ICT 3314

EMBEDDED SYSTEMS

Assignment 02

Y.M.G.D.L. Yapabandara

ITT/2020/119

1381

Department of Information & Communication Technology

Faculty of Technology

Rajarata University of Sri Lanka

Question 1

Question 1.

Q1.

prepare the SD card :

- Download the Rasberry Pi imager from the official Rasberry Pi website.
- Insert an SD card into your computer using a card reader
- use the Rasberry Pi imager to install the Rasberry Pi OS onto the SD card.

Insert SD card.

- Insert the prepared SD card into the SD card slot on the Rasberry Pi.

Connect the peripherals

- Plug in a keyboard and mouse into the USB ports on the Raspbery Pi.
- Connect a monitor or TV to the Raspberry Pi using an HDMI cable.

Power the Raspberry Pi

- Connect the power supply to the Raspberry Pi. It will turn on automatically when plugged in

Initial setup

- The Raspberry Pi will boot into the OS. Follow the on-screen instructions to set up language, Wi-Fi, and other setting.

02.

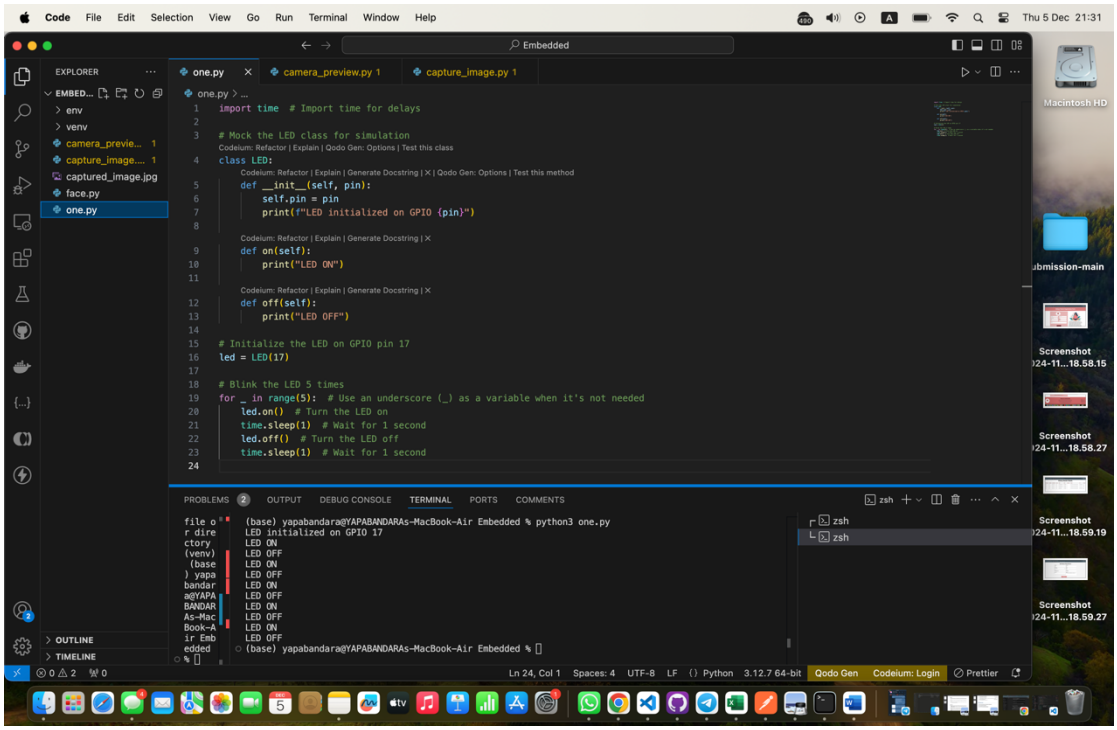
Updates add the latest security patches to potect the system from threats.

Upgrades improves speed and make the system work better.

Updates ensure the systems support new software and hardware.

Bug fixes prevent crashes and keep the system running smoothly.

03).




```
import time

class LED:
    def __init__(self, pin):
        self.pin = pin
        print(f"LED initialized on GPIO {pin}")
    def on(self):
        print("LED ON")
    def off(self):
        print("LED OFF")
led = LED(17)
 for _ in range(5):
    led.on()
    time.sleep(1)
    led.off()
    time.sleep(1)
```

Question 2

Question 02

01. Power off the Raspbery Pi - Always turn off the Raspberry Pi before connecting the camera module to avoid electrical damage.

