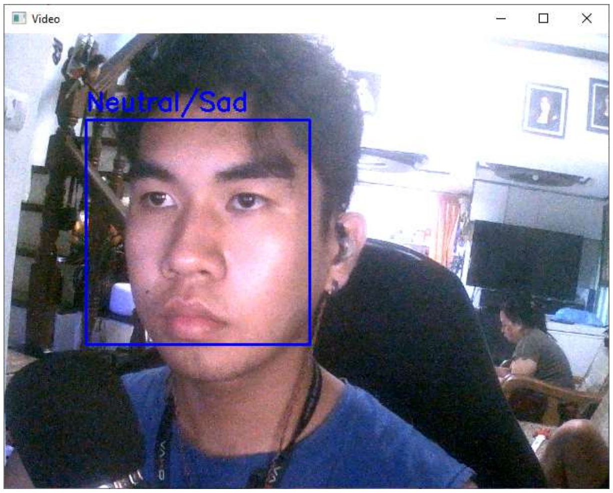
Using the knowledge and debugging on the previous activities we accomplished, I manage to execute this program and manage to accurately detect and recognize my face after trying other photos on my phone.

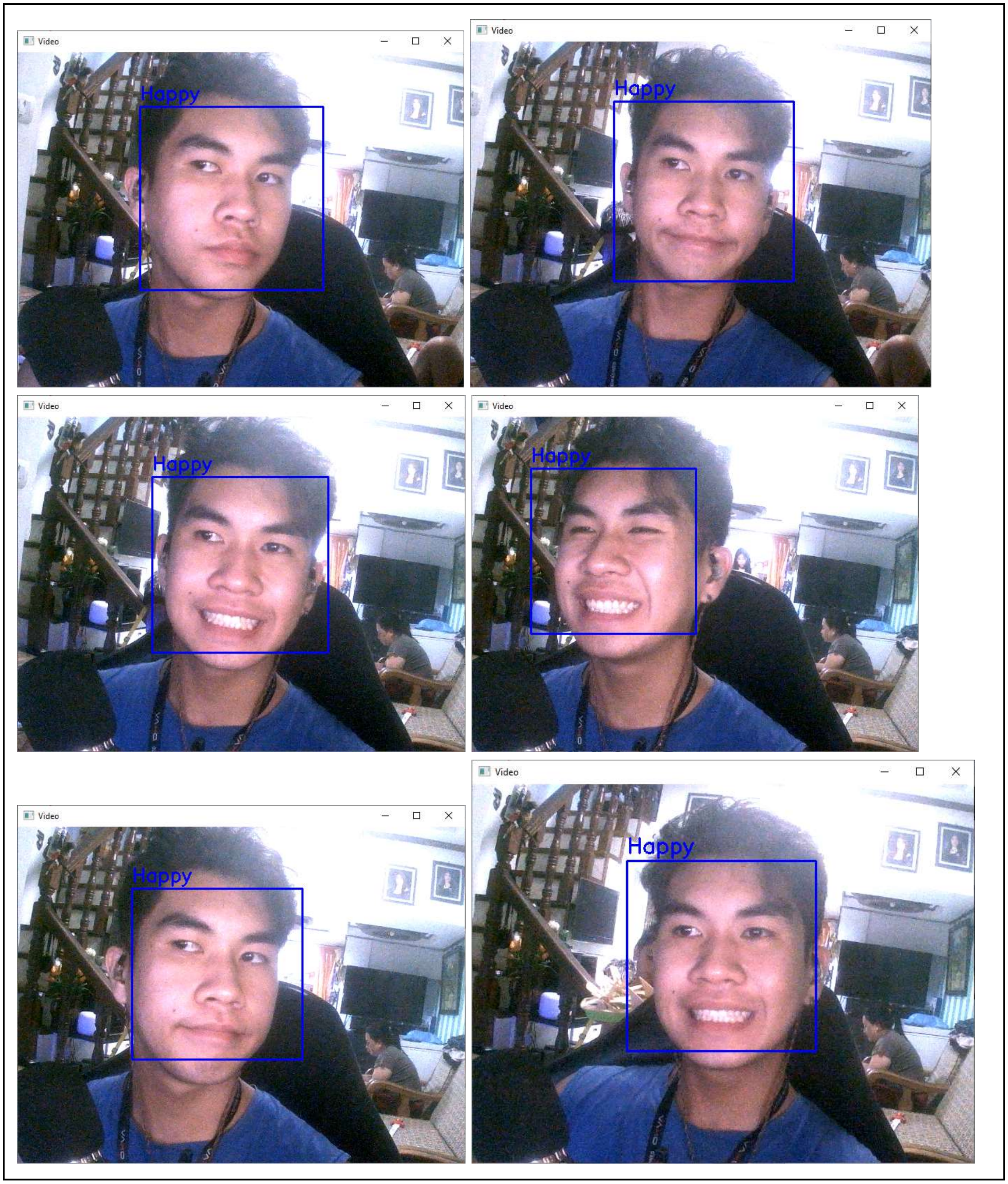

Spyder (Python 3.11)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\chris\Downloads\Midterm Exam
C:\Users\chris\Downloads\Midterm Exam\untitled7.py
untitled7.py*
Nam Type Size Value
Variable Explorer Plots Files
Console 1/A
IPython Console History
custom () Completions: custom LSP: Python Line 15, Col 47 UTF-8 CRLF RW Mem 45%

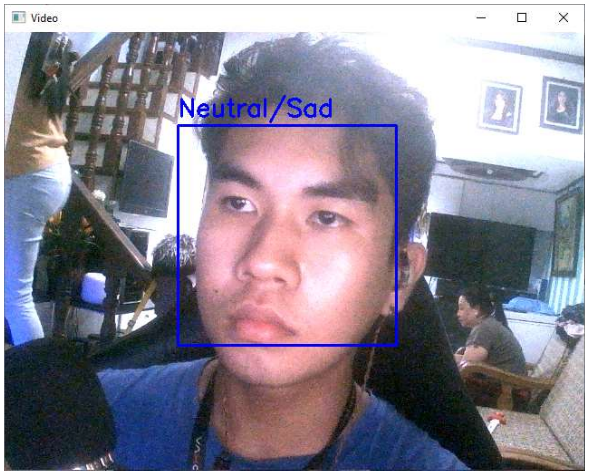
```python
# Import the module for tabulating the data
from tabulate import tabulate

# Create a list for content of the table
test_results = [
    ["1", "Efa, Not him", "Efa, Not him", 1],
    ["2", "Efa, Not him", "Efa, Not him", 1],
    ["3", "Efa, Not him", "Efa, Not him", 1],
    ["4", "Efa, Not him", "Efa, Not him", 1],
    ["5", "Efa, Not him", "Efa, Not him", 1],
    ["6", "Efa, Not him", "Efa, Not him", 1],
    ["7", "Efa, Not him", "Efa, Not him", 1],
    ["8", "Efa, Not him", "Efa, Not him", 1],
    ["9", "Efa, Not him", "Efa, Not him", 1],
    ["10", "Efa, Not him", "Efa, Not him", 1],
]

# Create a list for the headers of your table
header = ["Test #", "Expected", "Actual", "Score"]

# display table
print("Task 3A: Mood Detection using XYZ Algorithm")
print(tabulate(test_results, headers=header, tablefmt="grid"))

# Calculate for the accuracy
total = 0
for i in test_results:
    total += i[3]
print("Accuracy: ", round(total/len(test_results)*100,2))
```

