```
Fileobj = open("abc.txt", "w")
Fileobj.write("Computer Science Subjects".","."\n")
fileobj.write("DBMS\n Python\n Ds\n")
Fileobj.close()

Fileobj = open("abc.txt","r")
str1 = Fileobj.read()
Print("The output of read method :", str1)
File obj.close()
```

output:

```
>>> The output of read method : Computer Science
    Subject
    DBMS
    Python
    Ds
```

```
# readline()
Fileobj = open("abc.txt","r")
str2 = Fileobj.readline()
Print("The output of readline method :", str2)
Fileobj.close()
```

```
>>> The output of readline method : Computer Science Subjects.
```

```
# readlines
Fileobj = open("abc.txt","r")
str3 = Fileobj.readlines()
Print("The output of readlines method", str3)
Fileobj.close()
```

```
>>> The output of readlines method : Computer Science
    subject
    DBMS
    Python
    Ds
```


Sem II - Practical - 1.

- Objective :- Demonstrate the use of different file accessing mode, different attributes and Read Method

Step 1 - Create a file object using open method and use the write accessing mode followed up by writing some contents onto the file and than closing the file.

Step 2 - Now open file in read mode than use read(), readline(), readlines() and store the output in variable and display the contents of variable.

Step 3 - Now use the Fileobject for finding the Name of file, the file mode in which it is opened whether the file is still closed and finally the output of the softspace attribute.

Step 4 - Now open the fileobject in write mode write some another content close subsequently then again open the file object in 'wt' mode that is the update mode and write contents.

Step 5 - Open Fileobject in read mode, display the update written contents and close open again in r+ mode with parameter passed and display the output subsequently.

Step 6 - Now open fileobject in append mode. Open write method, write content, close the fileobject again. Open fileobject in read mode and display the append output.

Step 7 - Open the Fileobject in read mode, declare a variable and perform fileobject. tell method and store the output consequently in variable.

Step 8 - Use the seek method with the argume with opening the fileobject in read mode and closing subsequently.

Step 9 - Open fileobject with read mode also use the readlines method and store the output consequently in and print the same. for counting the length use the for condition statement and display the length.


```
# File attributes
a = Fileobj.name
print ("Name of file (name attribute):", a)
>>> (Name of file (name attribute, abc.txt)

b = Fileobj.close
print ("(close) attribute:", b)
>>> (close) attribute =, True

c = fileobj.mode
print ("file mode", c)
>>> ('file mode', 'r')

d = Fileobj.softspace
print ("softspace", d)
>>> ("Softspace:", 0)
```

```
# w+ mode
Fileobj.open ("abc.txt", "w+")
Fileobj.write ("Ayaan")
Fileobject.close()
```

```
# r+ mode
Fileobj = open ("abc.txt", "r+")
str1 = Fileobj.read (r)
print ("output of r+", str1)
File.object.close()
>>> ('output of r+', 'Ayaan)
```

```
# write mode
Fileobj=open ("abc.txt", "w"
Fileobj.write ("DBMS")
Fileobj.close ()
```

```
# read mode
Fileobj=open("abc.txt", "r"
str2 = Fileobj.read()
print("output of read
      mode", str2)
>>> ('output of read
      mode', 'Ayaan')
```

```
# append mode
Fileobj.open("abc.txt", "a")
Fileobj.write("Data Structure")
File.close()
Fileobject = open("abc.txt", "r")
str3 = Fileobj.read()
Print("output of append mode", str3)
File obj.close()
>>> ("output of append mode:", "Ayaan", "Data Struct
```

```
# tell()
Fileobj = open("abc.txt", "r")
pos = Fileobj.tell()
Print("tell():", pos)
Fileobject.close()
>>> ('tell():', pos)
```

```
# Seek()
Fileobj = open("abc.txt", "r")
str4 = Fileobj.seek(0,0)
Str8 = Fileobj.read(10)
Print("the begining of file is =", str8)
```

PRACTICAL 2.

Aim: Demonstrate the use of iterables and iter

Theory: In python iterator is an object which implements iterator class which has 2 methods namely -- iter() -- and nett()--

Q1. Write a program using iterable objects for displaying the odd numbers in range 1 to 10.

Algorith
Step 1 - Define a iter() with argument and initia the value and return the value.

Step 2 - Define the next() with an argument a compare the upper limit by using a conditional stat

Step 3 - Now create an object of the given, class and pass the object in the iter method.

Code

```
class odd:
    def __iter__(self):
        self.num = 1
        return self
    def next(self):
        if self.num <=10:
            num = self.num
            self.num += 2
            return num
        else:
            raise StopIteration
```

```
>>> y = odd()
>>> z = iter(y)
>>> z.next()    or    next(z)
1
>>> z.next()
3
>>> z.next()
5
>>> z.next()
7
>>> z.next()
9
>>> z.next()
```

– wise
Stop Iteration

Code:

```
class power:
    def __iter__(self):
        self.p = 0
        return self
    def next(self):
        if self.p <= 10
            num = self.p
            self.p += 1
            po = 2 ** num
            print("2**", self.p-1, "=", po)
            return po
        else:
            raise StopIteration
```

```
>>> p = power()
>>> x = iter(p)
>>> x = next()
2** 0 = 1
>>> x.next()
2** 1 = 2
>>> x.next()
2** 2 = 4
```


2. Write a program using an iterator for calculating the power of a given number. For instance number entered is 2 then value calculated should be $1, 2^1, 2^2, 2^3, 2^4$

Algorithm:

Step 1: Define iter() with argument and initialize value and return the value.

Step 2: Now define next() with an argument and compare the upper limit by using conditional statement.

Step 3: Now create an object of the given class and pass the object in the iter method.

3. Write a program using iterable concept to find factorial of number in range 1 to 10:

Algorith

step 1 - Define a iter() with argument and initialize the value and return the value.

step 2 - Define the next() with an argument and compare the upper limit by using a conditional statement.

step 3 - Now create an object of the given class and pass this object in the iter method.

24. Write a program using iterable concept to display multiple of 2 in range 1 to 10.

Algorith :

Step 1 : Define a iter () with argument and initialize the value and return the value.

Step 2 : Define the next () with an argument and compare the upper limit by using a condition statement.

