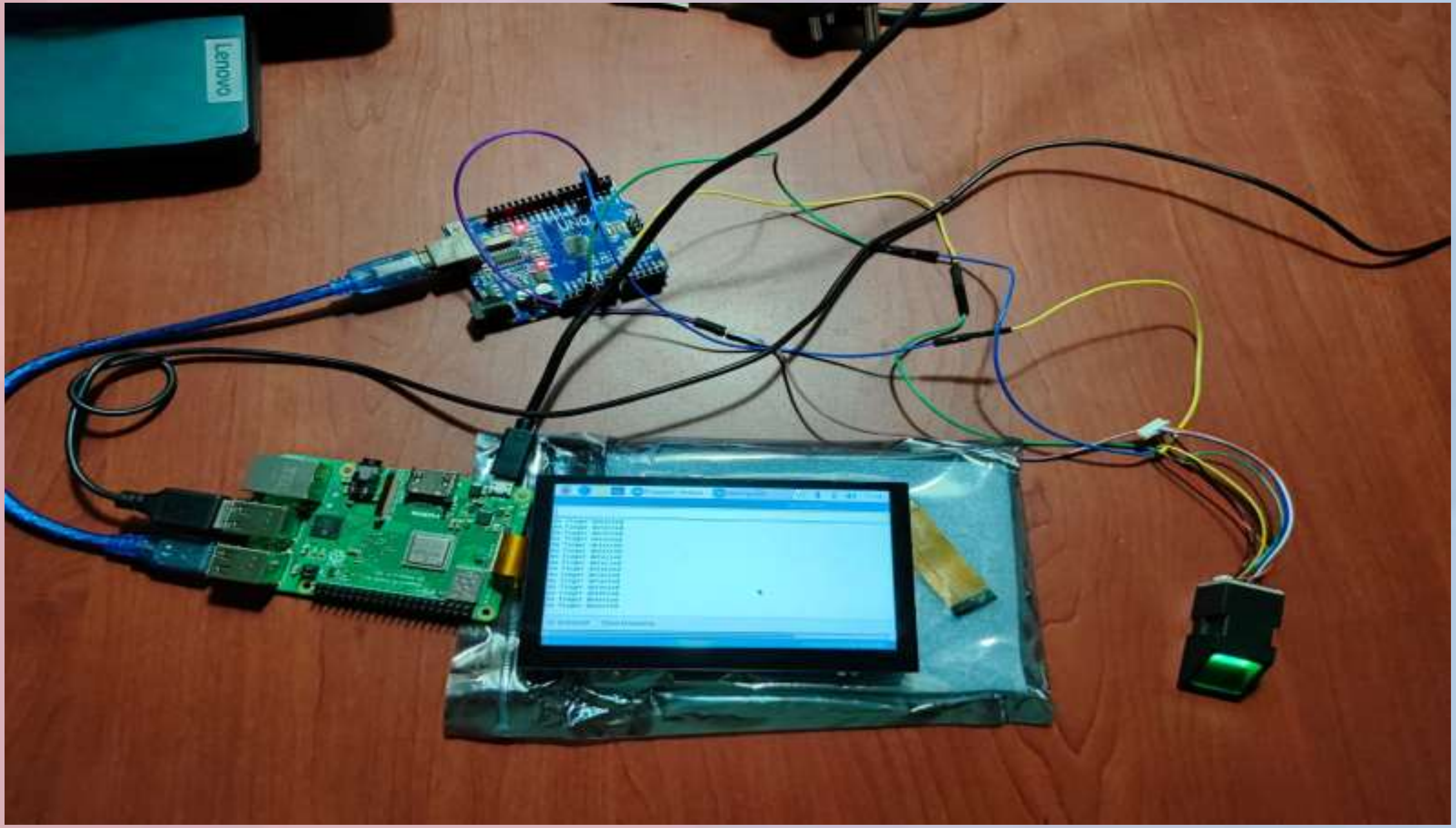
FINGERPRINT ATTENDANCE ALERTING SYSTEM



GROUP-5

- SUMANTH - (122EC0010)

AGENDA

1)Different parts and connections

2)Introduction video

3)How fingerprint codes and related codes are working

4)Email sending code

5)Telegram bot

INTRODUCTION

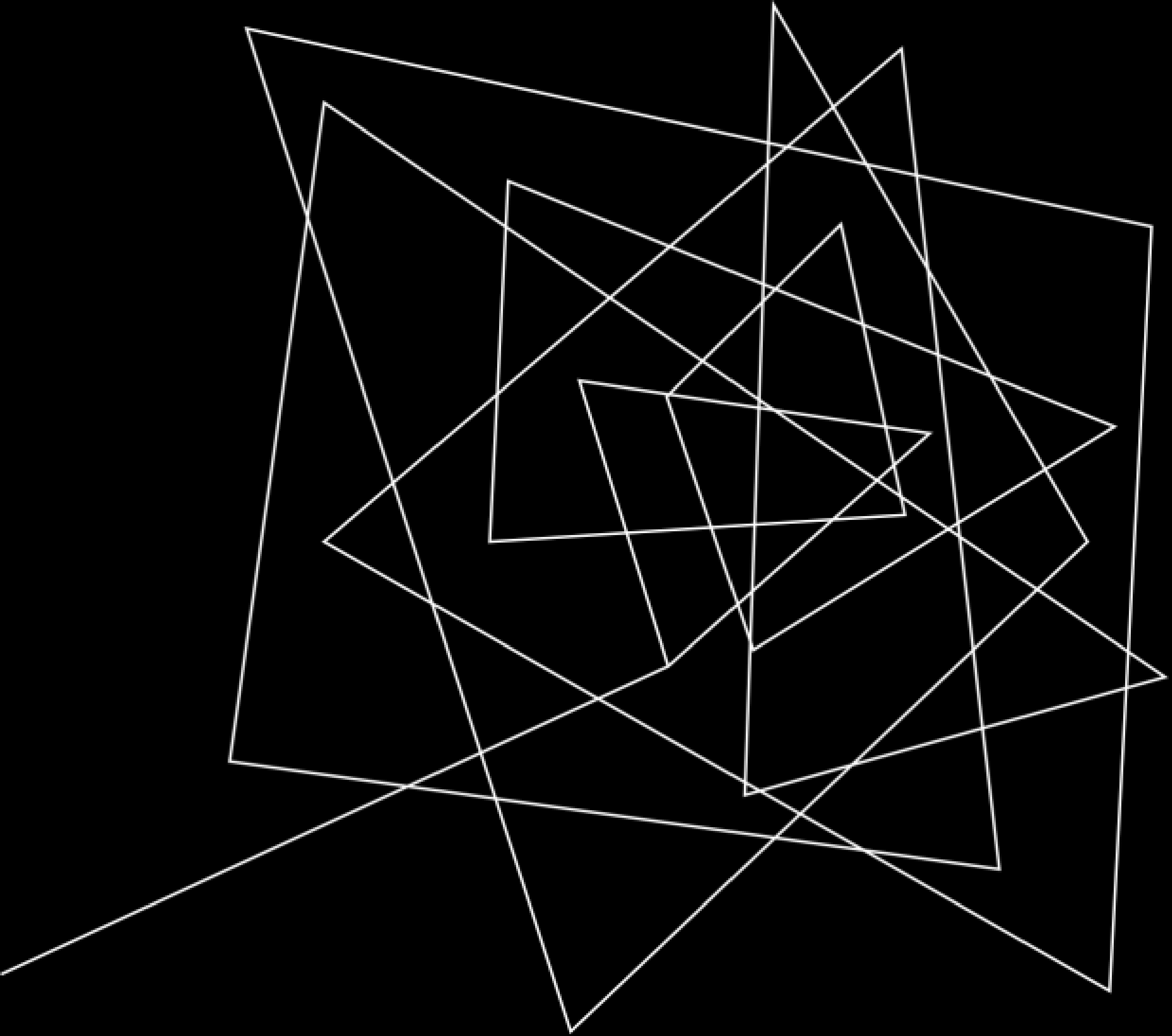
The project "Fingerprint Attendance Alerting System" aims to automate the attendance process by collecting fingerprints from users, comparing them with preloaded fingerprints, and generating email alerts for absentees. The system has several components, which are explained in detail in the following slides. Here is a summary of the main objectives and files involved:

Objectives:

Collect fingerprints from users.

Compare the collected fingerprints with preloaded fingerprints.

Generate email alerts for absentees based on the comparison



PARTS USED IN THE PROJECT

1)Arduino uno

2)Jumper wires

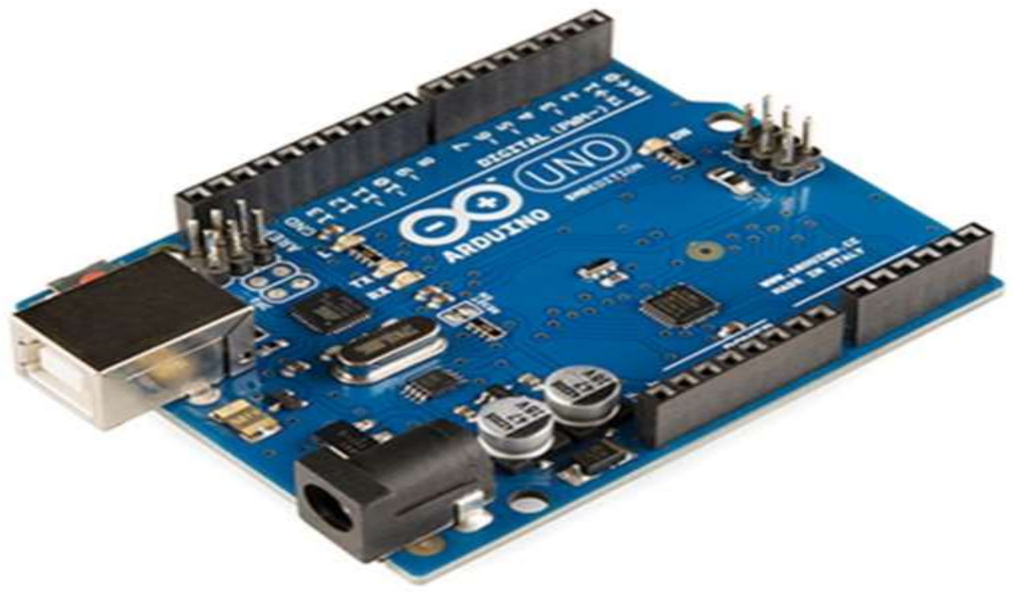
3)5 inch display for raspberry pi

4)Raspberry pi 3b+

5)Battery(power bank)

6)Fingerprint sensor

Arduino uno



Power

Power I/O Voltage	5V
Input voltage (nominal)	7-12V
DC Current per I/O Pin	20 mA

Processors

- Main Processor ATmega328P 16 MHz
- USB-Serial Processor ATmega16U2 16 MHz

Dimensions

Weight	25 g
Width	53.4 mm
Length	68.6 mm

Use of Arduino in the project :

1)It powers the fingerprint sensor
2)it connects the fingerprint sensor to the raspberry pi via serial port

Fingerprint sensor

Power

- Supply current <60mA
- Supply voltage 3.3V

Features

- Resolution: 500dpi
- Fingerprint image entry time: <1.0 seconds
- Window area: 15.3×18.2MM
- Communication Interface: USB/UART
- It nearly store fingerprint data upto 300

Dimensions

Weight	20 g
Height	24 mm
Width	21 mm
Length	46 mm

Use of fingerprint sensor in the project :

1)It takes the fingerprint data from the user
2)It stores the fingerprint data and process the fingerprints
3)it sends the data to the arduino that fingerprints are macthing or not

Raspberry pi 3B+

Use of Raspberry pi 3B+ in the project :

1)it is the main board it runs on the raspberry Linux OS
2)it runs the all python and Linux it stores the all the codes
3)it is hosting the Telegeram bot and e-mail sending server

Power

- 5V/2.5A DC via micro USB connector

Features

- Processor: Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
- Memory: 1GB LPDDR2 SDRAM
- Connectivity: 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2, 4 × USB 2.0 ports, HDMI

Dimensions

• Weight	50 g
• Height	17 mm
• Width	56 mm
• Length	85 mm

Waveshare 5 inch display

Power

- Voltage:3.3V (Supplied by Raspberry Pi display interface)
- Max Current :320mA

Features

- Resolution Ratio -800×480
- Video Interface-Raspberry Pi DSI
- Number of Touch Points-5
- Viewing Angle -60°/70°/70°/70°
- RGB888-16 Mega True Color
- Refresh Frequency-60Hz

Dimensions

Weight	211 g
Width	76 mm
Length	121 mm

Use of Display in the project :

1) The display is a visual output device that presents information, to the user

2)it is connected to raspberry shows the display it used for user friendly

UNO
ARDUINO

Power Bank

Power

- Micro Usb: Dc 5V/2.1A

Features

- Output voltage:5v
- Output current:3A

Dimensions

Weight	310 g
Height	2.4 cm
Width	4 cm
Length	7.4 cm

Use of Display in the project :

1) To power the raspberry pi and Arduino uno

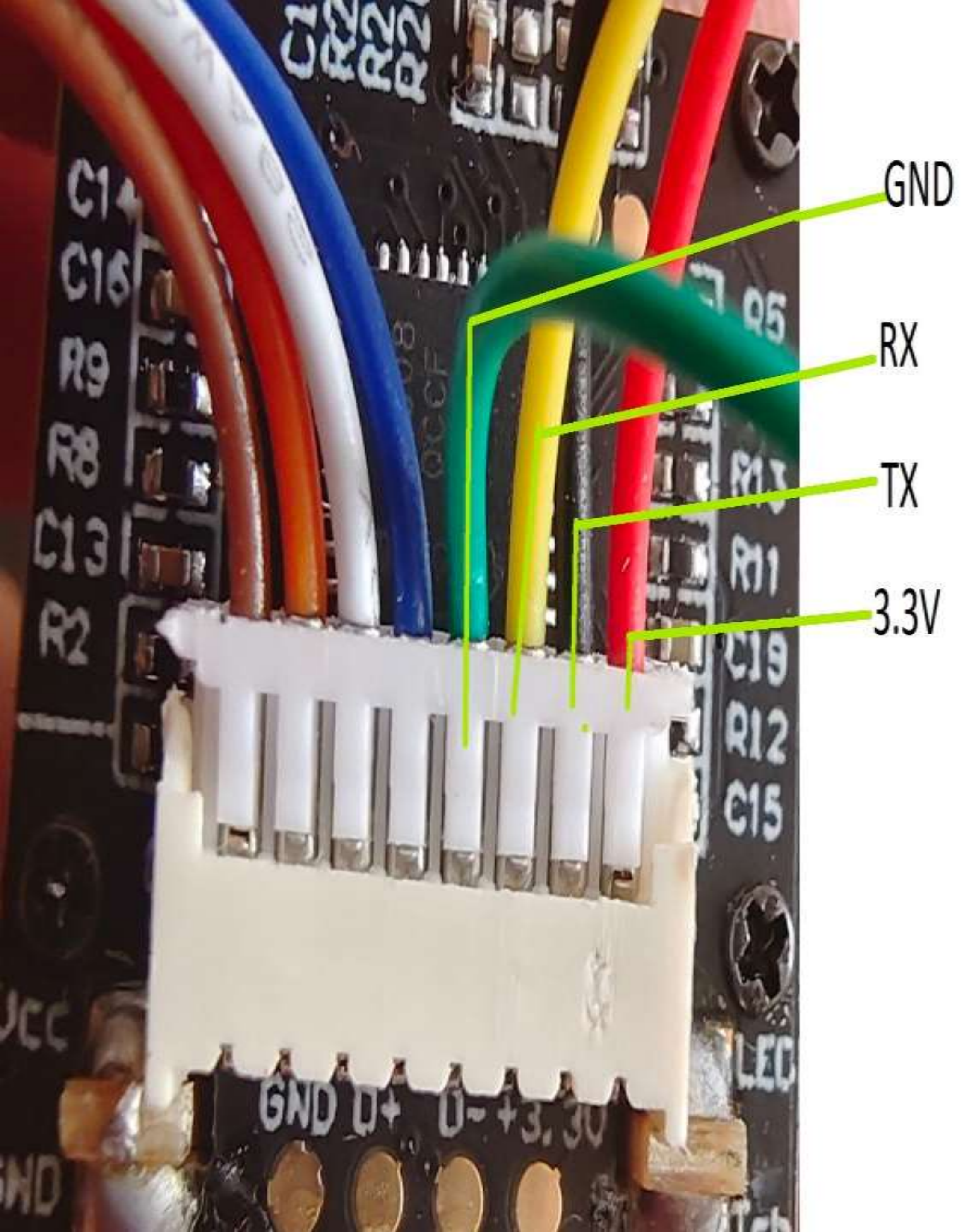
wires

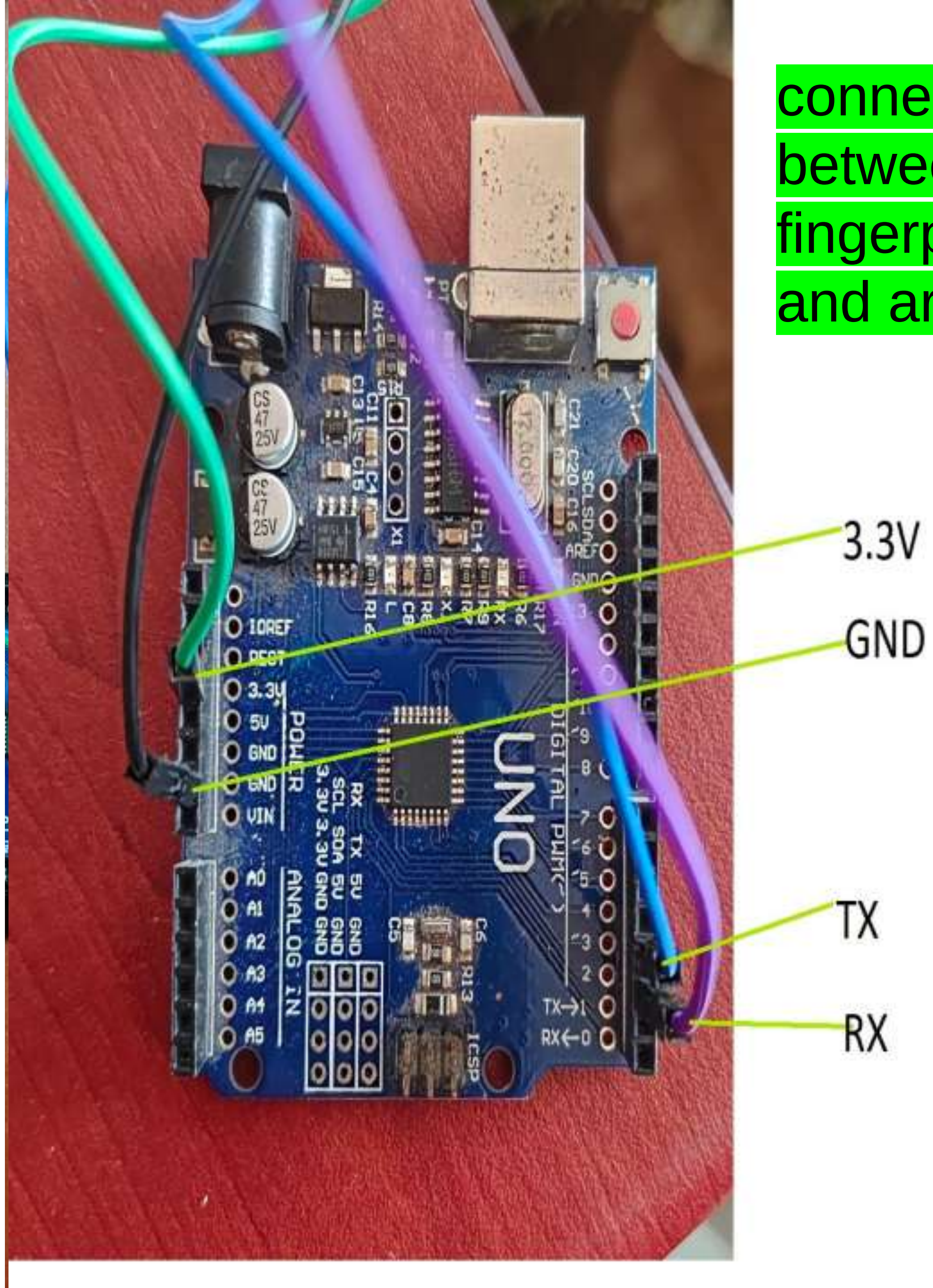
To connect Arduino to raspberry via usb

To connect the fingerprint sensor to the Arduino uno

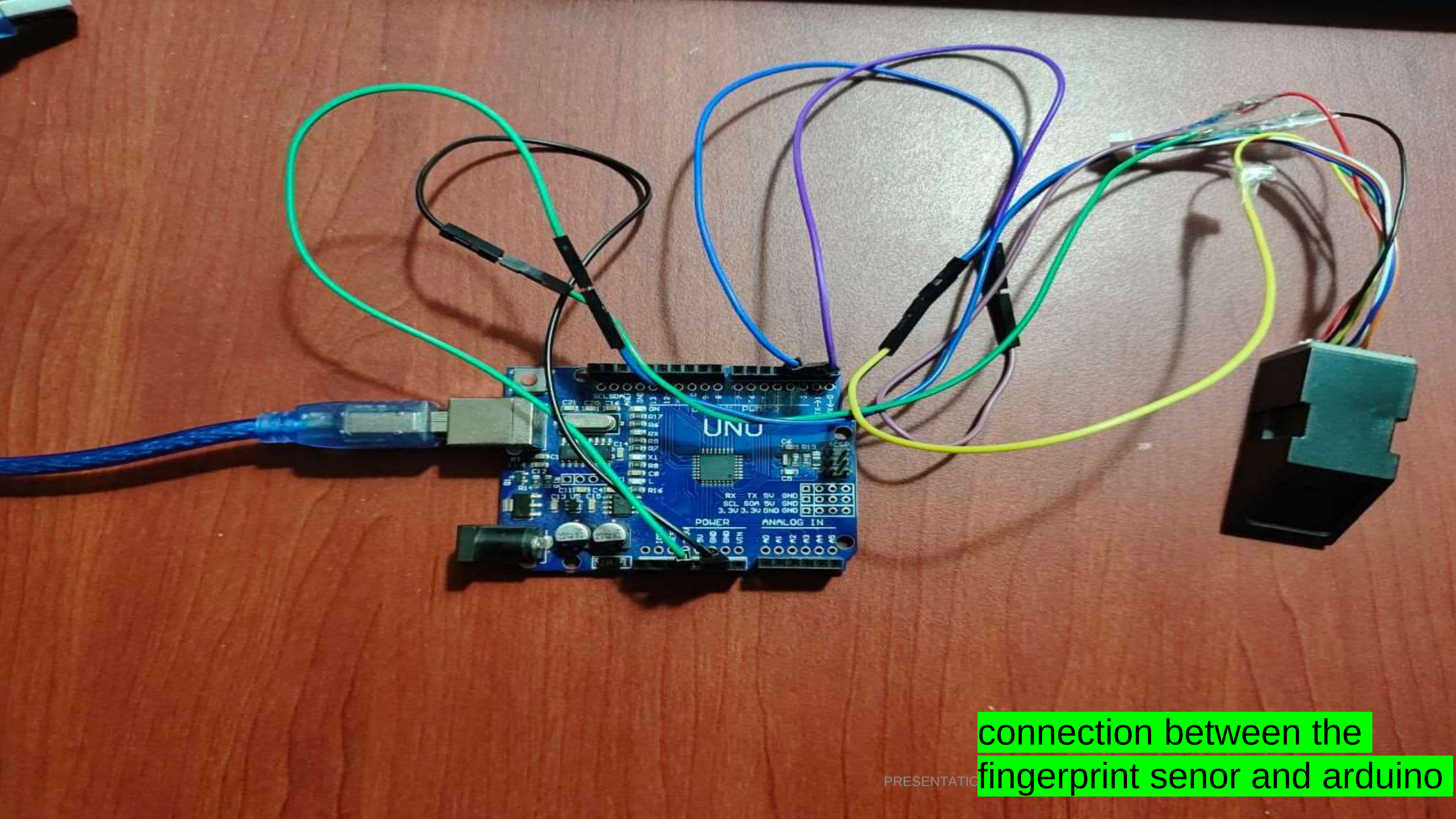
To connect the power bank to the raspberry pi

