6 variable length

```
def performa - operation (values, operation):
operation = input ("Enter the operation (add,
                    subtract, multiply, divide): ")
num-values = int (input ("Enter the number of
                    values: "))
values = [ ]
for i in range (num_values):
    value = float (input (("Enter value {i+1}:"))
    values. append (value)
if operation == 'add':
    result = sum (values)

elif operation == 'ubtract':
    result = values [0]
    for num in values [1:]:
        result == num

elif operation == 'ubtplay':
    result = 1
for num in values:
    result *= num

elif operation == 'divide':
result = values [0]
for num in values [1:]:
    res ult /= num
```

Teacher's Signature:


```
else :
    result = "invalid operation"
    print (f" The result of the operation is :
                        {result}")

perform_arithmethic_operation ()
```

OUTPUT

```
Enter the operation(add, subtract, multiply, divide):add
Enter the number of value : 2

Enter value 1 : 12
Enter value 2 : 12

The result of the operation : 24.0
```

Select option

1. Create File
2. Read File
3. Write File
4. Append File
5. Remove File

1

Enter File Name with Path : D://1.txt

Expt. No. 7 File Handling

Page No. 10
Date 2/7/24

AIM

```
import os

print (" Select options \n 1. create File \n 2. Read
       File \n 3. write File \n 4. Append File \n
       5. Remove File")
num = int (input ())
Fnm = input (" Enter File Name with Path:")
    if os.Path.exists (Fnm):
        count = 1
    if (num == 1 and count == 0):
        open (Fnm," x")
    elif (num == 2 and count == 1):
        FP = open (Fnm," r")
        print (FP.read ())
    elif (num == 3 and count == 1):
        FP = open (Fnm," w")
        data = input (" Enter Text:")
        FP. write (data)
    elif (num == 4 and count == 1):
        FP = open (Fnm," a")
        data = input (" Enter Text:")
        FP. write (data)
elif (num == 5 and count == 1):
        os. remove (Fnm)
```

```
Tuple : (1, 2, 3, 4, 5)
Converted to list : [1, 2, 3, 4, 5]
After push (append): [1, 2, 3, 4, 5, 6]
After pop (Popped element : 6) : [1, 2, 3, 4, 5]
After remove : [1, 2, 4, 5]
After sort : [1, 2, 4, 5]
After reverse : [5, 4, 2, 1]
Converted back to tuple : (5, 4, 2, 1)
```

Expt. No. 8

Page No. 11

Date 18/9/24

AIM

```
def tuple_operations():
    # creating a tuple
    my_tuple = (1, 2, 3, 4, 5)
    print("Tuple:", my_tuple)

    my_list = list(my_tuple)
    print("Converted to list:", my_list)

    my_list.append(6)
    print("After push (append):", my_list)

    popped_element = my_list.pop()
    print("After pop (Popped element:", popped_element
          "):", my_list)

    my_list.remove(3)
    print("After remove:", my_list)

    my_list.sort()
    print("After sort:", my_list)

    my_list.reverse()
    print("After reverse:", my_list)

    my_tuple = tuple(my_list)
    print("Converted back to tuple:", my_tuple)

tuple_operations()
```

vision

Teacher's Signature

Welcome mr / mrs / miss : poornam

Expt. No. 9 Function as object Page No. 12
Date 12/9/24

AIM :

```
def function_as_object(name):
    print('welcome mr/mrs/miss', name)

f = function_as_object('poornam')
```

Expt. No. 10 Page No. 13
Date 12/9/24

AIM:

```
# creating a dictionary

my_dict = {'name': 'varun', 'age': 30, 'city': 'mumbai'}
print ("original dictionary:", my_dict)

print ("Name:", my_did['name'])

my_dict['email'] = 'varun@example.com'
print ("After Adding Email:", my_dict)

my_dict['age'] = 31
print ("After Updating Age:", my_dict)

print ("Removed city:", my_dict.pop('city'))
print ("After Removing city:", my_dict)

print ("Remored Last Item:", my_dict.popitem())
print ("After Removing Last Item:", my_dict)

del my_dict['email']
print ("After Deleting Email:", my_dict)

print ("Name is in the dictionary" if 'name' in
       my_dict else "Name is not in dictionary")

print ("Keys:", list(my_dict.keys()))
print ("values:", list(my_dict.values()))
print ("key-value pairs:", list(my_dict.items()))
```

Teacher's Signature :