Step 3 : Now create an object of the given class and pass this object in the iter method.

```
# Code
class fact :
    def __iter__(self) :
        self.f = 1
        return self
    def next(self) :
        if self.f <= 10:
            num=self.f
            self.f += 1
            fac = 1
            for i in range ( 1, num+1) :
                fac = fac * i
            print ( self.f -1, "!=", fac)

        else :
            raise StopIteration
```

```
>>> f = fact()
>>> x = iter(f)
>>> x.next()
1! = 1
>>> x.next()
2! = 2
>>> x.next()
3! = 6
```

Program to display (Prac 2)

Code:

```
class mult :
    def __iter__ (self) :
        self.m = 1
        return self
    def next(self):
        if self.m <= 10 :
            num = self.m
            self.m += 1
            table = 2 * num
            print("2 *", num, "=", table)
        else:
            raise StopIteration
```

```
>>> m = mult()
>>> x = iter(m)
>>> x.next()
2 * 1 = 2
>>> x.next()
2 * 2 = 4
>>> x.next()
2 * 3 = 6
>>> x.next()
2 * 4 = 8
```


Practical 3:

Aim: Demonstrate the use of exception Handling.

Theory: An exception is an event which occurs during execution of program which disrupts the normal flow of program. Thus a exception represents object which represents an error. This object is derived from given class and when the python script raises an exception it must be handeled immidiately otherwise it will terminate and close the program.

2. Write a program to check the range of the age of the students in given class and if age does not fall in given range use value error exception otherwise return the valid no.

Algorith:

Step 1: Define a function which will accept the age of the stude from standard input

Step 2 : Use if conditional to check whether the input are falls in range and so return the age else use value error exception.

Step 3 : Define the while loop to check whether the boolean expression holds true. Use the try block to accept the age of student and terminate looping condition

Step 4 : Use except with value error and print message not a valid range.

Q2. Write a program to check whether the number in given class and if the number is floating point use Valueerror as exception for the given input.

Algorith :

Step 1 : Use try block and accept the input using input () and convert it into integer datatype and subsequently terminate the block.

Step 2 : Use the except block with exception as Valueerror and display appropriate message in suspicious code is part of the try block.

(Q 2)

code :

```
def accept-age()
    age = int(input("Enter your age"))
    if age > 30 or age < 16:
        raise ValueError
    else:
        print("your age is", age)

valid = False
while not valid:
    try:
        age = accept-age()
        valid = True
    except ValueError:
        print("Your age is not in range")
```

>>> Enter your age : 15
your age is not in range

Enter your age : 32
your age is not in range

Enter your age : 18
your age is 18.

(Q2)

Code

```
while True:
    try:
        a = int(input("Enter a number:"))
        print("valid number")
        break
    except ValueError:
        print("Not a valid Number! try again")
```

```
>>> Enter a number: 17.2
Not a valid number! Try again
Enter a number: 17
Valid Number.
```


Q3. Write a program to demonstrate use of zerodivision error

Algorith:

Step 1: Use the try block and accept the input using input() & then convert it into integer datatype.

Step 2: Define a function with 2 parameters to divide the numbers given by user.

Step 3: Define while loop to check whether the boolean expression holds true

Step 4: Use except with zerodivision error and print the message.

(03) Code :

```
def divide(a,b):
    ans = a/b
    return ans
while True:
    try:
        a = int(input("Enter first number."))
        b = int(input("Enter Second number:"))
        ans = divide(a,b)
        print("division of", a, "and", b, "is", ans)
        break
    except ZeroDivisionError:
        print("Error!")
```

```
>>> Enter first number : 1
>>> Enter second number : 1
    Division of 1 and 1 is 1
>>> Enter first number : 1
>>> Enter second number : 0
Error!
```

19/12/19

```
import re
string = " hello 1234 abc 567 "
result = re.findall ("\d+", string)
result2 = re.findall ("\D+", string)
print (result)
print (result 2)
```

output :

```
>>> ['1234', '5 6 7']
>>> [' hello', ' abc']
```

PRACTICAL 4 :

Aim : Demonstrate the use of regular expression

Theory : Regular expression represents the sequence of characters which is mainly used for finding and replacing the given pattern in a string and for this we import "re" module.
Regular expression involves :
1) Searching a given string
2) Finding a string
3) Breaking string into substring
4) Replacing part of string.

1] Write a regular expression reggreation numeric and alphabetic values from given string.

Algorithm :

Step 1 : Now display string and pattern in findall and display the output.

Step 2 : \d is used for matching all decimal digits whether \D is used to match non-decimal digits

Q2 Find matching string at Begining of a sequence

Algorithms :-

Step 1 :- Import re module and apply a string.

Step 2 :- Use search() with \A function and string as parameters to search for a parameter string.

Step 3 : Now display the output.

Step 4 :- Now, we use conditional statement for user to know whether the string is found or not.

```
import re
string = "python is important"
result = re.search ("\A Python", string)
print (result)
if result :
        print (" match found ")

    else :
        print (" match not found ")
```

output :

```
>>> (rematch object : span (0, 6)
                match = "python" >

>>> match found
```

```
import re
li = [" 99202 34212", "8412322334",
      "72913  19872", "92531210"]

for element in li:
    result = re.match("[8-9]{1}[0-9]{9}", element)
    if result:
        print("Correct mobile no")
        print(result.group(1))

    else:
        print("incorrect mobile no")
```

```
>>> Correct mobileno
99202 34212
```

Q3 Program to check whether given mobile no. starts with 8 or 9, the total length of digit should be almost to

Algo:

1. Import re module and apply a string of module no. 8
2. Now use for conditional statement to find if the no starts with 8 or 9 and total length should be 10 else match() inside for statement to find the match is given string
3. Use of conditional statement to know whether we have a match or not the o/p we have use of group() to display incorrect mobile no.

Q4. Expression for extracting a word from given string along with space character in b/w the word & subsequently extract word without space character

Algo:

Step 1: Import re module and apply a string

Step 2: Use findall() to extract a word from a given string.

Step 3: Use "/W*" to extract word along with space and "W+" to extract word without space.

```
import re
string = "Python is important"
result 1 = re.findall ("\W*", string)
result 2 = re.findall ("\W*", string)
print (result 1)
print (result 2)
```

output:

```
>>> ['Python', ' ', 'is', 'important']
['Python', 'is', 'important']
```

5.
```
import re
s = "python is impo
r = re.findall("\n/wt", string)
r1 = re.findall("\wf", string)
print(r)
print(r1)
>>> ['python']
>>> ['Important']
```

6.
```
import re
s = "Amit 201 24-12-2019"
s = re.findall ("\D{2}-\d{2}-\D{4}", string)
print (result)
>>> ['24-12-2019']
```

25. Write a reg exp for extracting first and last word from string.

Algo.

1. Import re. module and apply a string.
2. Use findall() in which use "\w" as one parameter to find first word of string then use "\w*" as parameter for last word.

26. Exp for extracting the date in format dd-mm-yyyy by using findall() where string has following format Amit 201 24-12-2019.

Algo

Step 1: Import re module and apply string.

Step 2: Use findall method and use '\d{2} \d{2} - \d{4}) as an parameter.

Step 3: Display output.

22. Regular exp for extracting
① username from email id
② hostname from email id
③ both username & hostname from email id.