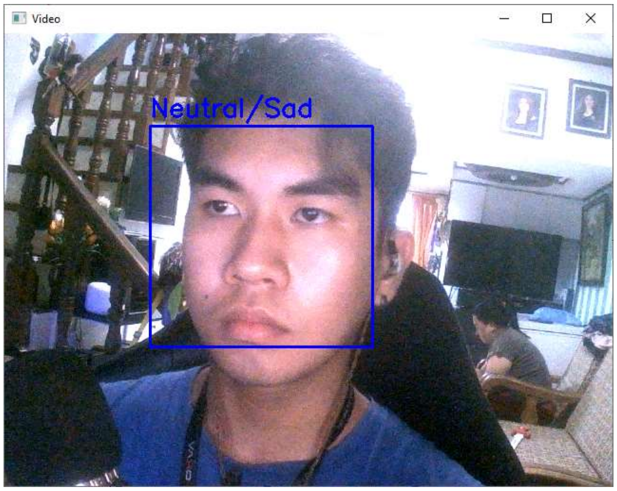
```
In [3]: runfile('C:/Users/chris/Downloads/Midterm Exam/untitled7.py',
wdir='C:/Users/chris/Downloads/Midterm Exam')
Task 3A: Mood Detection using XYZ Algorithm
+----------+--------------+--------------+---------+
|   Test # | Expected     | Actual       |   Score |
+==========+==============+==============+=========+
|        1 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        2 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        3 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        4 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        5 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        6 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        7 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        8 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|        9 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
|       10 | Efa, Not him | Efa, Not him |       1 |
+----------+--------------+--------------+---------+
Accuracy:  100.0

In [4]:
```