CONNECTIONS

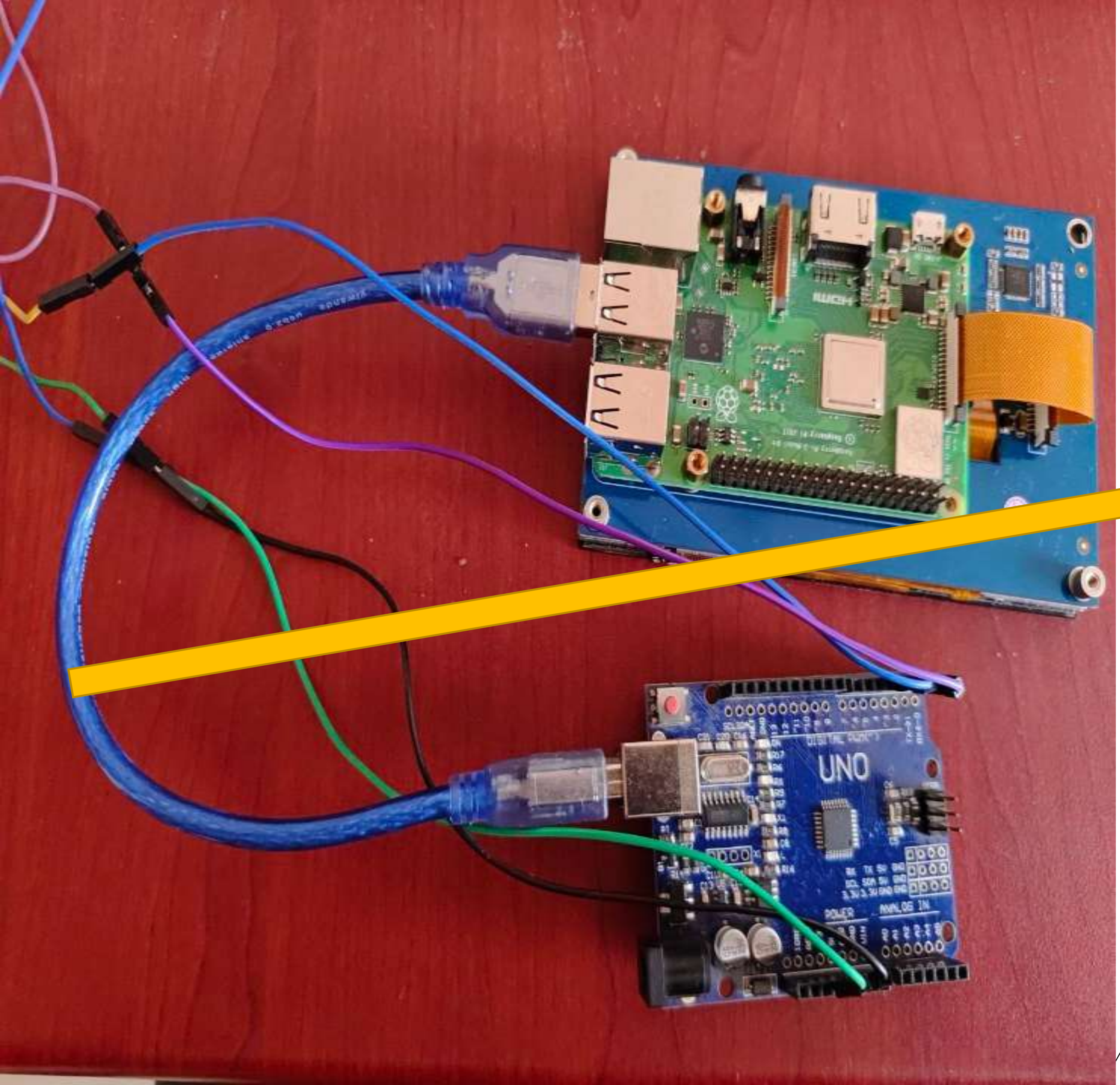




connection between the fingerprint senor and arduino



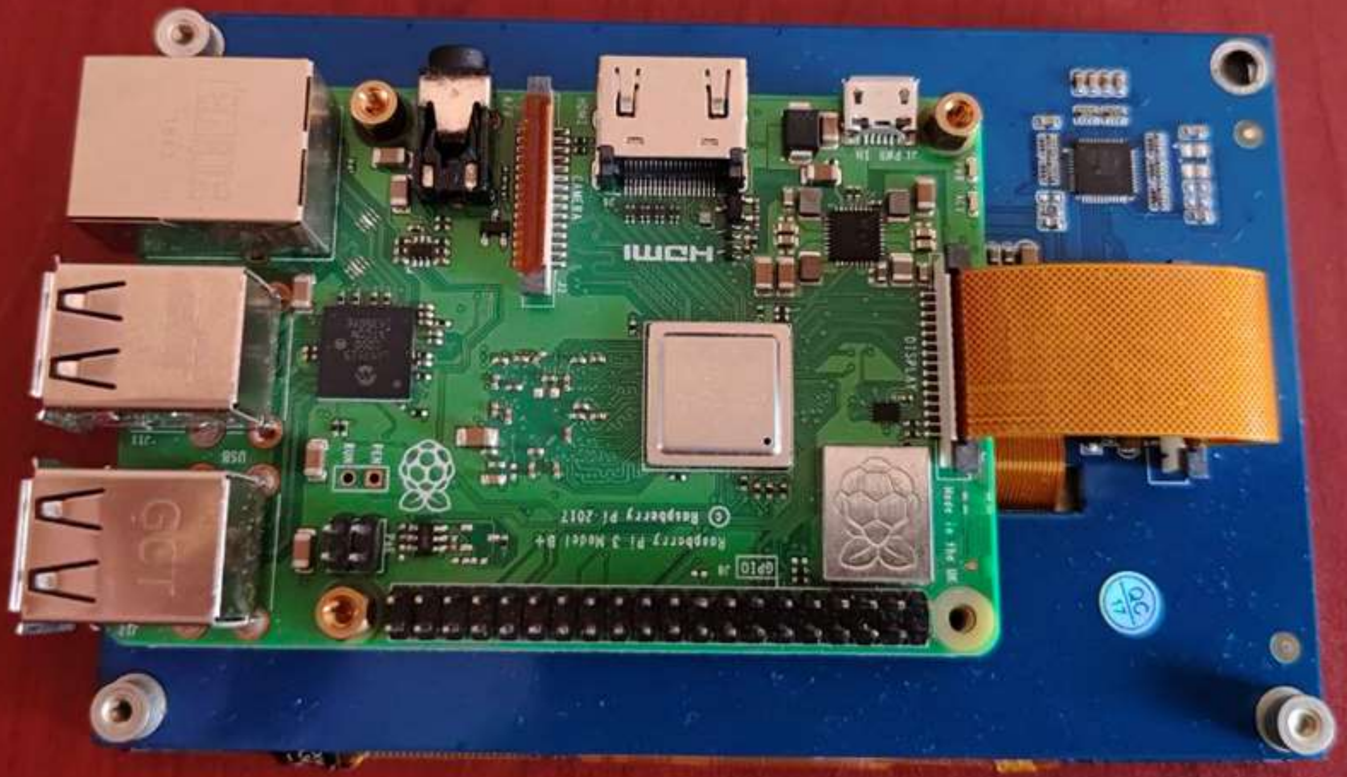
connection between the fingerprint senor and arduino



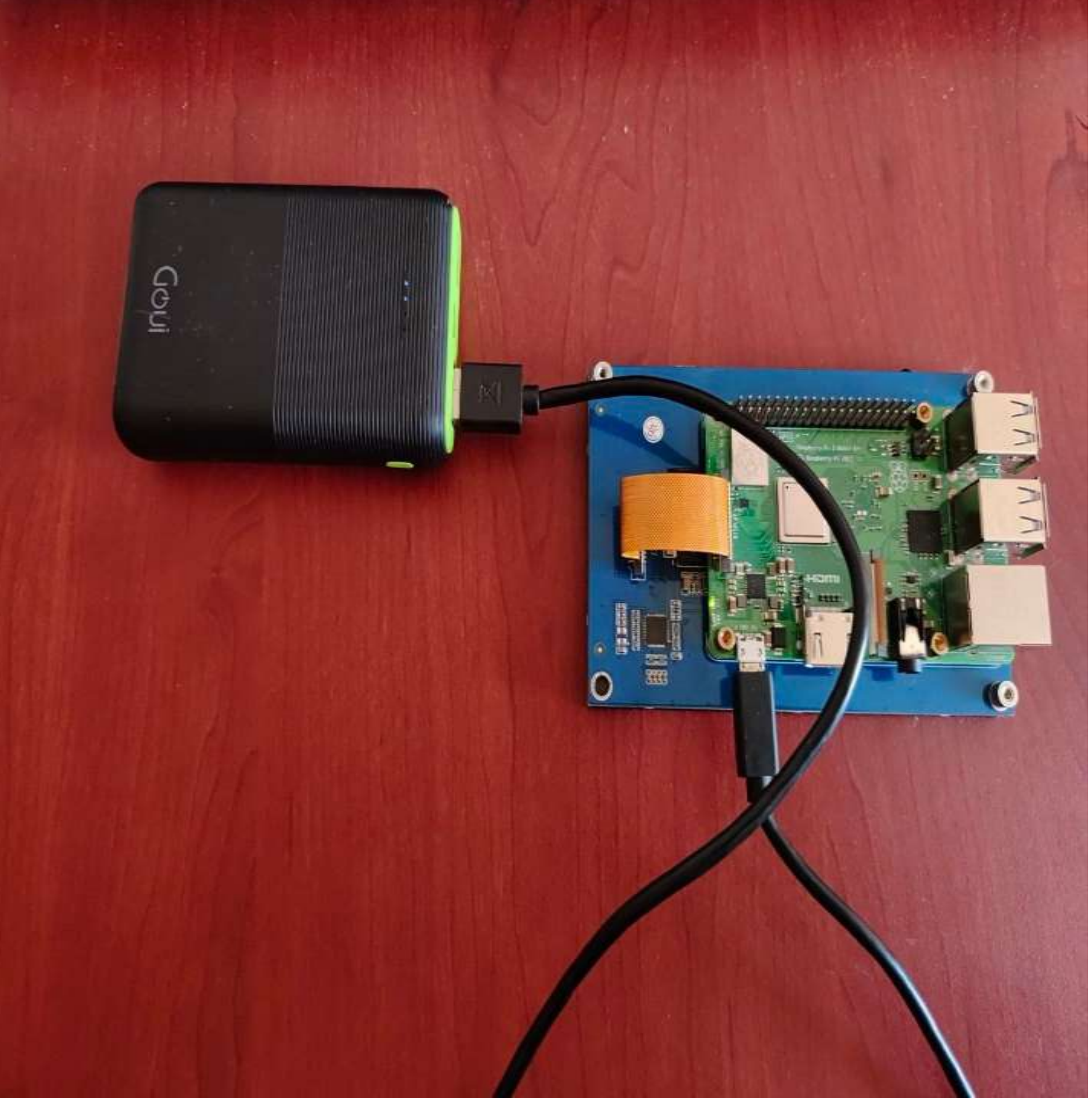
USB A TO ARDUINO UNO CONNECTOR

Connection between Arduino uno and raspberry pi using usb A to Arduino UNO connector

CONNECTION BETWEEEN THE RASPBERRY PI AND DISPLAY

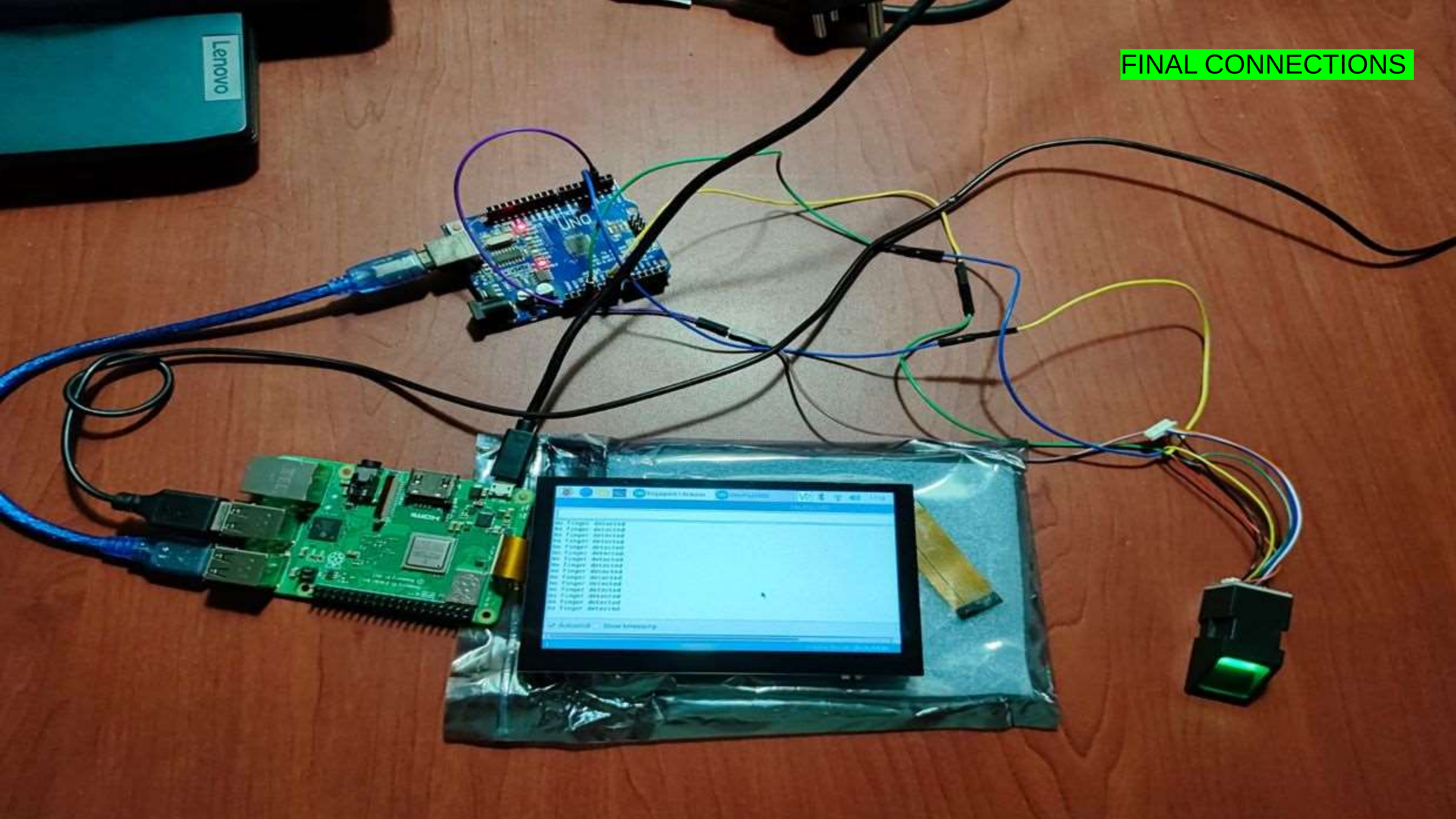


- RASPBERRY
- 15O-PIN
- DISPLAY



THE CONNECTION BETWEEN THE POWER BANK AND RASPBERRY

FINAL CONNECTIONS

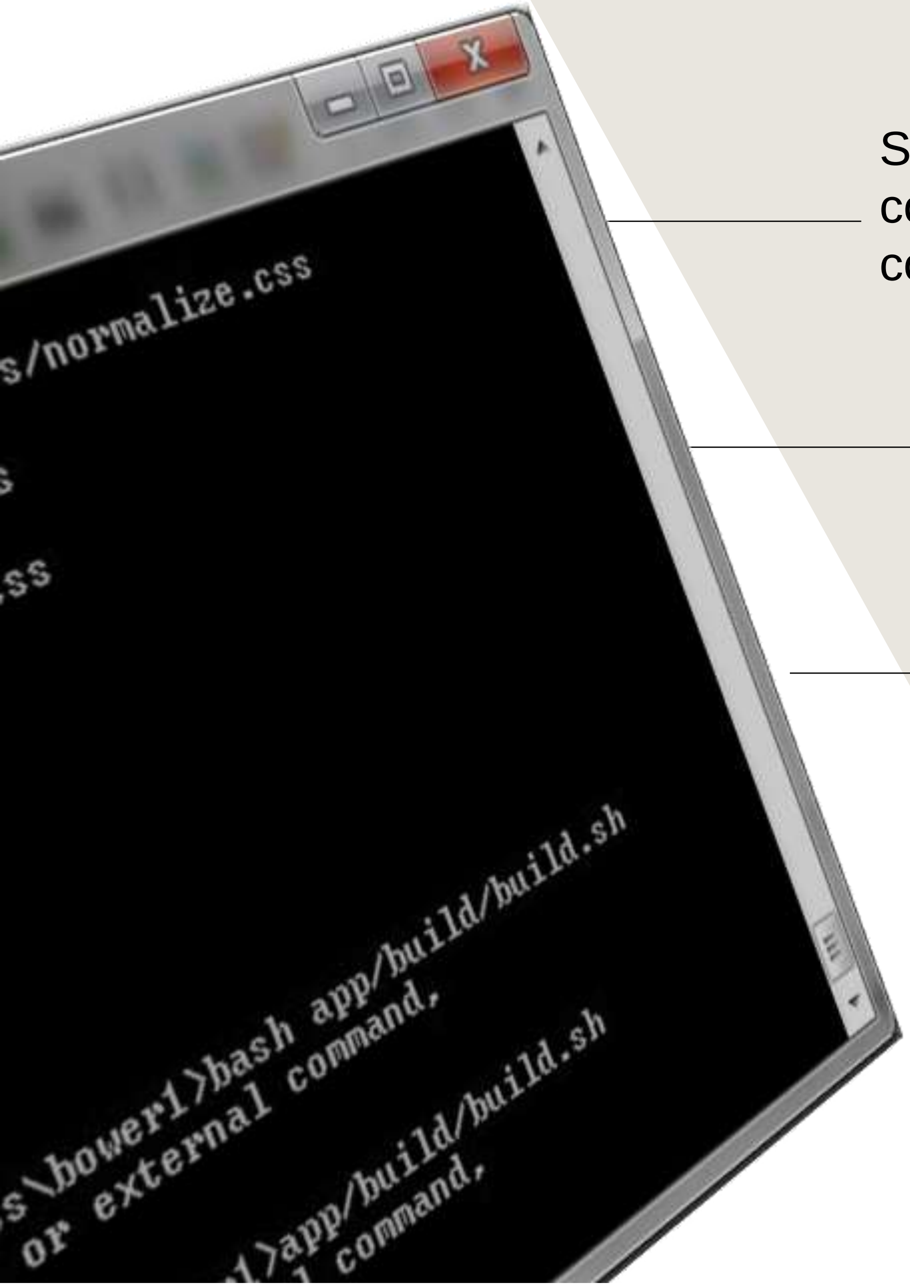


```
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
No finger detected
```

What is ts ?

- Ts means testcase
- We created new branch called ts
- This we created to show case the project with class strength 11 what will happen
- This will be used in the further slides
- Purpose of using 'ts': In this context, 'ts' signifies a 'test case'. We have preloaded the fingerprints of 11 test users (the group members) to demonstrate the working of the system.

SUMANTH	(122ts0001)
SHYAM PRASAD	(122ts0002)
SHAIK AFTHAB	(122ts0003)
SREE CHARAN REDDY	(122ts0004)
ANIRUDH	(122ts0005)
CH.SHYAM	(122ts0006)
KUSHALA	(122ts0007)
AMRUTHA	(122ts0008)
SUBHASH CHANDRA	(122ts0009)
ABHISHEK	(122ts0010)
KESHAVA SHARMA	(122ts0011)



```
s/normalize.css
s
.ss
s\bower1>bash app/build/build.sh
or external command,
1>app/build/build.sh
l command,
```

WHAT IS SH FILE ?

Shell scripts are plain text files that contain a series of commands that can be executed by the shell, which is the command-line interpreter in Linux

Shell scripts are used to automate tasks or to execute a sequence of commands

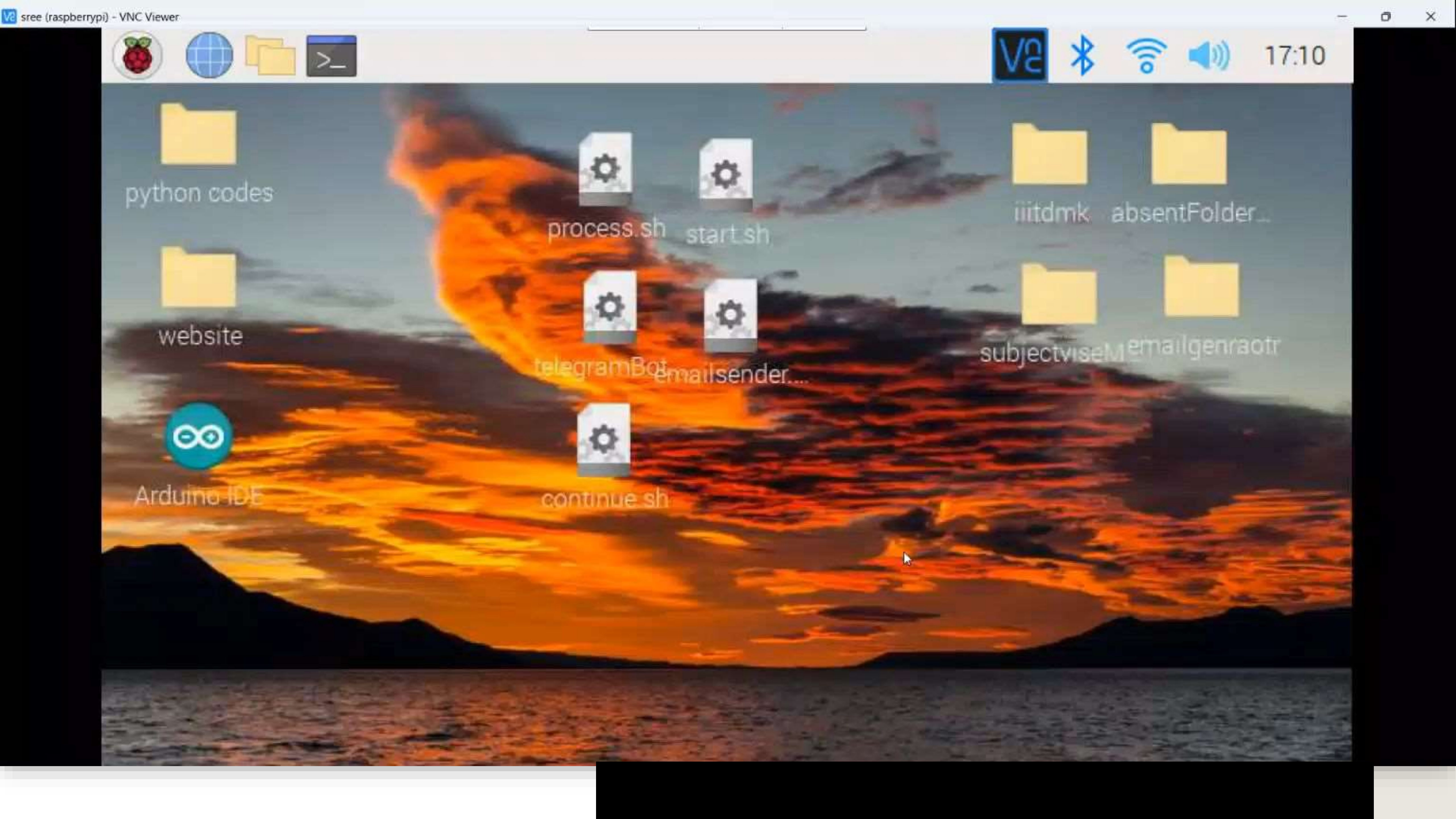
Shell scripts can include various types of commands, including system commands, control structures.

They allow users to combine multiple commands and create complex scripts to perform repetitive or specialized tasks.