Algo
Step 1 - Import re module and apply a string

Step 2 - Use findall() to find username hostname and both email id.

Step 3 - Use "^\w+" for username
Use "\w+\w+$" for hostname
Use "[\w\-]+" for both
parameter in findall().

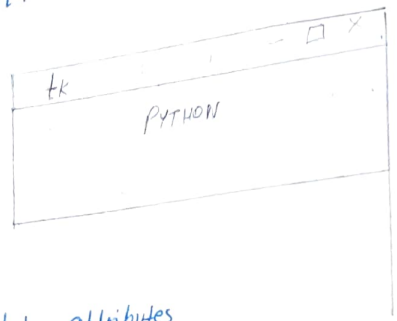
```
import re
s = "abc@tcsc.edu"
r = re.findall("^\w+", string)
r1 = re.findall("\w+\w+$", string)
r2 = re.findall("[\w\-]+", string)
print(r)
print(r1)
print(r2)
```

```
>>> ['abc']
>>> ['tcsc.edu']
>>> ['abc', 'tcsc.edu'].
```

Creating parent window.

```
from Tkinter import *
root = Tk()
l = label(root, text = "python")
l.pack()
root.mainloop()
```

output:



2 : label, attributes

```
From Tkinter import *
root = Tk()
l = label (root, text = " python")
l.pack()
l1 = label (root, text = "CS", bg = "grey", fg = "black", font="1
l1.pack (side = LEFT, padx = 20)
l2 = label (root, text = "CS", bg = "light blue", fg = "black", font=20
l2.pack (side = LEFT, pady = "30")
l3 = label (root, text = "CS!" bg = "yellow", fg = "black", font="
l3.pack (side=Top, ipadx=40)
```


PRACTICAL 5 : - A

Topic - Gui Components

Step 1 - Use the tkinter library for importing features of text widget.

step 2 - Create an object using the Tk()

Step 3 - Create a variable using the widget label and use the text method.

Step 4 - Use the mainloop() for triggering of the corresponding above mentioned events.

2 .

Step 1 : Use the tkinter library for importing the Feature of the text widget.

step 2 : Create a variable from the text method and position it on the parent window.

Step 3 : Use the pack() along with the object created from the text() and use the parameter.

1. Side = LEFT, padx = 20
2. Side = LEFT, pady = 30
3. Side = TOP, ipadx = 40
4. Side = Top, ipady = 50

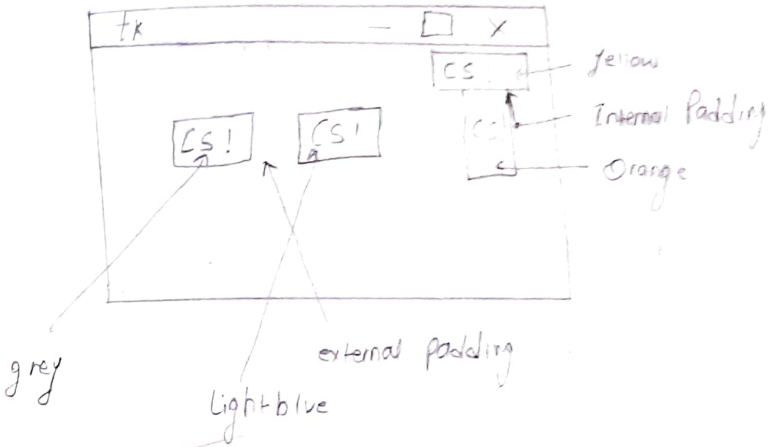
Step 4: Use the mainloop() for the triggering of the corresponding events.

Step 5: Now repeat above steps with the label() which takes the following arguments.

1) Name of the parent window.
2) Text attribute which defines the string.
3) The background colour (bg).
4) The foreground fg and than use the pack() with a relevant padding attributes.

```
L4 = label(root, text="CS!", bg="orange", fg="black", font="10")
L4.pack(side=TOP, ipady=50)
root.mainloop()
```


output.



Radio button

```
from tkinter import *
root = Tk
root.geometry ("500x500")
def select():
    Selection = "You just selected " + str(var.get())
    t1 = label(text = Selection, bg = "white", fg = "green")
    t1.pack(side = Top)
var = stringvar()
l1 = Listbox()
l1.insert(1, "list 1")
l2.insert (2, "list 2")
l1.pack(anchor = N)
r1 = RadioButton (root, text = "option 1", variable = var,
     value = "option 1", command = select)
r1.pack(anchor = N)
r2 = RadioButton (root, text = "option 2", variable = var,
     value = "option 2", command = select)
r2=pack(anchor = N)
root.mainloop()
```


Aim GUI Components.

1 :

Step 1 : Import the relevant methods from the tkinter library. Create an object with the parent window.

Step 2 : Use the parent window object along with the geometry() declaring specific pixel size of the parent window.

Step 3 : Now define a function which tells the user about the given selection mode from multiple option available

Step 4 : Now define the parent window and define the option with control variable.

Step 5 : Use the listbox() and insert options on the parent window along with the pack() with specifying anchor attribute.

Step 6 : Create an object from radio button which will take following attributes and parent window object, text variable which will take the values option no 1,2,3... variable argument, corresponding value and Trigger the function declared

Step 7. Now call the pack() for radio object so create and specify the argument using anchor attribute.

Step 8: Finally make use of the mainloop() along with parent object.

2:

Step 1: Import relevant methods from the Tkinter library.

Step 2: Create a parent object corresponding to the parent window.

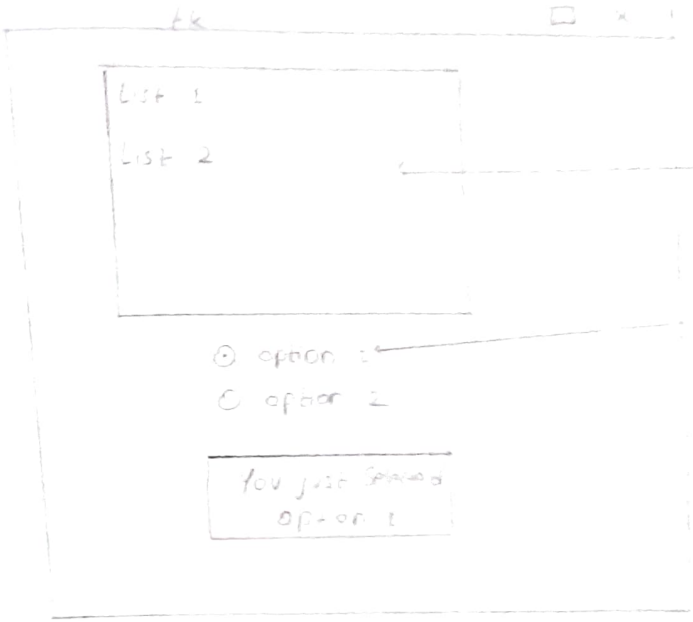
Step 3: Use the geometry() for laying of the window

step 4: Create an object and use the scrollbar()

step 5: Use the pack() along with the scrollbar object with side and fill attributes.