## Task 3: Mood Detection

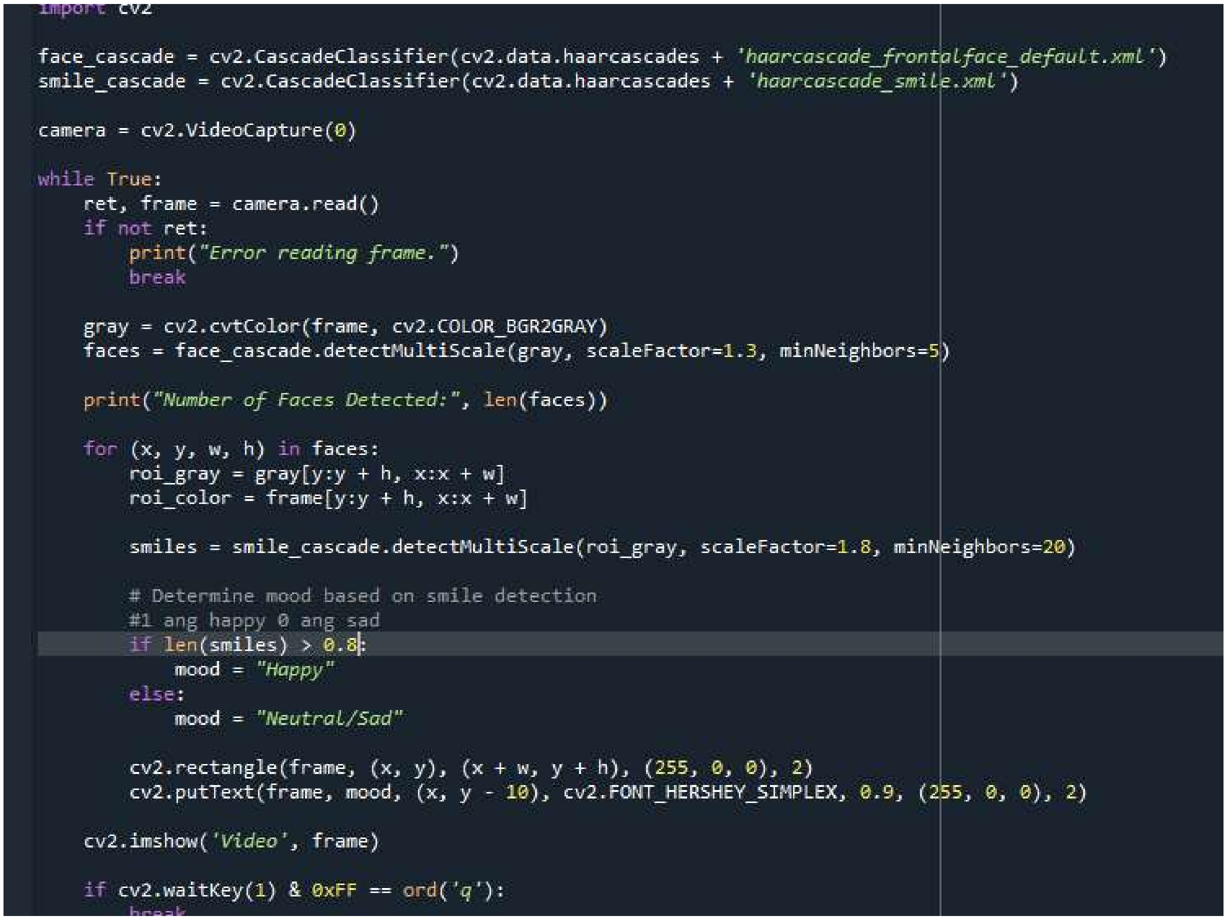
| Images |
|---|
|  <br> |


Video
Happy
Video
Happy
Video
Happy
Video
Happy
Video
Happy
Video
Happy



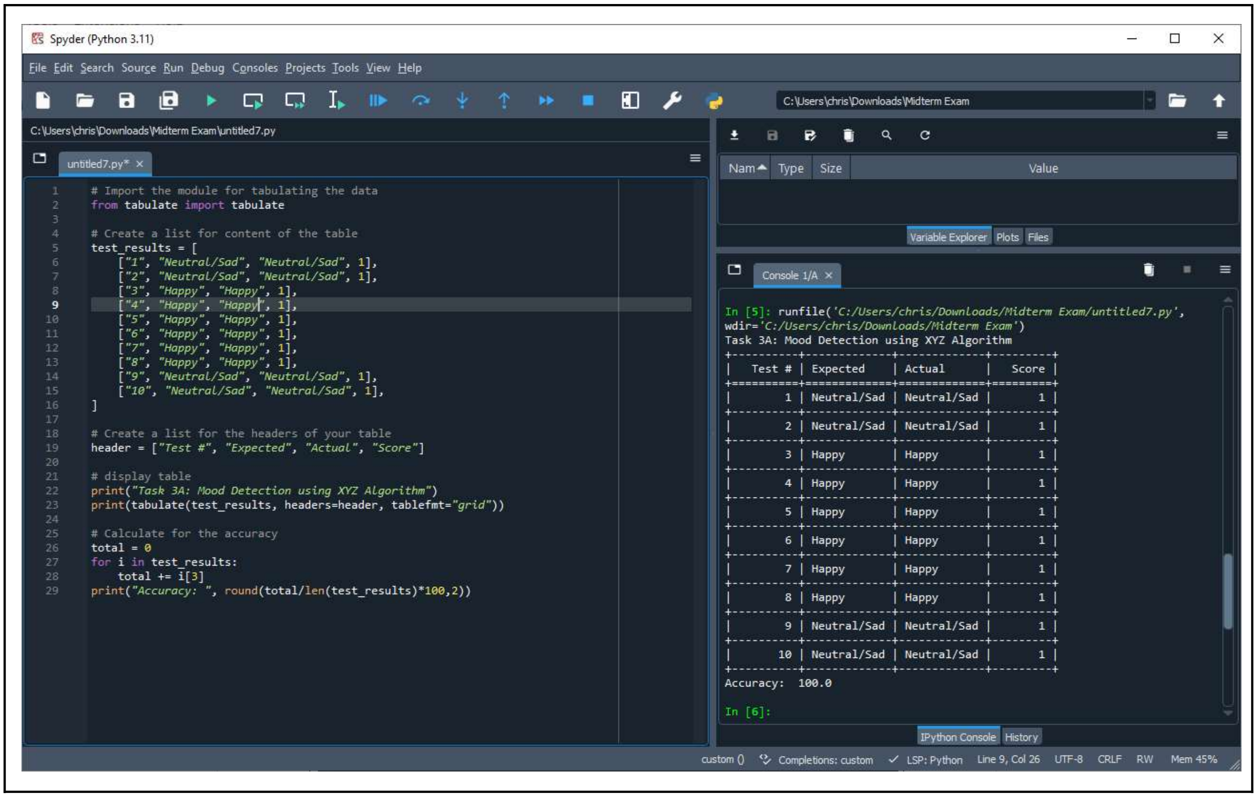


Codes



```
import cv2

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

camera = cv2.VideoCapture(0)

while True:
    ret, frame = camera.read()
    if not ret:
        print("Error reading frame.")
        break

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=5)

    print("Number of Faces Detected:", len(faces))

    for (x, y, w, h) in faces:
        roi_gray = gray[y:y + h, x:x + w]
        roi_color = frame[y:y + h, x:x + w]

        smiles = smile_cascade.detectMultiScale(roi_gray, scaleFactor=1.8, minNeighbors=20)

        # Determine mood based on smile detection
        #1 ang happy 0 ang sad
        if len(smiles) > 0.8:
            mood = "Happy"
        else:
            mood = "Neutral/Sad"

        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)
        cv2.putText(frame, mood, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (255, 0, 0), 2)

    cv2.imshow('Video', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

Analysis:
I tried other parameters to find a proper neutral expression but i changed it and made sad and neutral the same.


Spyder (Python 3.11)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\chris\Downloads\Midterm Exam
C:\Users\chris\Downloads\Midterm Exam\untitled7.py
untitled7.py*
Nam Type Size Value
Variable Explorer Plots Files
Console 1/A
IPython Console History
custom () Completions: custom LSP: Python Line 9, Col 26 UTF-8 CRLF RW Mem 45%

```python
# Import the module for tabulating the data
from tabulate import tabulate

# Create a list for content of the table
test_results = [
    ["1", "Neutral/Sad", "Neutral/Sad", 1],
    ["2", "Neutral/Sad", "Neutral/Sad", 1],
    ["3", "Happy", "Happy", 1],
    ["4", "Happy", "Happy", 1],
    ["5", "Happy", "Happy", 1],
    ["6", "Happy", "Happy", 1],
    ["7", "Happy", "Happy", 1],
    ["8", "Happy", "Happy", 1],
    ["9", "Neutral/Sad", "Neutral/Sad", 1],
    ["10", "Neutral/Sad", "Neutral/Sad", 1],
]

# Create a list for the headers of your table
header = ["Test #", "Expected", "Actual", "Score"]

# display table
print("Task 3A: Mood Detection using XYZ Algorithm")
print(tabulate(test_results, headers=header, tablefmt="grid"))

# Calculate for the accuracy
total = 0
for i in test_results:
    total += i[3]