Expt. No. _____ Page No. _____
Date _____

AIM :

```
copy_dict = my_dict.copy()
print("copied Dictionary:", copy_dict)

my_dict.clear()
print("After clearing Dictionary:", my_dict)
```

Original Dictionary : {'name' : 'varun', 'age' : 30
'city' : 'mumbai'}
Name : varun
After Adding Email : {'name' : 'varun', 'age' : 30
'city' : 'mumbai', 'email' : varun@exampe.com'}
After Updating Age : {'name' : 'varun', 'age' : 31,
'city' : 'mumbai', 'email' : 'varun@example.com'}
Removed City : Mumbai
After Removing city : {'name' : 'varun', 'age' : 31,
'email' : 'varun@example.com'}
Removed Last Item : ('email', 'varun@exampe.com')
After Removing Last Item : {'name' : 'varun', 'age' : 31}
After Deleting Email : {'name' : 'varun', 'age' : 31}
Name is in the dictionary
keys : ['name', 'age']
values : ['varun', 31]
key - value pairs : [('name', 'varun'), ('age', 31)]
Copied Dictionary : {'name' : 'varun', 'age' : 31}
After clearing Dictionary : {}

EXPT. NO. 11 NAME

Page No: 15 Date: 23/9/24

exception Handling

1
```
x = -1
if x < 0 :
  raise Exception("you are fail!")
```

2
```
x = 1
if not type(x) is int :
  raise Exception("Not Integer!")
```

3
```
x = "Hello"
assert x == "hello"
```

① you are fail

② raise exception("not integer!")
Exception : not integer!

③ assertException : hello

CLASS-ONE

EXPT. NO. 12 NAME

Inheritance

Page No.: 16 Date: 23/9/24

YOUVA

```
class Animal :
def speak (self) :
  Print ("Animal speaking")

class Dog (Animal) :
  def bark (self) :
   Print ("Dog barking")

class Dogchild (Dog) :
   def eat (self) :
   Print ("Dog eating")

d = Dogchild ()
d. bark ()
d. speak ()
d. eat ()
```

Animal speaking
Dog barking
Dog eating

EXPT. NO. 13 NAME

Page No. 17 YOUVA

Date 24/9/24

encapsulation

```
class Base :
  def __init__(self) :
    self.name = 18

class Derived (Base) :
  def __init__(self) :
    super().__init__()
  print("we will call the protected member of
        base class", self.name)
  self.name = 150
  print("we will call the modified protected members
        outside of the class", self.name)

ob1 = Derived()
ob2 = Base()
print("Access object1", ob1.name)
print("Access object2", ob2.name)
```

we will call the protected member of base class 78
we will call the modified protected members outside of the class 150
Access object1 150
Access object2 78

Expt. No. 17

Page No. 13

Date

AIM :

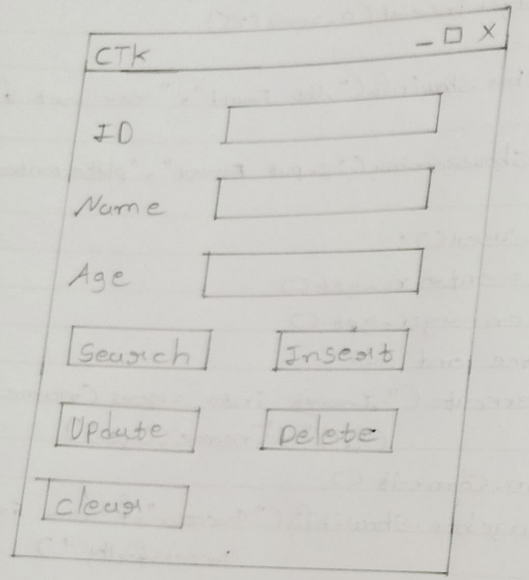
```
# CRUD Operation with GUI

import customtkinter as ctk
import sqlite3
from tkinter import messagebox

conn = sqlite3.connect("crud.db")
c = conn.cursor()
c.execute("""Create table if not exists users(
        id Integer primary key Autoincrement,
        name text,
        age integer
)""")

conn.commit()
root = ctk.CTk()
root.geometry("400x300")

def clear_entries():
    entryi.delete(0, ctk.END)
    entryn.delete(0, ctk.END)
    entrya.delete(0, ctk.END)

def Search_user():
    id = entryi.get()
    clear_entries()
    if id:
        c.execute("Select * from users WHERE id=?", (id,))
        user = c.fetchone()
        if user:
```

Teacher's Signature :

