PYTHON INTERNSHIP TRAINING PROGRAM

The industrial internship Training Program was organized by the Department of Basic sciences and humanities, TPCT'S college of Engineering, Osmanabad. It was arranged for all branches of F.Y.B.tech students. It was conducted by Fantasy Technologies in the period of 7th June to 2021 to 21/7/21 (45 Days) 90 Hrs. Founder of Fantasy technology are Mrs. Jyoti Tarange and trainer of this course were Mrs.Prarthana Bakshi. This industrial internship Training Program was held under the guidance of Honorable principal Dr.Vikramsingh Mane and coordinator of this program were Prof. Usha Wadne Head of BSH Dept.

Introduction to Python Programming

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale. Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library. Python interpreters are available for installation on many operating systems, allowing Python code execution on a wide variety of systems.

Introduction to Python Programming

- Why do we need Python?
- Program structure in Python

Execution steps

- Interactive Shell
- Executable or script files.
- User Interface or IDE

Memory management and Garbage collections

- Object creation and deletion
- Object properties

*

PYTHON INTERNSHIP TRAINING PROGRAM

Data Types and Operations

- Numbers
- Strings
- List
- Tuple
- Dictionary
- Other Core Types

Statements and Syntax in Python

- Assignments, Expressions and prints
- If tests and Syntax Rules
- While and For Loops
- Iterations and Comprehensions

File Operations

- Opening a file
- Using Files
- Other File tools

Functions in Python

- Function definition and call
- Function Scope
- Arguments
- Function Objects
- Anonymous Functions

Modules and Packages

- Module Creations and Usage
- Module Search Path
- Module Vs. Script
- Package Creation and Importing

