55

write methods.

```
file object = open("abc.txt","w")
file object.write("subject in" + "cs")
fileobject.write("\n c \n calculus\n stats\n python")
fileobject.close()
```

Read method

```
fileobject = open("abc.txt","r")
 str1 = file.object.read()
print("the output of read method is:",str1)
fileobject.close()
```

>>> the output of read method is : subject in cs
c
calculus
stats
python

readline method

```
fileobject = open("abc.txt","r")
 str2 = fileobject.readline()
print("the output of readline method is:", str2)
fileobject.close()
```

>>> the output of readline method is: subject in cs

readlines method

```
fileobject = open("abc.txt","r")
 str3 = fileobject.readlines()
print("the output of readlines method is:", str3)
```

>>> the output of readlines method is: ['subject in cs\n', 'c\n', 'calculus\n', 'stats\n', 'python']

Practical-1

Aim:- Demonstrate the use of different file accessing modes different attributes and and read methods.

Algorithm:-

Steps:-

1> Create a file object using open method & use the write access mode followed by writing some contents onto the file & then closing the file.

2) Now, open the file & read mode & then use read(), readline(), & readlines() & store the output in variable & finally display, the contents of variable.

3) Now, use the file object for finding the name of the file the filename, the file mode in the which it opened whether the file is still open as close and finally the output of the software attribute.

4) Now, open the file object in the write mode, write some another close subsequently then again open the fileobject in 'w' mode that is the update mode and write content.

5) open fileobject in read mode display the update written contents and close, again in 'r+' mode that is parameter passed and display the output subsequently.

6) Now, open fileobject in append mode open write method, write contents close the fileobject again, open the fileobject in read mode and display the append outputs.

file attributes

```
a=fileobj.name
b=fileobj.close
c=fileobj.mode
print("the file name is:",a)
print ("closed",b)
print("file mode:",c)
```

```
>>> The file name is: abc.txt
>>> closed: True
>>> filemode: r
```

write mode

```
fileobject=open("abc.txt","w")
fileobject.write("c")
fileobject.close()
```

read mode

```
fileobject=open("abc.txt","r")
  y=fileobject..read()
print("the output of read mode is:",y)
```

```
>>>the output of read mode is:c
```

w+ mode

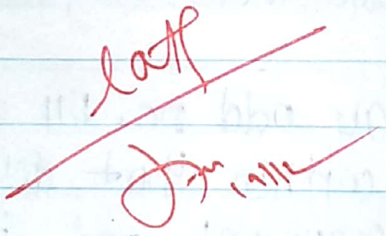
```
file=open("abc.txt","w+")
file.write("Aashish")
file.close()
```

```
#r+ mode
   fileobj = open ("abc.txt", "r+")
     s1 = fileobj.read(6)
   print ("output of r+ ", s1)
    fileobj.closec)
>>> ("output of r+", 'stats')
#Append mode
 fileobject = open ("abc.txt", "a")
 fileobject = write ("data structure")
  fileobject.close()
 fileobject.open ("abc.txt", "s")
      s3 = fileobject.read()
   print ("output of append mode:", s3)
    fileobject.close()
>>>output of append mode:
   #tell()
ileobject = open("abc.txt", "r")
 pos = fileobject.tell()
 print ("tell():", pos)
  fileobject.close()
>>> ("tell():", 0L)
#seek()
fileobject = open("abc.txt", "r")
 str4 = fileobject.seek(0,0)
 str8 = fileobject.read(10)
print("the beginning of
      file:", str8)
'python data structure'
#finding length of the different line exist within line
      fileobject = open ("abc.txt", "r")
       str9 = fileobject.readlines()
        print ("output:", str9)
      for line in str9:
                print(len(line))
               fileobject.close()
```

7. open the fileobject in read mode declare a variable and perform fileobject dot method tell to store the output consequently in variable.

8. Use the seek method with the argument with operating the fileobject in read and closing subsequently

9. open fileobject with read mode also use readlines() and store the output consequntly in and print the same for counting the statement and display the length.