Step 6: Use the mainloop with the parent object.

Output of # 1.

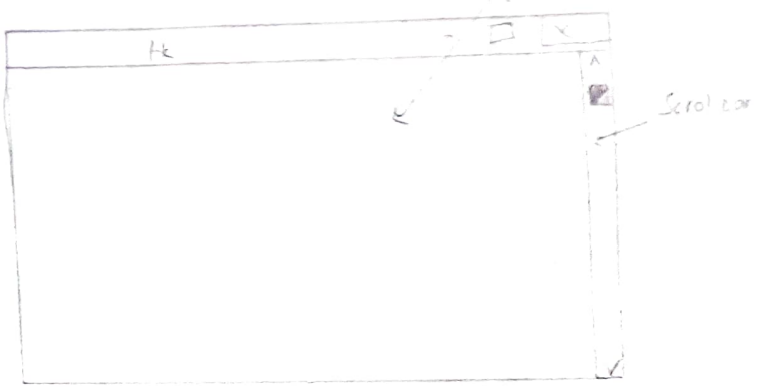


2

scrollbar()

```
from tkinter import *
root = Tk()
root.geometry("500x500")
s = scrollbar()
s.pack(side = "right", file = "y")
root.mainloop()
```

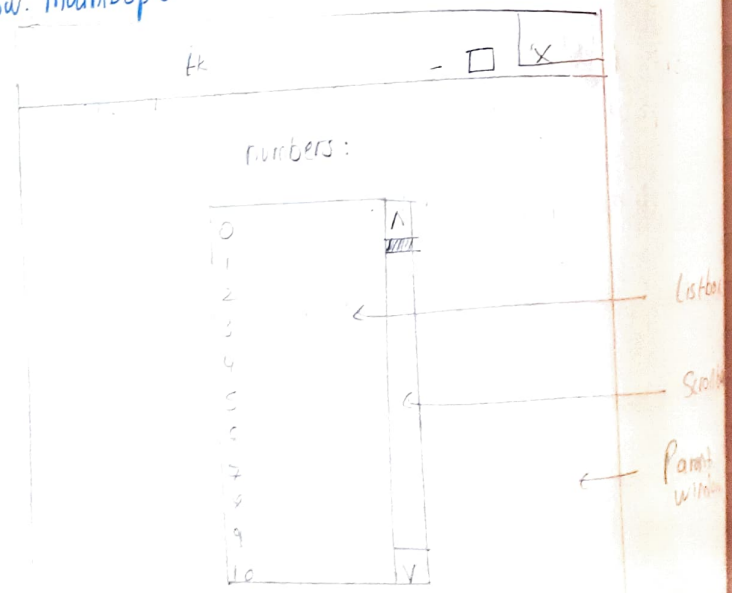
output:



#3:

```
# using frame widget
from tkinter import *
window = Tk()
window.geometry("680x500")
Label(window, text="number:").pack()
Frame = Frame(window)
Frame.pack()
ListNodes = Listbox(frame, width=20, height=20,
                    font=("Times New Roman", 10))
ListNodes.pack(side="left", fill='y')
Scrollbar = Scrollbar(frame, orient="vertical")
Scrollbar.config(command=ListNodes.yview)
Scrollbar.pack(side="right", fill="y")
for x in range(100):
    listnodes.insert(END, str(x))
window.mainloop()
```

output:



#3.

Step 1: Impat the relevant libranes from tkinter method.

Step 2: Create an corresponding object of the parent window.

Step 3: Vse the geometry manager with pixel size (680x500) or any other suitable pixel value.

Step 4: Use the label widget along with the parent object created and subsequently use the pack method.

Step 5: Use the frame widget along with the parent object created and use the pack method.

Step 6: Use the listbox method along with the attributes like width, height font. Do create a listbox method's object use pack() for the same.

Step 7: Use the scrollbar() with an object use the attribute of vertical. Then configure the same object created from the scrollbar() and use the pack().

Step 8: Trigger the events using mainloop.

#4.

Step 1: Import relevant methods from tkinter library.

Step 2: Define the object corresponding to parent window and define the size of parent window in terms of no. of pixels.

Step 3: Now define the frame object from the method and place it on the parent window.

Step 4: Create another frame object termed as the the left frame and put it on the parent window

Step 5: Similarly define the right frame and subsequently define the button object placed onto the given frame with the attribute as text, active background and foreground.

Step 6: Now use the pack() along with the side attribute.

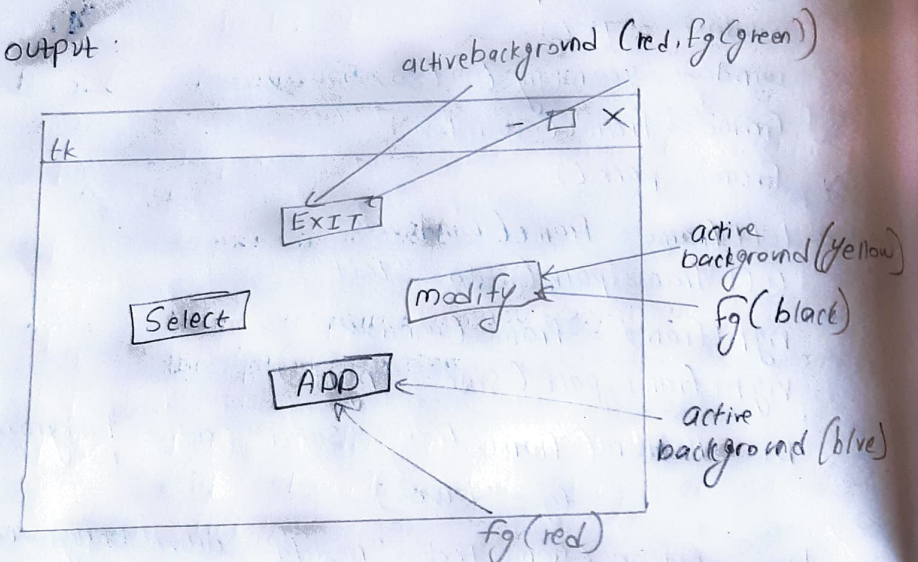
Step 7: Similarly create the button object corresponding to the MODIFY operation put it into frame object on side = "right".