*

PYTHON INTERNSHIP TRAINING PROGRAM

Classes in Python

• Classes and instances

• Classes method calls

• Inheritance and Compositions

• Static and Class Methods

• Bound and Unbound Methods

• Operator Overloading

• Polymorphism

Exception Handling in Python Programming

• Default Exception Handler

• Catching Exceptions

• Raise an exception

• User defined exception

Advanced Python Concepts

• Decorators

• Generators

• Iterators

• Co-routines

Standard Library Modules

Exercises

Roadmap with Python

PYTHON INTERNSHIP TRAINING PROGRAM

-: ADVANCED PYTHON TRAINING DETAILS :-

#MODULE1

Command Line arguments, Display Hooks

Standard data streams and Redirections

Osmodule , Sub-process module

Forking processes

Exec functions

Working with comprehensions

Working with Descriptors, Iterators, Generators and Decorators

The yield statement

range and x-range

Working with Context Managers

Wrapping Objects

Callback functions

Duck Typing, Monkey Patching in Python

Encapsulating Object Creation: Factory

MODULE 2

Introduction to Threads in python

thread module

threading module

Introduction to Pipes in python

anonymous pipes

named pipes, fifos

Introduction to Recursion

*

PYTHON INTERNSHIP TRAINING PROGRAM

Recursive functions in Python

Depth of Recursion

MODULE 3

CGI Programming

Introduction to WSGI

Introduction to PEP3333

Bottle Framework , Flask Framework

WebTest Framework

Create a basic Web Service in python

Working with Databases

Connecting with Cassandra DB, SQLite3, MySQL

Database Operations

MODULE 4

Network Programming

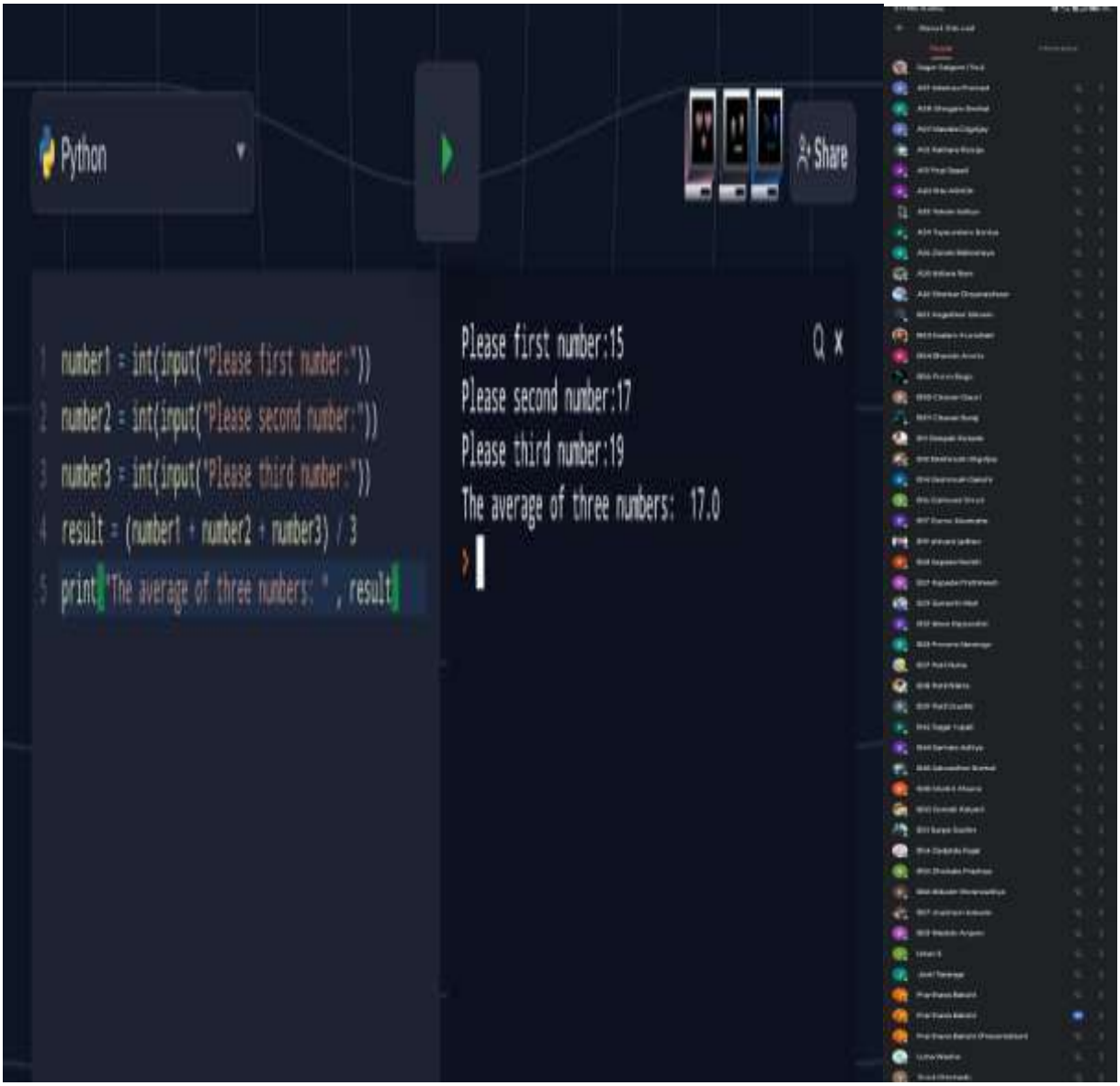
Working with XML Files

Developing GUIs

Working with SMTP

Integrating Python with other Languages

PYTHON INTERNSHIP TRAINING PROGRAM



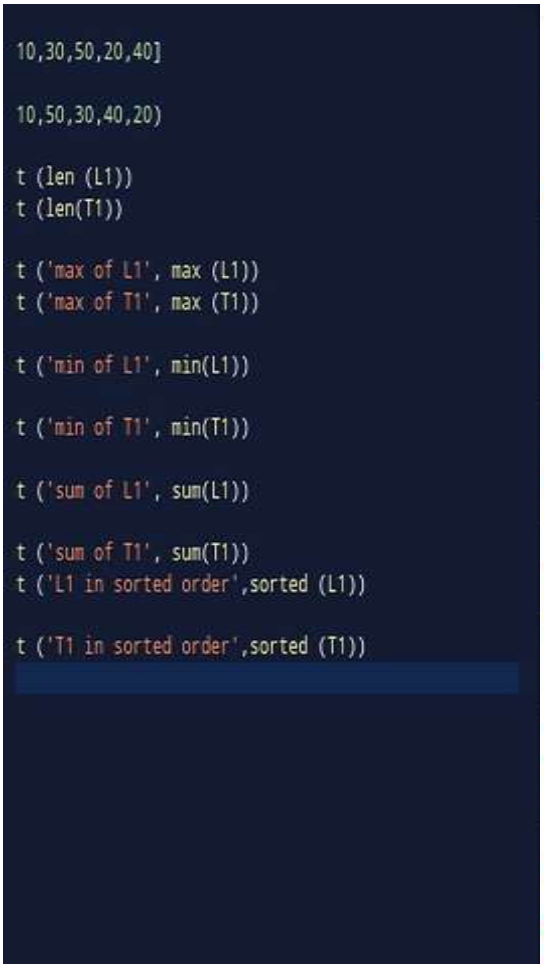
Class conduction and Attendance

*

PYTHON INTERNSHIP TRAINING PROGRAM

*

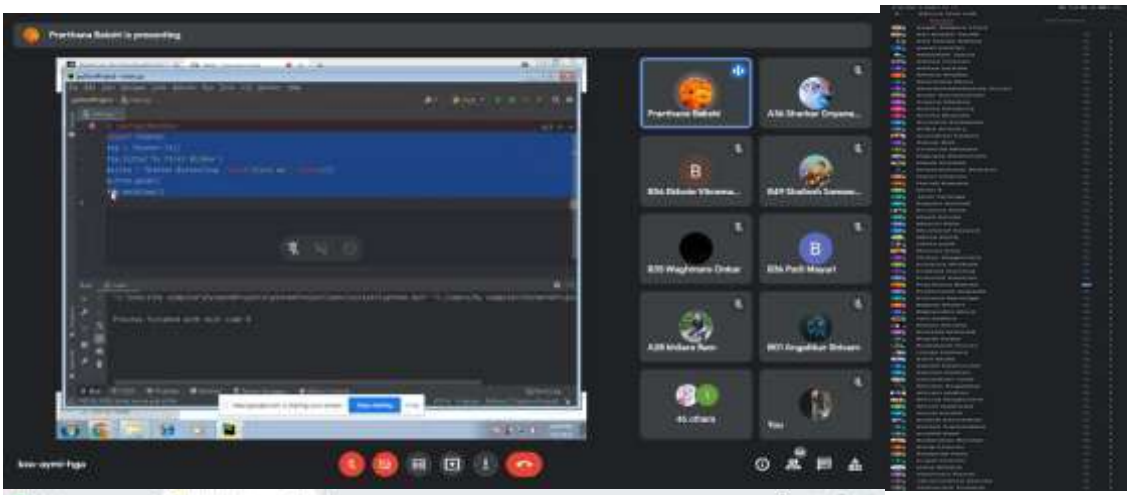
PYTHON INTERNSHIP TRAINING PROGRAM

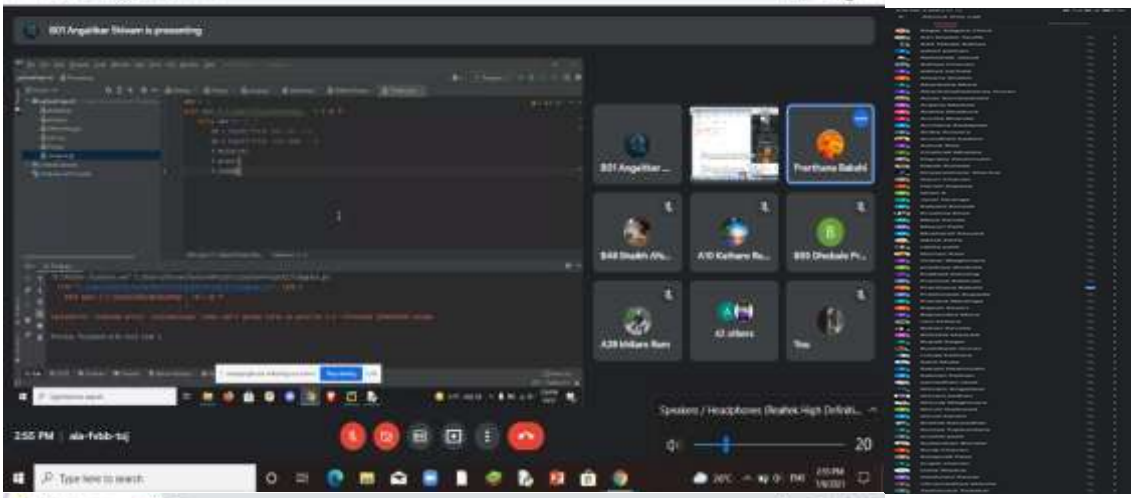


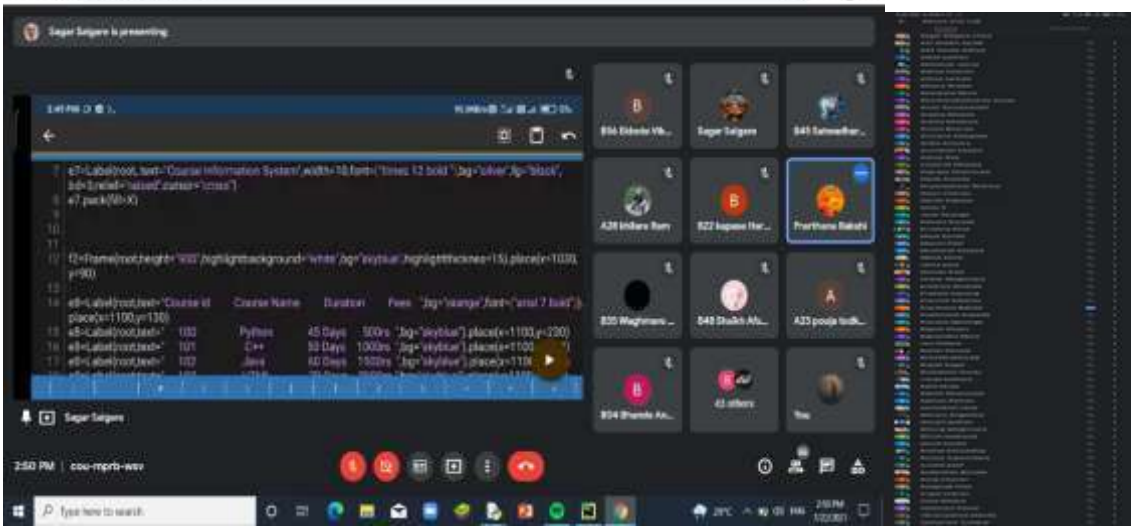
```
10,30,50,20,40]

10,50,30,40,20)

t (len (L1))
t (len(T1))

t ('max of L1', max (L1))
t ('max of T1', max (T1))

t ('min of L1', min(L1))

t ('min of T1', min(T1))

t ('sum of L1', sum(L1))

t ('sum of T1', sum(T1))
t ('L1 in sorted order',sorted (L1))

t ('T1 in sorted order',sorted (T1))
```