Locate the CSI Port - Find the camera serial interface port on the Raspberry Pi board. It is usualy near the HDMI Port and labeled "CAMERA".

open the CSI port connection - Gently lift the plastic clip on the CSI port to unelock it.

Insert the Ribbon cable - Align the metal contacts of the ribbon cable to face the CSI Port contacts.

push down the plastic clip to lock the cable in place securely.
Turn on the Raspberry Pi and ensure the camera module is detected using commands like vcgencmd get-camera

precautions.

Do not touch the camera lens to prevent smudges or scratches.

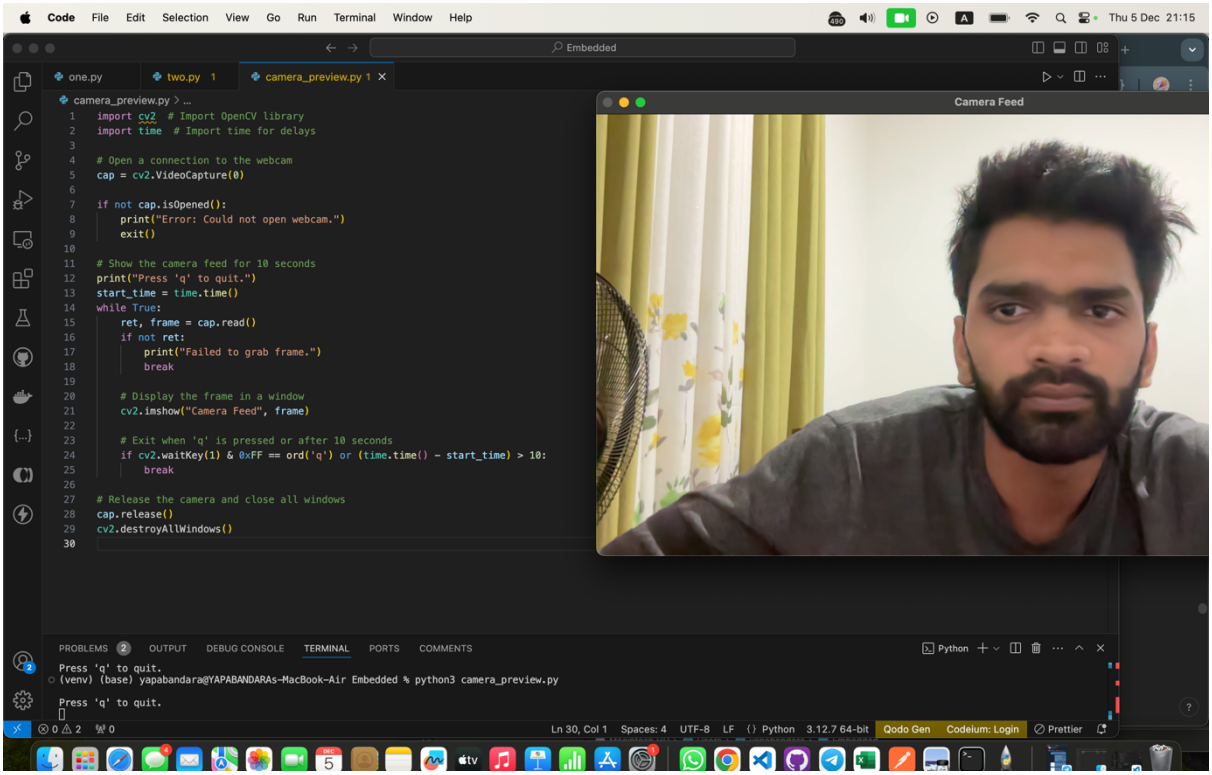
Do not bend or pull the cable too hard to avoid damage.

check the alignment of the ribbon cable to avoid improper connections.

prevent static Electricity

use correct port.