#4:

```
from tkinter import *
window = Tk()
window.geometry("680x500")
frame = Frame(window)
frame.pack()
leftframe = Frame(window)
leftframe.pack(side="left")
rightframe = Frame(window)
rightframe.pack(side="right")
b1 = Button(frame, text="Select", activebackground="red",
            fg="blue")
b2 = Button(frame, text="modify", activebackground="yellow",
            fg="black")
b3 = Button(frame, text="ADD", activebackground="blue",
            fg="red")
b4 = Button(frame, text="Exit", activebackground="red",
            fg="green")
b1.pack(side="left", padx=20)
b2.pack(side="right", padx=30)
b3.pack(side="bottom", pady=20)
b4.pack(side="top")
```

4.

output:



step 8: Create another button object and place it on the Right frame and label button as add.

Step 9: Add another button and put it on the top of frame and label it as Exit.

step 10: Use the pack() simultaneously for all the objects and finally use the mainloop ().

Practical 5-C

Aim: GUI Components.

Step 1: Import relevant method from tkinter library

Step 2: Impart TkMessage Box.

Step 3: Define a parent window object along with the parent window.

Step 4: Define a function which will use tkmessagebox with showinfo method along with info window attribute.

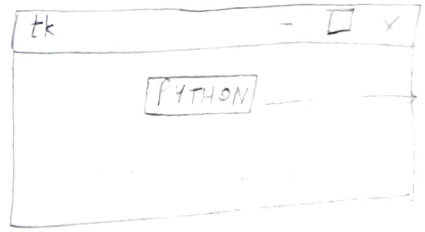
Step 5: Declare a button with parent window object along with the command attribute.

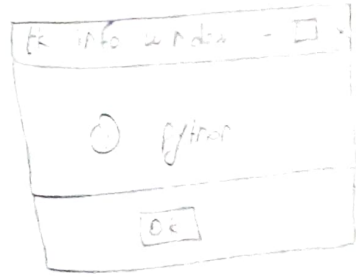
Step 6: Place the button widget onto the parent window and finally call mainloop() for triggering of the events called above.

message box

```
from Tkinter import *
  import tk MessageBox
root = Tk()
def function():
    tk MessageBox.showinfo("info window","python")
    b1 = Button(root, text = "python", command = funct
    b1.pack()
root.mainloop()
```


output:





```
# Multiple window
# Different button (Relief())

from Tkinter import *
root = Tk()
root.minsize(800, 300)
def main():
    top = Tk()
    top.config(bg = "black")
    top.title("HOME")
    top.minsize(300, 300)
    l = Label(top, text = "San FRANCISCO
    \n Places of Interest: \n golden gate
    Bridge \n Lombard street \n chinatown
    \n Coit Tower")
    l.pack()
    b1 = Button(top, text="next", command = second)
    b1.pack(side = RIGHT)
    b2 = Button(top, text= "exit", command = terminate)
    b2.pack(side = LEFT)
    top.mainloop()
```


Step 1: Import the relevant method from tkinter library along with parent window object declared.

Step 2: Use parentwindow object along with minsize function for window size.

Step 3: Define a function main, declare parent window object and use config(), title(), minsize(), Label() as well as button() and use pack() & mainloop() simultaneously.

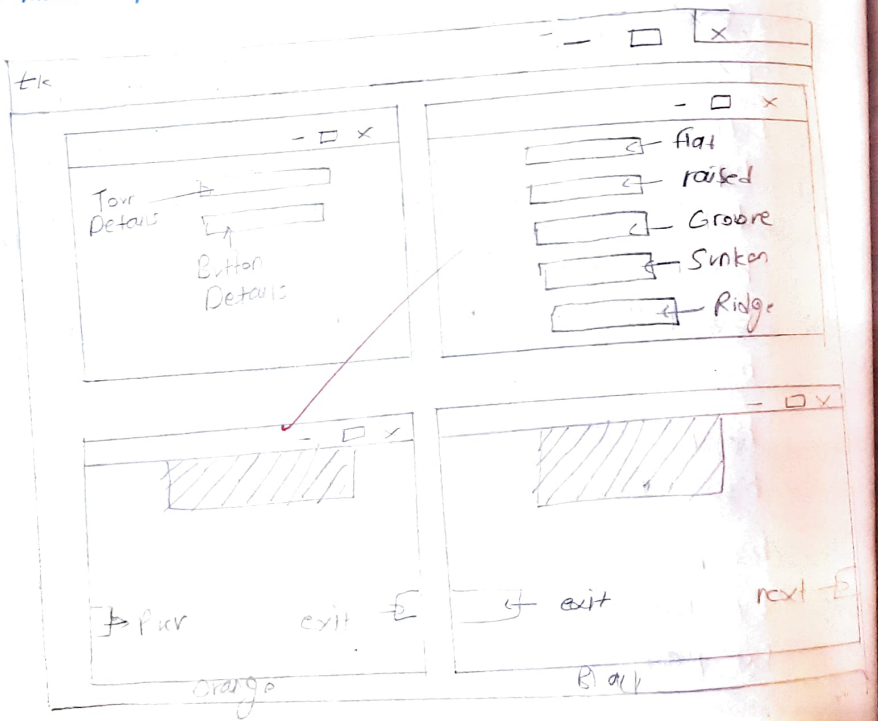
Step 4: Similary define the function second and use the attribute accordingly.

Step 5: Declare another function button along with parent object and declare button with attribute like FLAT, RIDGE, GROOVE, RAISED, SUNKEN along with relief widget.

Step 6: Finally call the mainloop() for event driven programming.

```
def second():
    top2 = Tk()
    top2.config(bg = "orange")
    top2.title("About us:")
    top2.minsize(300,300)
    l = label(top2, text="Created by: Ayaan Hyder In for more
                details contact official account")

    l.pack()
    b3 = Button(top2, text= "prev", command = main)
    b3.pack(side = LEFT)
    b2 = Button(top2, text = "exit", command = terminate)
    b2.pack(side = RIGHT)
    top2.mainloop().

def button():
    top3 = Tk()
    top3.geometry("300x300")
    b1 = Button(top3, text = "flat button", relief = FLAT)
    b1.pack()
    b2 = Button(top3, text = "groove button", relief = GROOVE)

    b2.pack()
    b3 = Button(top3, text = "raised button", relief = RAISED)

    b3.pack()
    b4 = Button(top3, text = "sunken button", relief = SUNKEN)
    b4.pack()
```

```
bs = Button (top3, text = "ridge button", relief = RIDGE,
    relief = RIDGE)
top3. mainloop()
 def terminate ():
    quit()
bs = Button (root, text = "TOUR DETAILS", Command = main)
bs. pack()
b6 = Button( root, text = "Button Details" (command = button
b6. pack()
root. mainloop().
```



Aim : GUI Components

Step 1 : Import relevant methods from tkinter library

Step 2 : Create parent window object and use the config method along with background color attribute specified.

Step 3 : Define a function finish with the MessageBox widget which will display a message i.e a warning message & subsequently terminates the program.