*

PYTHON INTERNSHIP TRAINING PROGRAM

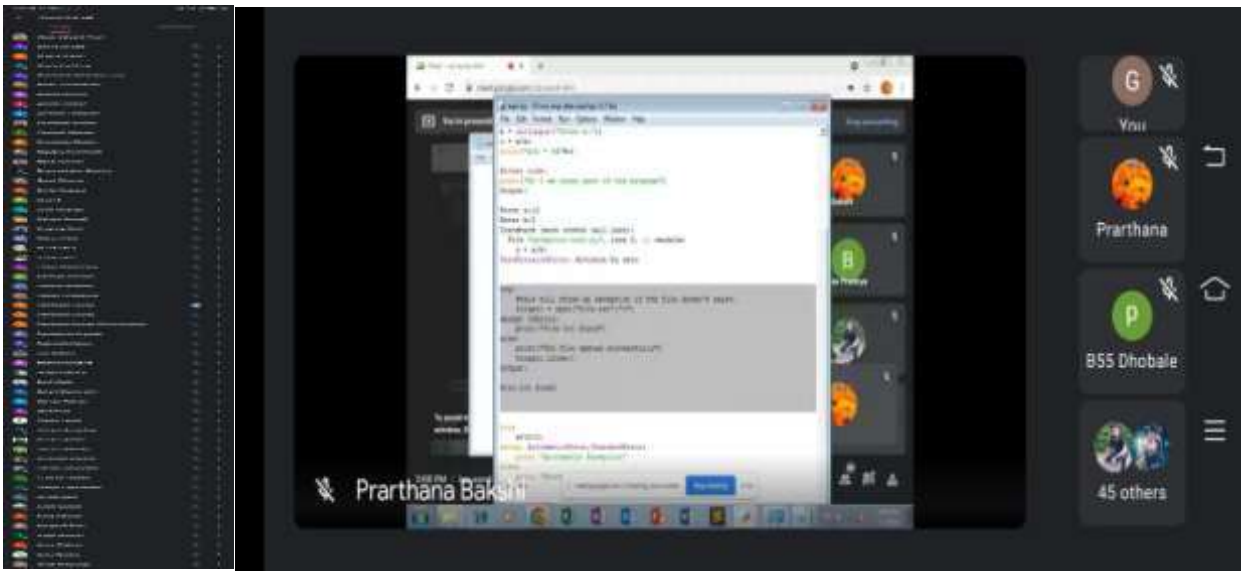


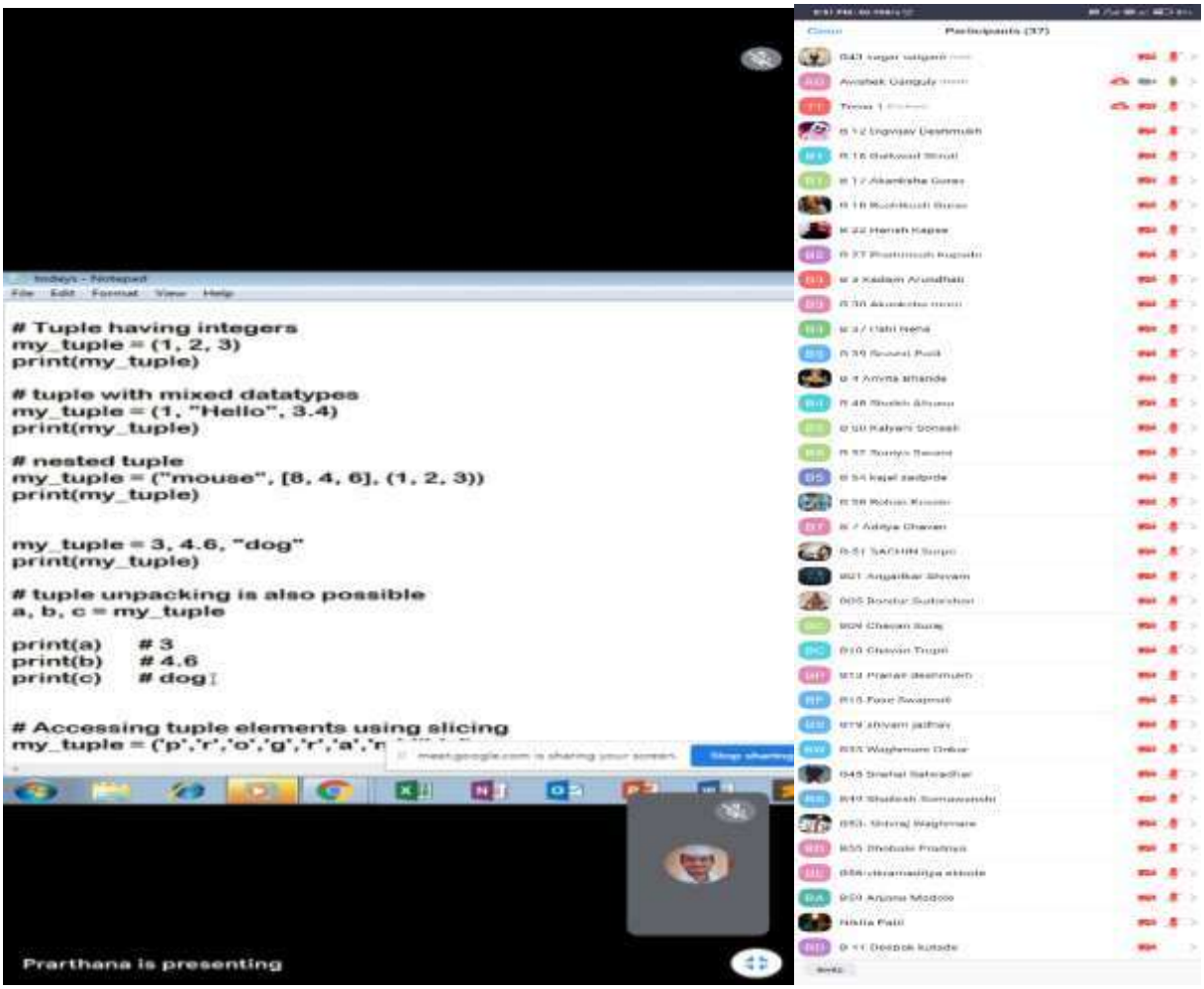




*

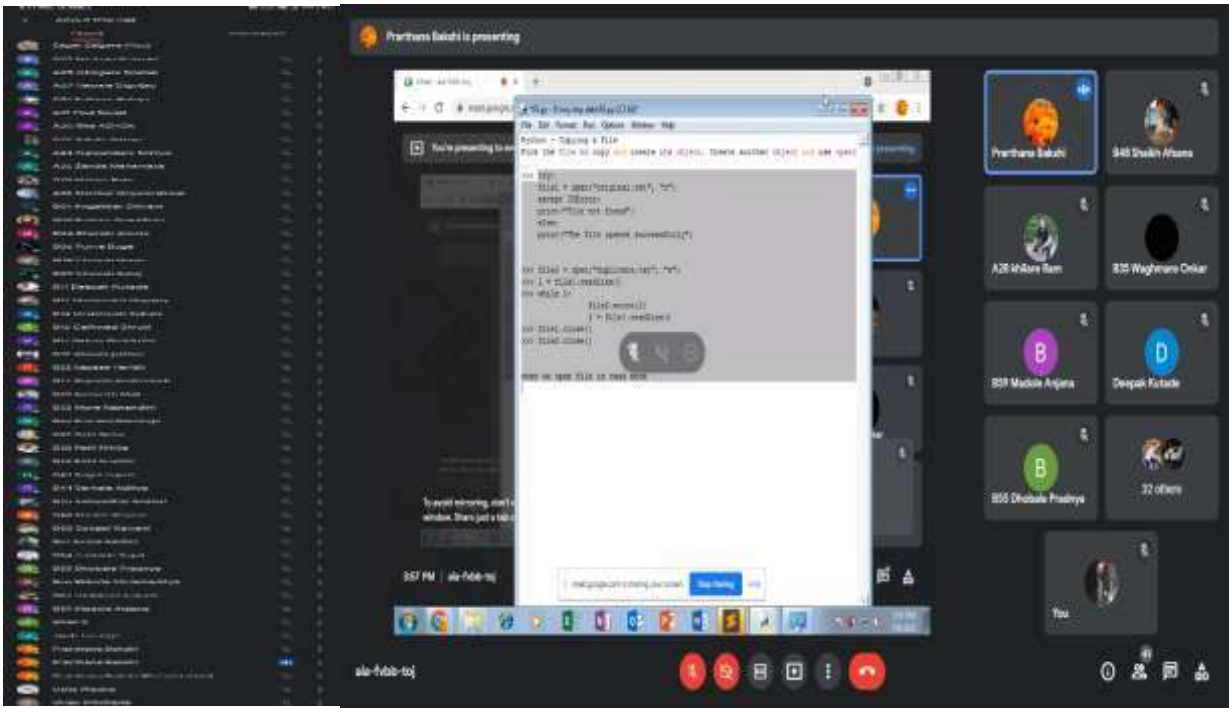
PYTHON INTERNSHIP TRAINING PROGRAM





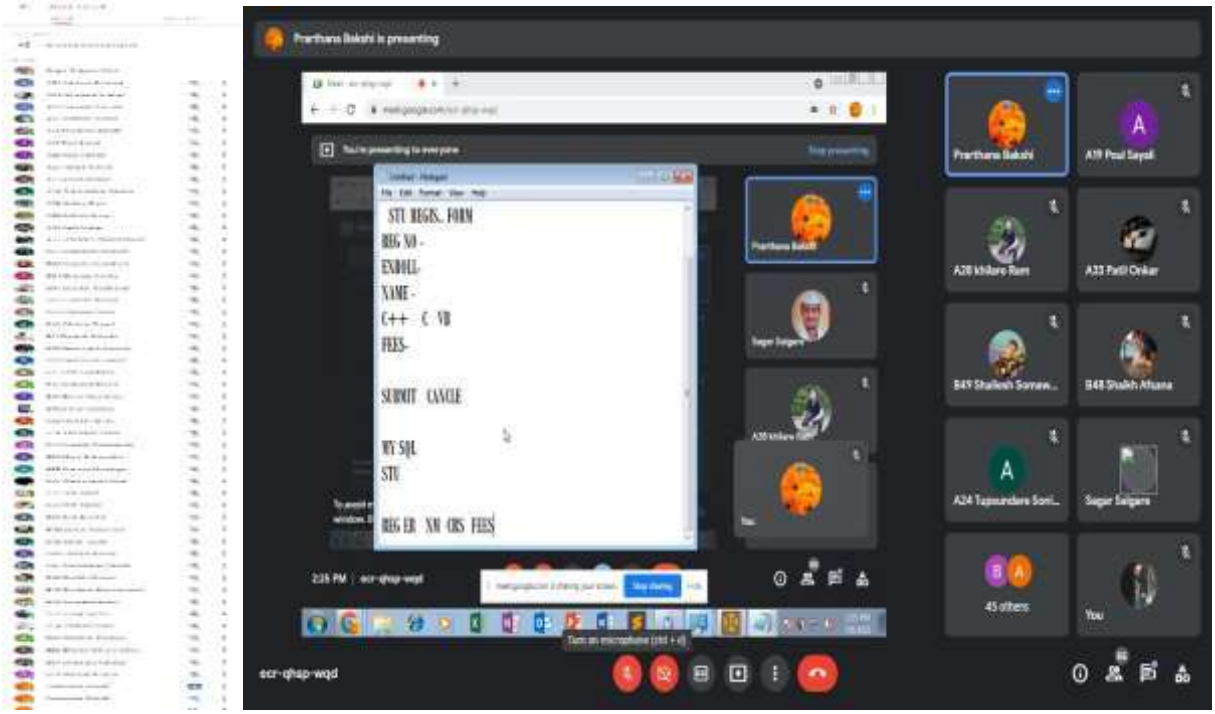
*

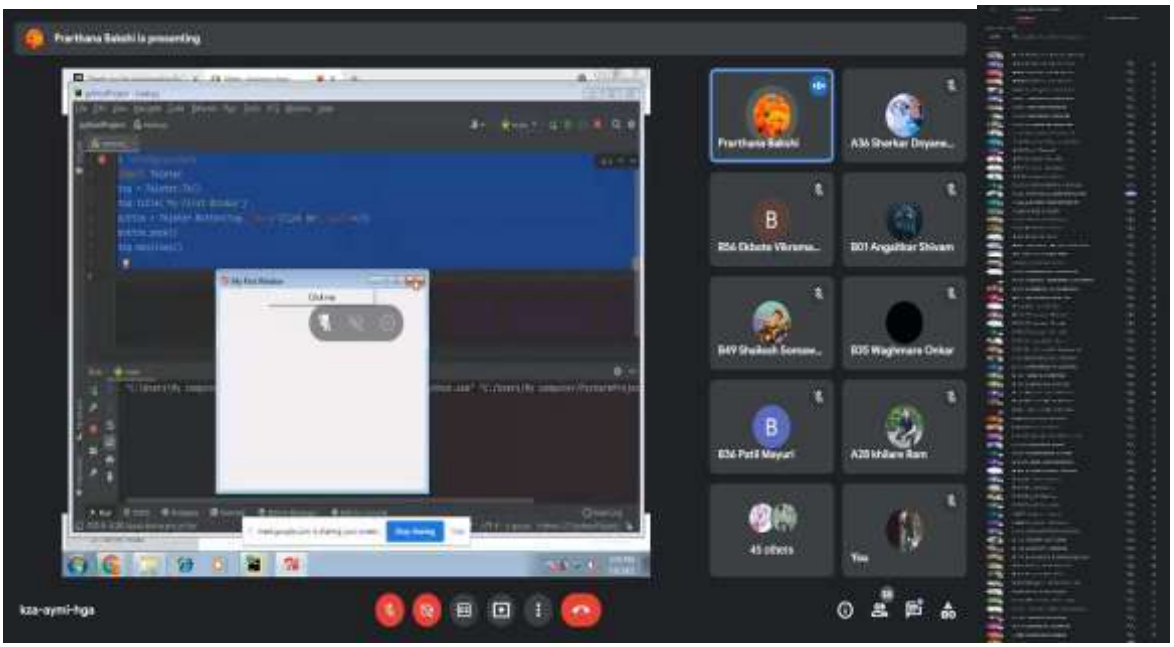