Practical 2

Aim:- Iterators

Program 1:
Algorithm:-
① Define a variable of list data type containing names of fruit
② Now, iterators over that variable and print each element using next method or some conditional loops statement.

Program 2: To display odd no. till 6 Algorithm.
① Create a class within that define a iter method with an argument and initialise the value and return that values.

② Define the next method with an argument.
③ Use if conditional statement to check whether the variable in 1st step is smaller than 16.
④ Increase the variable's value by 2 and return it.
⑤ Use else to stop iteration.
⑥ Create an object of given class and pass this object in an iter method.
⑦ Use while conditional loop to print the next method.

iter() and next()

```
mytuple1 = ("banana", "orange", "apple")
myiter1 = iter(mytuple)
print(next(myiter1))
myiter2 = iter(mytuple)
    print(next(myiter1))
myiter3 = iter(mytuple)
  print(next(myiter1))
```

```
>>> banana
    orange
    apple
```

For loop

```
mytuple1 = ("kevin", "stuart", "bob")
for x in mytuple1:
    print(x)
```

```
>>> kevin
    stuart
    bob
```

square and cube

```
def square(x):
    y = x*x
    return y
def cube(x):
    z = x*x*x
    return z
a = [square, cube]
```

```
for r in range(5):
    value = list[map(lambda x:x(r), func+1)]
    print(value)
>>> [0,0]
    [1,1]
    [4,8]
    [9,27]
    [16,64]
```

```
# map()
listnum = [0,4,5,7,9,11,13,15,20,19,25]
listnum = list(map(lambda x:x%5, listnum))
print(listnum)
def even(x):
    if (x%2==0):
        return "even"
    else:
        return "ODD"
    list[map(even,listnum)]
```

```
# odd numbers
Class odd:
    def __iter__(self):
        self.num=1
        return self
    def __next__(self):
        num.self=num
        num.self+=2
    def __next__(self):
        num = self.num
        self.num+=2
        return num
>>> Entero number:- 11
```

```
myobj = odd()
myiter = iter(myobj)
x=int(input("Enter a no:-"))
for i in myiter:
    if (i<x):
        print(i)
```

11

Program 3:- Factor of number till 10

Algorithm:-

① create a class within that define a iter method an argument, initialize a value and return it.

② Define a next method with an argument.

③ create a variable factorial with value 1 and use for conditional loop to calculate factorial till value of variable declared in step 1.

④ Print the fact's value and increase the value and increase the value of variable in step 1 by 1.

⑤ use if conditional statement to value stop iteration if the range of variable in step 1 is bigger than 10.

⑥ create a object this class and pass it to iter method and use while loop to use next method.

Prog 4:- Power of number taken from user.

① Create a class and define a iter method with an argument initialize a value and return it an also take a input from user.

② Define a next method with an argument.

③ If the initialised value is bigger than 5 then raise stop iteration or else return the multiplication of argument of itself.

④ create a object of given class and pass it to iter method and use while loop to use next method.

Programs:- To display the number in range 1 to 10.

Algorithm:-

① Create a class and define iter method with an argument, initialize a value and return it.

② Now define next method check whether the initialized value is smaller than 10, if yes then print the value.

③ Increase the value by 1, if the initialised value is bigger than 10 stop the iteration.

code of factorial:

```
class fact:
    def __iter__(M):
        M.value=1
        return M
    def __next__(M):
        facto = 1
        for i in range(1,M.value+1):
            facto=facto*(i)
        print("the factorial of", M.value, "is", facto)
        M.value+=1
        if (M.value==7):
            raise STOPIteration
        y=iter(fact())
        while True:
            next(y)
```

output:

```
the factorial of 1 is 1
the factorial of 2 is 2
the factorial of 3 is 6
the factorial of 4 is 24
the factorial of 5 is 120
the factorial of 6 is 720
```