WHEN THE RASPBERRY PI POWERED ON

The next slide contain a video that shows the basic function of the overall project

The video basically about function of all the codes used


sree (raspberrypi) - VNC Viewer
17:10
python codes
process.sh
start.sh
iiitdmk
absentFolder...
website
telegramBot
emailsender....
subjectviseM...
emailgenraotr
Arduino IDE
continue.sh

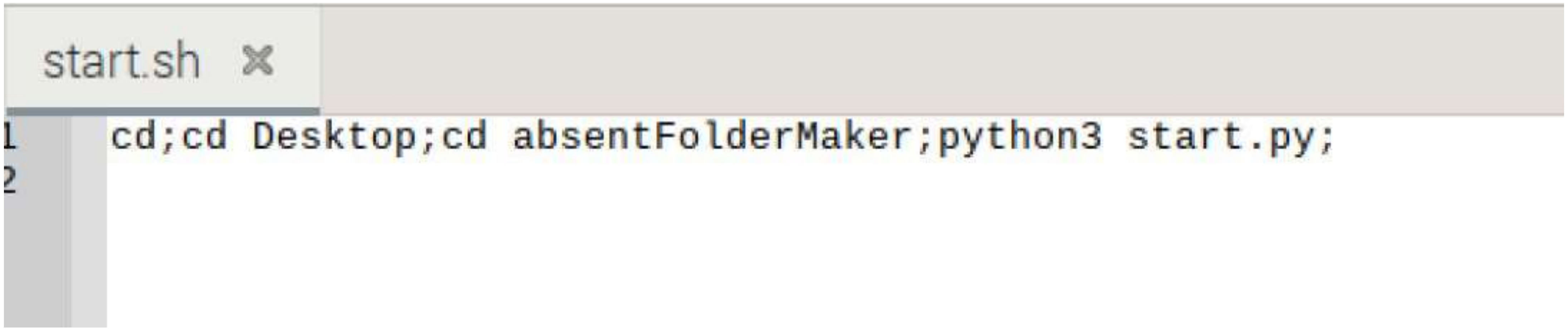
Fingerprint sensor driver

- https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library we took the code from the manufacture of the fingerprint sensor
- These codes directly installed on the linux systems and when we connect the Arduino with fingerprint sensor it automatically detects and available for use using the USB port serial which will be discussed in the further slides
- This driver helps us to connect the fingerprint to terminal so we can have data
- Driver is written in the c++ language
- The driver contains two types of code for enroll and checking the fingerprint

HOW THE START.SH WORKS ?

- First , it travels across the folder where there the start.py is present

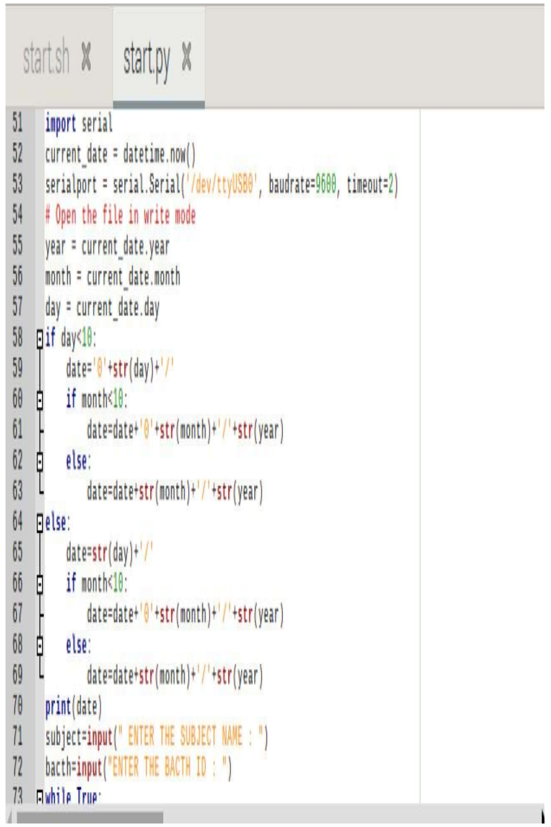
- Start.sh file will run a python code called start.py
- The start.py file will start taking the fingerprints and start taking the attendance



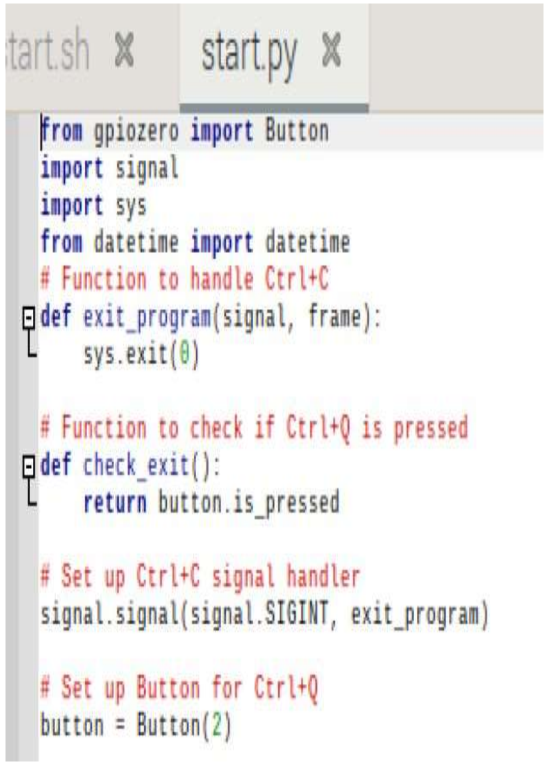
```
cd;cd Desktop;cd absentFolderMaker;python3 start.py;
```

This is the linux commands start.sh contains

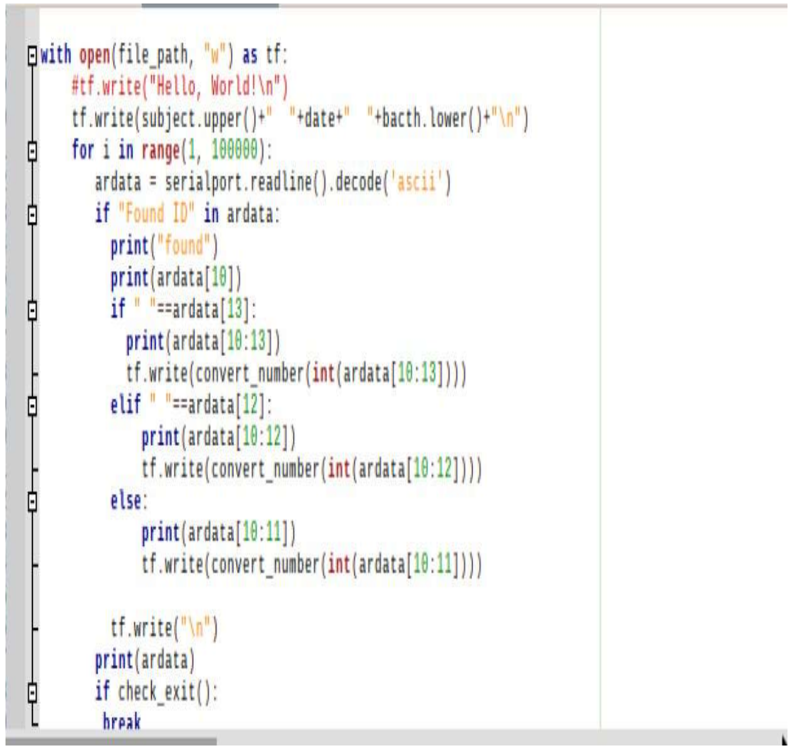
START.PY



```python
import serial
current_date = datetime.now()
serialport = serial.Serial('/dev/ttyUSB0', baudrate=9600, timeout=2)
# Open the file in write mode
year = current_date.year
month = current_date.month
day = current_date.day
if day<10:
    date='0'+str(day)+'/'
    if month<10:
        date=date+'0'+str(month)+'/'+str(year)
    else:
        date=date+str(month)+'/'+str(year)
else:
    date=str(day)+'/'
    if month<10:
        date=date+'0'+str(month)+'/'+str(year)
    else:
        date=date+str(month)+'/'+str(year)
print(date)
subject=input(" ENTER THE SUBJECT NAME : ")
bacth=input("ENTER THE BACTH ID : ")
while True:
```

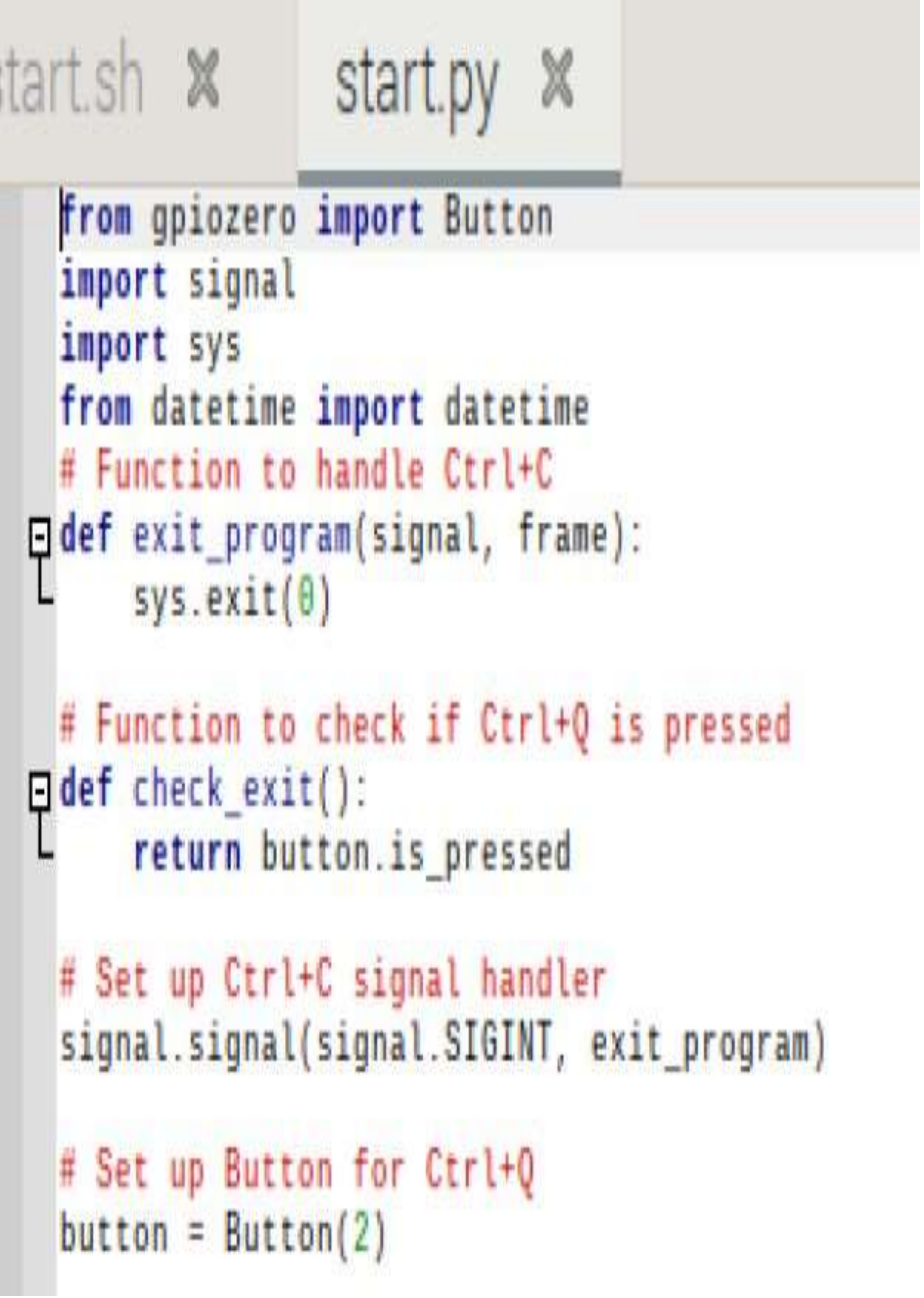


```python
from gpiozero import Button
import signal
import sys
from datetime import datetime
# Function to handle Ctrl+C
def exit_program(signal, frame):
    sys.exit(0)

# Function to check if Ctrl+Q is pressed
def check_exit():
    return button.is_pressed

# Set up Ctrl+C signal handler
signal.signal(signal.SIGINT, exit_program)

# Set up Button for Ctrl+Q
button = Button(2)
```



```python
with open(file_path, "w") as tf:
    #tf.write("Hello, World!\n")
    tf.write(subject.upper()+"  "+date+"  "+bacth.lower()+"\n")
    for i in range(1, 100000):
      ardata = serialport.readline().decode('ascii')
      if "Found ID" in ardata:
        print("found")
        print(ardata[10])
        if " "==ardata[13]:
          print(ardata[10:13])
          tf.write(convert_number(int(ardata[10:13])))
        elif " "==ardata[12]:
            print(ardata[10:12])
            tf.write(convert_number(int(ardata[10:12])))
        else:
            print(ardata[10:11])
            tf.write(convert_number(int(ardata[10:11])))

        tf.write("\n")
      print(ardata)
      if check_exit():
       break
```

- Start.py is a basically a python code
- For easy understanding we divided the code into three parts

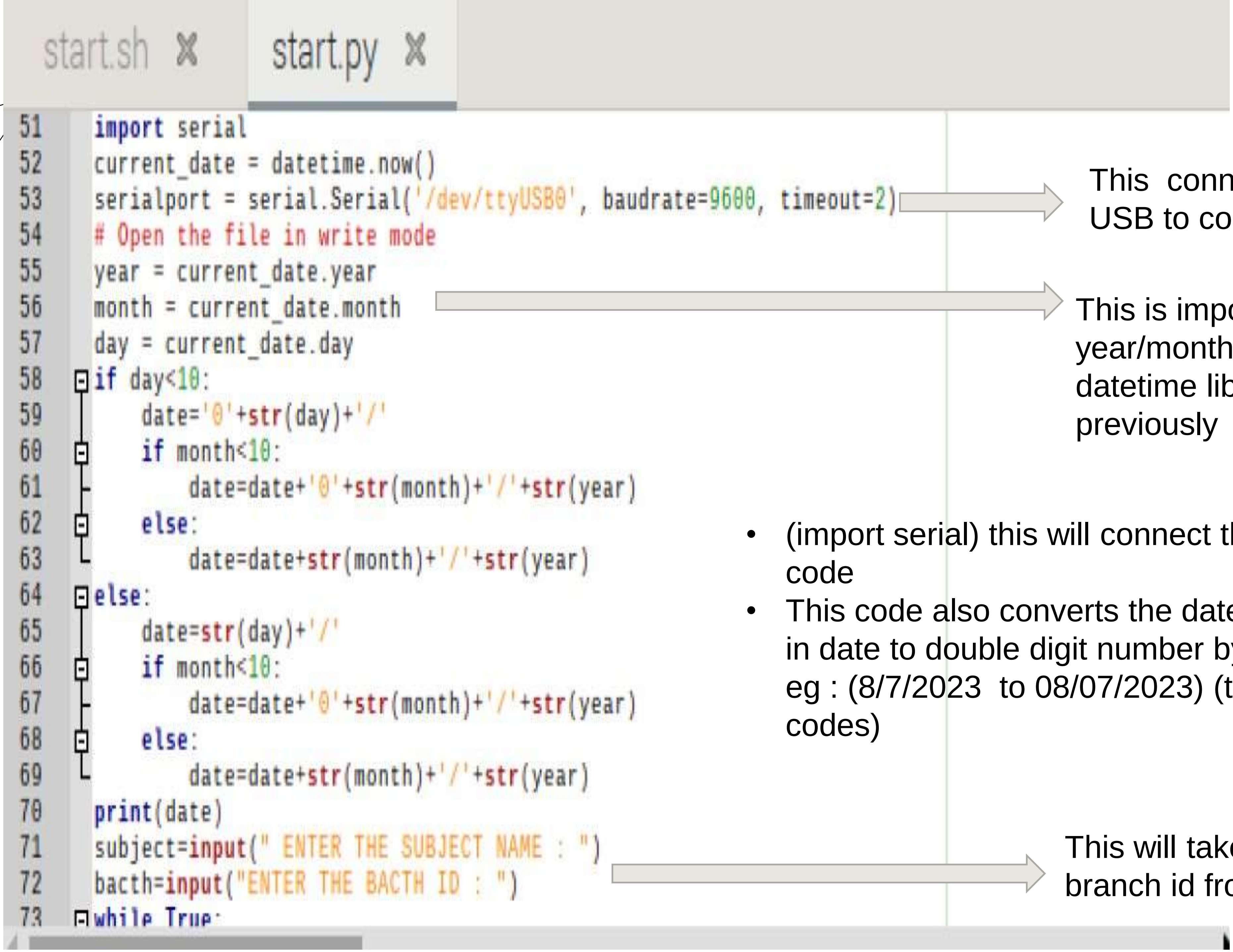

tart.sh ✖ start.py ✖

```
from gpiozero import Button
import signal
import sys
from datetime import datetime
# Function to handle Ctrl+C
def exit_program(signal, frame):
    sys.exit(0)

# Function to check if Ctrl+Q is pressed
def check_exit():
    return button.is_pressed

# Set up Ctrl+C signal handler
signal.signal(signal.SIGINT, exit_program)

# Set up Button for Ctrl+Q
button = Button(2)
```

START.PY

- (from datetime import datetime)It will imports the current date and time from the device
- (from gpiozero import Button) it will connect the keyboard the code
- (# Function to handle Ctrl+C) this function handle the program when we press ctrl+c it will end the program

START.PY



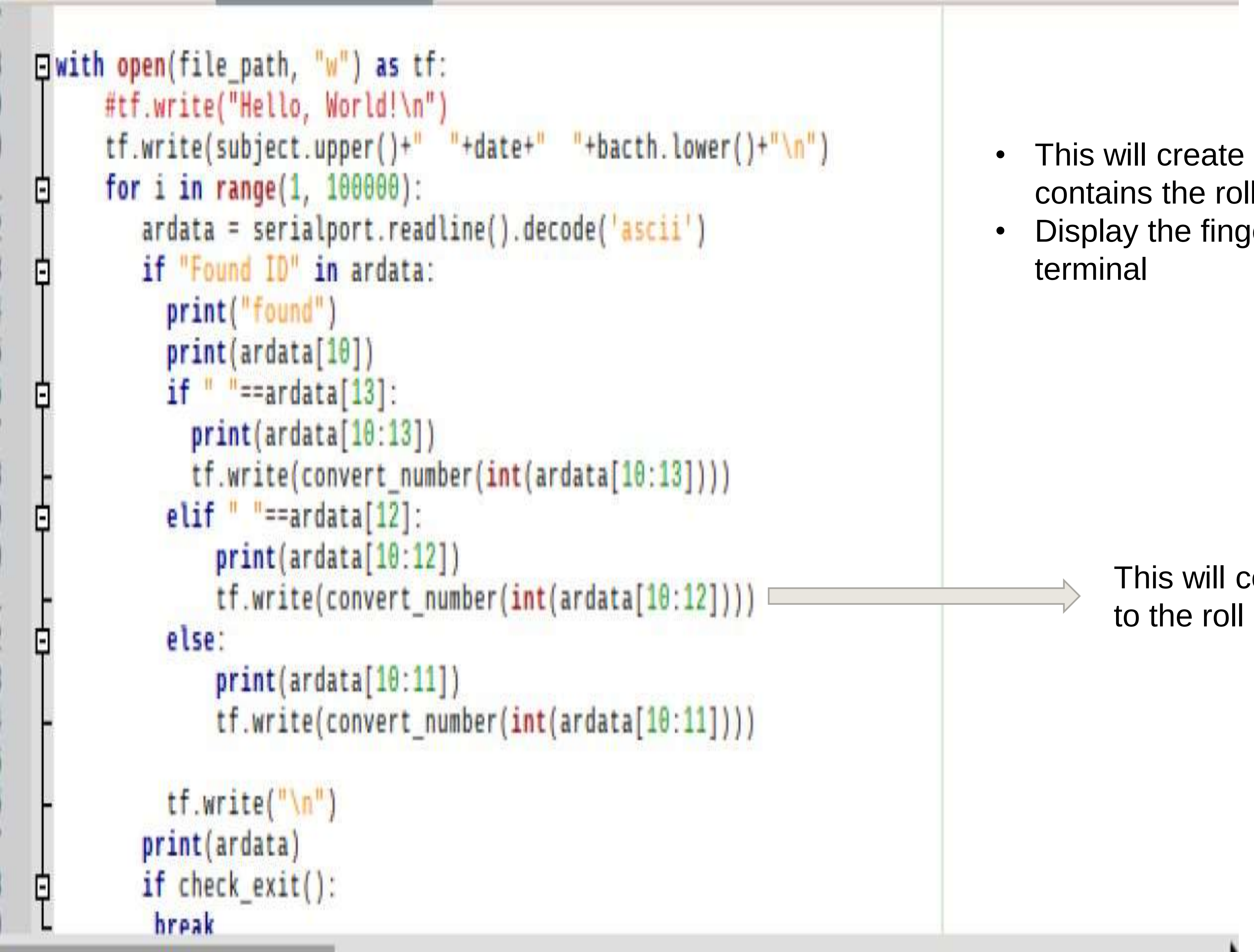
```
import serial
current_date = datetime.now()
serialport = serial.Serial('/dev/ttyUSB0', baudrate=9600, timeout=2)
# Open the file in write mode
year = current_date.year
month = current_date.month
day = current_date.day
if day<10:
    date='0'+str(day)+'/'
    if month<10:
        date=date+'0'+str(month)+'/'+str(year)
    else:
        date=date+str(month)+'/'+str(year)
else:
    date=str(day)+'/'
    if month<10:
        date=date+'0'+str(month)+'/'+str(year)
    else:
        date=date+str(month)+'/'+str(year)
print(date)
subject=input(" ENTER THE SUBJECT NAME : ")
bacth=input("ENTER THE BACTH ID : ")
while True:
```

This connects the specific USB to code

This is importing year/month/day from the datetime library that is imported previously

- (import serial) this will connect the serial port (usb port) to the code
- This code also converts the date without single digit number in date to double digit number by adding the zero eg : (8/7/2023 to 08/07/2023) (this helps int the further codes)

This will take the subject name and branch id fromo the user



```
with open(file_path, "w") as tf:
    #tf.write("Hello, World!\n")
    tf.write(subject.upper()+"  "+date+"  "+bacth.lower()+"\n")
    for i in range(1, 100000):
       ardata = serialport.readline().decode('ascii')
       if "Found ID" in ardata:
         print("found")
         print(ardata[10])
         if " "==ardata[13]:
           print(ardata[10:13])
           tf.write(convert_number(int(ardata[10:13])))
         elif " "==ardata[12]:
             print(ardata[10:12])
             tf.write(convert_number(int(ardata[10:12])))
         else:
             print(ardata[10:11])
             tf.write(convert_number(int(ardata[10:11])))

         tf.write("\n")
       print(ardata)
       if check_exit():
        break
```