PYTHON INTERNSHIP TRAINING PROGRAM



*

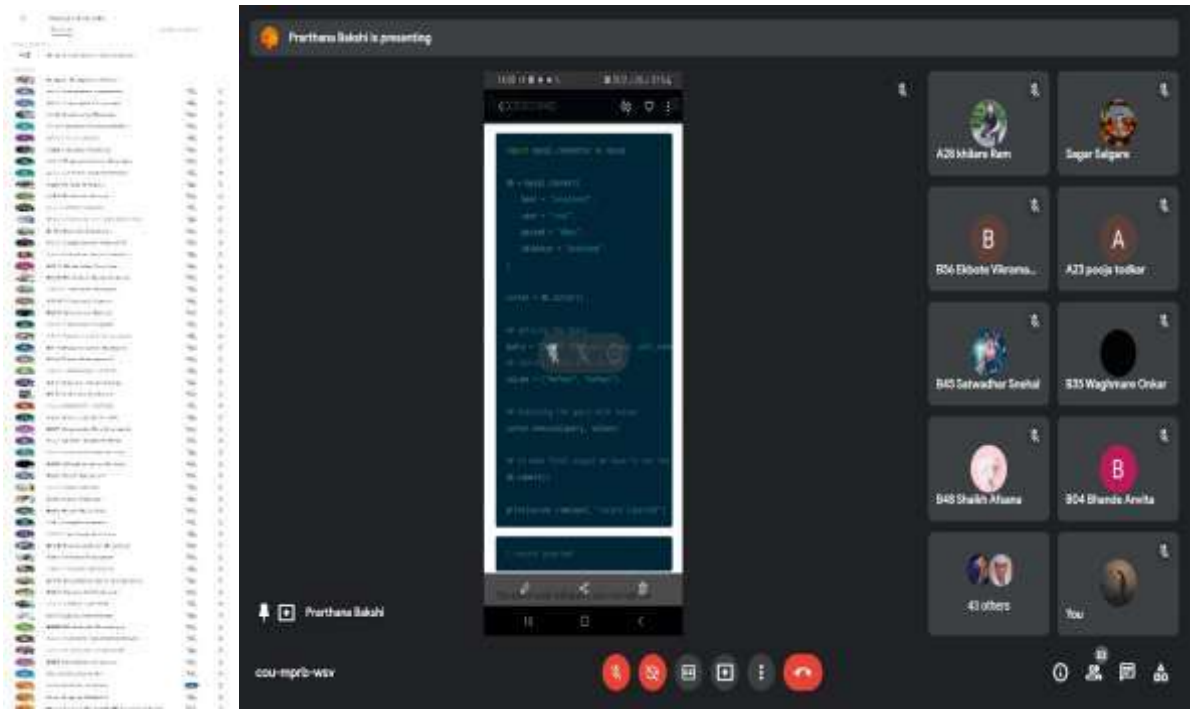
PYTHON INTERNSHIP TRAINING PROGRAM

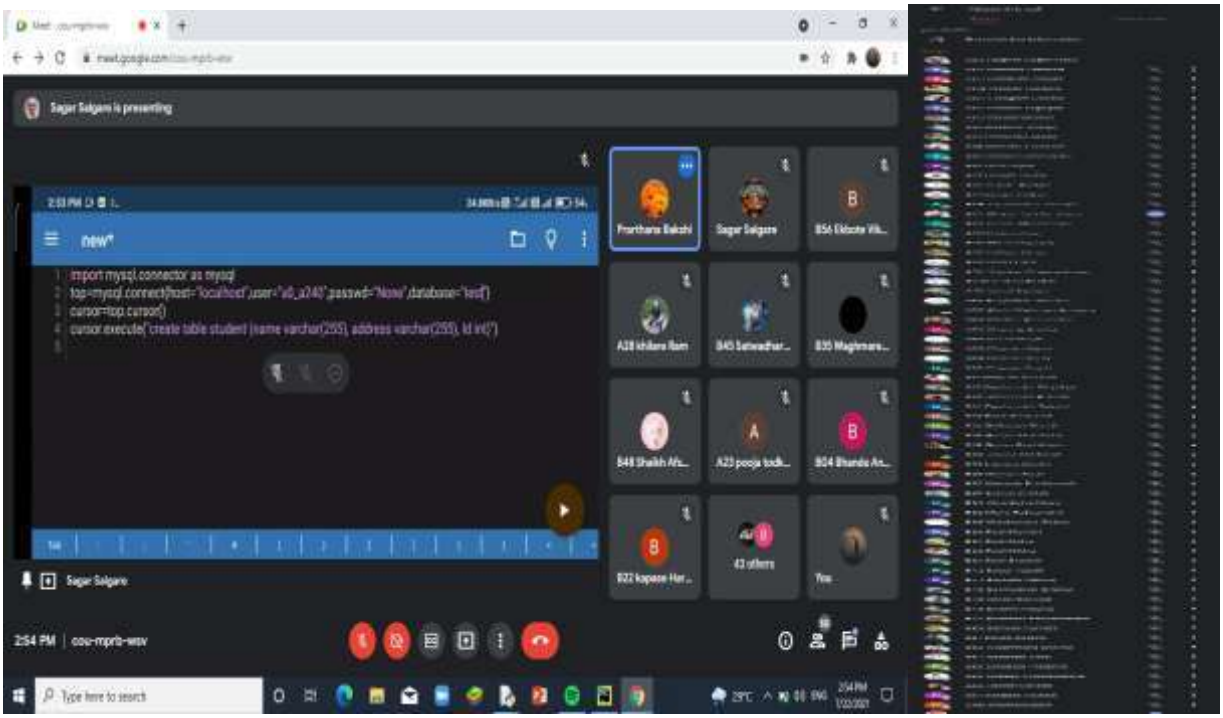




*

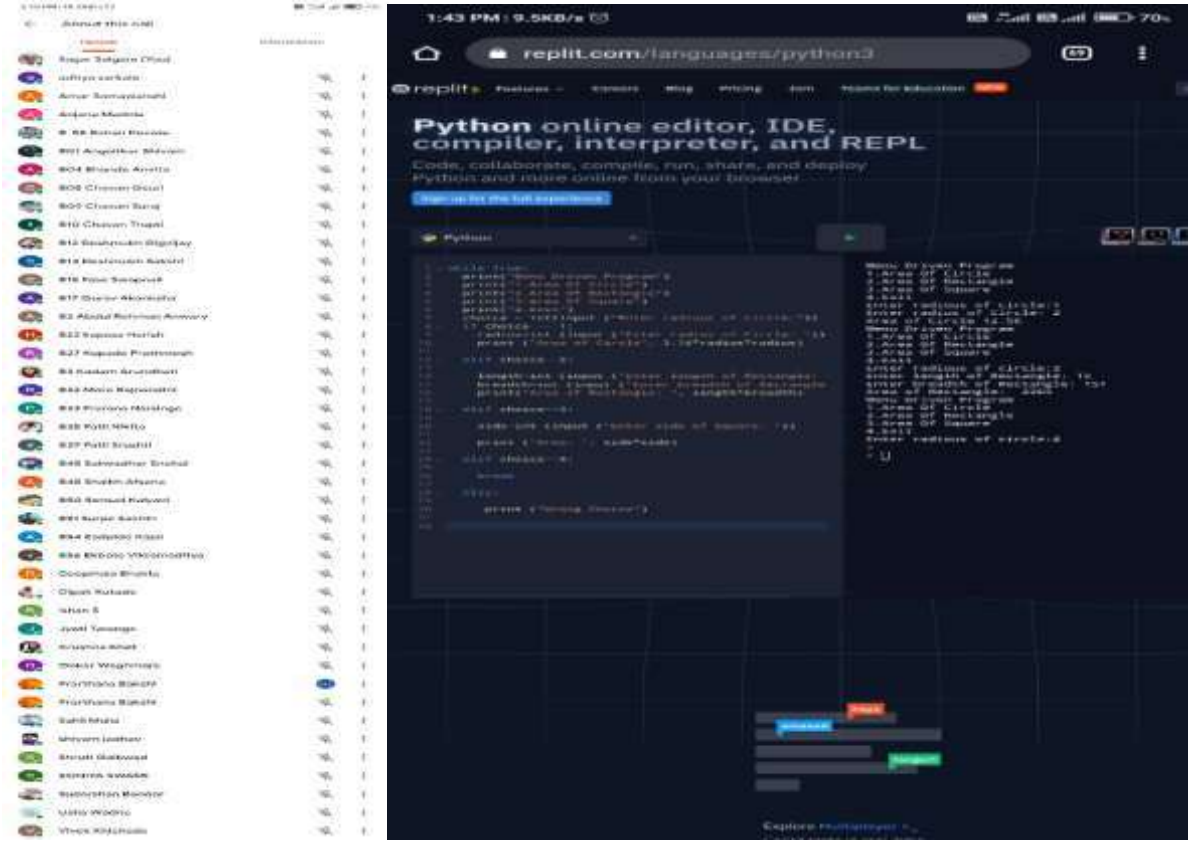
PYTHON INTERNSHIP TRAINING PROGRAM





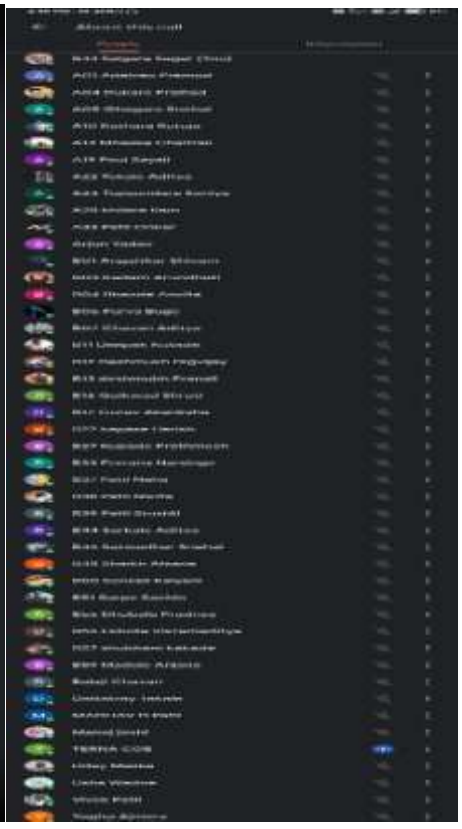
*

PYTHON INTERNSHIP TRAINING PROGRAM