code for number taken from user:

```
Class power:
    def __iter__(pow):
        pow.num=1
        pow.num=int(input("Enter number:"))
        return pow
    def __next__(pow):
        if pow.num<=6:
            nu=pow.nums**pow.num
            pow.num+=1
            return nu
```

```
else:
    raise STOP Iteration
    y = iter(power(1))
    while True:
        print(next(y))
```

output:

```
Enter number: 2
2
4
8
16
32
64
```

Code to display number upto range 10.

```
class range1:
    def __iter__(0):
        0.num = 1
        return 0
    def __next__(0):
        if 0.num <= 10:
            mu = 0.num
            0.num += 1
            return mu
        else:
            raise STOP Iteration
    y = iter(range1())
    while true:
        print(next(y))
```

output:

1 2 3 4 5 6 7 8 9 10

Practical no:3

Aim:- program to demonstrate exception handling.

1) write a program using the exception method of the nature arithmetic error.

step1:- use the try block and except the input using the raw input method and correct it into the integer datatype and subsequently terminate the block.

step2:- use the except block with the exception name as value error and display the appropriate message if the suspecious code is part of the try block.

2) write a program for accepting the file in a given mode and use the environment error as an exception for the given input.

step1:- within the try block open the file using th write mode and write mode and write som content on the file.

step2:- use the except block with IO error and dis the message regarding missing of the fil on incompability of the mode use the else b to display a message that the operation is co out successfully.

#program :-

```
while True:
    try:
        x = int(input("enter class:"))
        break
    except value Error
        print("Enter numeric value")
```

output:

```
enter class: 467
```

#program :-

```
try:
    fo = open("abc.txt", "w")
    fo.write("Abhinay Vish")
except IO Error
    print("Error writing on the file")
else:
    print("operation carried out successfully")
    fo close()
```

Output:

```
operation carried out successfully
```

30

#program

```
def assert_(n):
    assert(len(n)==0)
    print("list is empty")
var1=[ ]
print(assert_(var1))
```

output:

list is empty

#program

```
def acceptage():
    age=int(input("enter age:"))
    if age>30 or age<16:
        raise value Error
    return age
valid=False
while not valid:
    try:
        age=acceptage()
        valid=True
    except value Error:
        print("Not a valid age")
```

output:

```
Enter age:4
Not a valid age
Enter age:18
```

3) write a program to using the assert() to check if the list elements are empty.

step1:- Define a function which accepts an argument and check using the assert statement whether the given list is empty list and accordingly return the message

step2:- close the function and in the body of program and define certain element in list and take some appropriate action.

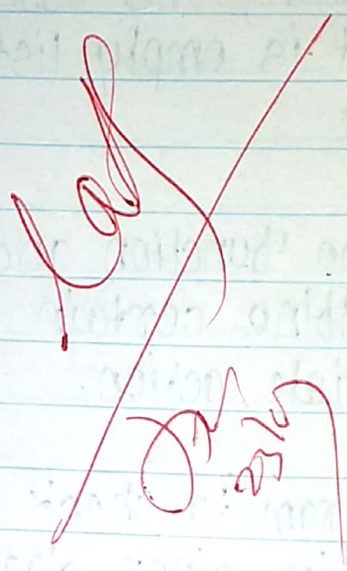
4) write a program to check the range of the age of the students in given class and if the age do not fall in given range else the value error exception otherwise return the valid number.

step1:- Define a function which will accept the age of the student from the standard input.

step2:- Use the if condition to check whether the input age falls in the range and so return the age else use the value error exception.

step3:- Define the while loop to check whether the boolean expression holds true use the try block to accept the age of student and terminate the looping condition.