START.PY

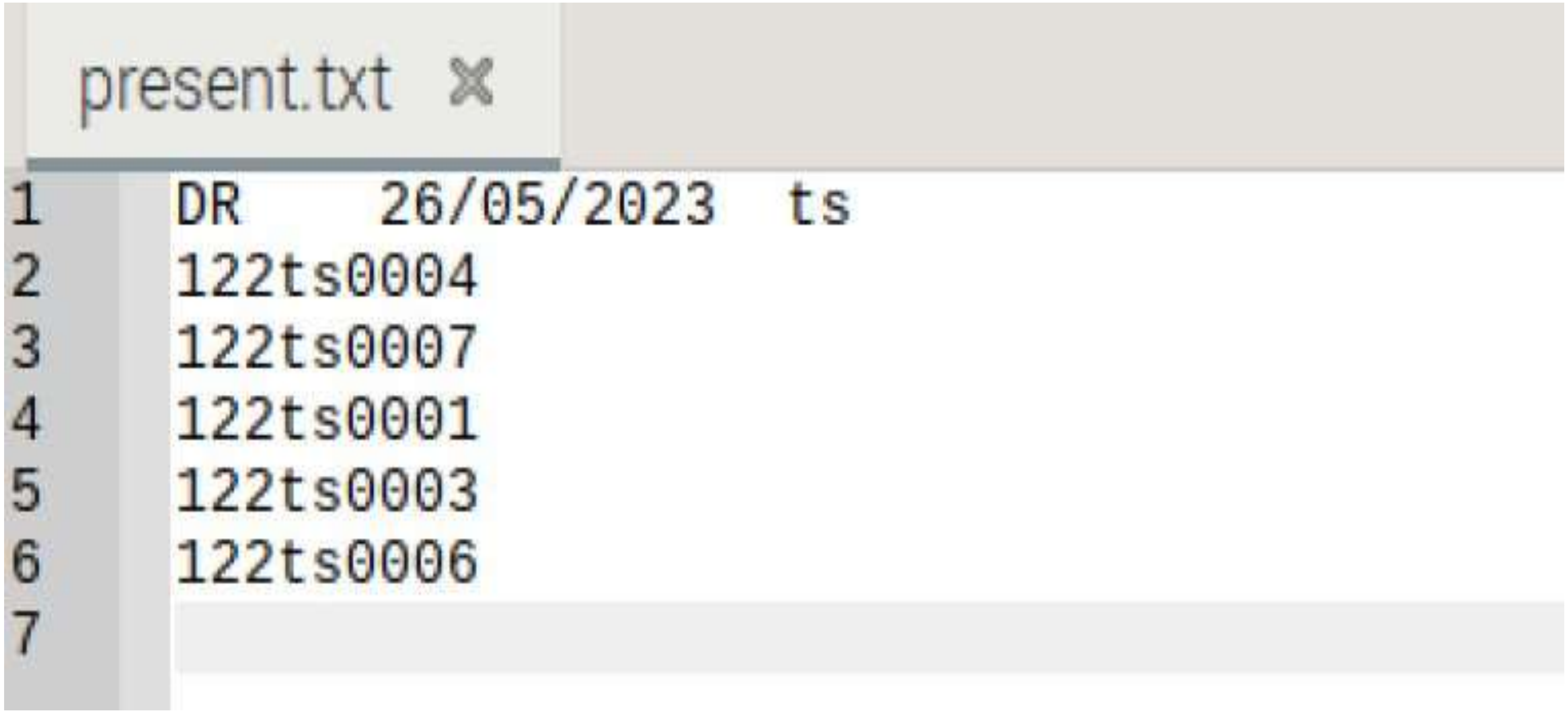
- This will create present.txt in which it contains the roll numbers who are present
- Display the fingerprint id if it is found in the terminal

This will convert the fingerprint id to the roll number

START.PY

- After running the program it will create the present.txt

Present.txt will look like this:

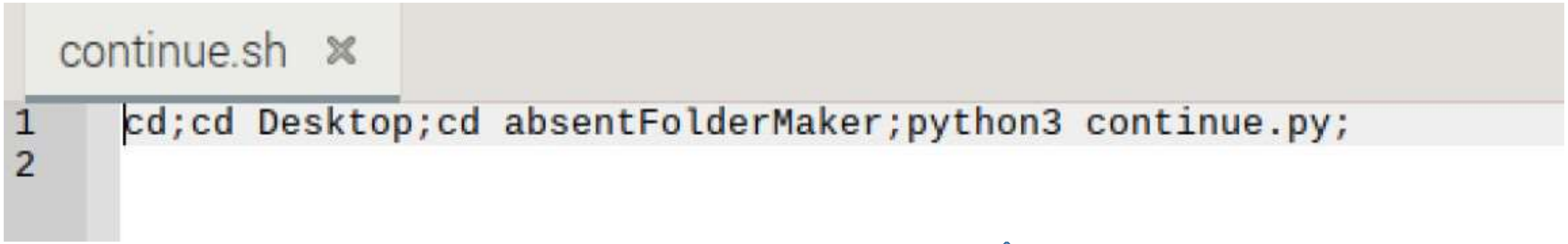


```
DR    26/05/2023  ts
122ts0004
122ts0007
122ts0001
122ts0003
122ts0006
```

HOW THE CONTINUE.SH WORKS ?

- First , it travels across the folder where there the continue.py is present

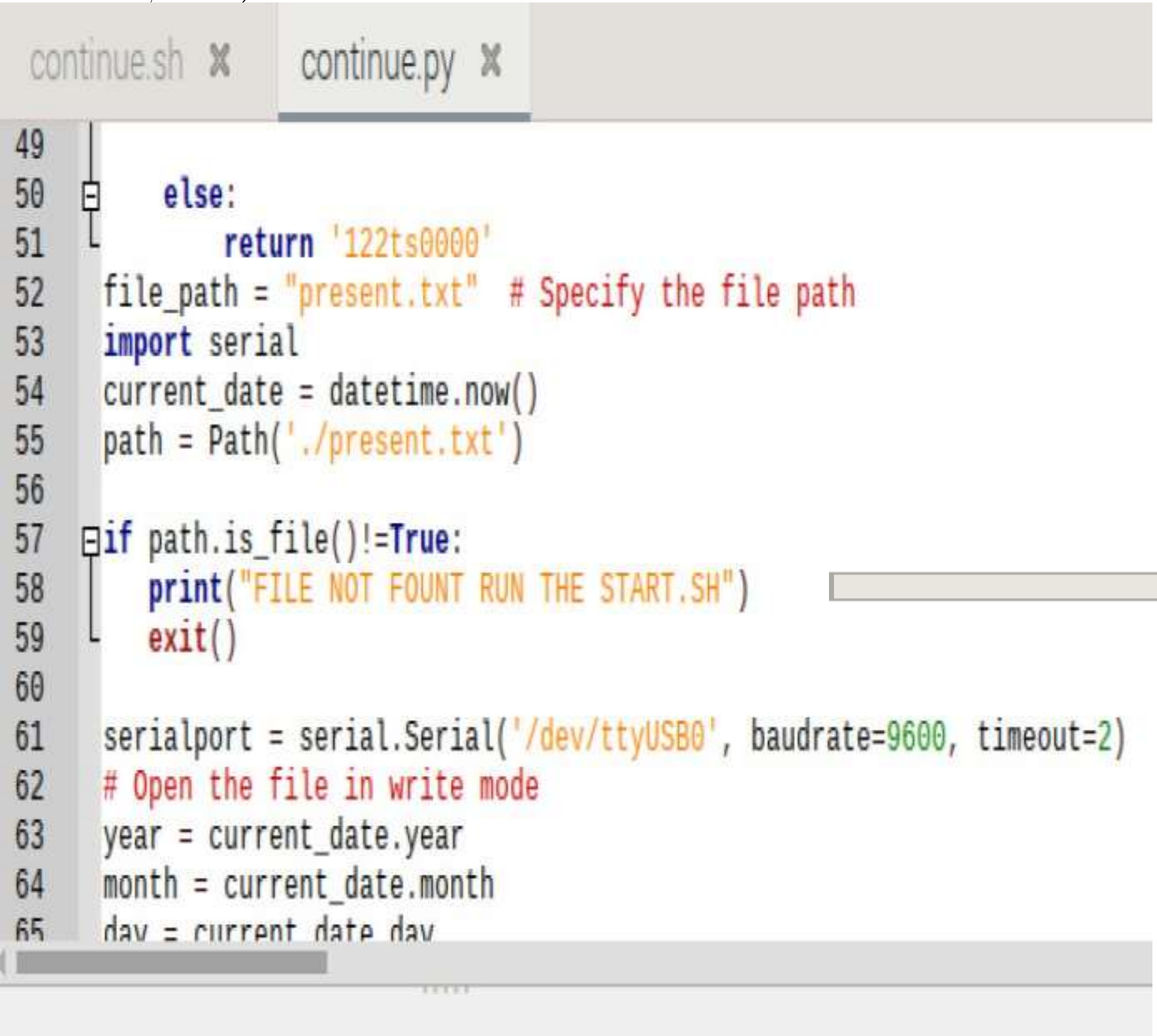
- Start.sh file will run a python code called continue.py
- The continue.py file will start taking the fingerprints and start taking the attendance



```
cd;cd Desktop;cd absentFolderMaker;python3 continue.py;
```

This is the linux commands continue.sh contains

DIFFERENCE BETWEEN CONTINUE.PY AND START.PY


continue.sh
continue.py

```
    else:
        return '122ts0000'
file_path = "present.txt"  # Specify the file path
import serial
current_date = datetime.now()
path = Path('./present.txt')

if path.is_file()!=True:
    print("FILE NOT FOUNT RUN THE START.SH")
    exit()

serialport = serial.Serial('/dev/ttyUSB0', baudrate=9600, timeout=2)
# Open the file in write mode
year = current_date.year
month = current_date.month
day = current_date.day
```

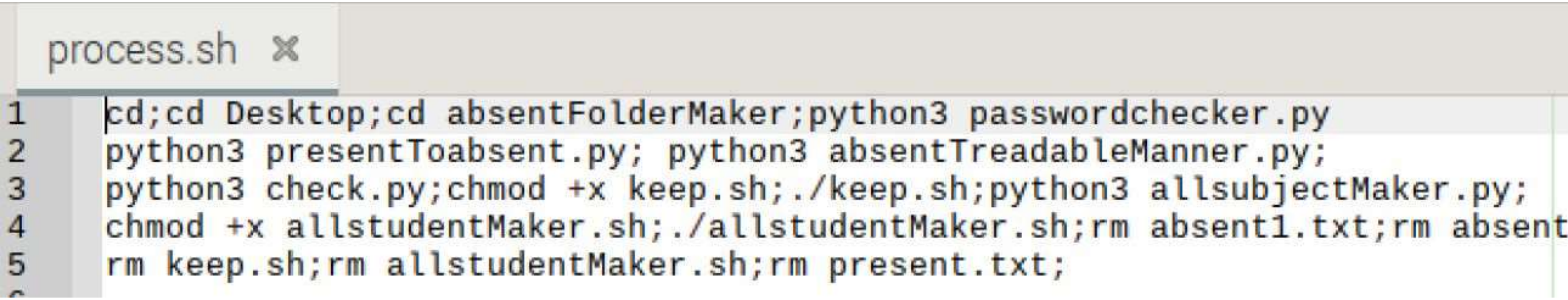
- It contains same code as start.py but it contains some extra functions (this only runs if start.py already run before because it will check present.txt is present or not)
- If present only it will start adding roll numbers

This function will check that present.txt is found or not

HOW THE PROCESS.SH WORKS ?

- First , it travels across the folder where there the all codes present

- process.sh file will run a 5 python code
- The process.sh file process the data



process.sh

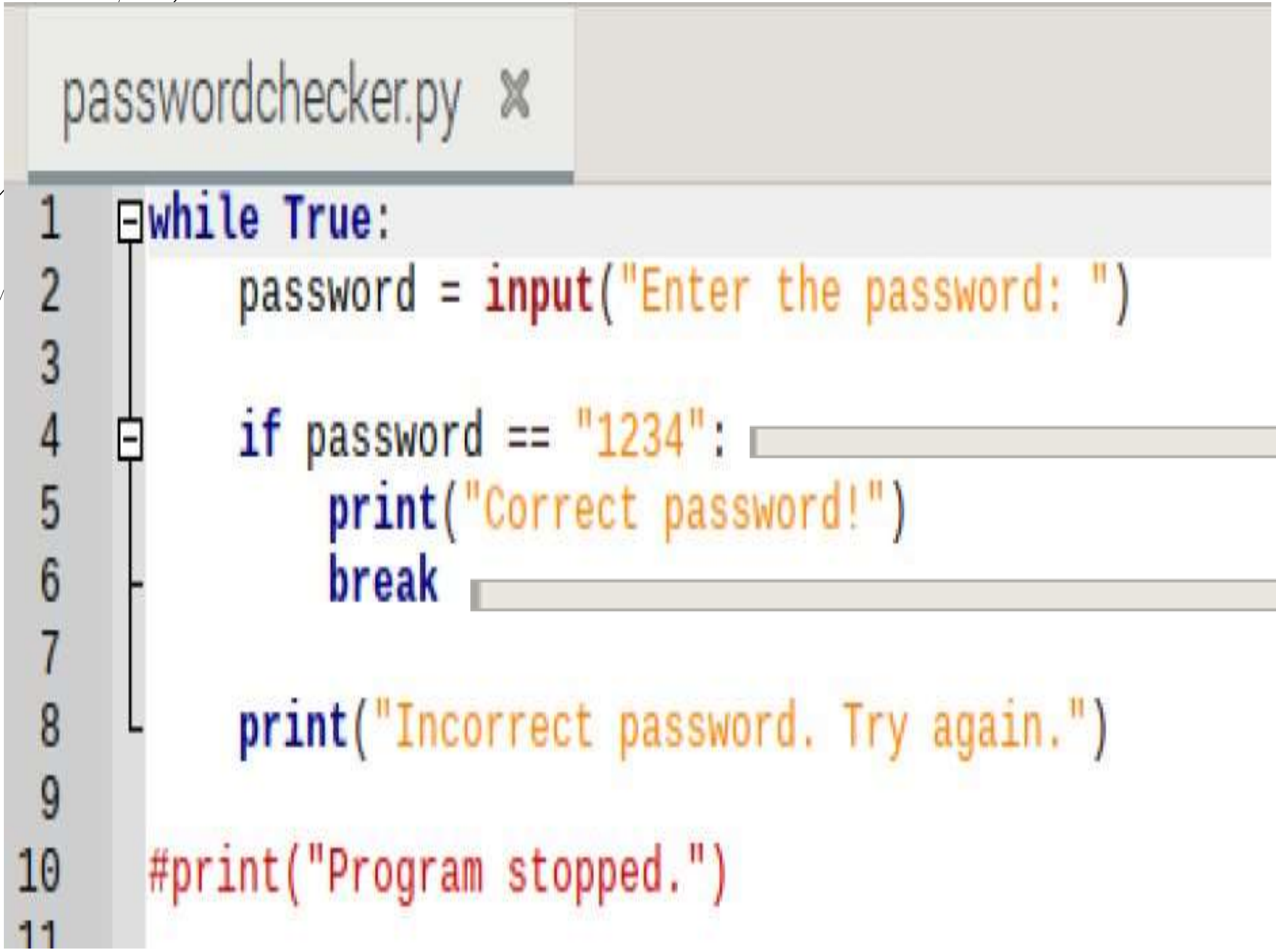
```
cd;cd Desktop;cd absentFolderMaker;python3 passwordchecker.py
python3 presentToabsent.py; python3 absentTreadableManner.py;
python3 check.py;chmod +x keep.sh;./keep.sh;python3 allsubjectMaker.py;
chmod +x allstudentMaker.sh;./allstudentMaker.sh;rm absent1.txt;rm absent
rm keep.sh;rm allstudentMaker.sh;rm present.txt;
```

This is the linux commands process.sh contains

WHAT CODES THE PROCESS.SH FILE RUNS

- Passwordchecker.py
- PresentToabsent.py
- AbsentTreadableManner.py
- Check.py
- AllsubjectMaker.py

Passwordchecker.py



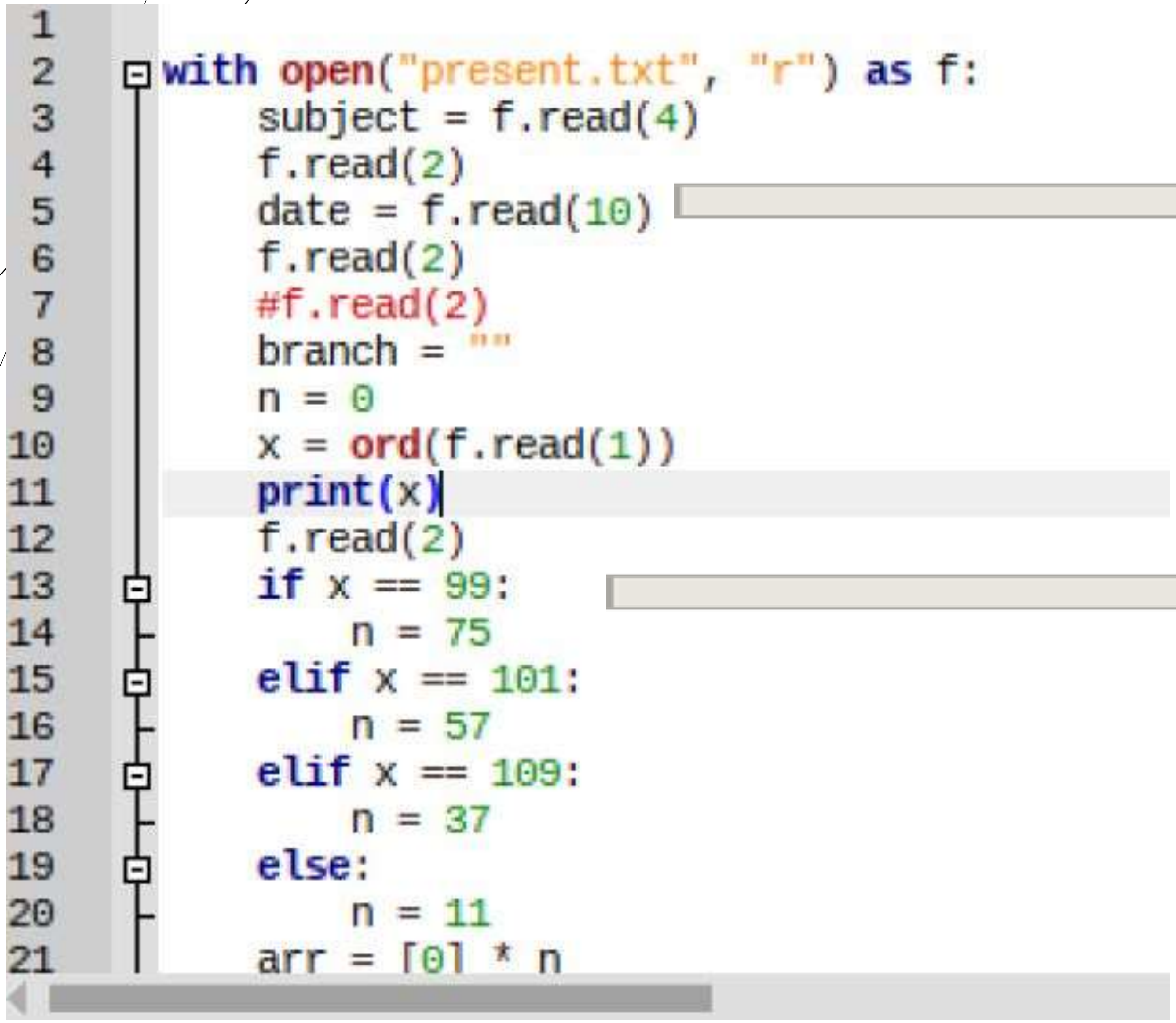
```
while True:
    password = input("Enter the password: ")

    if password == "1234":
        print("Correct password!")
        break

    print("Incorrect password. Try again.")

#print("Program stopped.")
```

- Passwordchecker.py it contains infinite while loop it will check the password for infinite times until user presses the correct password

Password

It will stop code if it is entered correct code

PresentToabsent.py



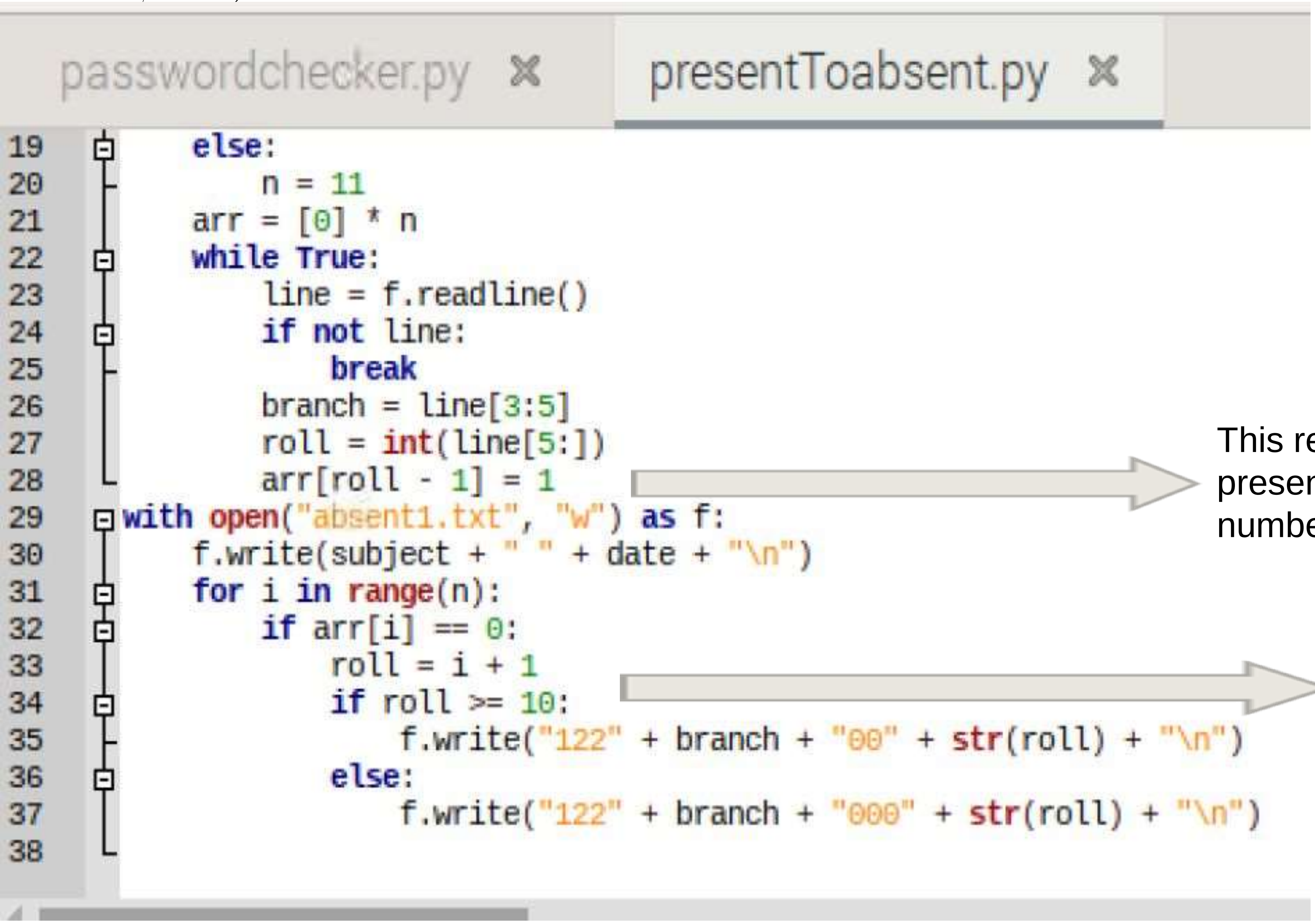
```
with open("present.txt", "r") as f:
    subject = f.read(4)
    f.read(2)
    date = f.read(10)
    f.read(2)
    #f.read(2)
    branch = ""
    n = 0
    x = ord(f.read(1))
    print(x)
    f.read(2)
    if x == 99:
        n = 75
    elif x == 101:
        n = 57
    elif x == 109:
        n = 37
    else:
        n = 11
    arr = [0] * n
```

It reads the subject, date and brand id

- The function of the code is that the it converts the present roll numbers to the absent roll numbers

It decides the array length based on the branch code. The array contains person Is absent or present