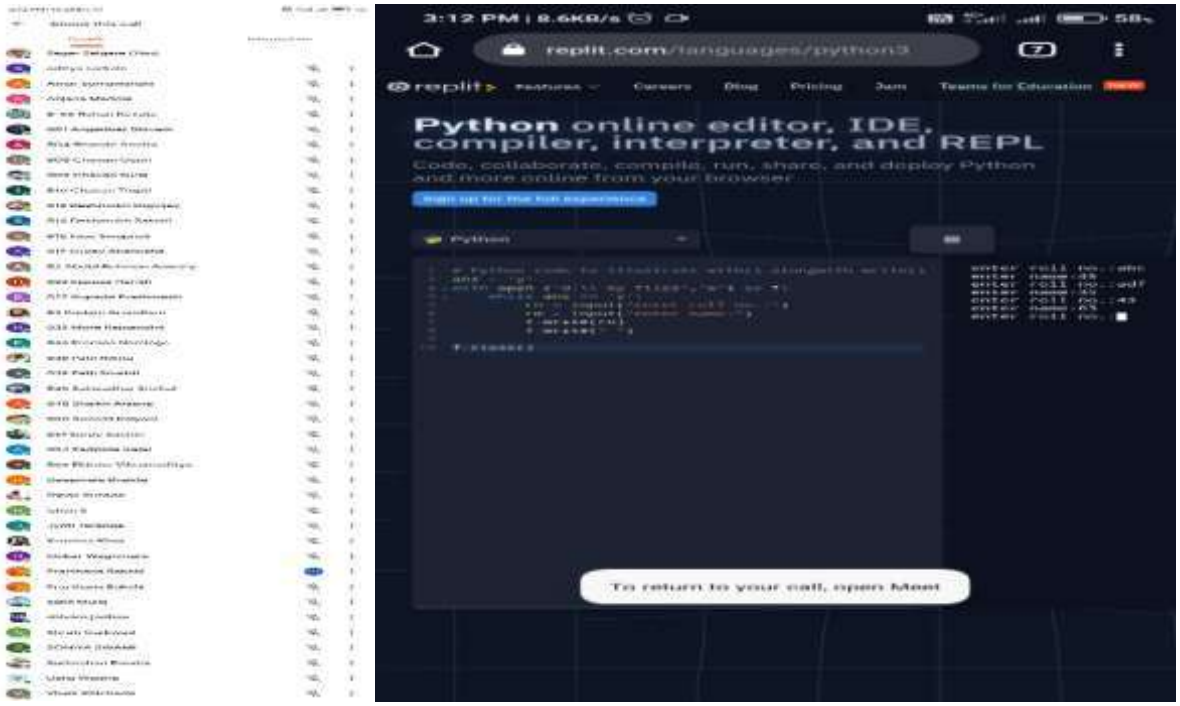
*

PYTHON INTERNSHIP TRAINING PROGRAM



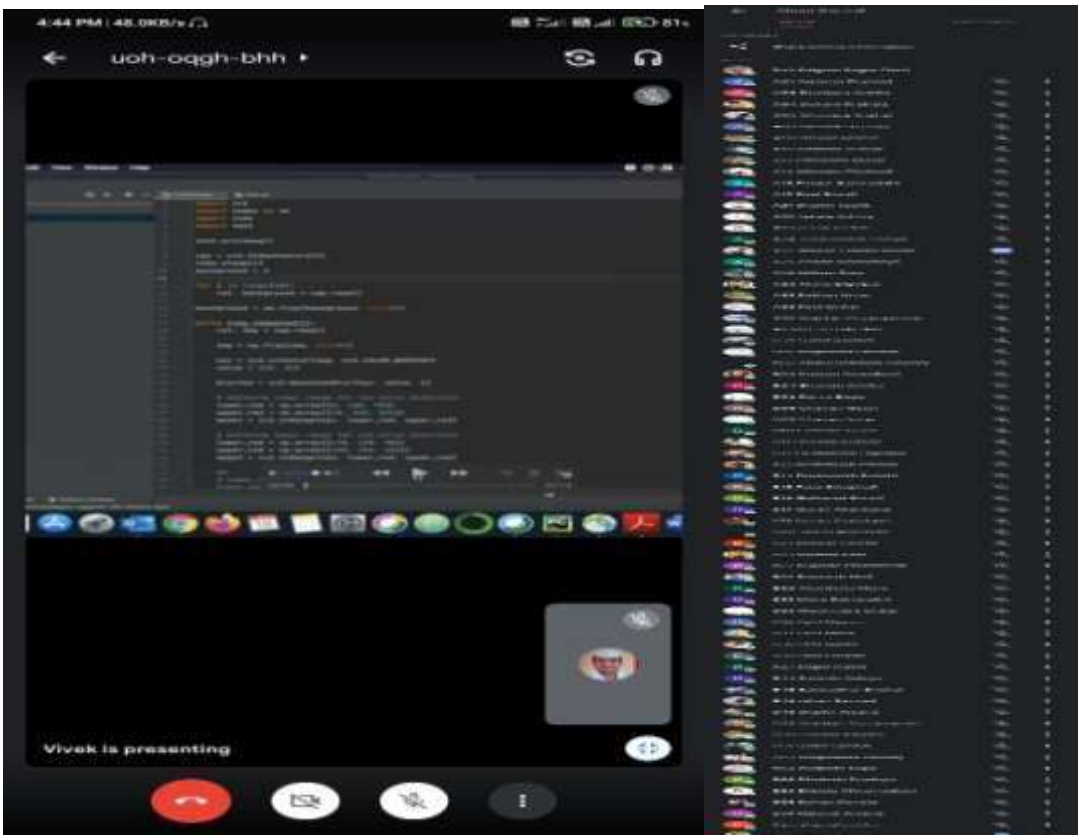
*

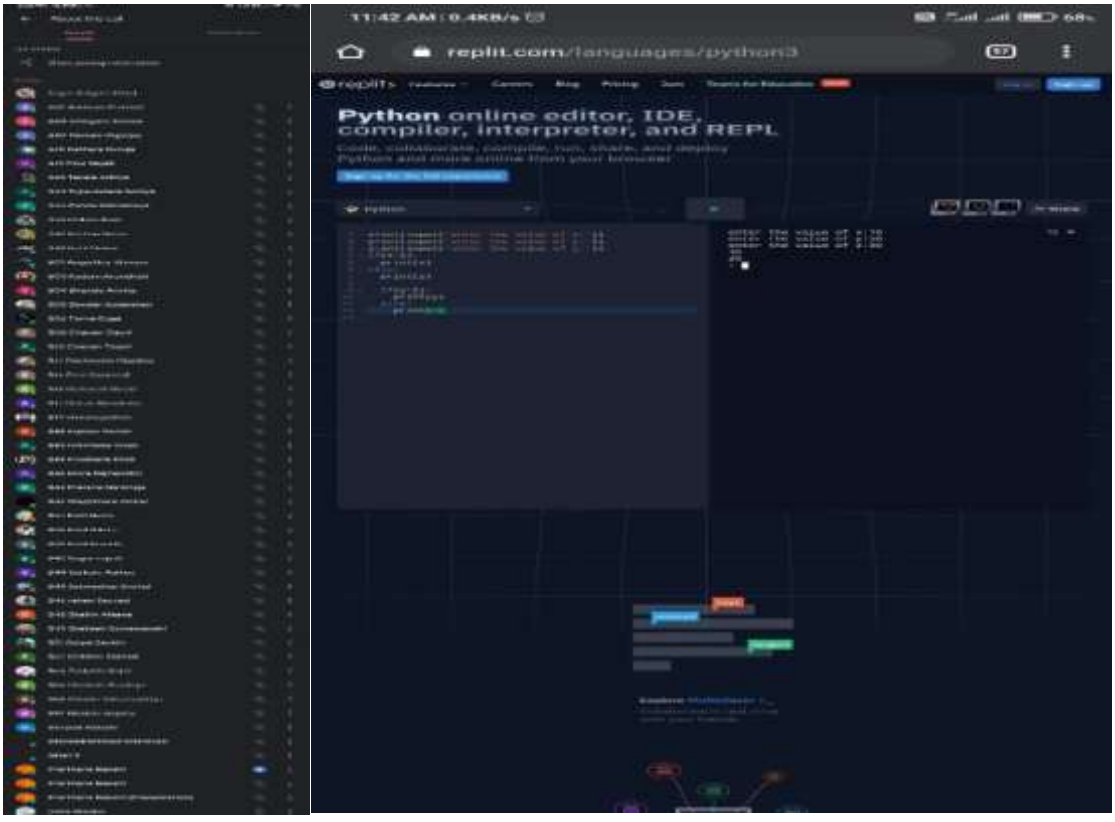
PYTHON INTERNSHIP TRAINING PROGRAM



*

PYTHON INTERNSHIP TRAINING PROGRAM

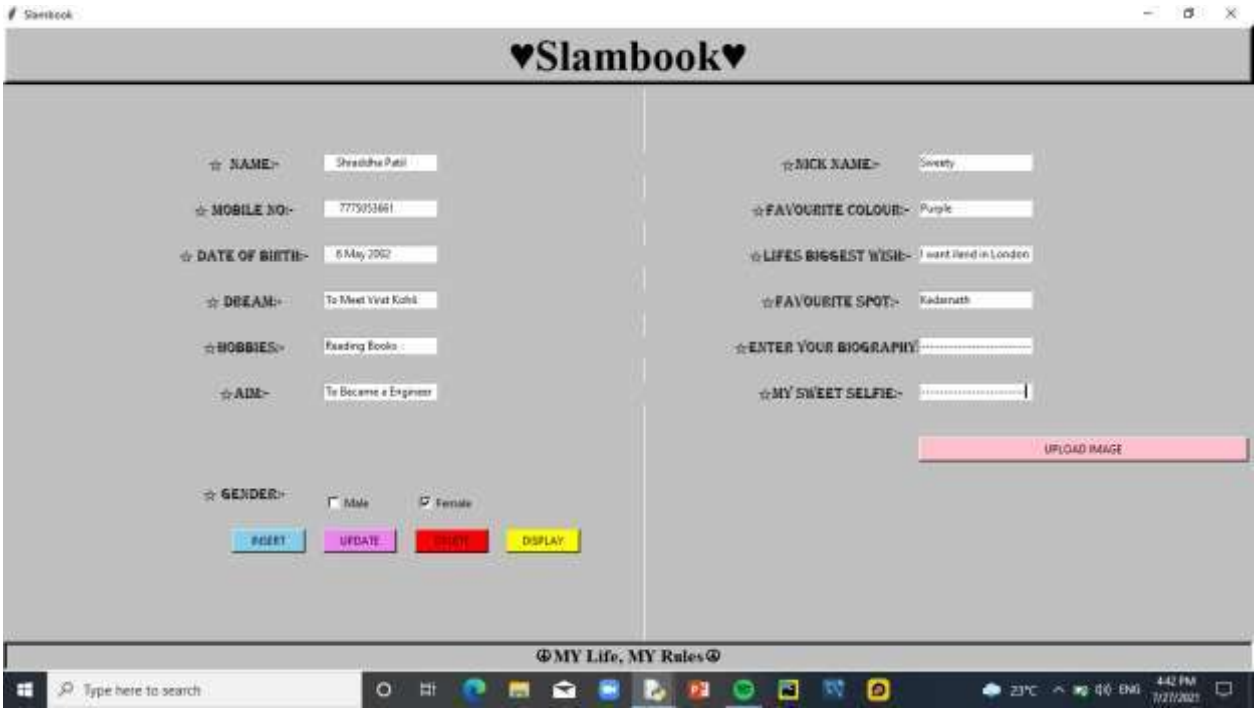


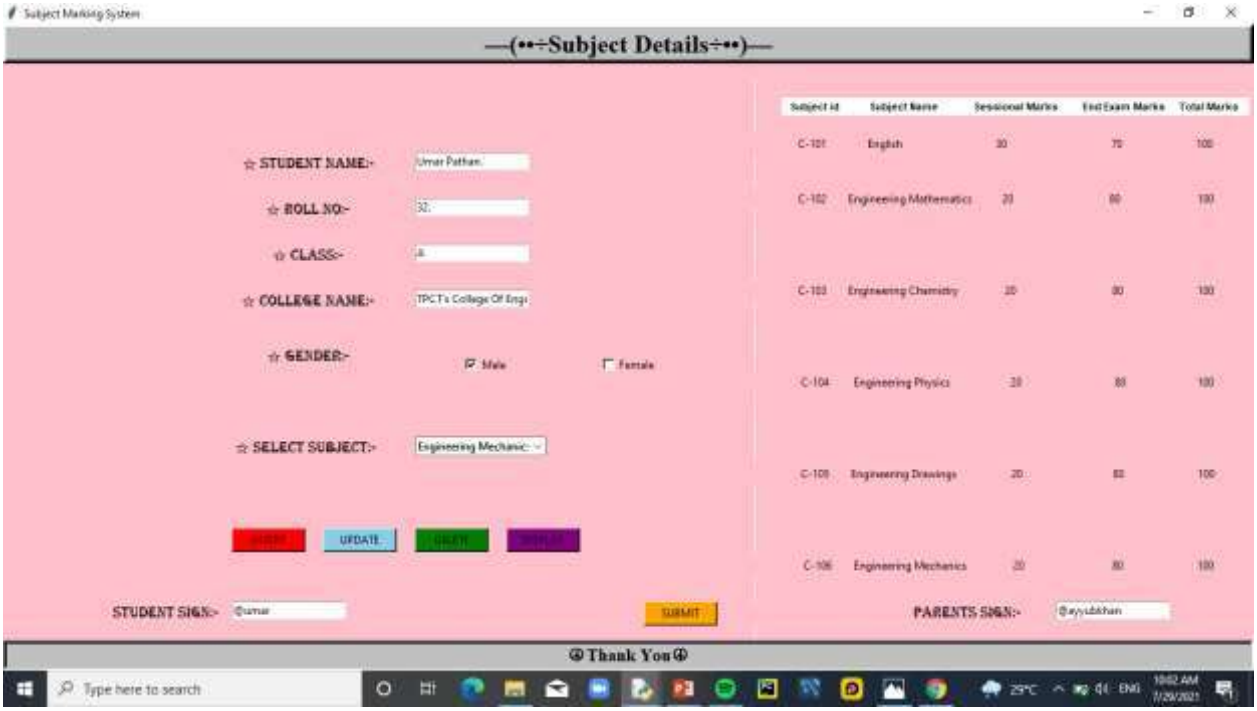