AIM :

```
        entryi.insert(0, id)
        entryn.insert(0, user[1])
        entryc.insert(0, user[2])
    else:
        messagebox.showinfo("Not Found", "user not found")
else:
    messagebox.showwarning("Input Error", "plese enter an ID")

def insert_user():
    name = entryn.get()
    age = entryc.get()
    if name and age:
        c.execute("Insert into users (name, age) values
                  (?,?)", (name, age))
        conn.commit()
        messagebox.showinfo("Success", f"user inserted
                            successfully")
        clear_entries()
    else:
        messagebox.showwarning("Input Error", "please enter
                               all fields")

def update_user():
    id = entryi.get()
    name = entryn.get()
    age = entryc.get()
    if name and age:
        c.execute("Update users Set name=?, age=?
                  Where id=?", (name, age, id))
```

Output :-



Expt. No. _____ Page No. 21

Date _____

AIM :

```
        Conn.Commit()
        Messagebox.showinfo("Success", f"User updated Successfully")
        clear_entries()
    else:
        Messagebox.showwarning("Inputs Error","please enter
                                all fields")

def delete_user():
    id = entryi.get()
    if id:
        C.execute("Delete from users where id= ?", (id))
        Conn.Commit()
        Messagebox.showwarning("Inputs Error","please enter
                                an ID")

label = ctk.CTkLabel(root, text="ID")
label.grid(row=0, column=0, padx=10, pady=10)
entryi = ctk.CTkEntry(root)
entryi.grid(row=0, column=1, padx=10, pady=10)

labeln = ctk.CTkLabel(root, text="Name")
labeln.grid(row=1, column=0, padx=10, pady=10)
entryn = ctk.CTkEntry(root)
entryn.grid(row=1, column=1, padx=10, pady=10)

labela = ctk.CTkLabel
labela.grid(row=2, column=0, padx=10, pady=10)
labela = ctk.CTkEntry(root)
labela.grid(row=2, column=1, padx=10, pady=10)
```