Step4:- Use except with value error and print the message not a valid range.



```
#match()
import re
pattern=r"FYCS"
sequence="FYCS represents computer science stream"
if re.match(pattern,sequence):
        print("Matched pattern found!")
else:
    print("NOT FOUND!")
```

```
>>>Matched pattern found:
```

```
#numerical values (segregation)
  import re
  pattern=r'\d+'
  string='hello123,howdy789,45hours'
  output=re.findall(pattern,string)
  print(output)
>>>['123','789','45']
```

```
#split()
  import re
  pattern=r'\d+'
  string='hello123,howdy789,45hours'
  output=re.split(pattern,string)
  print(output)
>>>['hello',',howdy',',','hours']
```

Practical no:4

Topic:- Regular expression.

step1:- Import re module declare pattern and declare sequence use match method with declare arguments if arguments matched then print the same otherwise print pattern NOT FOUND!

step2:- Import re module declare pattern with literal and meta character declare string value. Use the findall() with arguments and print the same.

step3:- Import re module declare pattern with meta character use the split() and print the output.

step4:- Import re module declare string and accordingly declare pattern replace to the blank space with no-space use sub() with 3 arguments and print the string without spaces.

step5:- Import re module declare a sequence use search method for finding subsequently use ch group () with dot operator ds search() gives memory creation using group() it will show up the method string.

step6:- Import re module declare list with numbers. Use the conditional statement here we have used up the for condition statement. use if condition for checking first number is either 8 or 9 and

next number are in range of 0 to 9 and check whether the entered numbers are equal to 10. if criteria matches print all number matches otherwise print failed.

step7:. import re module declare a string use the module with findall() for finding the vowels in the string and declare the same.

step8:. Import re module declare the host and domain name declare pattern for separating the host and domain name. use the findall() and print the output respectively.

step9:. Import re module enter a string use pattern of the particular string use findall() declare two variables with initial value as zero use for condition and subsequently use the if condition check whether condition satisfy add up the or else increament value and display the values subsequently.

```
#nospace:
import re
string = 'abc def ghi'
pattern = r'\s+'
replace = ''
  v1 = re.sub(pattern, replace, string)
print(v1)
>>> abcdefghi
```

```
# group()
import re
seqeuree = 'python is an interesting language'
   V = re.search ("Apython', sequeuee)
print(v)
   v1 = V.group()
   print(v1)
>>> <_sre.SRE_Match object at 0x02S101=00>
    python
```

```
#verifying the given set of phone.numbers
import re
list1 = ['8004567891', '9145673210', '7865432981',
        '9876543201']
for value in list1:
   if re.match(r'[8-9]{1}[0-9]{9};
      value and len(value) == 10);
       print("criterial matched for cell numbers!")
   else:
       print("criteria failed!")
>>criteria matched for cell number
criteriad matched for cell number
criteria failed:
criteria matched for cell number
```

```
#vowels
import re
str1 = 'plant is life overall'
output = re.findall(r'\b[aeiOUAEIOU]\w+', str1)
print(output)
>>> ['is', 'overall']
```

```
# host and domain
import re
seq = 'abc.+csc@edu.com xyz@gmail.com'
pattern = r'[\w\.-]+[\w\.-]'
output = re.findall(pattern, seq)
print(output)
>>> ['abc.+csc', 'edu.com', 'xyz', 'gmail.com']
```

```
# counting of first two letters
import re
s = 'mr.a, ms.b, ms.c, mr.t'
p = r'[/ms/mr/]+
o = re.findall(p,s).
print(o)
m=0
f=0
for v in o:
if(v=='ms'):
    f=f+1
else:
    m=m+1
print("No. of males is:",m)
print("No. of females is:",f)
>>>['mr','ms','ms','mr']
('No. of males is:',2]
('No. of females is:',2)
```

Practical 5

Aim:- GUI component

step 1: use the tkinter library for importing the feature of text widget.

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

step 3:- use the pack method along with the object created from text method.

step 4:- use the main loop method for triggering of corresponding events.

step 5:- use the tkinter library for importing the feature of text widget.

step 6:- create a variable from text method and position it onto parent window.