PresentToabsent.py

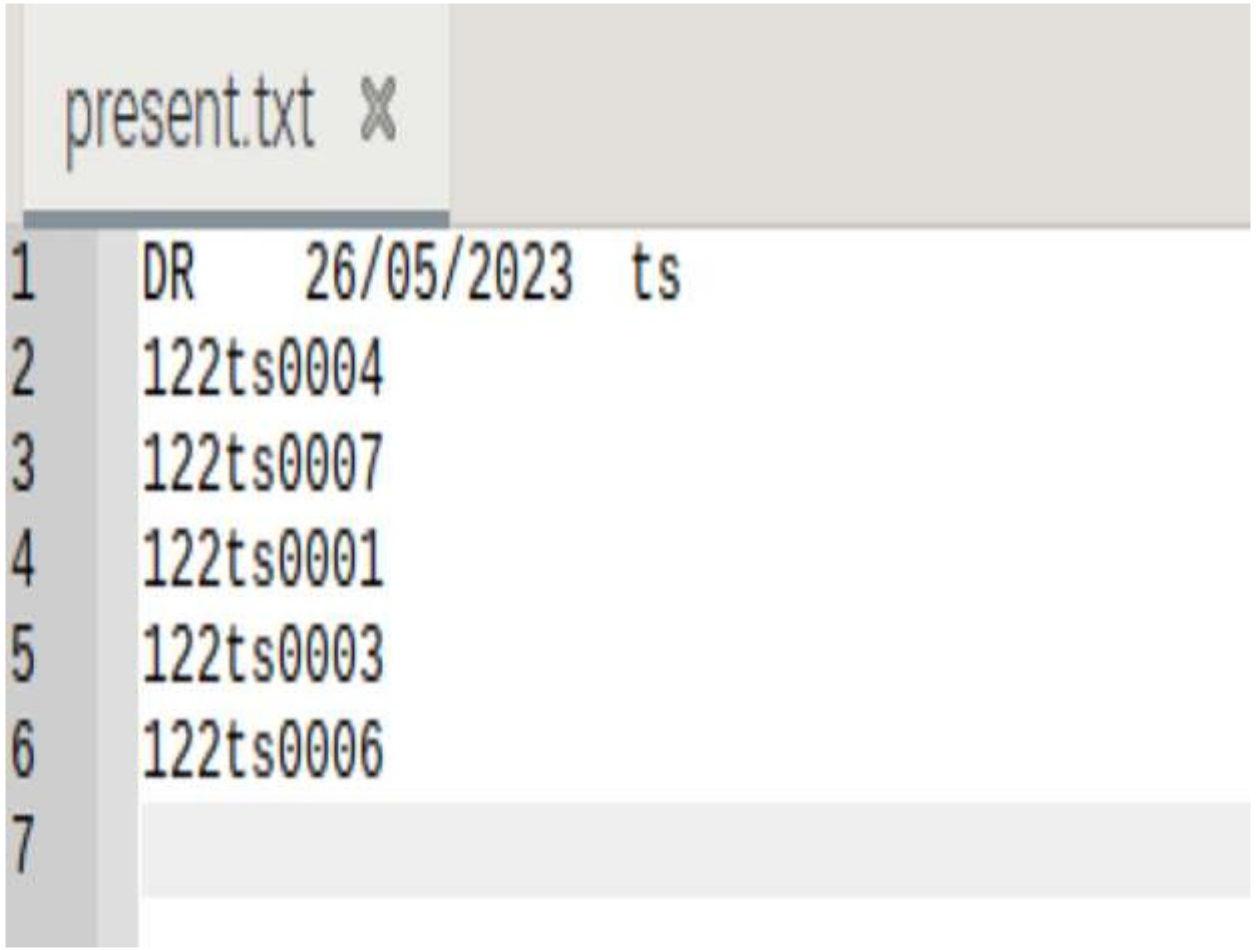


```
    else:
        n = 11
    arr = [0] * n
    while True:
        line = f.readline()
        if not line:
            break
        branch = line[3:5]
        roll = int(line[5:])
        arr[roll - 1] = 1
with open("absent1.txt", "w") as f:
    f.write(subject + " " + date + "\n")
    for i in range(n):
        if arr[i] == 0:
            roll = i + 1
            if roll >= 10:
                f.write("122" + branch + "00" + str(roll) + "\n")
            else:
                f.write("122" + branch + "000" + str(roll) + "\n")
```

This reads the roll no from present.txt keep the 1 at roll number in the array

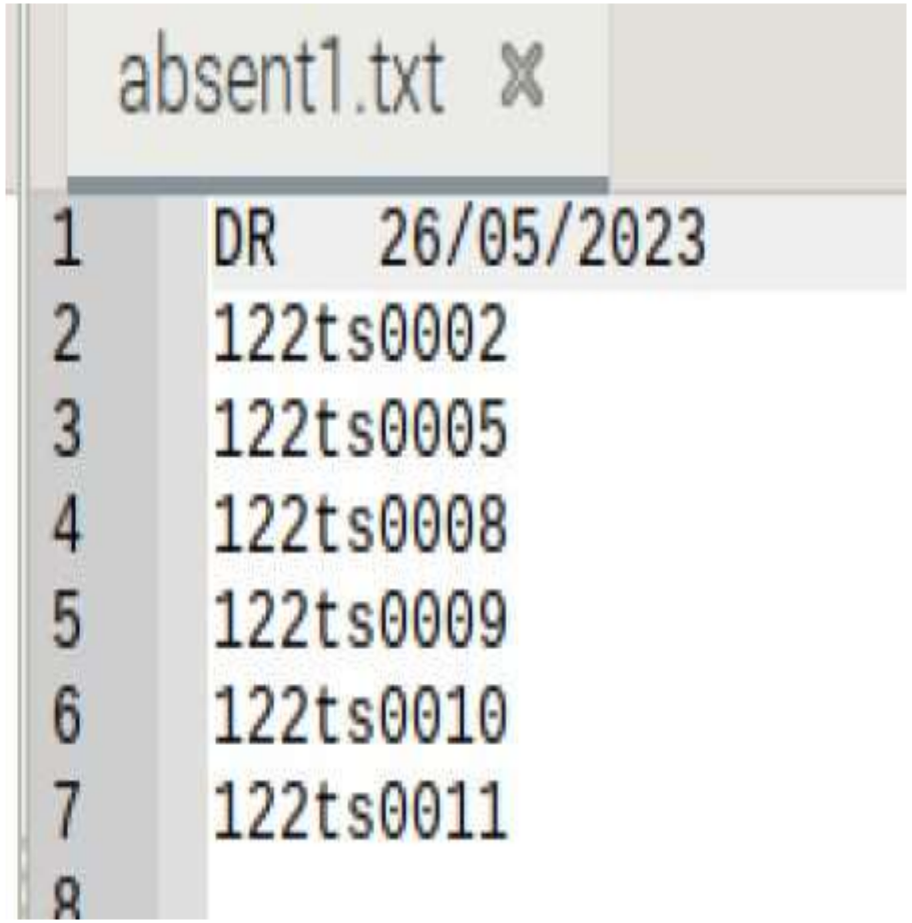
This will create absent1.txt in that only absent members roll numbers will be present

PresentToabsent.py



present.txt

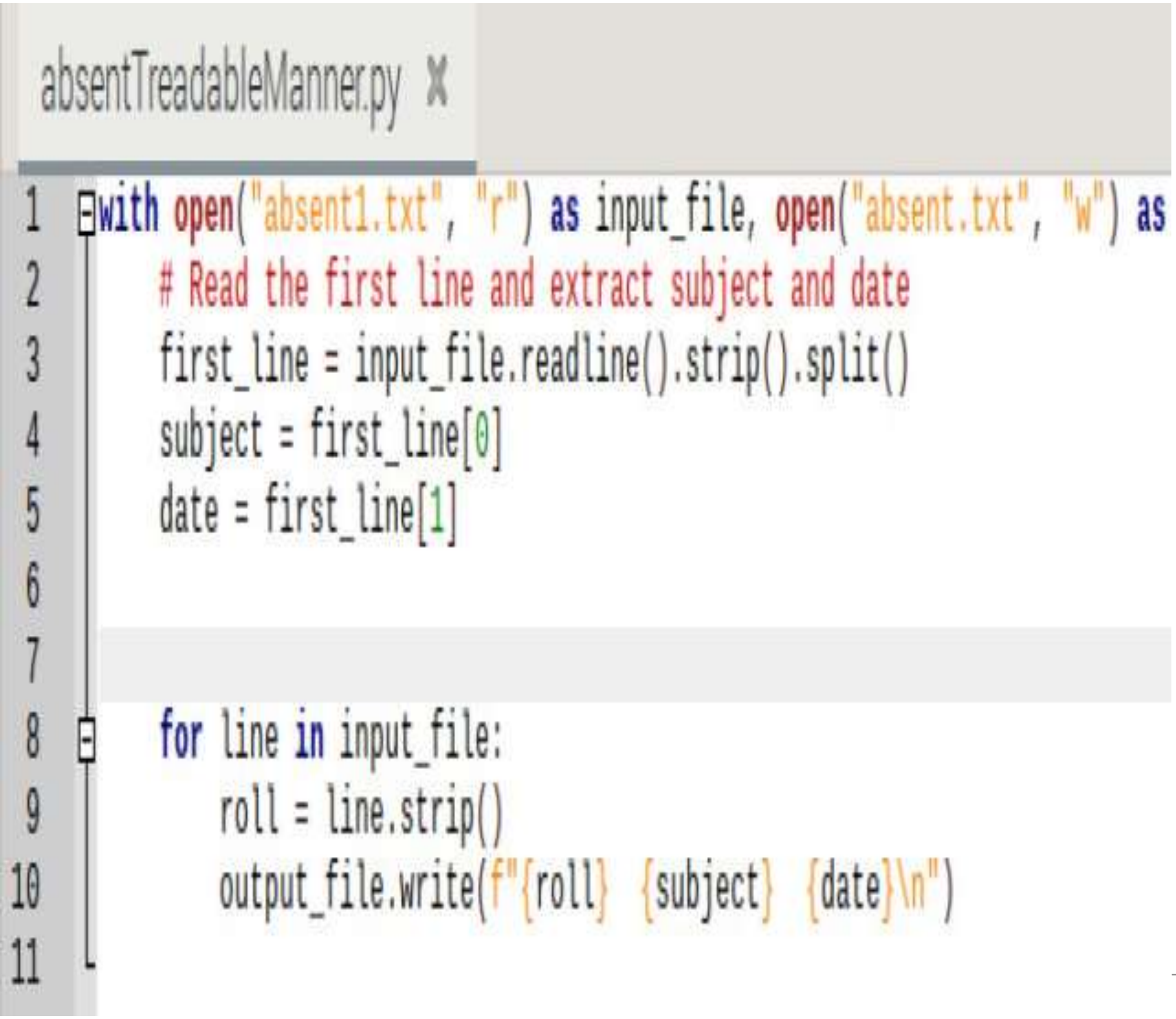
```
DR    26/05/2023  ts
122ts0004
122ts0007
122ts0001
122ts0003
122ts0006
```



absent1.txt

```
DR    26/05/2023
122ts0002
122ts0005
122ts0008
122ts0009
122ts0010
122ts0011
```

AbsentTreadableManner.py



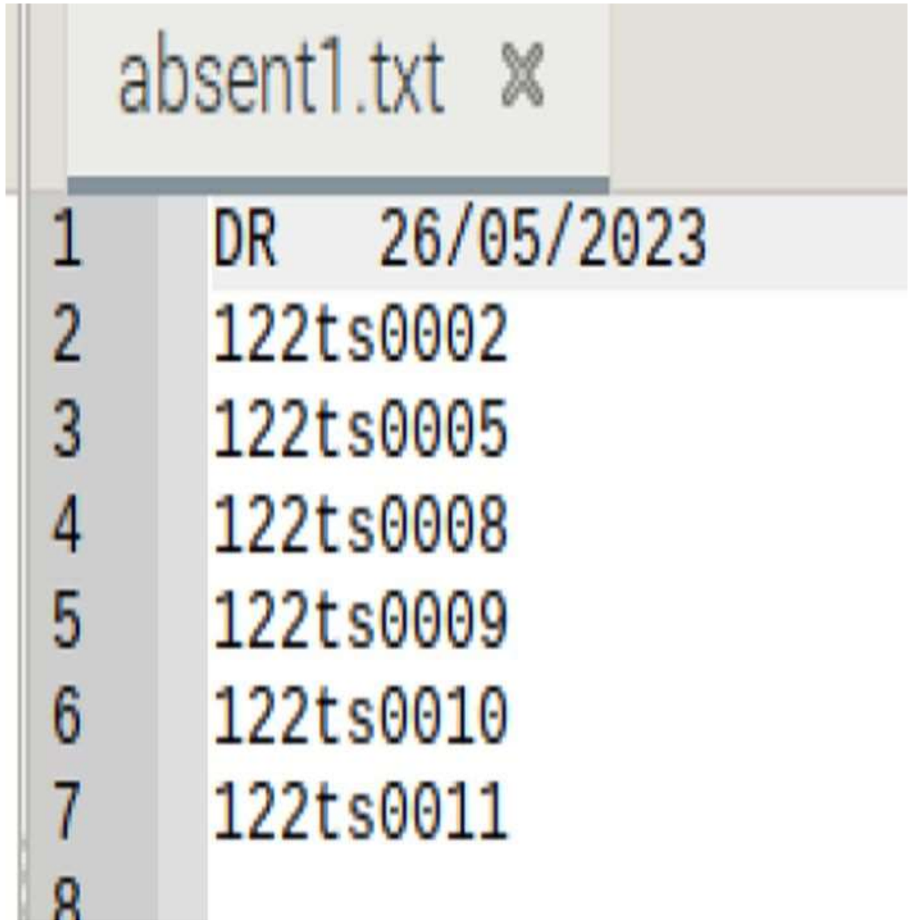
absentTreadableManner.py

```
with open("absent1.txt", "r") as input_file, open("absent.txt", "w") as
    # Read the first line and extract subject and date
    first_line = input_file.readline().strip().split()
    subject = first_line[0]
    date = first_line[1]


    for line in input_file:
        roll = line.strip()
        output_file.write(f"{roll} {subject} {date}\n")
```

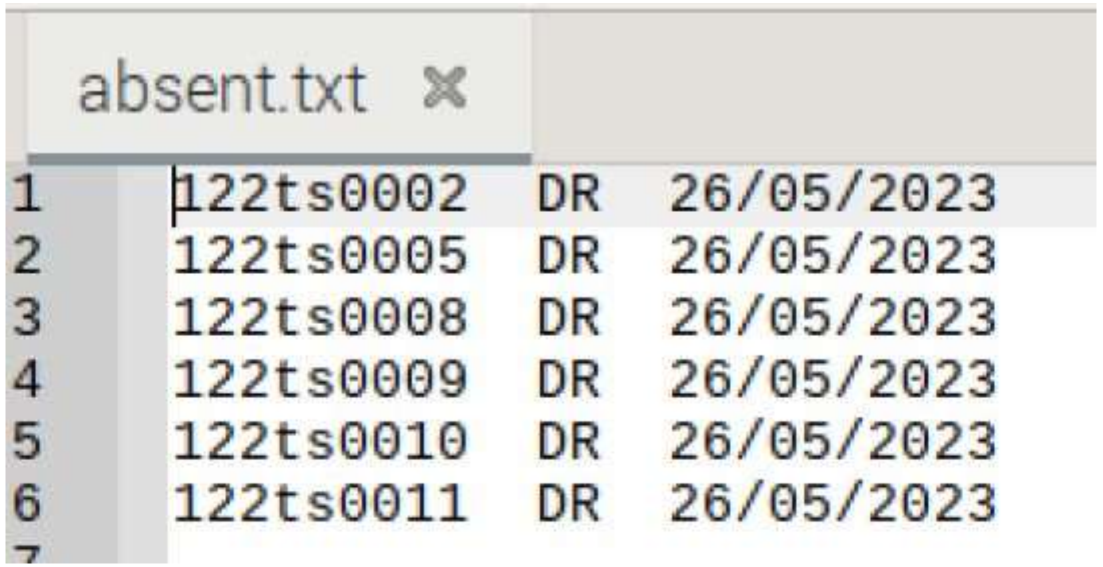
- The function of the code is convert the absent roll numbers into straight and meaning full manner because next code input will be taking the multiple subjects at a time so not confusion between the subjects

- It will take input absent1.txt
- The output will be absent.txt

AbsentTreadableManner.py

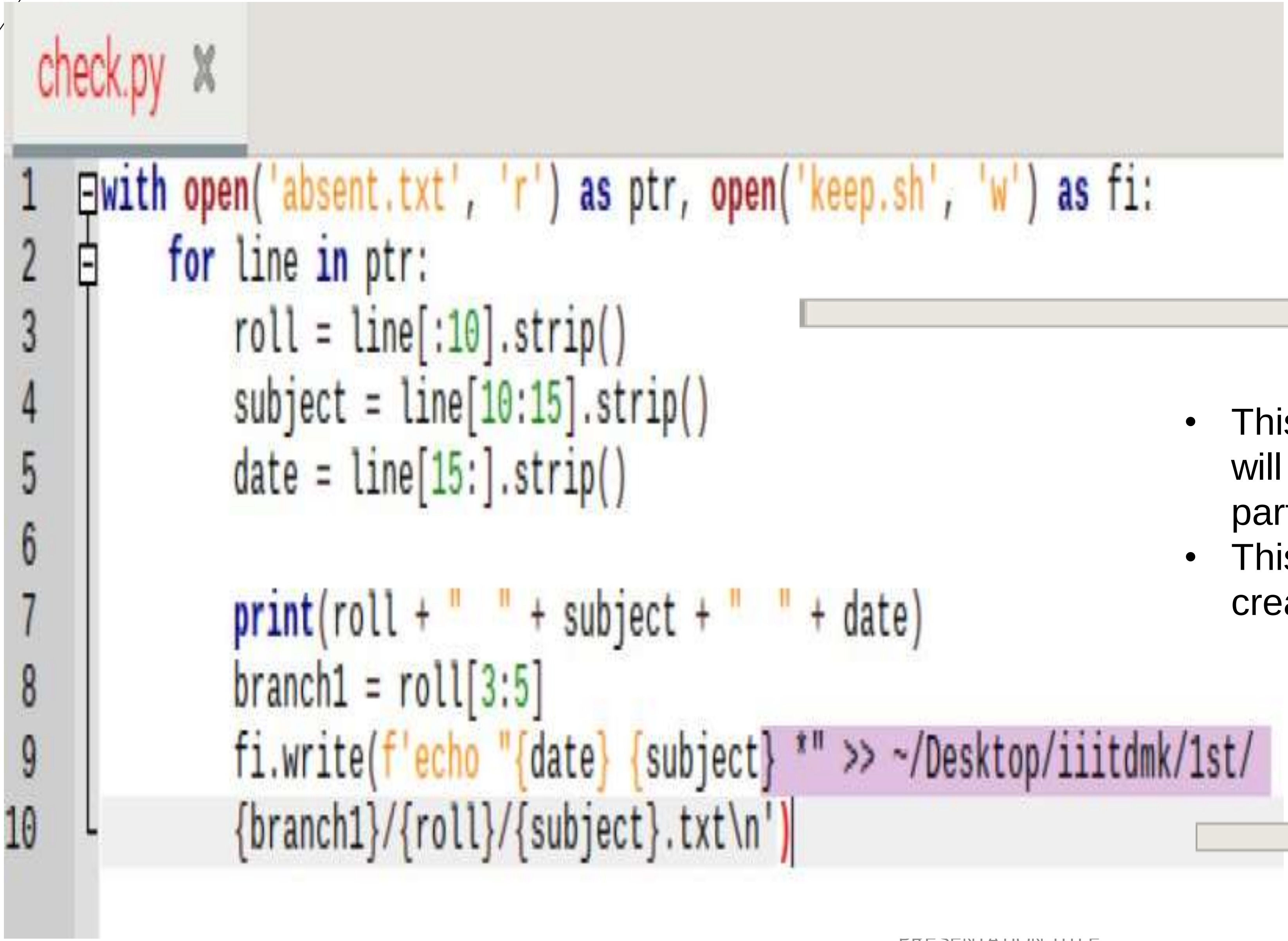

absent1.txt

```
DR   26/05/2023
122ts0002
122ts0005
122ts0008
122ts0009
122ts0010
122ts0011
```


absent.txt

```
122ts0002   DR   26/05/2023
122ts0005   DR   26/05/2023
122ts0008   DR   26/05/2023
122ts0009   DR   26/05/2023
122ts0010   DR   26/05/2023
122ts0011   DR   26/05/2023
```

check.py

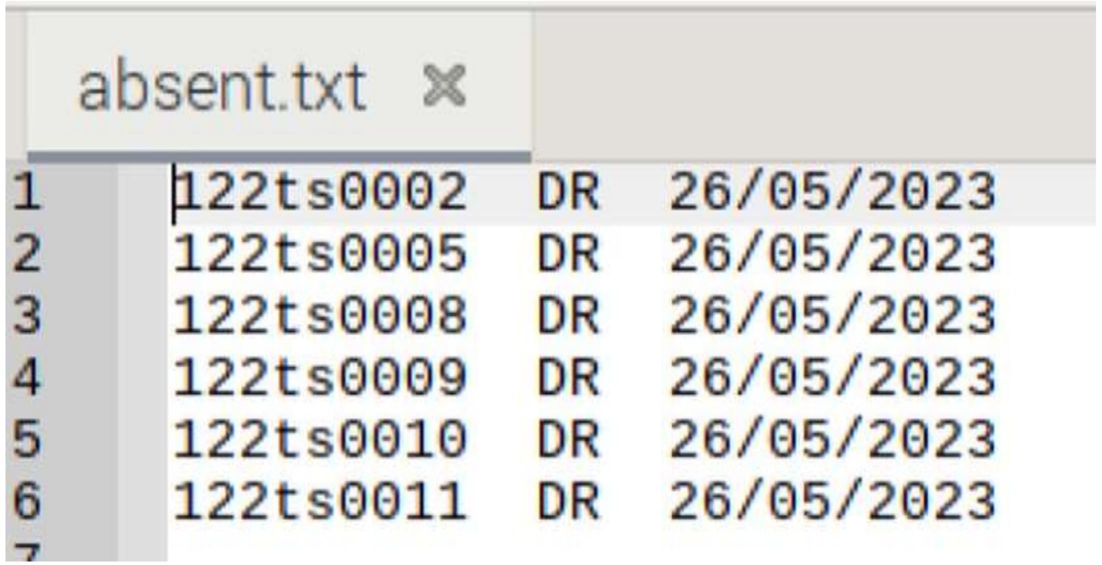


```python
with open('absent.txt', 'r') as ptr, open('keep.sh', 'w') as fi:
    for line in ptr:
        roll = line[:10].strip()
        subject = line[10:15].strip()
        date = line[15:].strip()

        print(roll + "  " + subject + "  " + date)
        branch1 = roll[3:5]
        fi.write(f'echo "{date} {subject} *" >> ~/Desktop/iiitdmk/1st/
        {branch1}/{roll}/{subject}.txt\n')
```

This will read the roll numbers ,dates and subject

- This creates the linux command that will add the absent dates to the particular roll no
- This will take the input absent.txt and creates the keep.sh

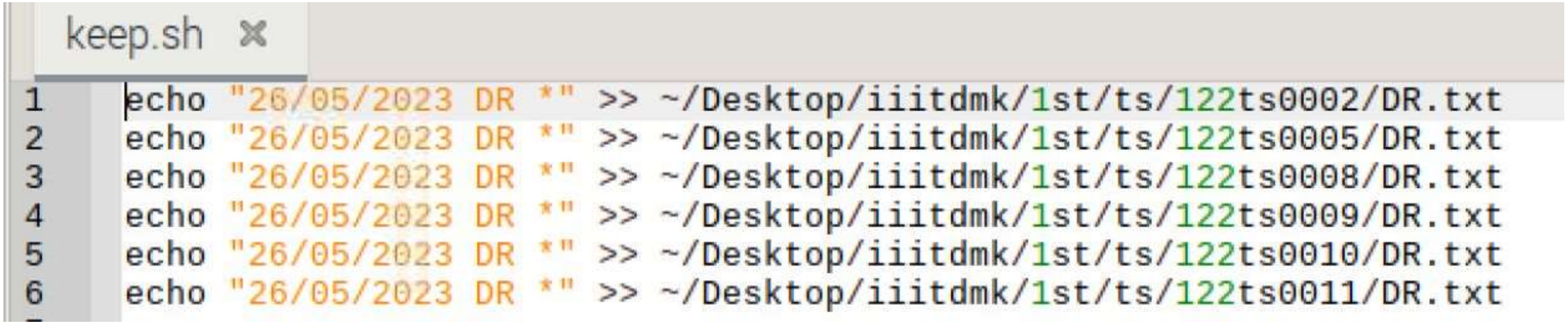
This is the linux command



absent.txt

```
122ts0002   DR   26/05/2023
122ts0005   DR   26/05/2023
122ts0008   DR   26/05/2023
122ts0009   DR   26/05/2023
122ts0010   DR   26/05/2023
122ts0011   DR   26/05/2023
```

check.py

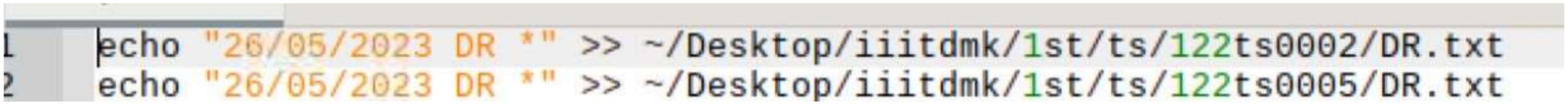


keep.sh

```
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0002/DR.txt
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0005/DR.txt
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0008/DR.txt
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0009/DR.txt
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0010/DR.txt
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0011/DR.txt
```

check.py

- After creating the keep.sh, to activate the keep.sh we should type the chmod +x keep.sh in the terminal .After activating the file we can run the file by clicking it or run the command ./keep.sh



```
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0002/DR.txt
echo "26/05/2023 DR *" >> ~/Desktop/iiitdmk/1st/ts/122ts0005/DR.txt
```