Step 4 : Define a function info use a listbox widget along with the objelt of the same. Use the listbox object along with insert method and insert the same and finally use the grid() with ipadx attribute.

Step 5 : Define a function about us with babel widget and text attribute and subsequently use the grid().

Step 6 : Use photoimage widget with file and filename with gif attribute.

Step 7 : Create a frame object along with the frame along with parent window object height and width specified and subsequently use the grid() with row and column attribute specified.

Step 8 : Similarly create another frame object as declared by step 7.

Step 9 : Create another object and use the subsample (5/4).

Step 10 : Use label widget along with the frame object relief attribute and subsequently else use grid().

Step 11 : Now create button object dealing with different sections of frame.


```
from
root = Tk()
root = config(bg = "grey")
def finish():
    messagebox.askokcancel("Warning", This will end the
                                        program")
    quit()
def info():
    list1 = listbox()
    List1.insert(1, "Co.Name: Apple")
    List1.insert(2, "Products: iphone")
    list1.insert(3, "Language: Swift")
    List1.insert(4, "OS: IOS")
    list1.grid(ipadx = 30)
def aboutus():
    list2 = Label(text = "About us")
    List2.grid(ipadx = 30)
    List3 = label(text = "Steve jobs theater March 2020")
    List3.grid(ipadx = 24)
P1 = PhotImage(file = "download.gif")
f1 = Frame(root, height = 35, width = 5)
F1.grid(row = 1, column = 0)
f2 = frame(root, height = 250, width = 500)
F2.grid(row = 1, column = 1)
```

```
P2 = p1.subsample(5,4)
L1 = label(f1, image=p2, relief=FLAT)
l1.grid(row=1, column=0, padx=20, pady=15)
l2 = label(f2, image=p1, relief=SUNKEN)
l2.grid(padx=25, pady=10)
b1 = Button(f1, text="Information", relief=SUNKEN,
      command=info)
b1.grid(row1, column=0)
b2 = Button(f1, text="About us", relief=SUNKEN,
      command=aboutus)
b2.grid(row=1, column=2, padx=5)
b3 = Button(f1, text="EXIT", relief=RAISED,
      command=finish)
b3.grid(row=2, column=1, ipadx=15)
root.mainloop().
```

Practical - 5 - e.

Aim: GUi Components

#1. SpinBox:

Step 1: Import relevant ()'s from the tkinter library.

Step 2: Create parent window object along with tk().

Step 3: Create an object from spinbox method and use attribute parent window object from and to attributes

Step 4: Subsequently call the pack method along with spinbox object and call the mainloop ().

#2. PANED WINDOW

Step 1: Import relevant methods from tkinter library also create a parent window object

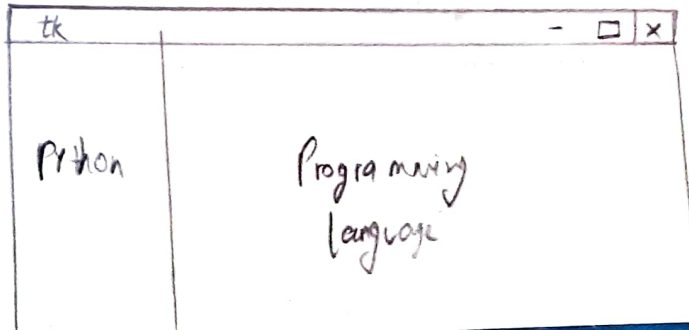
Step 2: Create an object along with paned window and subsequently use the pack() along with attributes like fill and expand.

```
from Tkinter import *
master = Tk()
s = Spinbox(master, from_=0, to=10)
s.pack()
master.mainloop()
```


output:

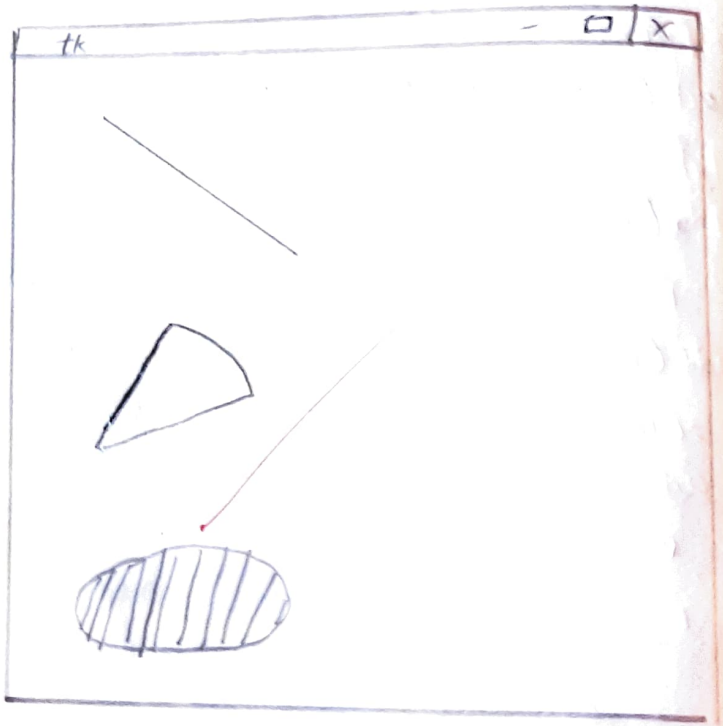

```
from Tkinter import *
root = Tk()
p = panedwindow()
p.pack(fill=BOTH, expand=10)
l = label(p, text="PYTHON")
p.add(L)
p1 = panned Window(p, orient=VERTICAL, bg="black")
p.add(p1)
l1 = Label(p1, text="programming, language")
p1.add(l1)
root.mainloop()
```

output:



#3 From tkinter import *

```
root = Tk()
c = Canvas (root, height = 100, width = 200, bg = "
carc = c. create_arc ( 10, 20, 30, 40, start = 10,
                       extent = 50, fill = "black")
coval = c. create_oval (20, 31, 45, 50, fill = "
cline = c. create_line (10, 15, 30, 20, fill = "blue")
c. pack()
root. mainloop()
```



Canvas

Step 3: Create an label object along with label use parent window object, orient and background colour

Step 4: Similarly create another label method and use the add () subsequently.

Step 5: Finally called the mainloop() for event trigreation.

#3: Canvas:

Step 1: Import relevant ()'s from tkinter library declare a parent window object

step 2: Create a canvas object along with the canvas() with attributes parent window object, height, width background colour.

Step 3: Use Create_arc () along with canvas object declared along with start and extent and co-ordinates.

Step 4: Similarly for oval and line use the pack() and called mainloop () for event driven programming.

Practical 6.

Aim: Demonstrate the use of GUI by creating a human face and converting celsius into fahrenheit.