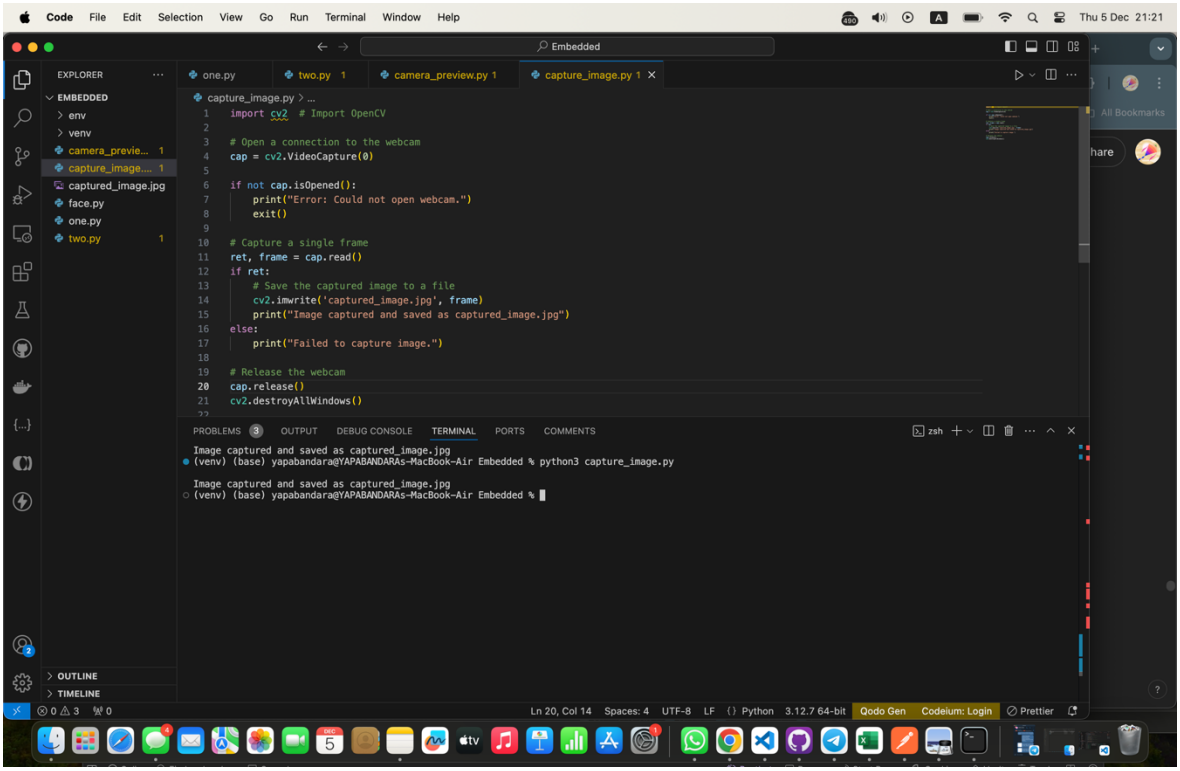
02)




```
import cv2
import time
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: Could not open webcam.")
    exit()
print("Press 'q' to quit.")
start_time = time.time()
while True:
ret, frame = cap.read()
    if not ret:
        print("Failed to grab frame.")
        break
```