*

PYTHON INTERNSHIP TRAINING PROGRAM

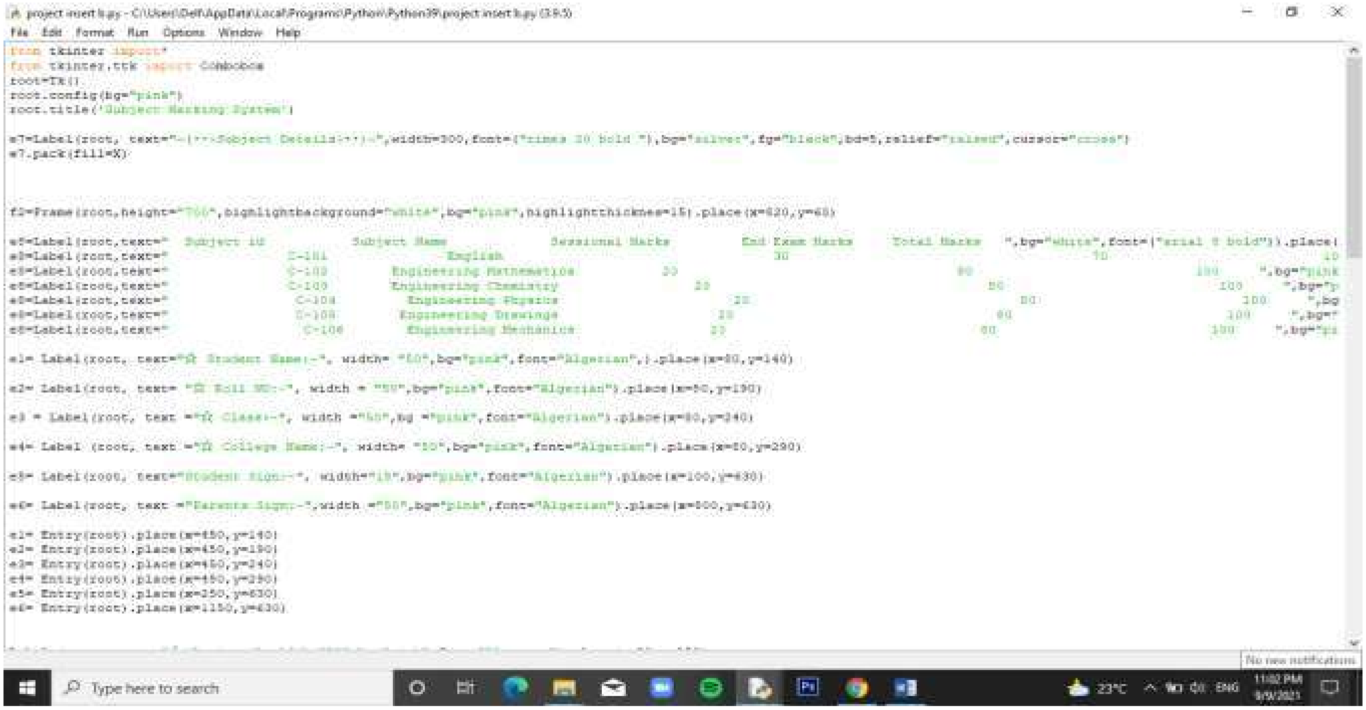
Some Front End Windows





CODING FOR THIS FRONT END WINDOW :-

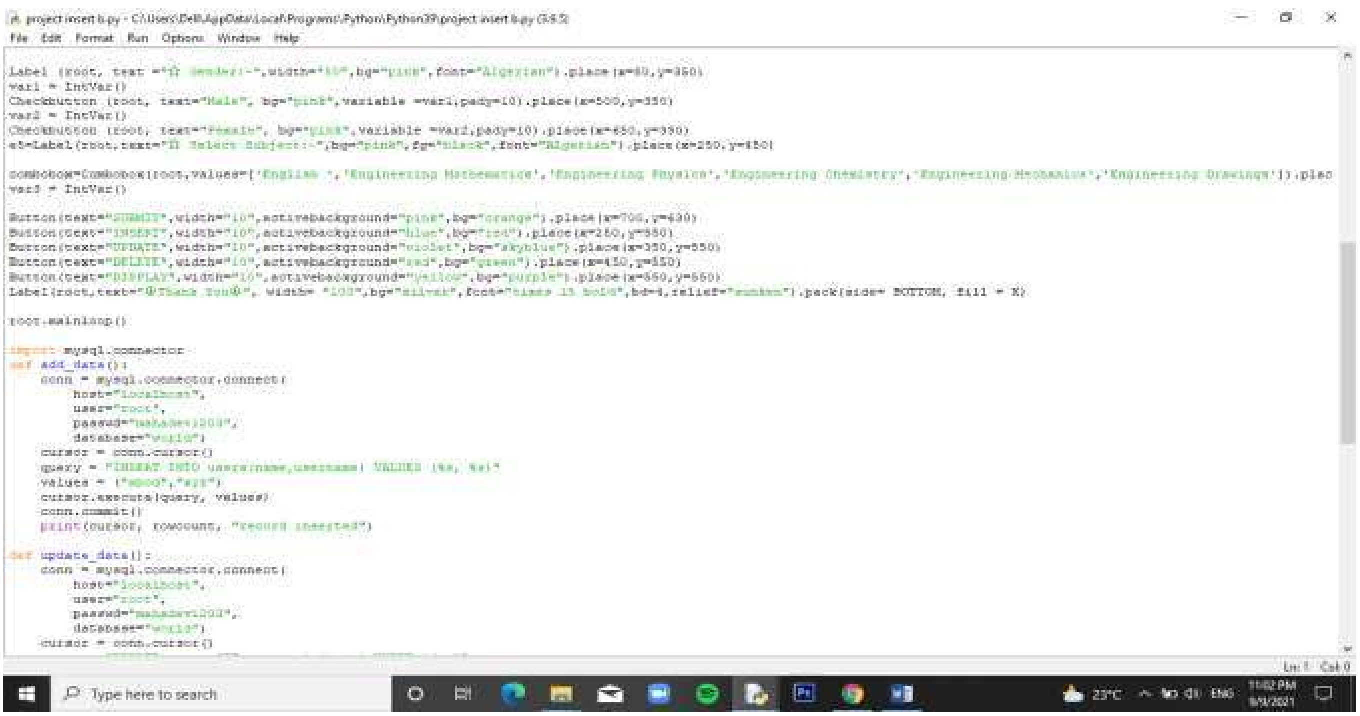
*



project insert b.py - C:\Users\Dell\AppData\Local\Programs\Python\Python39\project insert b.py (3.9.5)