Teacher's Signature : _____

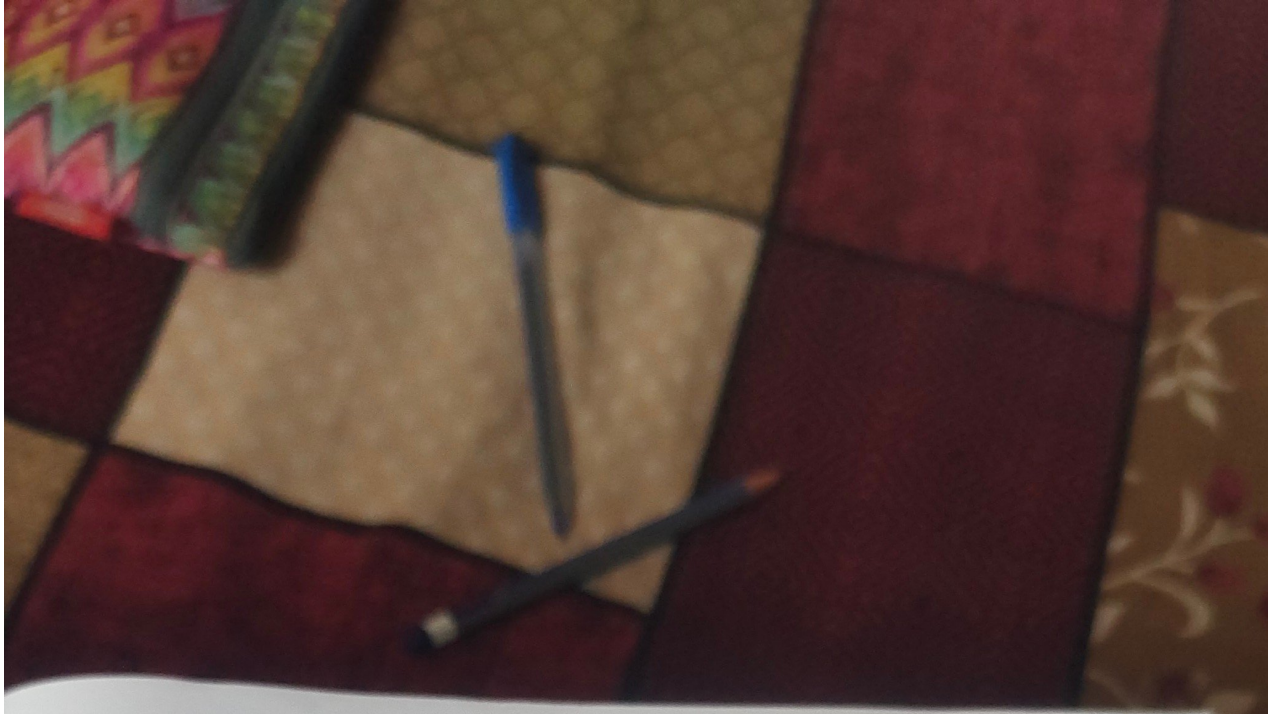
Expt. No. _____ Page No. 22

Date _____

AIM :

```
search b= ctk.CTkButton(root, text="Search",
                        command=search_user)
search b.grid(row=3, column=0, padx=10, pady=10)

insert b=ctk.CTkButton(root, text="Insert",
                       command=insert_user)
insert b.grid(row=[redacted], column=0, padx=10, pady=10)

update b=ctk.CTkButton(root, text="update",
                       command=update_user)
update b.grid(row=4, column=0, padx=10, pady=10)

delete b=ctk.CTkButton(root, text="delete",
                       command=delete_user)
delete b.grid(row=4, column=0, padx=10, pady=10)

clear b=ctk.CTkButton(root, text="clear",
                      command=clear_entries)
clear b.grid(row=5, column=0, padx=10, pady=10)

root.mainloop()
```

Teacher's Signature : _____

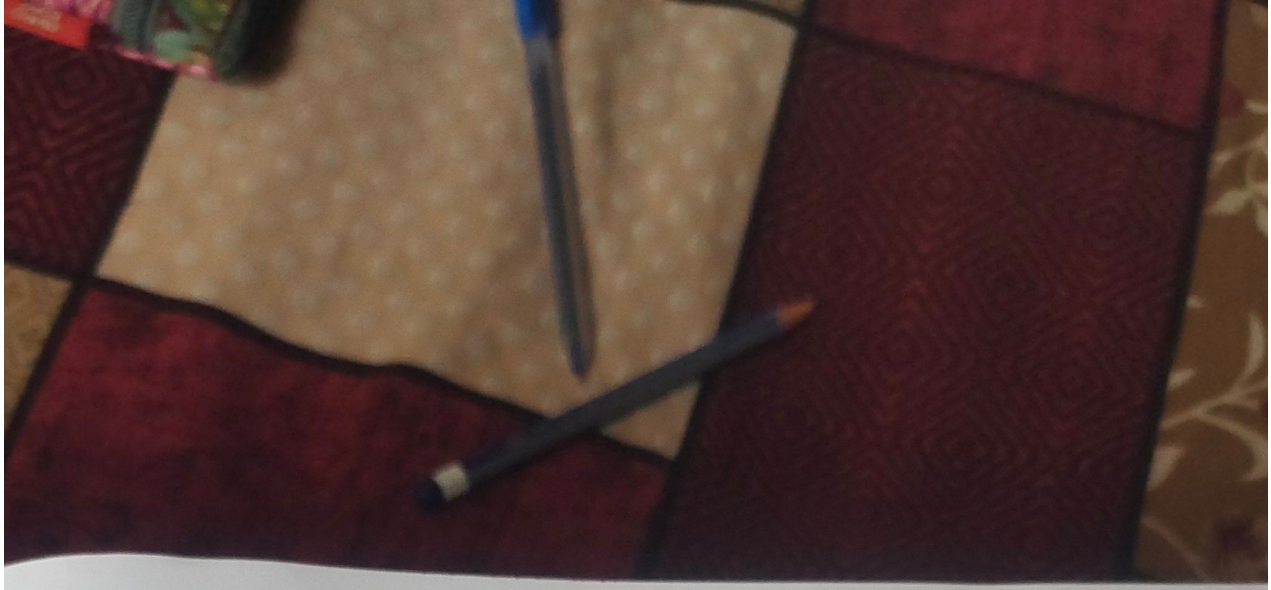


EXPT NO. NAME

Page No. 22
Date 25/9/24

15 knowing id address

```
from urllib.request import urlopen
import re as r
d = str(urlopen('http://checkip.dyndns.com').read())
id = print(r.compile(r'Address: (\d+\.\d+\.\d+\.\d+)'
        .search(d).group(1))
Print(id)
```



EXPT. NO. | NAME

16 Downloading web page

Page No. 23 | Date: 25/9/19 | YOUVA

```
from urllib.request import urlopen
import os
d = str(urlopen('http://checkip.dyndns.com/').read
fnm = "D:/TEST/test.html"
if os.Path.exists(fnm):
    Print("File Already exists")
else:
    open(fnm,"x")
    fp = open(fnm,"w")
    fp.write(d)
```