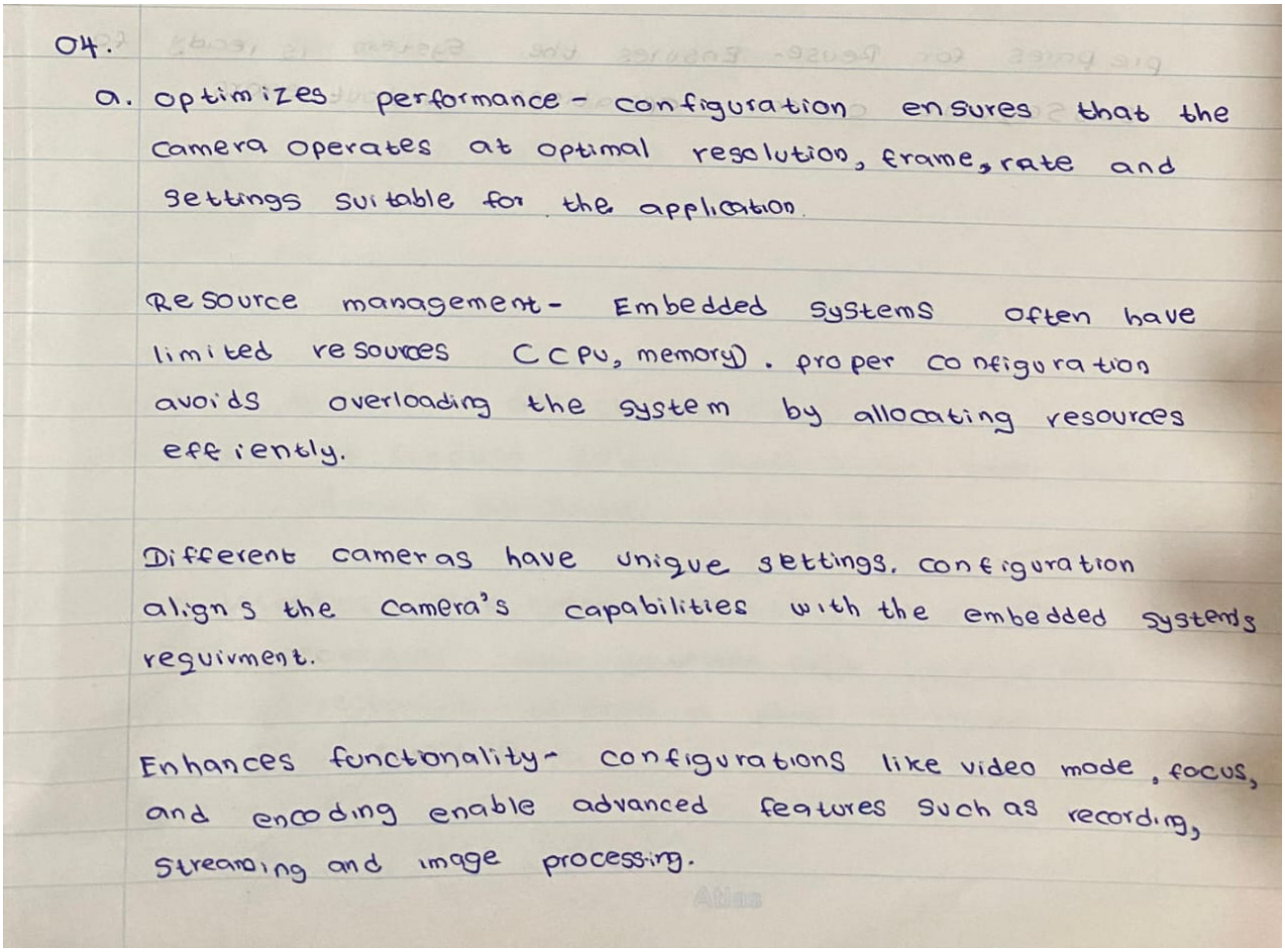
```
    cv2.imshow("Camera Feed", frame)
    if cv2.waitKey(1) & 0xFF == ord('q') or (time.time() - start_time) > 10:
        break
cap.release()
cv2.destroyAllWindows()
```

03).




```
import cv2  # Import OpenCV
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: Could not open webcam.")
    exit()
```

```
ret, frame = cap.read()
if ret:
    cv2.imwrite('captured_image.jpg', frame)
    print("Image captured and saved as captured_image.jpg")
else:
    print("Failed to capture image.")
cap.release()
cv2.destroyAllWindows()
```



04.

a. optimizes performance - configuration ensures that the camera operates at optimal resolution, frame, rate and settings suitable for the application.

Resource management - Embedded systems often have limited resources (CPU, memory). proper configuration avoids overloading the system by allocating resources efficiently.

Different cameras have unique settings. configuration aligns the camera's capabilities with the embedded systems requirment.

Enhances functionality - configurations like video mode, focus, and encoding enable advanced features such as recording, streaming and image processing.

No : Date: __/__/____

b.

picam2.start():
prepares the camera module by allocating resources and initializing the hardware.
Actives the camera, allowing it to start capturing images or videos.
Enable streaming- Necessary for live video or image capturing functionality.

picam2.stop():
Releases Resources- Frees up system resources (memory, processing power) used by the camera module

Ensures safety- properly shuts down the camera to prevent hardware damage or data corruption.

prepares for Reuse- Ensures the system is ready for subsequent camera operations without errors.