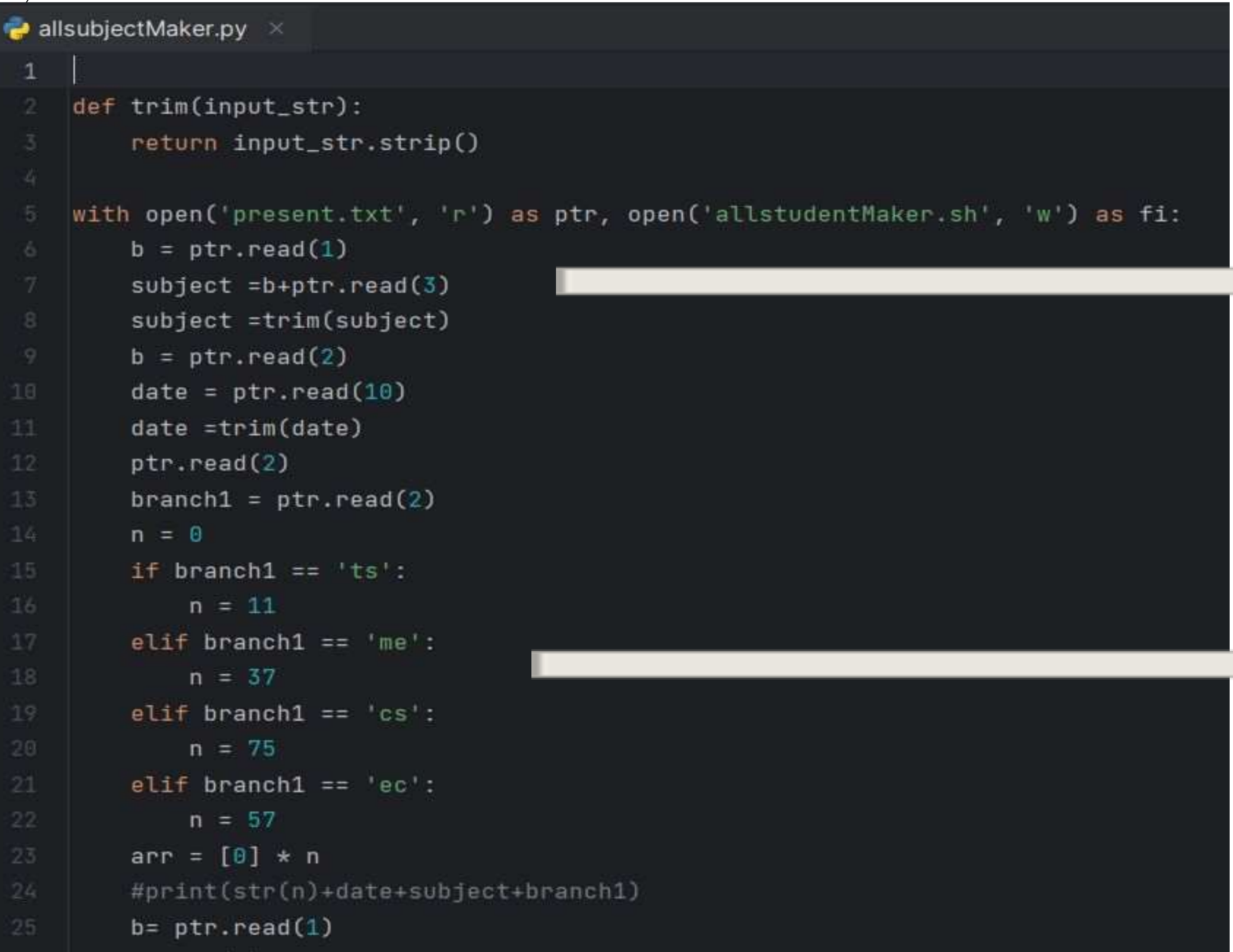
Linux command

keyword

Text that what we want to keep

Location where we want to keep the text

AllsubjectMaker.py



allsubjectMaker.py

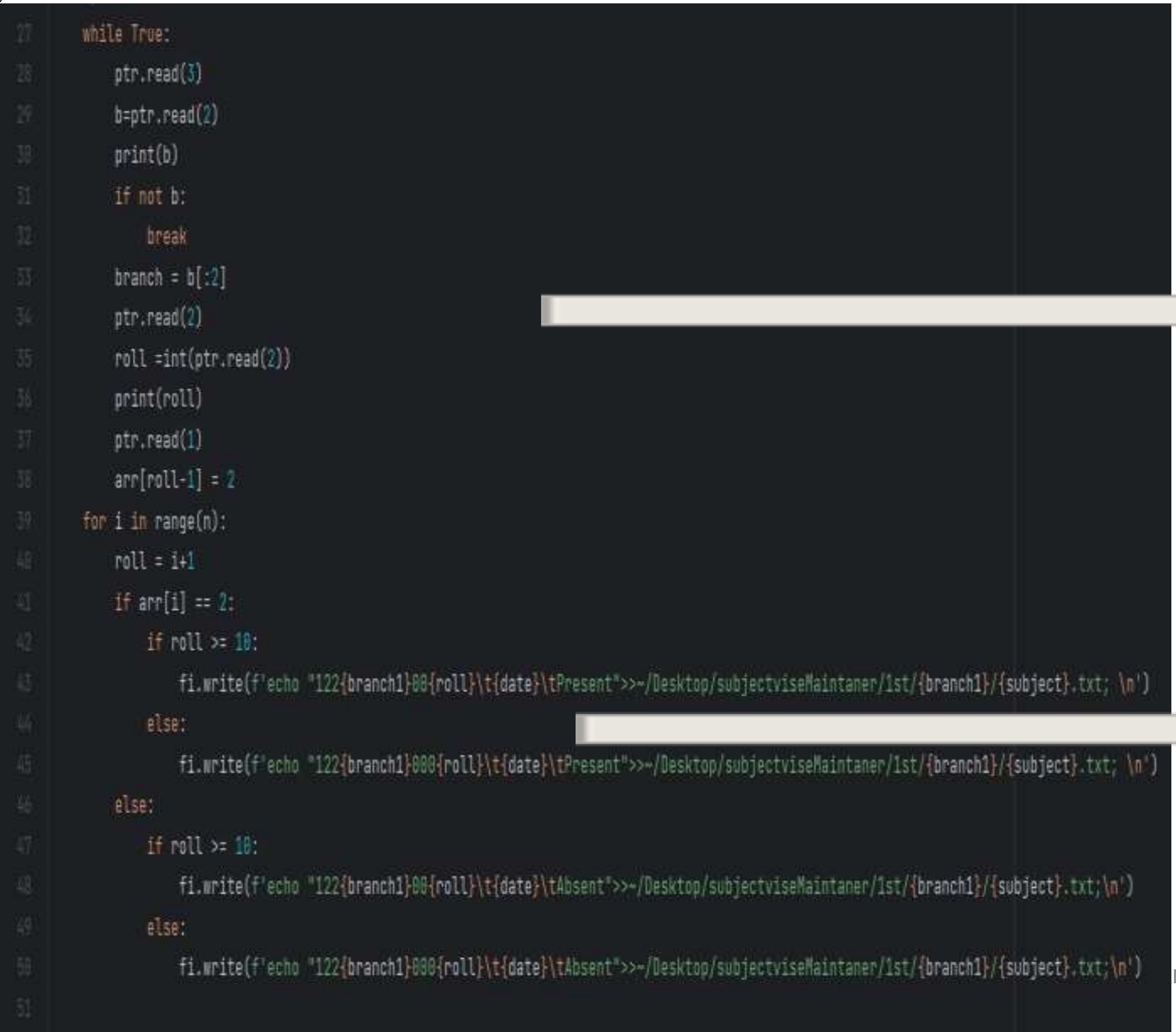
```python

def trim(input_str):
    return input_str.strip()

with open('present.txt', 'r') as ptr, open('allstudentMaker.sh', 'w') as fi:
    b = ptr.read(1)
    subject =b+ptr.read(3)
    subject =trim(subject)
    b = ptr.read(2)
    date = ptr.read(10)
    date =trim(date)
    ptr.read(2)
    branch1 = ptr.read(2)
    n = 0
    if branch1 == 'ts':
        n = 11
    elif branch1 == 'me':
        n = 37
    elif branch1 == 'cs':
        n = 75
    elif branch1 == 'ec':
        n = 57
    arr = [0] * n
    #print(str(n)+date+subject+branch1)
    b= ptr.read(1)
```

- This reads the subject name , date, branch code

- This will take input present.txt
- Gives the output allsubjectmaker.sh

- This will create the array length based on the branch id

AllsubjectMaker.py

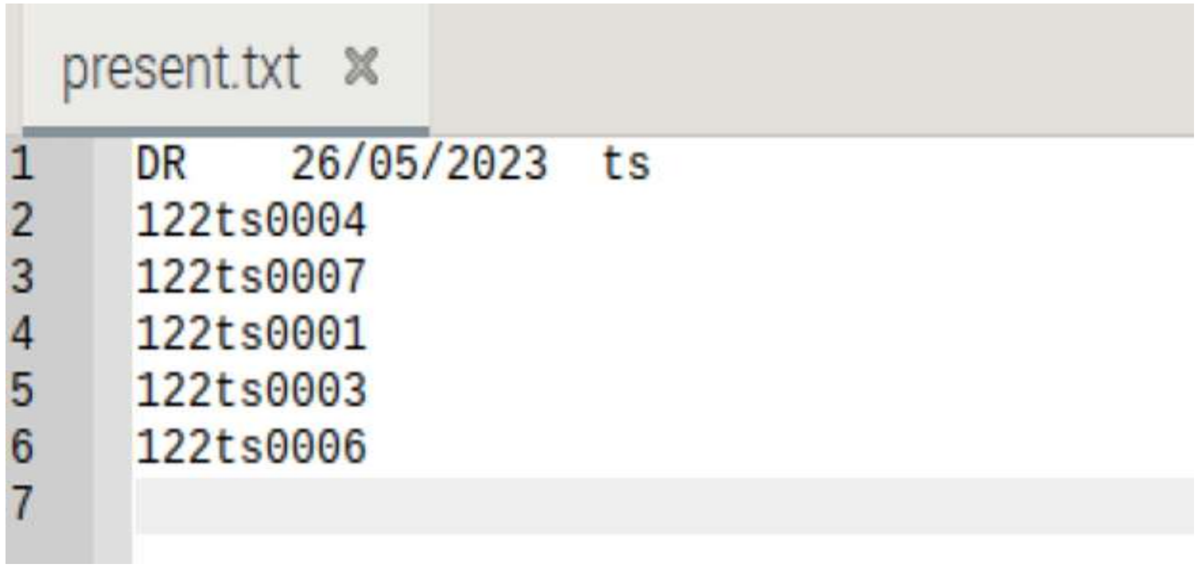


```
while True:
    ptr.read(3)
    b=ptr.read(2)
    print(b)
    if not b:
        break
    branch = b[:2]
    ptr.read(2)
    roll =int(ptr.read(2))
    print(roll)
    ptr.read(1)
    arr[roll-1] = 2
for i in range(n):
    roll = i+1
    if arr[i] == 2:
        if roll >= 10:
            fi.write(f'echo "122{branch1}00{roll}\t{date}\tPresent">>~/Desktop/subjectviseMaintaner/1st/{branch1}/{subject}.txt; \n')
        else:
            fi.write(f'echo "122{branch1}000{roll}\t{date}\tPresent">>~/Desktop/subjectviseMaintaner/1st/{branch1}/{subject}.txt; \n')
    else:
        if roll >= 10:
            fi.write(f'echo "122{branch1}00{roll}\t{date}\tAbsent">>~/Desktop/subjectviseMaintaner/1st/{branch1}/{subject}.txt;\n')
        else:
            fi.write(f'echo "122{branch1}000{roll}\t{date}\tAbsent">>~/Desktop/subjectviseMaintaner/1st/{branch1}/{subject}.txt;\n')
```

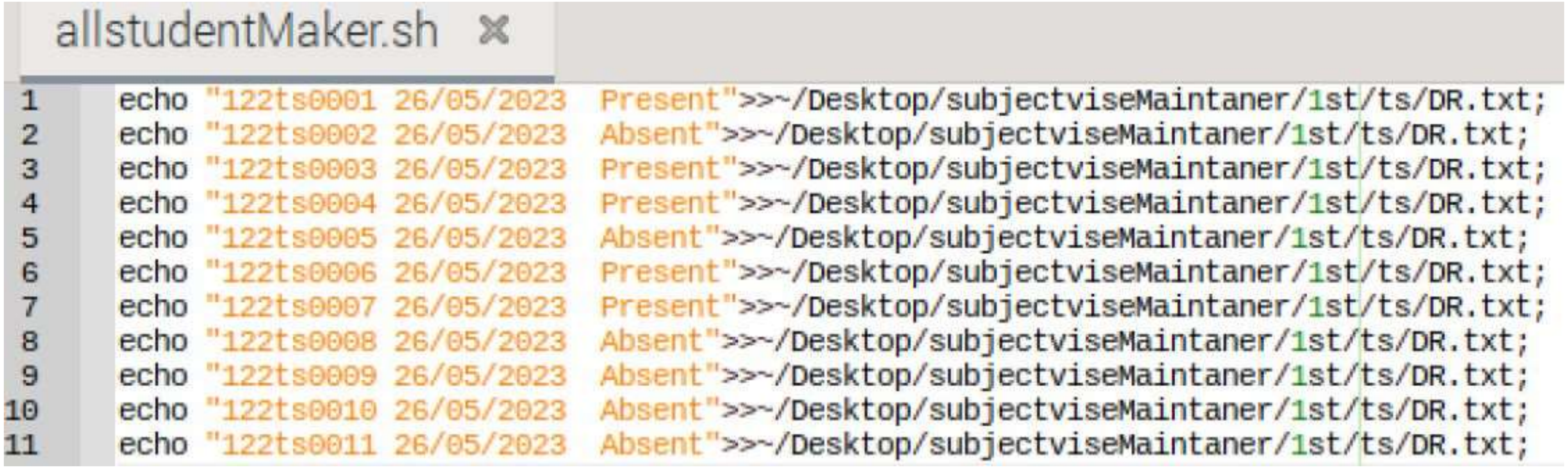
Reading the data of the present students

This is linux command

AllsubjectMaker.py

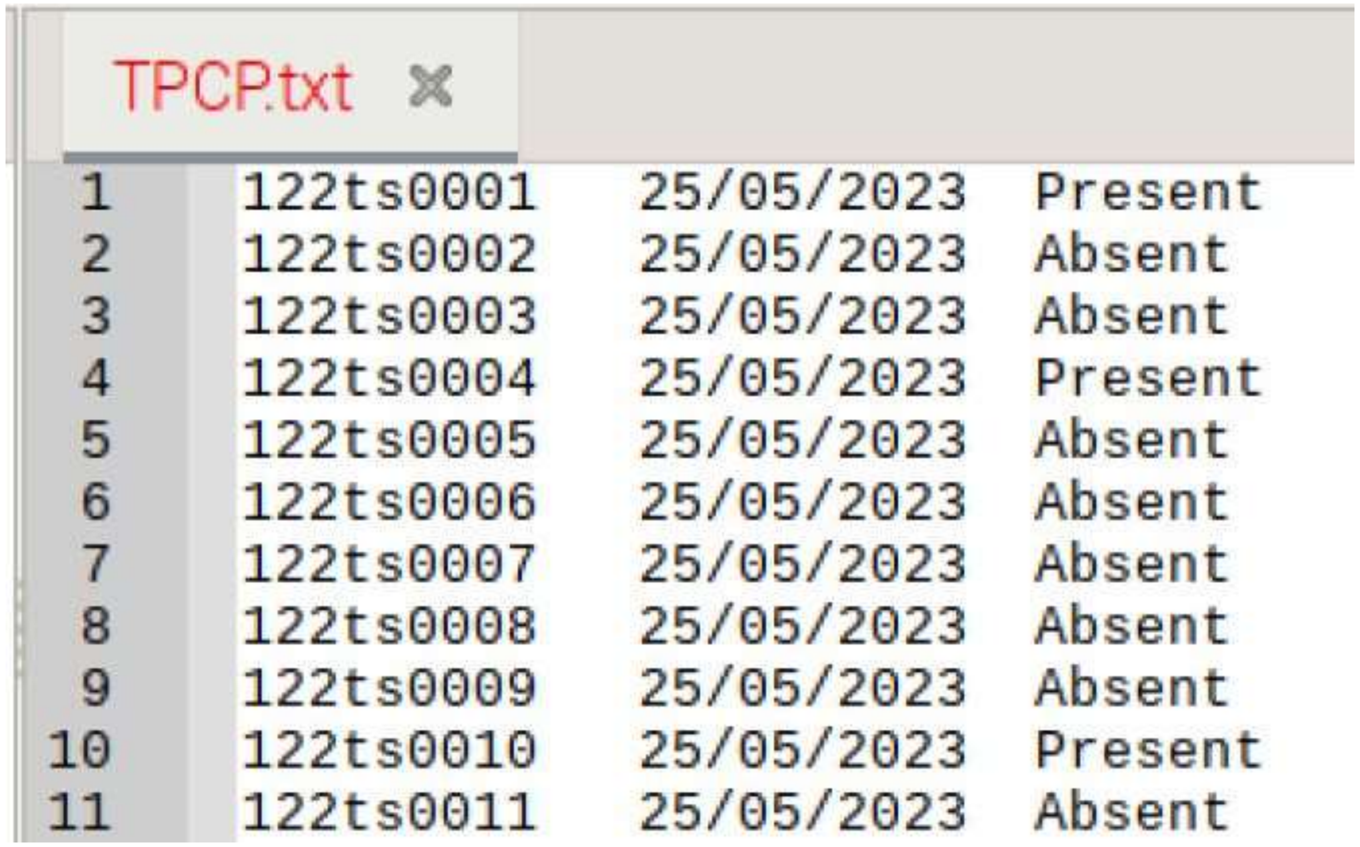


```
DR    26/05/2023  ts
122ts0004
122ts0007
122ts0001
122ts0003
122ts0006

```



```
echo "122ts0001 26/05/2023  Present">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0002 26/05/2023  Absent">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0003 26/05/2023  Present">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0004 26/05/2023  Present">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0005 26/05/2023  Absent">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0006 26/05/2023  Present">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0007 26/05/2023  Present">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0008 26/05/2023  Absent">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0009 26/05/2023  Absent">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0010 26/05/2023  Absent">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
echo "122ts0011 26/05/2023  Absent">>~/Desktop/subjectviseMaintaner/1st/ts/DR.txt;
```

AllsubjectMaker.py


TPCP.txt

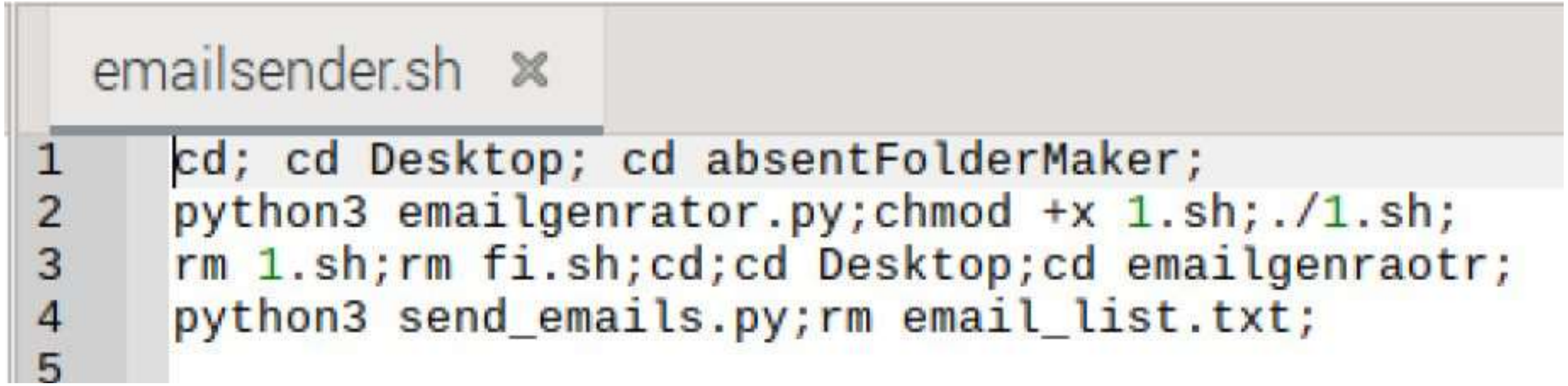
```
122ts0001   25/05/2023  Present
122ts0002   25/05/2023  Absent
122ts0003   25/05/2023  Absent
122ts0004   25/05/2023  Present
122ts0005   25/05/2023  Absent
122ts0006   25/05/2023  Absent
122ts0007   25/05/2023  Absent
122ts0008   25/05/2023  Absent
122ts0009   25/05/2023  Absent
122ts0010   25/05/2023  Present
122ts0011   25/05/2023  Absent
```

HOW THE EMAILGENERATOR.SH WORKS ?