EXPT. NO. 17 NAME

Page No. 24 YOUVA
Date 26/9/24

Dowloading an image

```
import requests
image_url =
                    ಇಲ್ಲಿ ಚಿತ್ರವಿದೆ
r = requests.get(image_url)
with open("Python_logo.png", 'wb') as f:
    f.write(r.content)
```

Example :

```
import urllib.request
r =
                    ಇಲ್ಲಿ ಚಿತ್ರವಿದೆ
urllib.request.urlretrieve(r, "c:/TEST/Test.jpg")
```

Teacher's Signature:

EXPT NO. 18 NAME

Page No. 25 YOUVA
Date 26/9/24

TCP / IP Server & client

* Server

```
import sockets
tcp_socket = socket.socket (socket.AF_INET, Socket.
                            SOCK_STREAM)
server_address = ('localhost', 81)
tcp_socket.bind (server_address)
tcp_socket.listen (1)
while True:
    print ("waiting for connection")
    connection, client = tcp_socket.accept()
    print ("connected to client ip: {}", format (client))
    while True:
        data = connection.recv (1024)
        print ("Recieved data: {}", format (data))
        if not data:
            break
```

* client

```
import socket
tcp_socket = socket.create_connection (('localhost', 81))
data = str.encode ("Hi, I am client")
tcp_socket.sendall (data)
tcp_socket.close ()
```

Teacher's Signature:

UDP is online

```
C:\client message is: b'hello UDP server'
C:\client IP is: ('127.0.0.1', 62001)
```

EXPT. NO. | NAME: 19 UDP server & client | Page No. 26 | Date: 27/9/24 | YOUVA

* server

```
import socket
localIP = "127.0.0.1"
localPort = 20001
bufferSize = 1024
msgFromServer = "Hello this is UDP server"
bytesToSend = str.encode(msgFromServer)
UDPServerSocket = socket.socket(family=socket.
    AF_INET, type=socket.SOCK_DGRAM)
UDPServerSocket.bind((localIP, localPort))
Print("UDP server is running and listening")
while(True):
    bytesAddressPair = UDPServerSocket.recv from
                                    (buffersize)
    message = bytesAddressPair[0]
    address = bytesAddressPair[1]
    clientmsg = "message from client : {}",format
                                    (message)
    clientIP = "client IP Address : {}", format
                                    (address)
    Print(clientmsg)
    Print(clientIP)
    UDPServer.socket.sendto(bytesTosend, address)
```

Teacher's Signature

(message from server : %y", b' Hello UDP client")

EXPT. NO. | NAME | Page No. 27 | Date | YOUVA

* client

```
import socket
msgfromclient = "Hello UDP Server"
bytesTosend = str.encode(msgfromclient)
serverAddressPort = ("127.0.0.1", 20001)
bufferSize = 1024

UDPclientsocket = socket.socket(family = socket.
                  AF_INET, type = socket.SOCK_DGRAM)
UDPclientsocket.sendto(bytesTosend, serverAddress,
                  Port)
msgfromserver = UDPclientsocket.recvfrom(buffer
                  Size)
msg = "message from server : %y".format
                  (msgfromServer[0])
Print(msg)
```

Server is listening...

EXPT. NO. | NAME

20 File server & client

Page No. 25 | YOUVA

Date 27/9/24

* server

```
import socket
Port = 60000
s = socket.socket()
host = socket.gethostname()
s.bind((host, Port))
s.listen(5)
Print("server is listening...")
while True:
 con, addr = s.accept()
 Print("Got connection From:", addr)
 data = conn.recv(1024)
 Print("Server Received:", repr(data))
 filename = "C:/TEST/1.txt
 f = open(filename, "rb")
 l = f.read(1024)
 while (l):
  conn.send(l)
  Print("sent", repr(l))
  l = f.read(1024)
  f.close()
  Print("Done sending")
  Conn.close()
```

Receiving file....

Receiving Data
File Received

EXPT. NO | NAME

Page No. 29 | YOUVA
Date

* client

```
import socket
s = socket.socket()
host = socket.gethostname()
port = 60000
s.connect((host, port))
s.send("Hello server!".encode())
with open('C:\TEST\received_file1.txt', 'wb')
                                    as f :
    print("Receiving file...")
    data = s.recv(1024)
    if not data :
        break
    f.write(data)
f.close()
print("File Received")
s.close()
```