Q1. Write a program to draw human face using GUI.

Algorithm:

Step 1: Import relevant methods from tkinter library.

Step 2: Create an object corresponding to the parent window from tk().

Step 3: Create an object from canvas() and place it on the parent window along with height and width.

Step 4: Now use pack() for positioning of widget on the parent window.

Step 5: Now create an object face and use object. Create_oval() with coordinates 50, 50, 350, 350 and outline = 'black', fill = "yellow" as attribute to create face.


```
from tkinter import *
root = Tk()
c = Canvas(root, width = 500, height = 500)
c pack()
face = c.create_oval(50, 50, 350, 350, outline = "black",
                     fill = "yellow")
eye1 = c.create_oval(125, 125, 175, 175, fill = "black")
eye2 = c.create_oval(225, 125, 275, 175, fill = "black")
mouth = c.create_arc(125, 225, 275, 275, start = 0, extent = -180,
                     width = 5, fill = "red")
root.mainloop().
```

output:

output :

Step 6 : Now create eye 1 object & again use object. Create_oval () with appropriate coordinates along with fill as attribute to create left eye.

Step 7 : Now repeat the same step 6 to create right eye.

Step 8 : Create an object mouth and use object. Create_arc() with appropriate coordinates, start = 0 extent = -180 and fill = "red"; width = 5 as attribute to create mouth

Step 9 : Finally use the mainloop().

Q2. Write a program to convert Celsius into Farhenheit using G

Algorith :

Step 1 : Import all the relevant methods in tkinter lib

Step 2 : Create object corresponding to the parent window From tk().

Step 3 : Now initialize fahrenheit as Doublevar() and set it to 320.

Step 4 : Now define a function convert with argument Celcius to convert celcius into fahemheit using set().

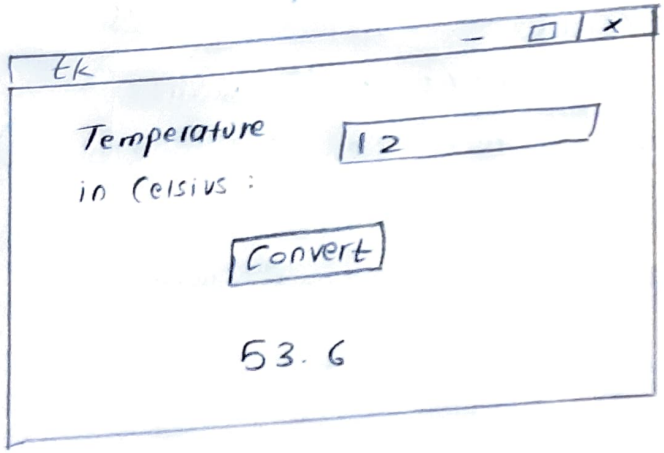
Step 5 : Now create an object L2 using label() and place it onto parent window and use text attribute as enter a no :

Step 6 : Now use grid() for position the object onto the parent window. Initialize celcius as integer using IntVar().

Step 7 : Create another object and use entry widget to enter the input and place it onto the parent window.

```
from tkinter import *
window = Tk()
Fahrenheit = DoubleVar()
fahrenheit.set(32.0)
def convert(celsius):
        fahrenheit.set((9.0/5.0) * Celsius + 32)
L1 = Label(window, text="Temperature in Celsius")
L1.grid(row=0, column=0)
e = entry(window, textvariable = Celsius)
e.grid(row = 0, column = 1)
Celsius = IntVar()
L2 = Label(window, textvariable = Fahrenheit)
L2.grid(row=2, column=0, columnSpan=2)
B = Button(window, text="Calculate", command = lambda
        convert(celcius.get())
B.grid(row = 1, column = 0, columnspan=2)
window.mainloop()
```

output :



Step 8. Now use grid() for positioning the object onto parent window with text variable attribute.

Step 9 : Now again use label() along with text variable attribute to display output and use grid() for positioning.

Step 10 : Finally use mainloop().

Practical 7

Aim: Write a program to find factorial of number and use arthmetic operations of on two numbers using GUI.

Q1. Write a program to find factorial of number using GUI

Algorith:

Step 1: Import relevant methods from tkinter library.

Step 2: Now define a function factorial to calculate factorial using recursive function.

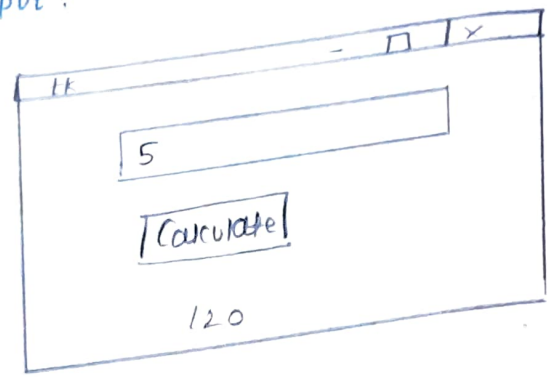
Step 3: Define another function calculate to call factorial function.

Step 4: Now create an object with entry() and use pack() for positioning on parent window.

Step 5: Now create an object with button() along with command = attribute to calculate factorial


```
from tkinter import *
def factorial(n):
    if n==0 or n==1:
        return 1
    else:
        return n * factorial(n-1)
def calculate():
    result = factorial(int(entrytext.get()))
    info.config(text = result)
root = Tk()
entrytest = entry(root)
entrytest.pack()
btn = Button(root, text = "calculate", command = calculate)
btn.pack()
info = label(root text = "factorial")
info.pack()
root.mainloop()
```

output :



Code :

```
from tkinter import *
def calculate():
    if int(v.get()) == 1:
        res = int(e1.get()) + int(e2.get())
        l3.config(text = res)
    elif int(v.get()) == 2:
        res = int(e1.get()) - int(e2.get())
        l3.config(text = res)
    elif int(v.get()) == 3:
        res = int(e1.get()) * int(e2.get())
        l3.config(text = res)
    else:
        res = int(e1.get()) / int(e2.get())
        l3.config(text = res)
```


Step 6 : Now again create an object with label() to show output

Step 7 : Finally use the mainloop()

Q2 : Write a program to perform arithmetic operation on 2 numbers using GUI

Algorith :

Step 1 : Import relevant methods from tkinter library

Step 2 : Now create an object corresponding to parent window.

Step 3 : Now define a function calculate to carry out arithmetic operations on 2 numbers.

Step 4 : Now create object with label() as num 1 and num 2 and use grid() to place it onto parent window.

Step 5 : Create objects with entry() to take input from user().

Step 6 : Now initialize v as integer using IntVar().

Step 7 : Now create 4 objects with Radiobutton () to choose any one of arithmetic operators and use grid () for positioning onto parent window.