File Edit Format Run Options Window Help

```python
from tkinter import*
from tkinter.ttk import Combobox
root=Tk()
root.config(bg="pink")
root.title('Subject Marking System')

e7=Label(root, text="~(**-Subject Details-**)~",width=300,font=("times 30 bold "),bg="silver",fg="black",bd=5,relief="raised",cursor="cross")
e7.pack(fill=X)



f2=Frame(root,height="700",highlightbackground="white",bg="pink",highlightthicknes=15).place(x=620,y=60)

e5=Label(root,text="  Subject id            Subject Name            Sessional Marks          End Exam Marks      Total Marks   ",bg="white",font=("arial 9 bold")).place(
e8=Label(root,text="                    C-101                  English                                    30                     70                    10
e8=Label(root,text="                    C-102          Engineering Mathematics           20                                 80                   100    ",bg="pink
e5=Label(root,text="                    C-103          Engineering Chemistry              20                                 80                   100    ",bg="p
e8=Label(root,text="                     C-104           Engineering Physics                  20                                80                  100    ",bg
e8=Label(root,text="                     C-105          Engineering Drawings               20                                 80                  100    ",bg="
e8=Label(root,text="                      C-106           Engineering Mechanics              20                                80                  100    ",bg="p

e1= Label(root, text="☆ Student Name:-", width= "50",bg="pink",font="Algerian",).place(x=80,y=140)

e2= Label(root, text= "☆ Roll NO:-", width = "50",bg="pink",font="Algerian").place(x=80,y=190)

e3 = Label(root, text ="☆ Class:-", width ="50",bg ="pink",font="Algerian").place(x=80,y=240)

e4= Label (root, text ="☆ College Name:-", width= "50",bg="pink",font="Algerian").place(x=80,y=290)

e5= Label(root, text="Student Sign:-", width="15",bg="pink",font="Algerian").place(x=100,y=630)

e6= Label(root, text ="Parents Sign:-",width ="50",bg="pink",font="Algerian").place(x=800,y=630)

e1= Entry(root).place(x=450,y=140)
e2= Entry(root).place(x=450,y=190)
e3= Entry(root).place(x=450,y=240)
e4= Entry(root).place(x=450,y=290)
e5= Entry(root).place(x=250,y=630)
e6= Entry(root).place(x=1150,y=630)
```

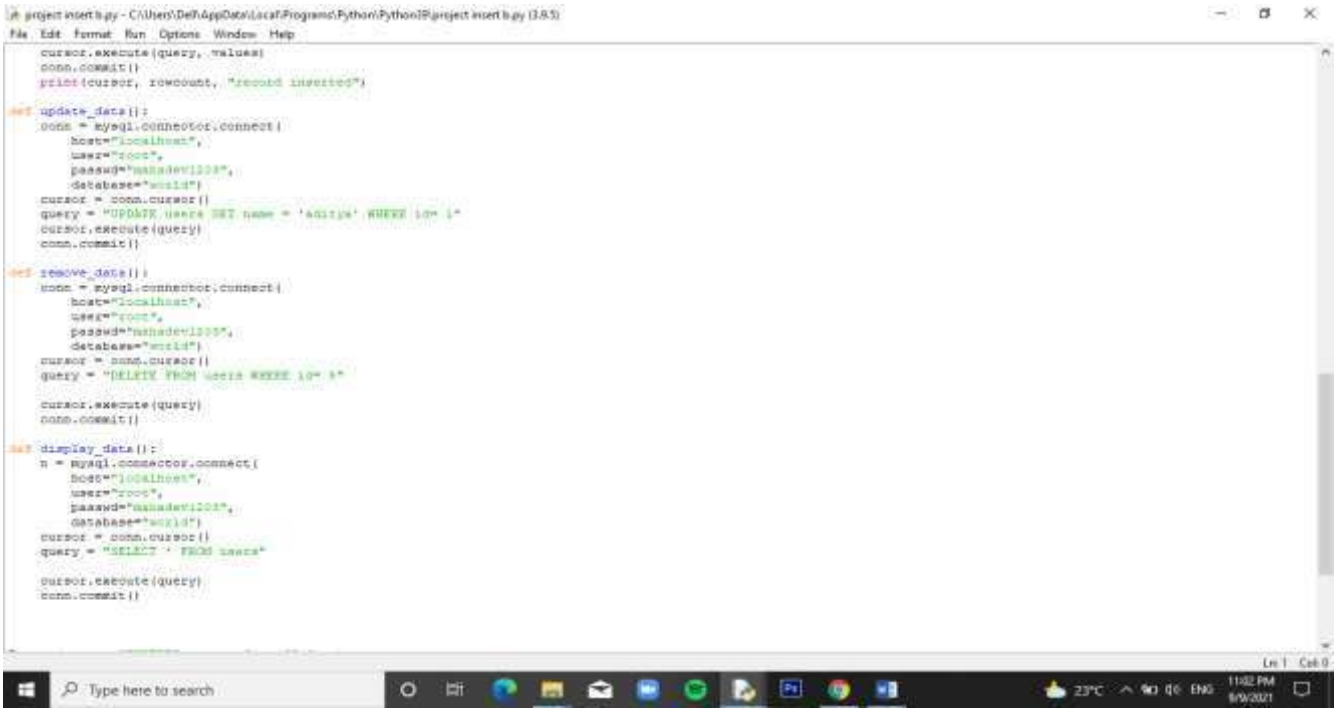


project insert b.py - C:\Users\Dell\AppData\Local\Programs\Python\Python39\project insert b.py (3.9.5)

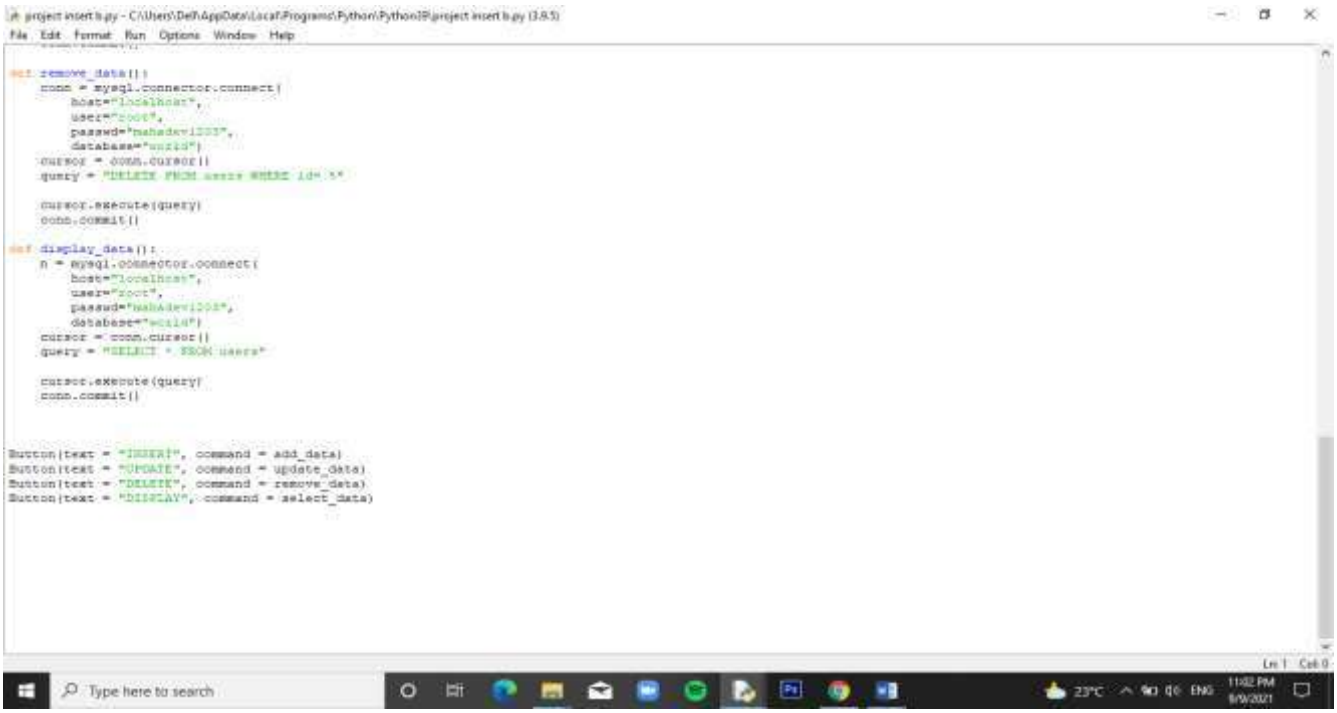
File Edit Format Run Options Window Help

```python
Label (root, text ="☆ Gender:-",width="50",bg="pink",font="Algerian").place(x=80,y=350)
var1 = IntVar()
Checkbutton (root, text="Male", bg="pink",variable =var1,pady=10).place(x=500,y=350)
var2 = IntVar()
Checkbutton (root, text="Female", bg="pink",variable =var2,pady=10).place(x=650,y=350)
e5=Label(root,text="☆ Select Subject:-",bg="pink",fg="black",font="Algerian").place(x=250,y=450)

combobox=Combobox(root,values=['English ','Engineering Mathematics','Engineering Physics','Engineering Chemistry','Engineering Mechanics','Engineering Drawings']).plac
var3 = IntVar()

Button(text="SUBMIT",width="10",activebackground="pink",bg="orange").place(x=700,y=630)
Button(text="INSERT",width="10",activebackground="blue",bg="red").place(x=250,y=550)
Button(text="UPDATE",width="10",activebackground="violet",bg="skyblue").place(x=350,y=550)
Button(text="DELETE",width="10",activebackground="red",bg="green").place(x=450,y=550)
Button(text="DISPLAY",width="10",activebackground="yellow",bg="purple").place(x=550,y=550)
Label(root,text="☺Thank You☺", width= "100",bg="silver",font="times 15 bold",bd=4,relief="sunken").pack(side= BOTTOM, fill = X)

root.mainloop()

import mysql.connector
def add_data():
    conn = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
    query = "INSERT INTO users(name,username) VALUES (%s, %s)"
    values = ("smooc","xyz")
    cursor.execute(query, values)
    conn.commit()
    print(cursor, rowcount, "record inserted")

def update_data():
    conn = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
```