Question 3

Question 03

01) Image processing - supports operation like filtering, edge detection, image transformation, and object recognition.

Video Analysis- includes motion detection, background subtraction, and object tracking.
supports reading and processing video streams from cameras or pre-recorded files.

Computer vision Algorithms- implements popular algorithm for tasks like feature detection (SIFT, SURF), optical flow, and stereo vision.

Real time performance- optimized for speed, making it suitable for real time applications like robotics augmented reality and surveillance.

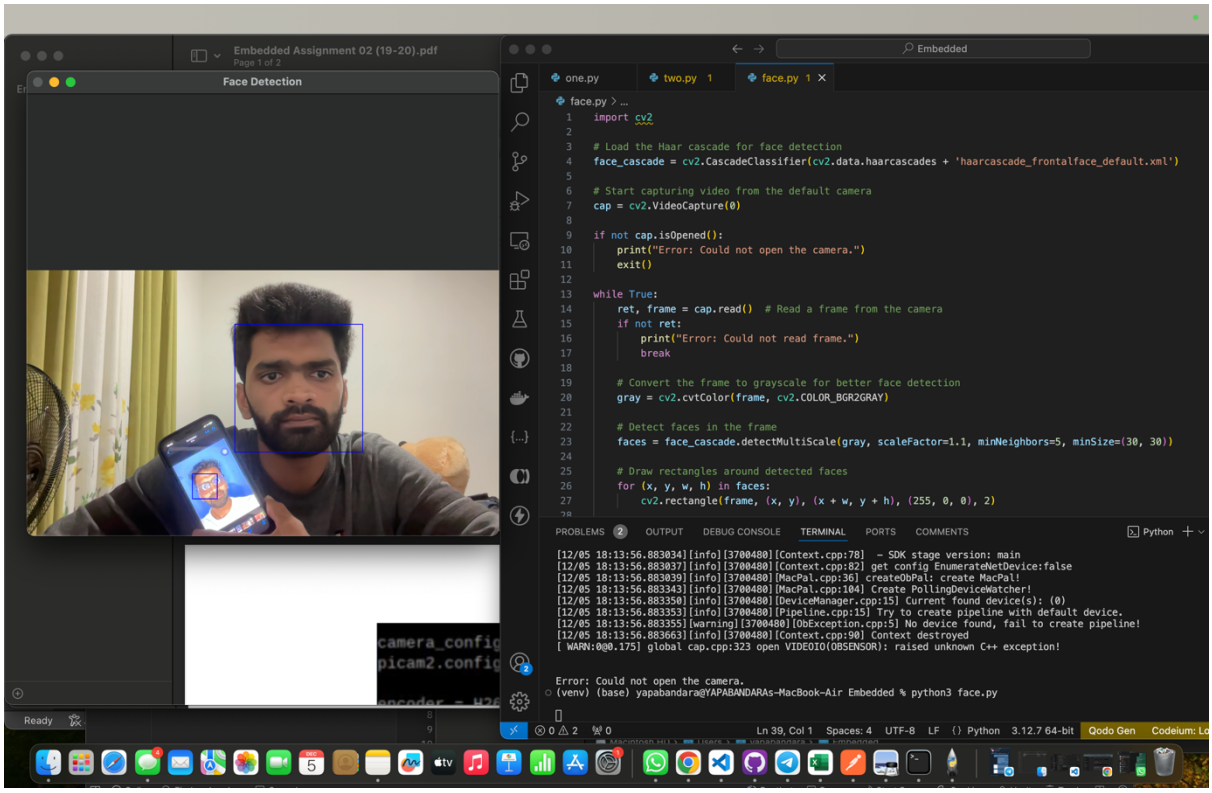
open source and extensible.

03)

function name - detect_face(img)
purpose- The function detect faces in an image and draws rectangles around them.

face detection - face_cascade.detectMultiScale(img)
Drawing rectangles- each detected face function uses cv2.rectangle to draw a white rectangle around the face.

02).




```
import cv2  # Import OpenCV
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: Could not open webcam.")
    exit()
ret, frame = cap.read()
if ret:
    cv2.imwrite('captured_image.jpg', frame)
    print("Image captured and saved as captured_image.jpg")
else:
    print("Failed to capture image.")
cap.release()
cv2.destroyAllWindows()
```