- First , it travels across the folder where there the all codes present

- emailgenerator.sh file will run a 5 python code
- The emailgenerator.sh file process the data
- It runs two python codes they are

1. emailgenrator.py
2. send_emails.py


emailsender.sh

```
cd; cd Desktop; cd absentFolderMaker;
python3 emailgenrator.py;chmod +x 1.sh;./1.sh;
rm 1.sh;rm fi.sh;cd;cd Desktop;cd emailgenraotr;
python3 send_emails.py;rm email_list.txt;

```

This is the linux commands emailgenerator.sh contains

EMAILGENERATOR.PY

```
subject=input("ENTER THE SUBJECT NAME: ")
branch=input("ENTER THE BRACH CODE : ")
if branch=='cs':
   n=75
if branch=='ts':
  n=12
if branch=='ec':
 n=56
if branch=='me':
 n=36
print(n)
with open('1.sh', 'w') as ptr:
  ptr.write("cp ~/Desktop/absentFolderMaker/fi.sh ~/Desktop/subjectviseMaintaner/1st/"+branch+";cd;cd Desktop;cd subjectviseMaintaner;cd 1st;cd "+branch+";chmod +x fi.sh;./fi.sh;rm fi.sh;")
with open('fi.sh', 'w') as p:
  i=1
  while (i<=n):
   if i<10:
     p.write("#!/bin/bash \njohn_percent=$(awk -F \"\t\" '{if ($1 == \"122"+branch+"000"+str(i)+"\" && $3 == \"Present\") present++; if($1 == \"122"+branch+"000"+str(i)+"\")count++} END
{print (present/count)*100}' "+subject+".txt)\n\nif (( $(echo \"$john_percent < 75\" | bc -l) )); then\necho \"122"+branch+"000"+str(i)+"@iiitk.ac.in\">> ~/Desktop/emailgenraotr/email.txt
\nfi;\n")
   else:
     p.write("#!/bin/bash \njohn_percent=$(awk -F \"\t\" '{if ($1 == \"122"+branch+"00"+str(i)+"\" && $3 == \"Present\") present++; if($1 == \"122"+branch+"00"+str(i)+"\")count++} END {print
(present/count)*100}' "+subject+".txt)\n\nif (( $(echo \"$john_percent < 75\" | bc -l) )); then\necho \"122"+branch+"00"+str(i)+"@iiitk.ac.in\">> ~/Desktop/emailgenraotr/email.txt\nfi;\n")
   i=i+1
```

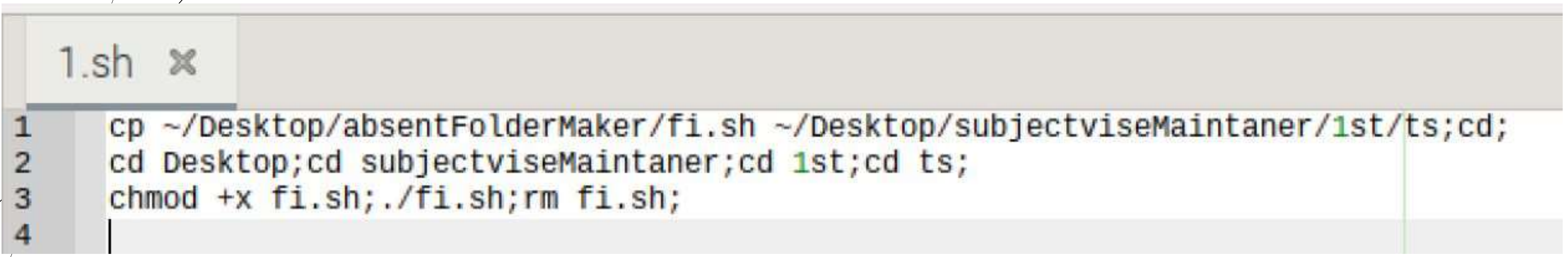
Taking input from the user subject name and branch code

- This creates the file like 1.sh and fi.sh

1.sh

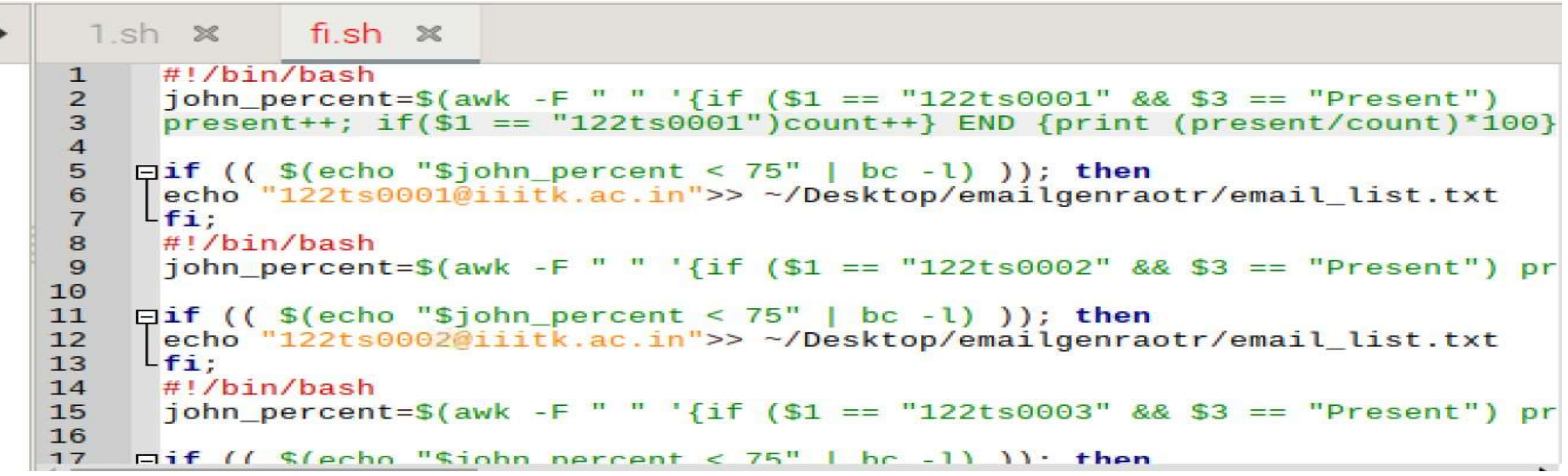
Unix code

EMAILGENERATOR.PY



1.sh

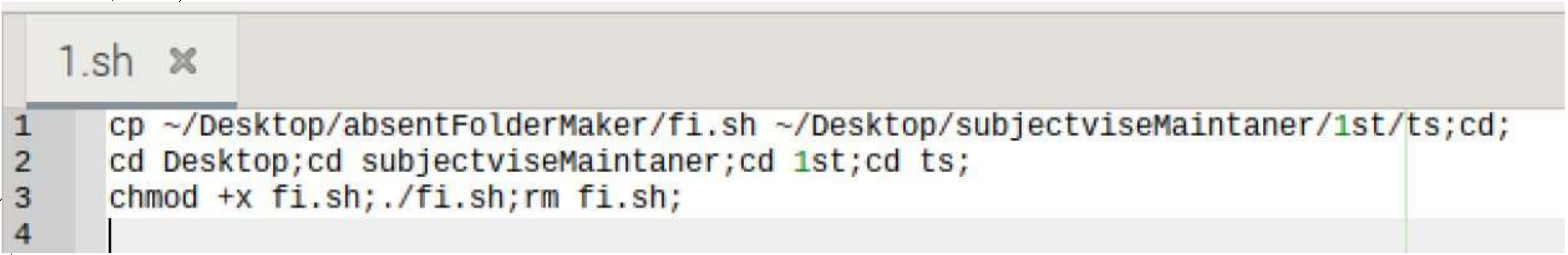
```
cp ~/Desktop/absentFolderMaker/fi.sh ~/Desktop/subjectviseMaintaner/1st/ts;cd;
cd Desktop;cd subjectviseMaintaner;cd 1st;cd ts;
chmod +x fi.sh;./fi.sh;rm fi.sh;
```



1.sh fi.sh

```
#!/bin/bash
john_percent=$(awk -F " " '{if ($1 == "122ts0001" && $3 == "Present")
present++; if($1 == "122ts0001")count++} END {print (present/count)*100}

if (( $(echo "$john_percent < 75" | bc -l) )); then
echo "122ts0001@iiitk.ac.in">> ~/Desktop/emailgenraotr/email_list.txt
fi;
#!/bin/bash
john_percent=$(awk -F " " '{if ($1 == "122ts0002" && $3 == "Present") pr

if (( $(echo "$john_percent < 75" | bc -l) )); then
echo "122ts0002@iiitk.ac.in">> ~/Desktop/emailgenraotr/email_list.txt
fi;
#!/bin/bash
john_percent=$(awk -F " " '{if ($1 == "122ts0003" && $3 == "Present") pr

if (( $(echo "$john_percent < 75" | bc -l) )); then
```

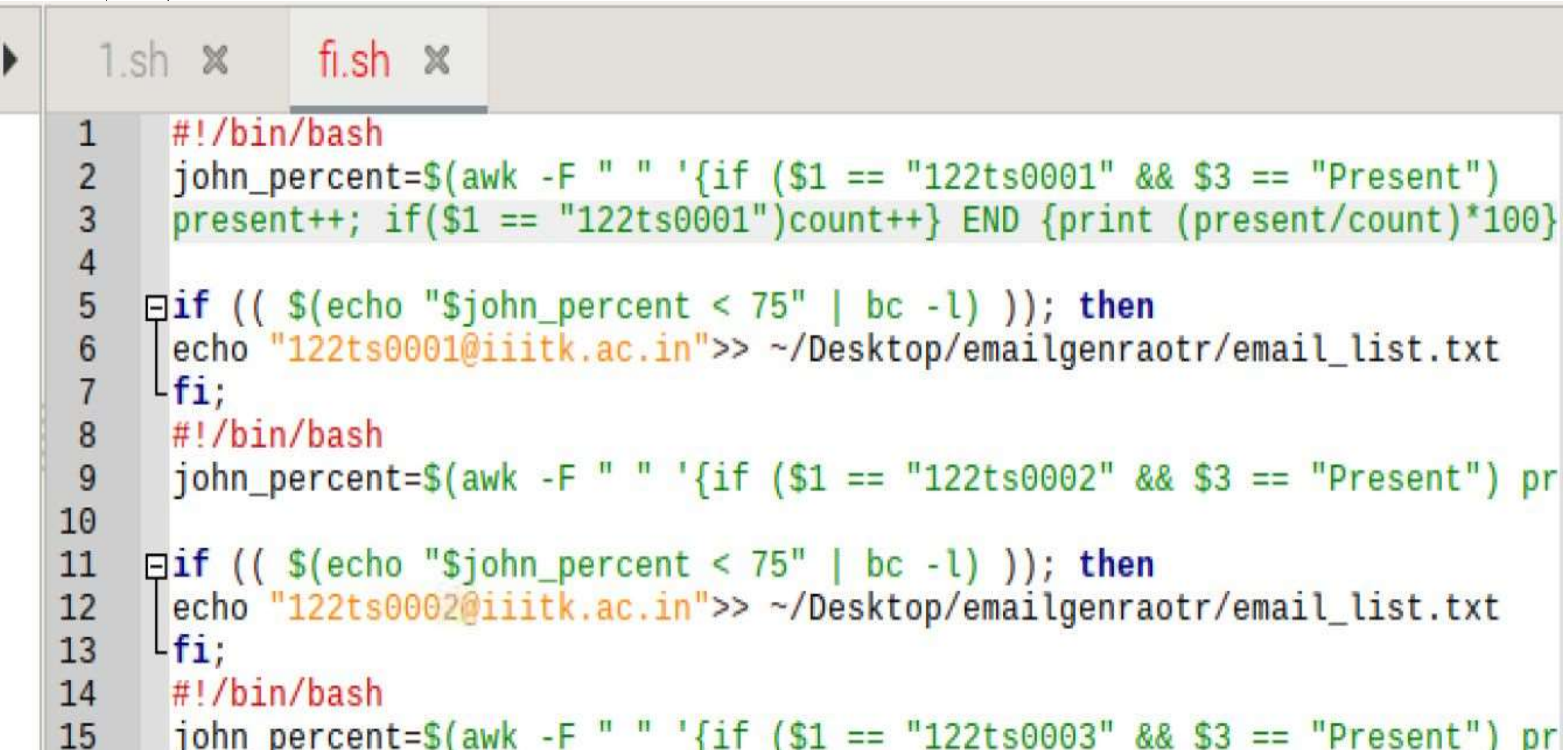
EMAILGENERATOR.PY


1.sh

```
cp ~/Desktop/absentFolderMaker/fi.sh ~/Desktop/subjectviseMaintaner/1st/ts;cd;
cd Desktop;cd subjectviseMaintaner;cd 1st;cd ts;
chmod +x fi.sh;./fi.sh;rm fi.sh;

```

- This 1.sh will help the fi.sh file to run at the particular at desired location and activate the fi.sh
- Automatically delete the fi.sh after the run was completed

EMAILGENERATOR.PY



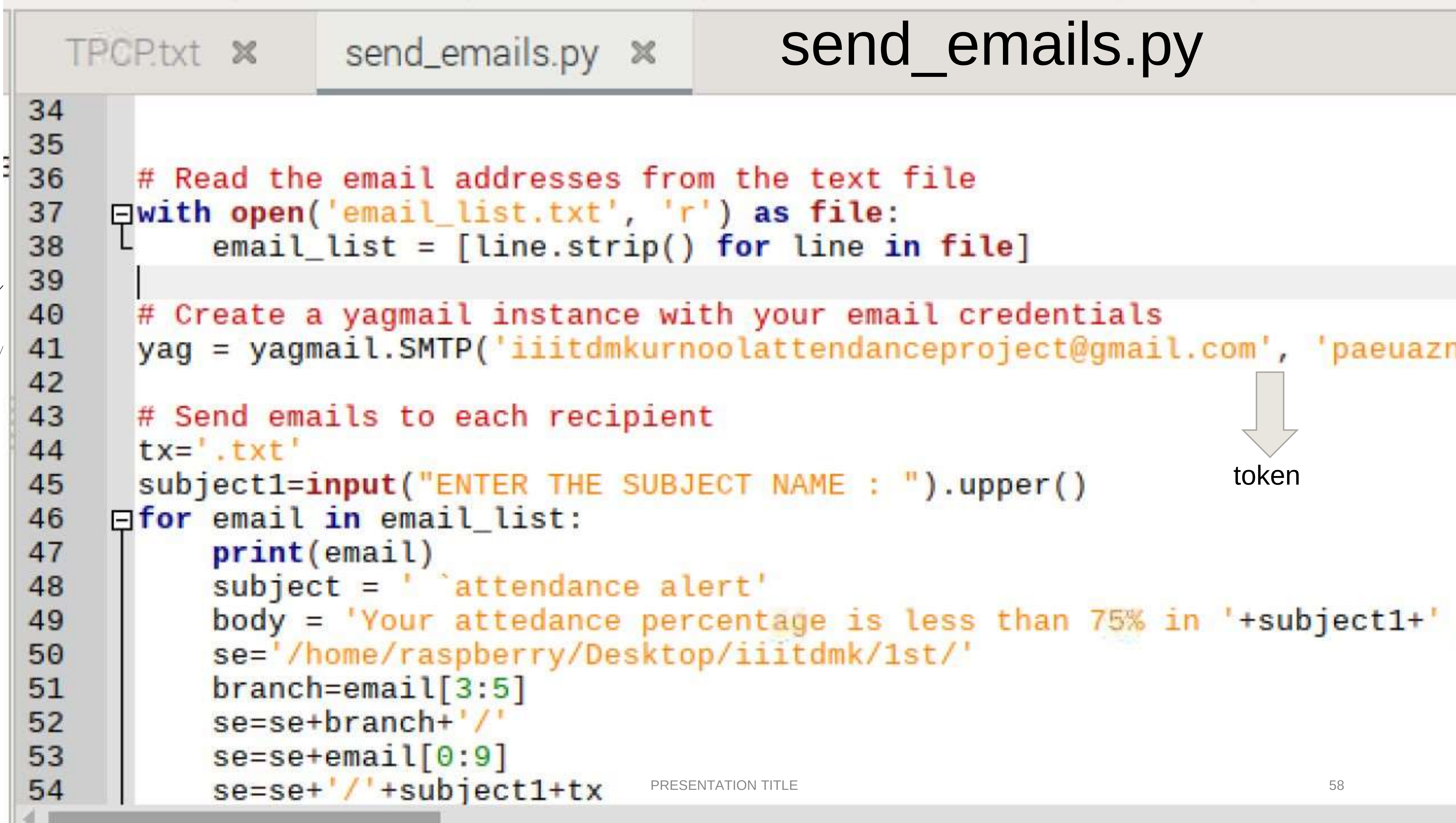
```
#!/bin/bash
john_percent=$(awk -F " " '{if ($1 == "122ts0001" && $3 == "Present")
present++; if($1 == "122ts0001")count++} END {print (present/count)*100}

if (( $(echo "$john_percent < 75" | bc -l) )); then
echo "122ts0001@iiitk.ac.in">> ~/Desktop/emailgenraotr/email_list.txt
fi;
#!/bin/bash
john_percent=$(awk -F " " '{if ($1 == "122ts0002" && $3 == "Present") pr

if (( $(echo "$john_percent < 75" | bc -l) )); then
echo "122ts0002@iiitk.ac.in">> ~/Desktop/emailgenraotr/email_list.txt
fi;
#!/bin/bash
john_percent=$(awk -F " " '{if ($1 == "122ts0003" && $3 == "Present") pr
```

EMAILGENERATOR.PY

```
#!/bin/bash
jpercent=$(awk -F "              " '{if ($1 == "122ts0002" && $3 == "Present") present++;
if($1 == "122ts0002")count++} END {print (present/count)*100}' DR.txt)

if (( $(echo "$jpercent < 75" | bc -l) )); then
echo "122ts0002@iiitk.ac.in">> ~/Desktop/emailgenraotr/email_list.txt
fi;
```

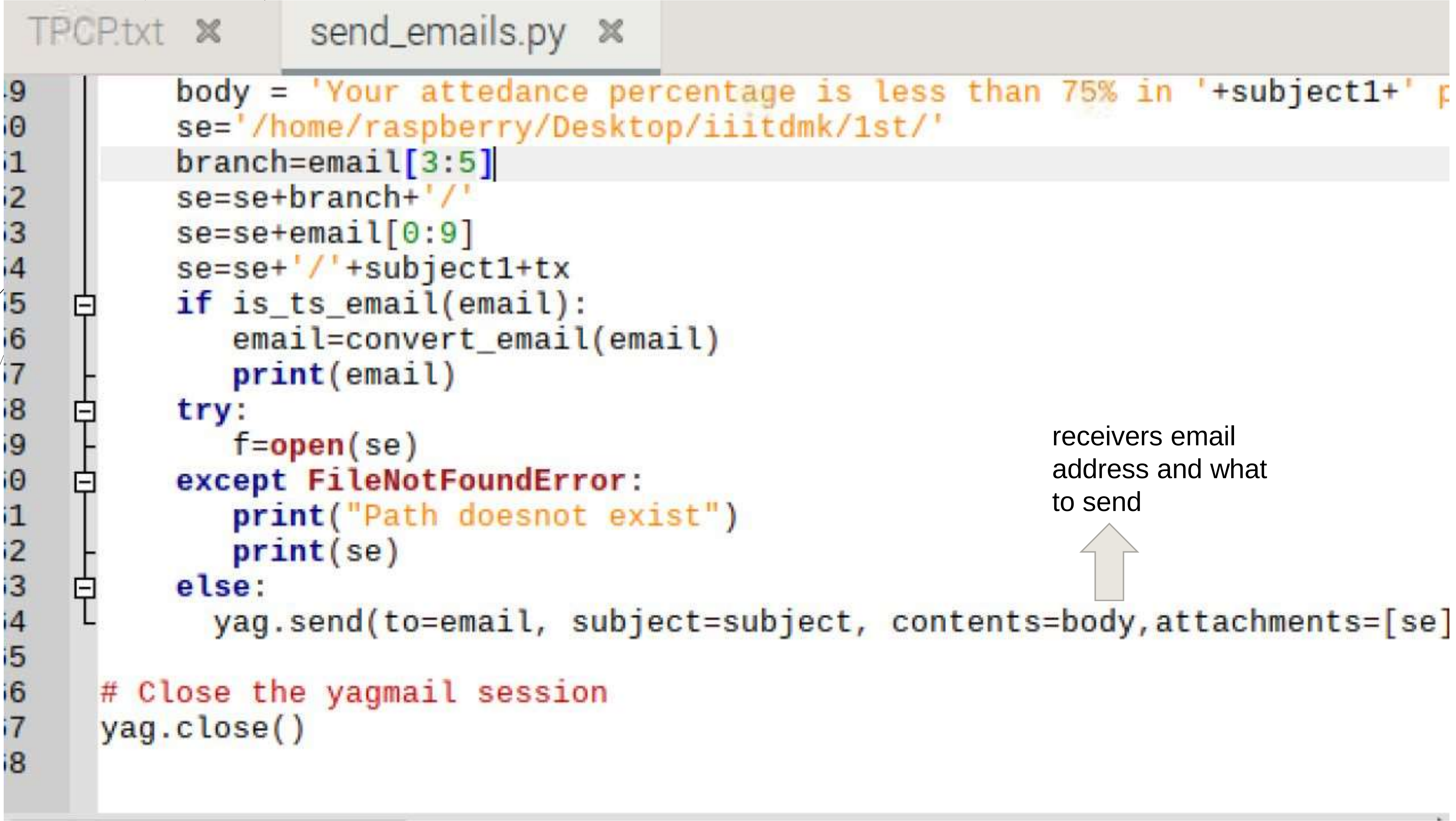
- Here the variable jpercent was created and it contains the attendance percentage
- If your attendance is less than the 75% your email will be added to queue where they ready to send emails

send_emails.py



```
# Read the email addresses from the text file
with open('email_list.txt', 'r') as file:
    email_list = [line.strip() for line in file]

# Create a yagmail instance with your email credentials
yag = yagmail.SMTP('iiitdmkurnoolattendanceproject@gmail.com', 'paeuazn

# Send emails to each recipient
tx='.txt'
subject1=input("ENTER THE SUBJECT NAME : ").upper()
for email in email_list:
    print(email)
    subject = ' `attendance alert'
    body = 'Your attedance percentage is less than 75% in '+subject1+'
    se='/home/raspberry/Desktop/iiitdmk/1st/'
    branch=email[3:5]
    se=se+branch+'/'
    se=se+email[0:9]
    se=se+'/'+subject1+tx
```

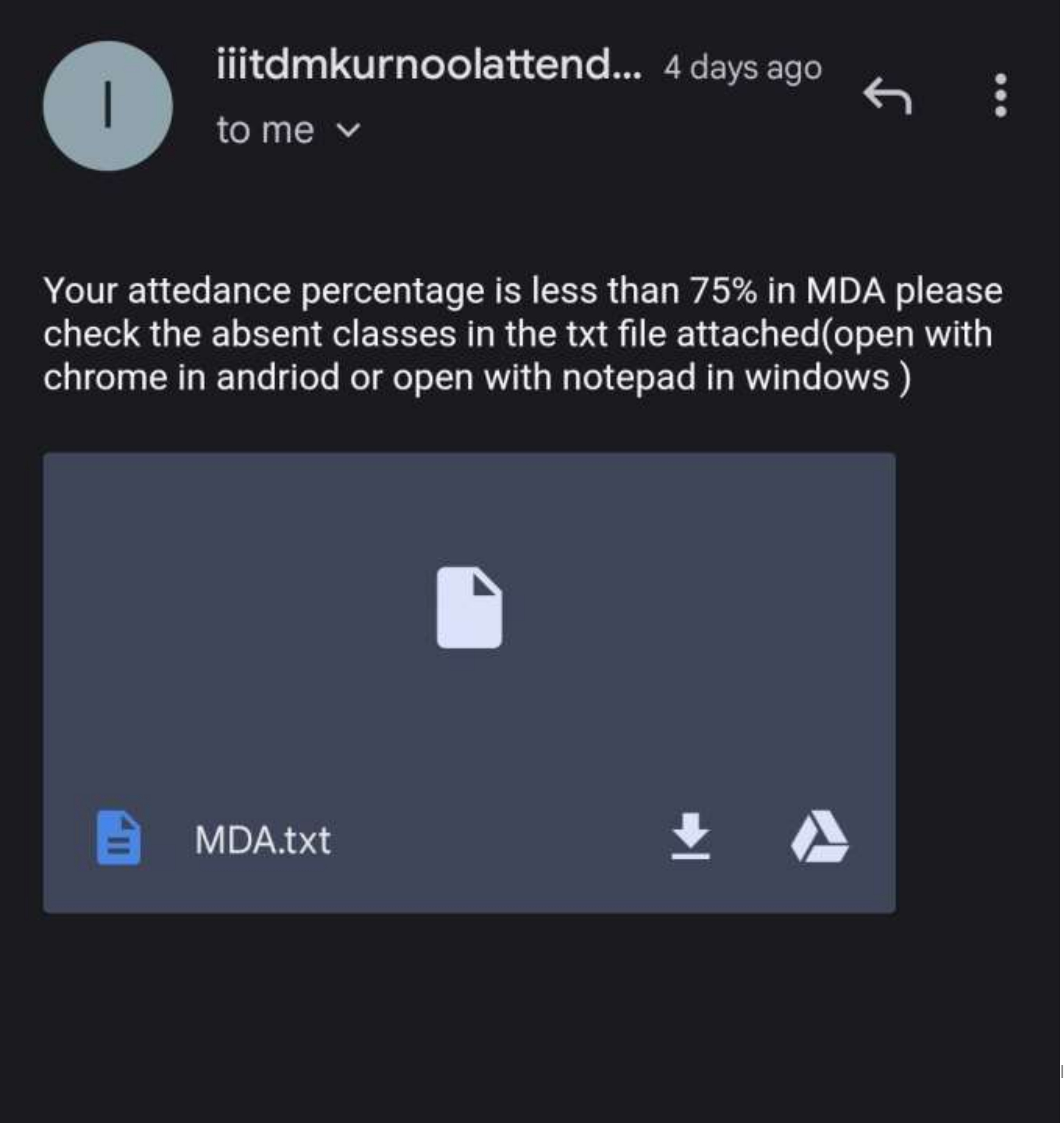
token



TPCP.txt ✖ send_emails.py ✖

```
    body = 'Your attedance percentage is less than 75% in '+subject1+' p
    se='/home/raspberry/Desktop/iiitdmk/1st/'
    branch=email[3:5]
    se=se+branch+'/'
    se=se+email[0:9]
    se=se+'/'+subject1+tx
    if is_ts_email(email):
       email=convert_email(email)
       print(email)
    try:
       f=open(se)
    except FileNotFoundError:
       print("Path doesnot exist")
       print(se)
    else:
      yag.send(to=email, subject=subject, contents=body,attachments=[se]

# Close the yagmail session
yag.close()
```

receivers email address and what to send


iiitdmkurnoolattend... 4 days ago

to me

Your attedance percentage is less than 75% in MDA please check the absent classes in the txt file attached(open with chrome in andriod or open with notepad in windows)