Step 8 : Now create a object with button () along with command attribute to carry out arithmetic operation of users choice.

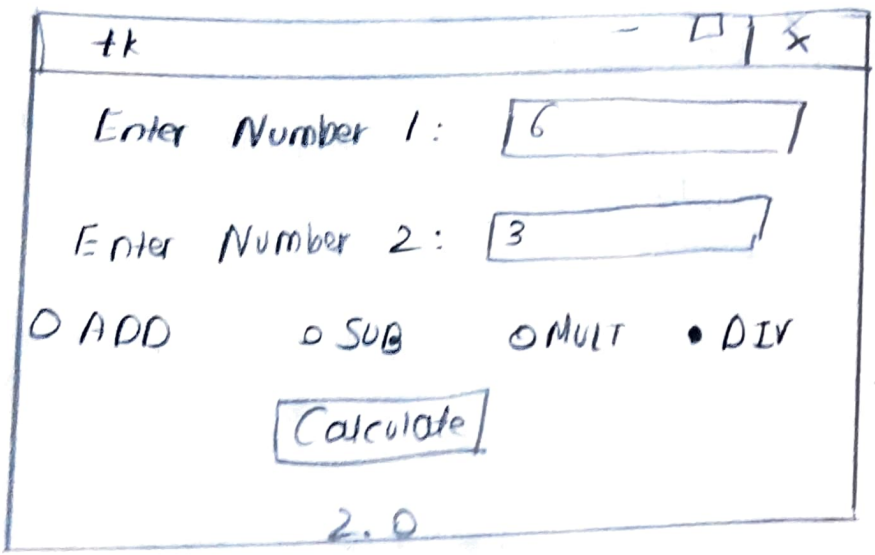
Step 9 : Now create a object with label () to show output.

Step 10 : Finally use the mainloop ().

```
root = Tk()
L1 = Label (root, text = " Enter a no: ")
L1.grid (row = 0, column = 0)
e1 = entry (root)
e1.grid (row = 0, column = 1)
L2 = Label (row = 0, column = 1)
L2.grid (row = 1, column = 0)
e2 = entry (root)
e2.grid (row = 1, column = 1)
r = IntVar()
r1 = Radiobutton (root, text = "add", variable = V, value = 1)
r1.grid (row = 2, column = 0)
r2 = Radiobutton (root, text = "Sub", variable = V, value = 2)
r2.grid (row = 2, column = 1)
r3 = Radiobutton (root, text = "Mult", variable = V, value = 3)
r3.grid (row = 2, column = 2)
r4.grid (row = 2, column = 3)
B = Button (root, text = "Calculate", command = calculate)
B.grid (row = 3, column = 1, columnspan = 2)
```

```
l3 = Label(root)
l3.grid(row=4, column=1)
root.mainloop()
```

output :



Code :

```
import socket
def server_program:
    host = socket get hostname()
    port = 5000
    server_socket = socket.socket()
    Server_socket.blind((host, port))
Server_socket.listen(2)
Conn, address = server_socket.accept()
Print("connection from:" + str(address))
```

```
while true:
    data = conn.recv(1024).decode()
    if not data:
        break
    print("From connected user: " + str(data))
    data = input(" -> ")
    conn.send(data.encode())
conn.close()
```

(Now run the program and now right write client program)

Code:

```
import socket
def client_program():
    host = socket.gethostname()
    port = 5000
    client_socket = socket.socket()
    client_socket.connect((host, port))
    message = input(" -> ")
    while message.lower().strip() != 'bye'
        client_socket.send(message.encode())
        data = client_socket.recv(1024).decode()
    print("Received from Server:" + data)
```

Practical 8:

Aim: Demonstrate the use of socket module and server client programs.

Write a program to demonstrate use of socket module and server client programs.

Algorith:

Step 1: Import the socket module to import relevant methods.

Step 2: Define a function as server_program to get hostname

Step 3: Now get value for port variable to initialize port no. above 1024.

Step 4: Use .socket() to get instance

Step 5: Now use bind() function to bind host address and port together to configure how many client the server can list simultaneously.

Step 6: Now use accept() to accept new connection.

Step 7: Now print the address.

Step 8: Use while loop as True to receive data stream.

Step 9: Now close the program.

Algorith :

Step 1 : Import socket module to import methods that are relevant.

Step 2 : Define a function client_program get the hostname and give port a value 5000.

Step 3 : Now again initiate by using socket.socket()

Step 4 : Use connect() to connect the server.

Step 5 : Now take the input ("→ ").

Step 6 : Use while conditional loop to send a message.

Step 7 : Now use decode to receive response.

Step 8 : Now show the data.

Step 9 : Again take input

Step 10 : Close the program by using close().

```
message = input ("→ ")
client_ socket. close ()
```

Output for socket_Program

```
$ python.3.4 socket-server. py
Connection from : ('127.0.0.1', 57822)
From connected user : Hi
→ Hello
from connected user : Awesome !
→ Ok then, bye !
```

output for client_program

```
$ python 3.6 Socket_client . py
→ Hi
Received from server : Hello
→ How are you?
Received from server : Good
→ Awesome !
Received from server : Ok then bye !
→ Bye.
```

* CODE IN SHELL ENVIRONMENT

```
>>> import sqlite3
>>> conn = sqlite3.connect("Student1.db")
>>> cur = conn.cursor()
>>> cur.execute('create table student
        (roll_no int(5) primary key not null, name
        varchar(50) not null, address varchar
        (50) not null, class varchar(50), dob date)')
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute('insert into student values
        (101,"Ayaan","Malad","FYCS","14/03/2001")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute('insert into student values
        (102,"Daren","Malad","FYCS","28/02/2001")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute("select * from student")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.fetchall()
[(101,'Ayaan','Malad','FYCS','14/03/2001')
 (102,'Daren','Malad','FYCS','28/02/2001')]
>>> cur.execute("update student set dob="13/09/2003"
        where roll_no=101")
<sqlite3.cursor object at 0x0322EBED.
```

Practical 9.

Aim: Demonstrate the use of database connectivity.

Algorithm:

Step 1: Import sqlite3 module to import relevant methods.

Step 2: Now initialize a variable conn to by using connect() to a new database using extension .db

Step 3: Now initialize a variable to connect to cursor()

Step 4: Now use cur.execute() to create a table, insert values into table and use DML, DDL statements to manipulate the data in this database.

Step 5: Use fetchall() to show the output.

Step 6: Use commit to save all changes

Step 7: Use close() to terminate the program.

```
>>> cur.execute('select * from student where address = "Malad")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.fetchall().
[(102, 'Ayaan', 'Malad', 'FYCS', 13/08/2003)]
>>> cur.execute('commit')
<sqlite3.cursor object at 0x0322EBED>
>>> cur.close().
```