print("Accuracy: ", round(total/len(test_results)*100,2))
```

```
In [5]: runfile('C:/Users/chris/Downloads/Midterm Exam/untitled7.py',
wdir='C:/Users/chris/Downloads/Midterm Exam')
Task 3A: Mood Detection using XYZ Algorithm
+----------+-------------+-------------+---------+
|   Test # | Expected    | Actual      |   Score |
+==========+=============+=============+=========+
|        1 | Neutral/Sad | Neutral/Sad |       1 |
+----------+-------------+-------------+---------+
|        2 | Neutral/Sad | Neutral/Sad |       1 |
+----------+-------------+-------------+---------+
|        3 | Happy       | Happy       |       1 |
+----------+-------------+-------------+---------+
|        4 | Happy       | Happy       |       1 |
+----------+-------------+-------------+---------+
|        5 | Happy       | Happy       |       1 |
+----------+-------------+-------------+---------+
|        6 | Happy       | Happy       |       1 |
+----------+-------------+-------------+---------+
|        7 | Happy       | Happy       |       1 |
+----------+-------------+-------------+---------+
|        8 | Happy       | Happy       |       1 |
+----------+-------------+-------------+---------+
|        9 | Neutral/Sad | Neutral/Sad |       1 |
+----------+-------------+-------------+---------+
|       10 | Neutral/Sad | Neutral/Sad |       1 |
+----------+-------------+-------------+---------+
Accuracy:  100.0

In [6]:
```

# Analysis

For the three different techniques you used in face detection, provide an in-depth analysis.

To do this, you must:

- Test the face detection, face recongition, and mood detection functions 10 times each. Only the mood detection will have components for 10 tests for each different technique used.
- Create a table containing the 10 tests (like shown below) for each task.
- Analyze each output by identifying the accuracy and providing your observations.

# Summary and Lessons Learned

This section must be answered concisely. Do not engage in unmeaningful writing for the summary and lessons learned. Provide your brief reflection only.

**After doing this task it is very difficult to debug and find the problems such as face is not detected and manage to implement and execute the program, Having the previous activities to refresh my mind and research about the codes that i havent explored yet to have a better understanding and deep learning through face detection,face recognition and also mood detection.**