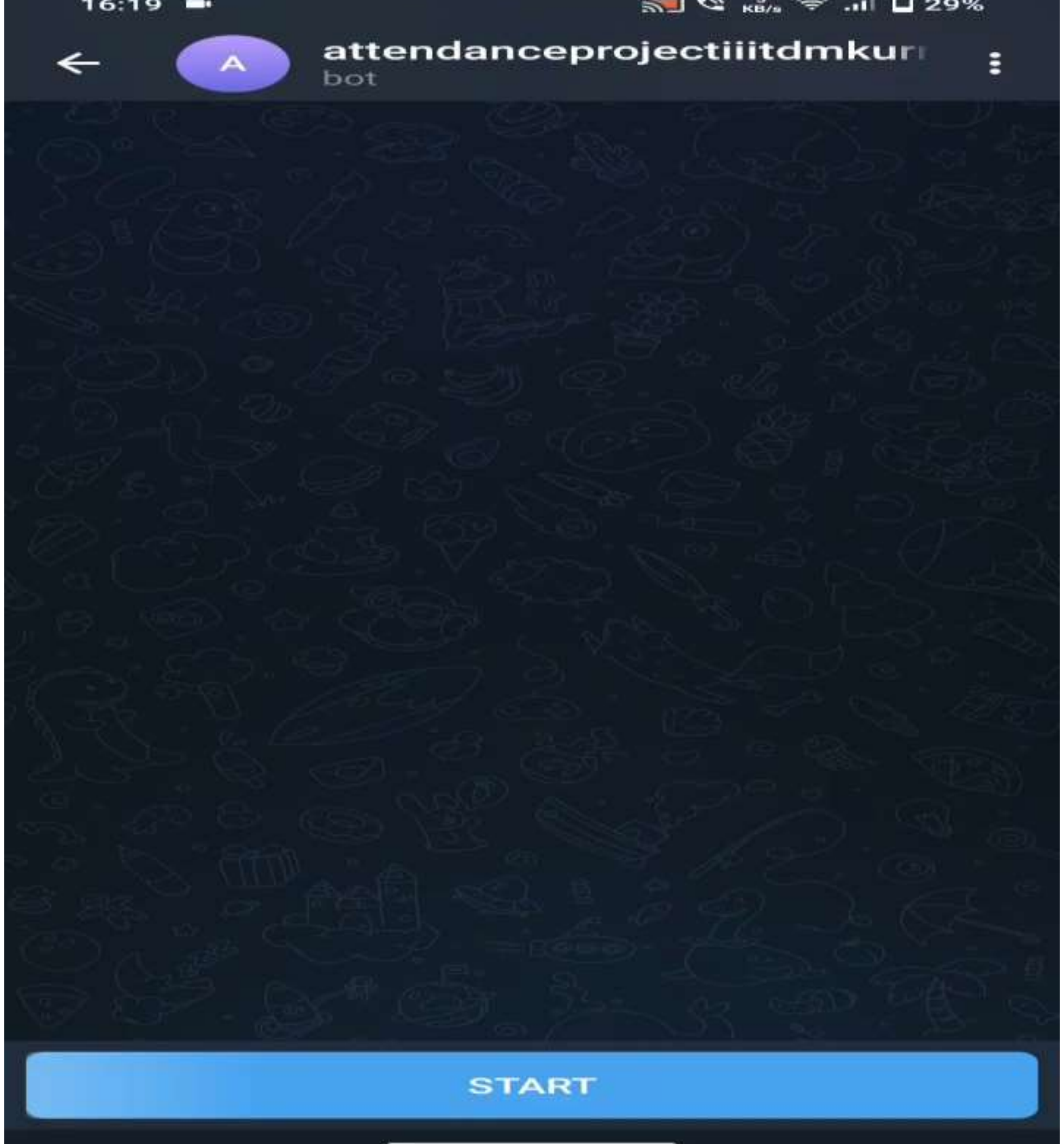
MDA.txt

- In this case the emails will be sent if your attendance is less than 75% with absent dates text file
- Yagmail is the library In the python that we can use for the email sending using the token
- We can send the multiples at a time within minutes it can send upto 100 emails individual

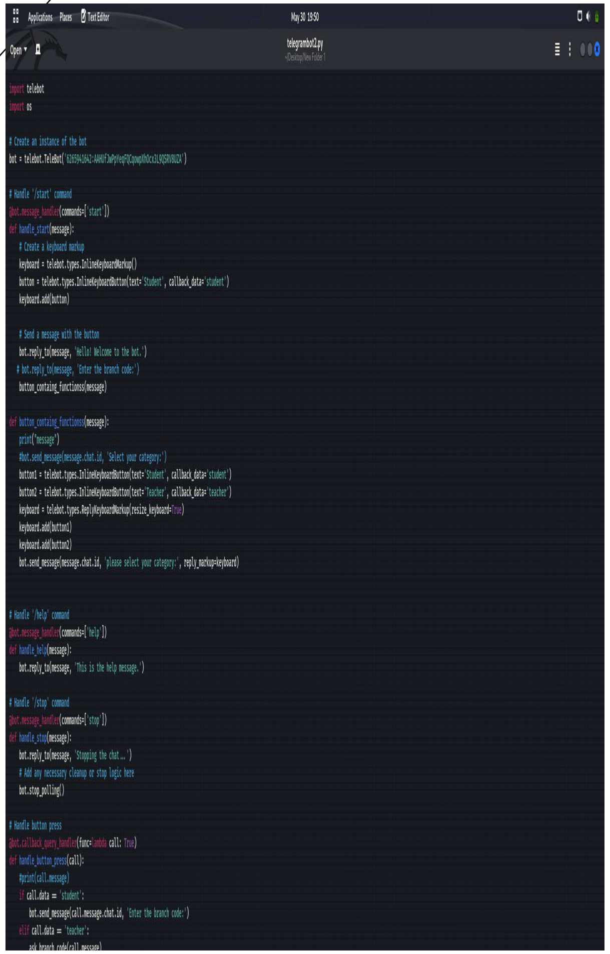
TELEGRAM BOT

- The telegram bot used for the to find the indivial dates that are absent
- Telegram bot is used as a supplementary tool for individual students to access attendance specifications.
- This will work by taking some of the user data
- This bot server is located on the raspberry this work only when raspberry pi is powered on
- This bot was written in the python using the telebot library that hosts the telegram bot

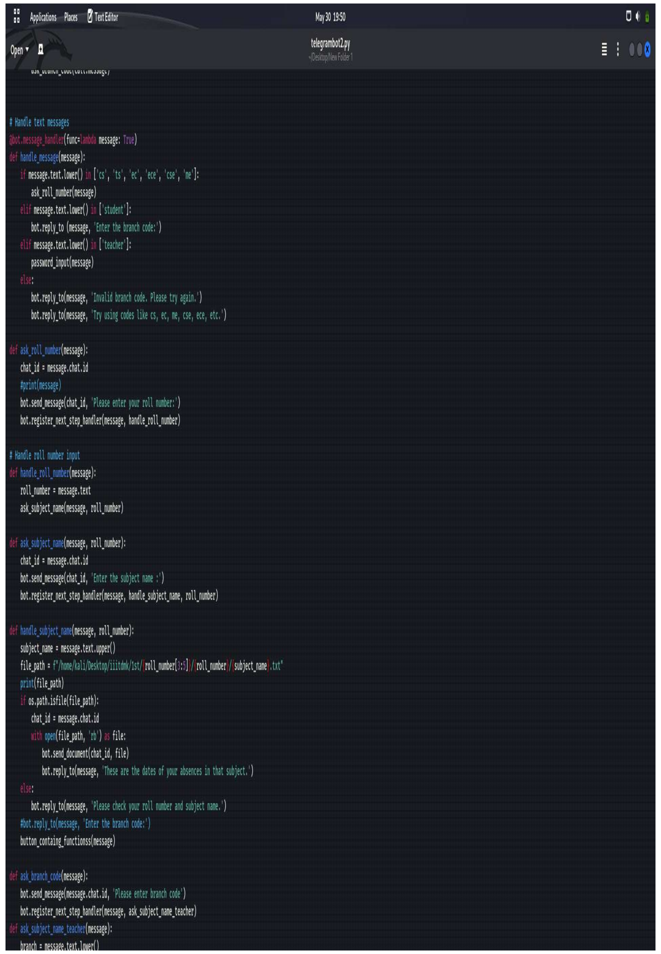
TELEGRAM BOT



CODE FOR TELEGRAM BOT



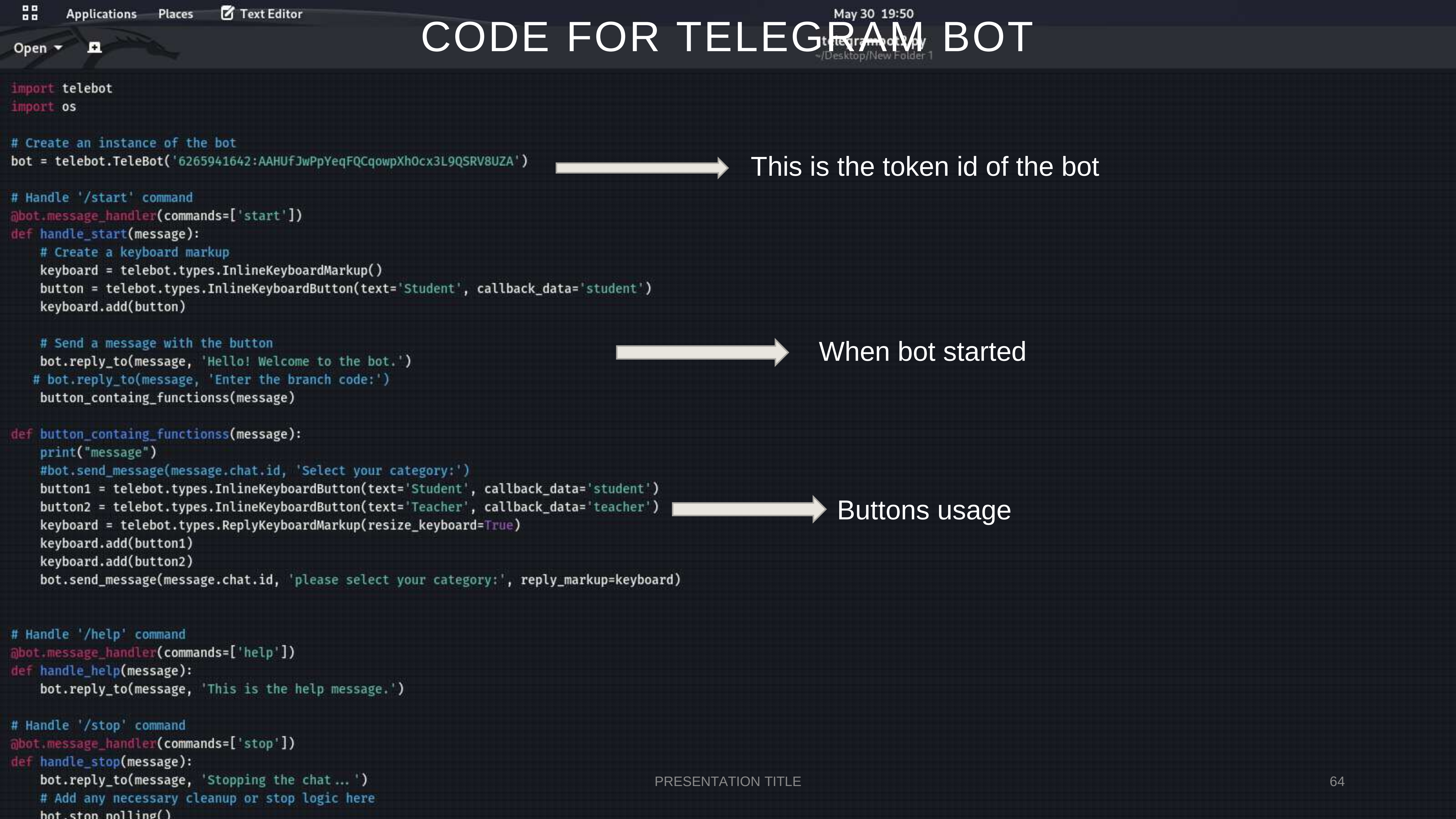
```python
import telebot
import os

# Create an instance of the bot
bot = telebot.TeleBot('6265941642:AAHUfJwPpYeqFQCqowpXhOcx3L9QSRV8UZA')

# Handle '/start' command
@bot.message_handler(commands=['start'])
def handle_start(message):
    # Create a keyboard markup
    keyboard = telebot.types.InlineKeyboardMarkup()
    button = telebot.types.InlineKeyboardButton(text='Student', callback_data='student')
    keyboard.add(button)


    # Send a message with the button
    bot.reply_to(message, 'Hello! Welcome to the bot.')
   # bot.reply_to(message, 'Enter the branch code:')
    button_containg_functionss(message)

def button_containg_functionss(message):
    print("message")
    #bot.send_message(message.chat.id, 'Select your category:')
    button1 = telebot.types.InlineKeyboardButton(text='Student', callback_data='student')
    button2 = telebot.types.InlineKeyboardButton(text='Teacher', callback_data='teacher')
    keyboard = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
    keyboard.add(button1)
    keyboard.add(button2)
    bot.send_message(message.chat.id, 'please select your category:', reply_markup=keyboard)


# Handle '/help' command
@bot.message_handler(commands=['help'])
def handle_help(message):
    bot.reply_to(message, 'This is the help message.')

# Handle '/stop' command
@bot.message_handler(commands=['stop'])
def handle_stop(message):
    bot.reply_to(message, 'Stopping the chat...')
    # Add any necessary cleanup or stop logic here
    bot.stop_polling()

# Handle button press
@bot.callback_query_handler(func=lambda call: True)
def handle_button_press(call):
    #print(call.message)
    if call.data == 'student':
        bot.send_message(call.message.chat.id, 'Enter the branch code:')
    elif call.data == 'teacher':
        ask_branch_code(call.message)
```



```python
        ask_branch_code(call.message)


# Handle text messages
@bot.message_handler(func=lambda message: True)
def handle_message(message):
    if message.text.lower() in ['cs', 'ts', 'ec', 'ece', 'cse', 'me']:
        ask_roll_number(message)
    elif message.text.lower() in ['student']:
        bot.reply_to (message, 'Enter the branch code:')
    elif message.text.lower() in ['teacher']:
        password_input(message)
    else:
        bot.reply_to(message, 'Invalid branch code. Please try again.')
        bot.reply_to(message, 'Try using codes like cs, ec, me, cse, ece, etc.')

def ask_roll_number(message):
    chat_id = message.chat.id
    #print(message)
    bot.send_message(chat_id, 'Please enter your roll number:')
    bot.register_next_step_handler(message, handle_roll_number)

# Handle roll number input
def handle_roll_number(message):
    roll_number = message.text
    ask_subject_name(message, roll_number)

def ask_subject_name(message, roll_number):
    chat_id = message.chat.id
    bot.send_message(chat_id, 'Enter the subject name :')
    bot.register_next_step_handler(message, handle_subject_name, roll_number)

def handle_subject_name(message, roll_number):
    subject_name = message.text.upper()
    file_path = f"/home/kali/Desktop/iiitdmk/1st/{roll_number[3:5]}/{roll_number}/{subject_name}.txt"
    print(file_path)
    if os.path.isfile(file_path):
        chat_id = message.chat.id
        with open(file_path, 'rb') as file:
            bot.send_document(chat_id, file)
            bot.reply_to(message, 'These are the dates of your absences in that subject.')
    else:
        bot.reply_to(message, 'Please check your roll number and subject name.')
    #bot.reply_to(message, 'Enter the branch code:')
    button_containg_functionss(message)

def ask_branch_code(message):
    bot.send_message(message.chat.id, 'Please enter branch code')
    bot.register_next_step_handler(message, ask_subject_name_teacher)
def ask_subject_name_teacher(message):
    branch = message.text.lower()
```

```python
[illegible]
```

CODE FOR TELEGRAM BOT



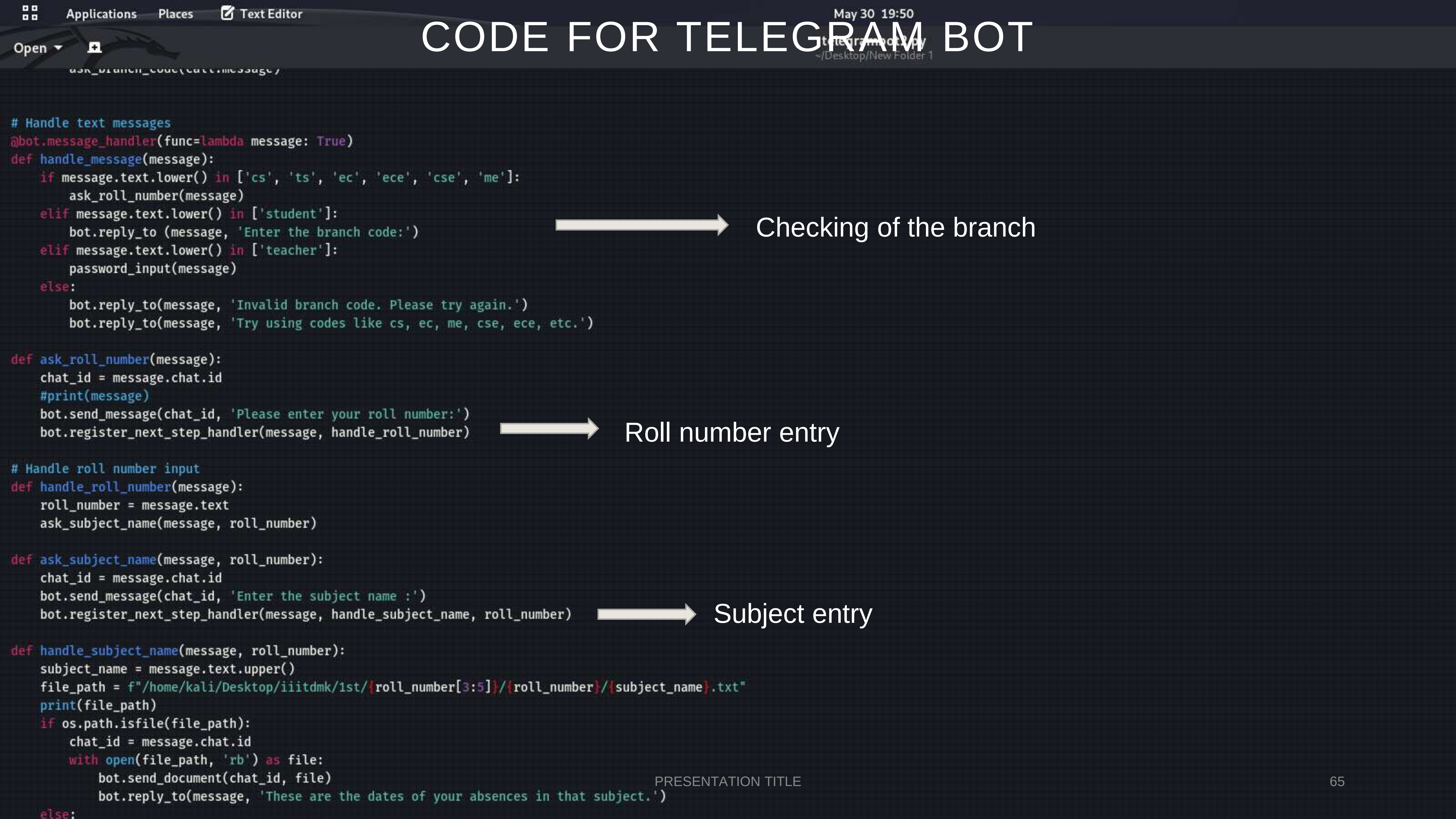
```python
import telebot
import os

# Create an instance of the bot
bot = telebot.TeleBot('6265941642:AAHUfJwPpYeqFQCqowpXhOcx3L9QSRV8UZA')

# Handle '/start' command
@bot.message_handler(commands=['start'])
def handle_start(message):
    # Create a keyboard markup
    keyboard = telebot.types.InlineKeyboardMarkup()
    button = telebot.types.InlineKeyboardButton(text='Student', callback_data='student')
    keyboard.add(button)

    # Send a message with the button
    bot.reply_to(message, 'Hello! Welcome to the bot.')
   # bot.reply_to(message, 'Enter the branch code:')
    button_containg_functionss(message)

def button_containg_functionss(message):
    print("message")
    #bot.send_message(message.chat.id, 'Select your category:')
    button1 = telebot.types.InlineKeyboardButton(text='Student', callback_data='student')
    button2 = telebot.types.InlineKeyboardButton(text='Teacher', callback_data='teacher')
    keyboard = telebot.types.ReplyKeyboardMarkup(resize_keyboard=True)
    keyboard.add(button1)
    keyboard.add(button2)
    bot.send_message(message.chat.id, 'please select your category:', reply_markup=keyboard)


# Handle '/help' command
@bot.message_handler(commands=['help'])
def handle_help(message):
    bot.reply_to(message, 'This is the help message.')

# Handle '/stop' command
@bot.message_handler(commands=['stop'])
def handle_stop(message):
    bot.reply_to(message, 'Stopping the chat ... ')
    # Add any necessary cleanup or stop logic here
    bot.stop_polling()
```

This is the token id of the bot

When bot started

Buttons usage

CODE FOR TELEGRAM BOT



```
        ask_branch_code(call.message)


# Handle text messages
@bot.message_handler(func=lambda message: True)
def handle_message(message):
    if message.text.lower() in ['cs', 'ts', 'ec', 'ece', 'cse', 'me']:
        ask_roll_number(message)
    elif message.text.lower() in ['student']:
        bot.reply_to (message, 'Enter the branch code:')
    elif message.text.lower() in ['teacher']:
        password_input(message)
    else:
        bot.reply_to(message, 'Invalid branch code. Please try again.')
        bot.reply_to(message, 'Try using codes like cs, ec, me, cse, ece, etc.')

def ask_roll_number(message):
    chat_id = message.chat.id
    #print(message)
    bot.send_message(chat_id, 'Please enter your roll number:')
    bot.register_next_step_handler(message, handle_roll_number)

# Handle roll number input
def handle_roll_number(message):
    roll_number = message.text
    ask_subject_name(message, roll_number)

def ask_subject_name(message, roll_number):
    chat_id = message.chat.id
    bot.send_message(chat_id, 'Enter the subject name :')
    bot.register_next_step_handler(message, handle_subject_name, roll_number)

def handle_subject_name(message, roll_number):
    subject_name = message.text.upper()
    file_path = f"/home/kali/Desktop/iiitdmk/1st/{roll_number[3:5]}/{roll_number}/{subject_name}.txt"
    print(file_path)
    if os.path.isfile(file_path):
        chat_id = message.chat.id
        with open(file_path, 'rb') as file:
            bot.send_document(chat_id, file)
            bot.reply_to(message, 'These are the dates of your absences in that subject.')
    else:
```

Checking of the branch

Roll number entry

Subject entry

CONCLUSION

- this system has versatile utility in educational and corporate establishments, automating attendance and reducing time. enhancing the efficiency of organizations and institutes.

THANK YOU