PYTHON INTERNSHIP TRAINING PROGRAM



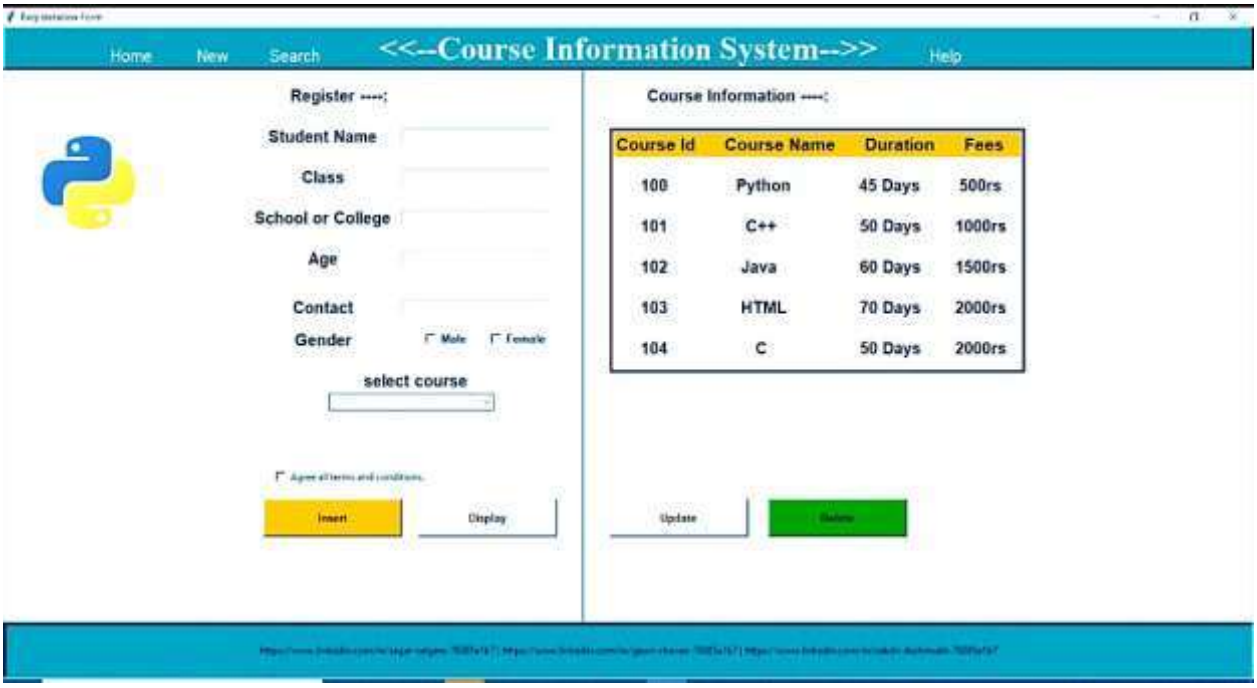
```
project insert b.py - C:\Users\Dell\AppData\Local\Programs\Python\Python39\project insert b.py (3.9.5)
File Edit Format Run Options Window Help
    cursor.execute(query, values)
    conn.commit()
    print(cursor, rowcount, "record inserted")

def update_data():
    conn = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
    query = "UPDATE users SET name = 'aditya' WHERE id= 1"
    cursor.execute(query)
    conn.commit()

def remove_data():
    conn = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
    query = "DELETE FROM users WHERE id= 1"

    cursor.execute(query)
    conn.commit()

def display_data():
    n = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
    query = "SELECT * FROM users"

    cursor.execute(query)
    conn.commit()
```



```
project insert b.py - C:\Users\Dell\AppData\Local\Programs\Python\Python39\project insert b.py (3.9.5)
File Edit Format Run Options Window Help

def remove_data():
    conn = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
    query = "DELETE FROM users WHERE id= 1"

    cursor.execute(query)
    conn.commit()

def display_data():
    n = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="mahadev1203",
        database="world")
    cursor = conn.cursor()
    query = "SELECT * FROM users"

    cursor.execute(query)
    conn.commit()


Button(text = "INSERT", command = add_data)
Button(text = "UPDATE", command = update_data)
Button(text = "DELETE", command = remove_data)
Button(text = "DISPLAY", command = select_data)
```

*

PYTHON INTERNSHIP TRAINING PROGRAM



*

PYTHON INTERNSHIP TRAINING PROGRAM

OUTCOMES

MODULE1

1. Exec functions :-

exec () function is used for the dynamic execution of Python program which can either be a string or object code. If it is a string, the string is parsed as a suite of Python statements which is then executed unless a syntax error occurs and if it is an object code, it is simply executed. We must be careful that the return statements may not be used outside of function definitions not even within the context of code passed to the exec() function. It doesn't return any value, hence returns None.

Syntax:

exec(object[, globals[, locals]])
It can take three parameters:

object: As already said this can be a string or object code
globals: This can be a dictionary and the parameter is optional
locals: This can be a mapping object and is also optional
Now let's see how this function works. In the following code, we have used an object code and executed it using exec() function. We have just taken the object parameter and omitted the other two fields.

Example:

```
prog = 'print("The sum of 5 and 10 is", (5+10))'
exec(prog)
```

Output:
The sum of 5 and 10 is 15

MODULE 2

1. Thread module :-

There are two modules which support the usage of threads in Python:
thread
and
threading

Please note: The thread module has been considered as "deprecated" for quite a long time. Users have been encouraged to use the threading module instead. So,in Python 3 the module "thread" is not available anymore. But that's not really true: It has been renamed to "_thread" for backwards incompatibilities in Python3.

The module "thread" treats a thread as a function, while the module "threading" is implemented in an object oriented way, i.e. every thread corresponds to an object.

Example for a Thread in Python:

```
from thread import start_new_thread

def heron(a):
   """Calculates the square root of a"""
eps = 0.0000001
old = 1
new = 1
while True:
old,new = new, (new + a/new) / 2.0
print old, new
if abs(new - old) <eps:
break
return new

start_new_thread(heron,(99,))
start_new_thread(heron,(999,))
start_new_thread(heron,(1733,))

c = raw_input("Type something to quit.")
```

The raw_input() in the previous example is necessary, because otherwise all the threads would be exited, if the main program finishes. raw_input() waits until something has been typed in.

We expand the previous example with counters for the threads.

```
from thread import start_new_thread

num_threads = 0
def heron(a):
globalnum_threads
num_threads += 1

   # code has been left out, see above
num_threads -= 1
return new

start_new_thread(heron,(99,))
start_new_thread(heron,(999,))
start_new_thread(heron,(1733,))
start_new_thread(heron,(17334,))

whilenum_threads> 0:
pass
```

MODULE 3

1 Database Operations :-

Questions
How can I access databases from programs written in Python?

Objectives
Write short programs that execute SQL queries.

Trace the execution of a program that contains an SQL query.

Explain why most database applications are written in a general-purpose language rather than in SQL.

Example

```
import sqlite3

connection = sqlite3.connect("survey.db")
cursor = connection.cursor()
```

```
cursor.execute("SELECT Site.lat, Site.long FROM Site;")
results = cursor.fetchall()
for r in results:
print(r)
cursor.close()
connection.close()
```

Output

```
(-49.85, -128.57)
(-47.15, -126.72)
(-48.87, -123.4)
```

MODULE 4

1Developing GUIs :-

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.
To create a tkinter app:

Importing the module – tkinter
Create the main window (container)
Add any number of widgets to the main window
Apply the event Trigger on the widgets.
Importing tkinter is same as importing any other module in the Python code. Note that the name of the module in Python 2.x is 'Tkinter' and in Python 3.x it is 'tkinter'.

```
importtkinter
```

There are two main methods used which the user needs to remember while creating the Python application with GUI.

Tk(screenName=None, baseName=None, className='Tk', useTk=1): To create a main window, tkinter offers a method 'Tk(screenName=None, baseName=None, className='Tk', useTk=1)'. To change the name of the window, you can change the className to the desired one. The basic code used to create the main window of the application is:

m=tkinter.Tk() where m is the name of the main window object
mainloop(): There is a method known by the name mainloop() is used when your application is ready to run. mainloop() is an infinite loop used to run the application, wait for an event to occur and process the event as long as the window is not closed.

```
m.mainloop()
importtkinter
m = tkinter.Tk()
'''
widgets are added here
'''
m.mainloop()
```

tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

pack() method:It organizes the widgets in blocks before placing in the parent widget.
grid() method:It organizes the widgets in grid (table-like structure) before placing in the parent widget.
place() method:It organizes the widgets by placing them on specific positions directed by the programmer.
There are a number of widgets which you can put in your tkinter application. Some of the major widgets are explained below:

Button:To add a button in your application, this widget is used.
The general syntax is:
w=Button(master, option=value)
master is the parameter used to represent the parent window.
There are number of options which are used to change the format of the Buttons. Number of options can be passed as parameters separated by commas. Some of them are listed below.

activebackground: to set the background color when button is under the cursor.
activeforeground: to set the foreground color when button is under the cursor.
bg: to set he normal background color.
command: to call a function.
font: to set the font on the button label.
image: to set the image on the button.
width: to set the width of the button.
height: to set the height of the button.

```
importtkinter as tk
r = tk.Tk()
r.title('Counting Seconds')
button = tk.Button(r, text='Stop', width=25, command=r.destroy)
button.
```