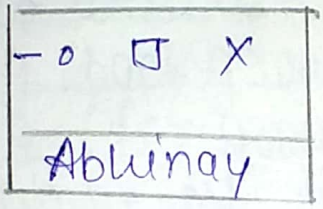
step 7:- use the pack method along with the object created from text method and use the parameter.

step 8:- side = LEFT , padx = 20
side = LEFT , pady = 30
side = TOP , padx = 40
side = TOP , pady = 50

CODE:-

```
from tkinter import *
    root=TK()
  l=label(root,text="Abhinay")
  l.pack
  root.mainloop()
```

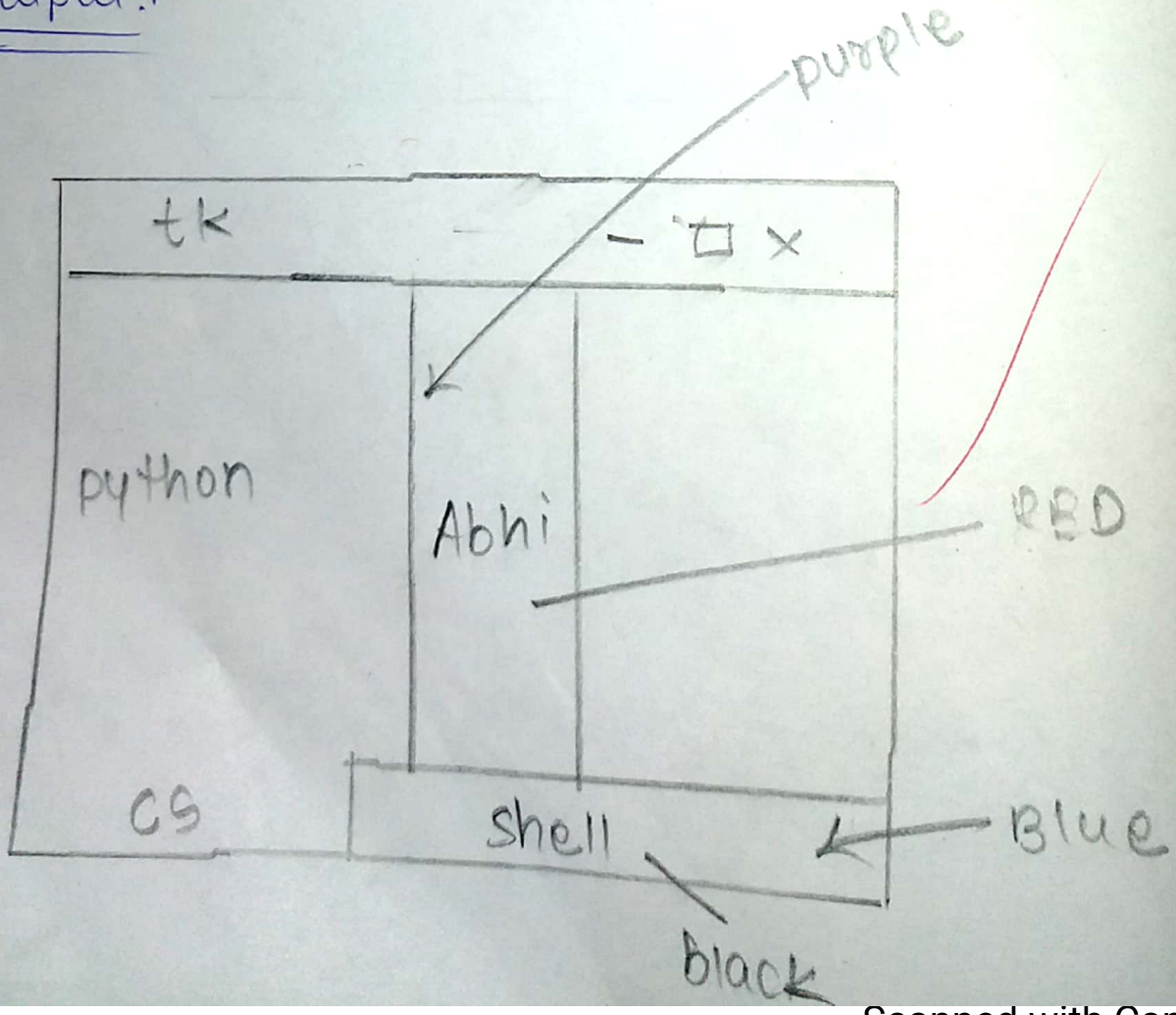
output:



CODE:-

```
from tkinter import *
root = Tk()
d = label(root, text = "python")
d.pack(side = LEFT, pady = 30)
d1 = label(root, text = "Abhi", bg = "purple", fg = "red"
d1.pack(side = Top, ipady = 40)
m = label(root, text = "cs")
m.pack(side = LEFT, padx = 20)
m2 = label(text = "shell", bg = "blue", fg = "black")
m2.pack(side = LEFT, ipadx = 50)
root.mainloop()
```

output:-



step9: Use the mainloop method for triggering of corresponding events.

step10: Now repeat the above steps with the label method which takes the following arguments.

(1) text attribute which defines string.
(2) bg (background) colour
(3) fg (for ground) colour
(4) Name of the parent window
(5) use pack method with the releavant padding attribute.

Radio button

W.A.P making use of the control variable and button widget for selection of the given option.

step 1: use the tkinter import relevant method.

step 2: Define a function which tells the user about the given selection need of the multiple options available.

step 3: use the configuration method along with the label object and call the variable as an argument within the method.

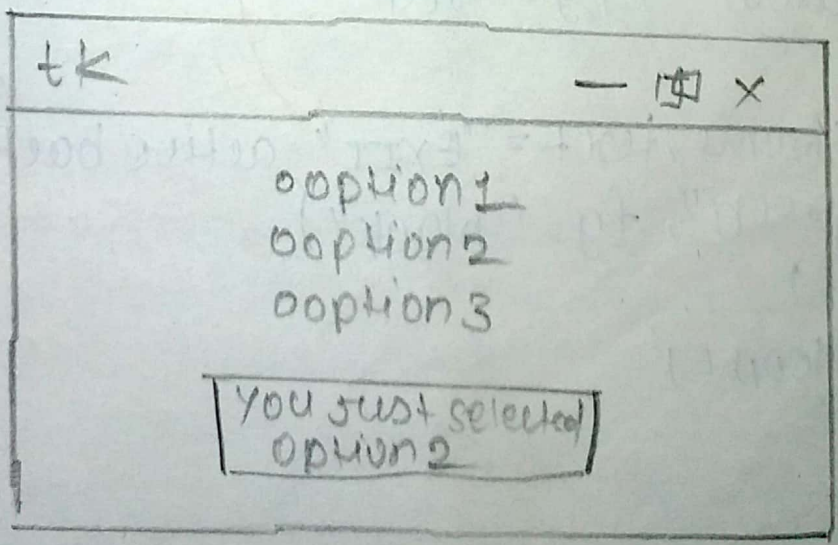
step 4: Now define the parent window and define the option using control variable.

step 5: Now create an object from the radio button method which will take the all arguments
1] positioning on parent window
2] Defining the text variable [1, 2, 3, 4]
3] Define the variable argument
4] corresponding value and trigger the given function.

* source code:-

```
from tkinter import *
root = Tk()
def sel():
    selection = "you selected the option" + str(var.get())
    l = label.config(root, text = "selection", justify=LEFT)
    l.pack(anchor=s)
    v = IntVar()
r1 = Radiobutton(text = "option 1", variable = "v",
    value = '1', command = sel)
r1.pack()
r2 = Radiobutton(text = "option 2", variable = "v",
    value = "2", command = sel)
r2.pack()
r3 = Radiobutton(text = "option 3", variable = "v",
    value = "3", command = sel)
r3.pack()
root.mainloop()
```

output:-



Source code:

```
#frame object:
from tkinter import *
    top = TK()
top.geometry('100 x 200')
    frame.frame(top)
     frame.pack()
   left frame=frame(top)
    left frame.pack(side=LEFT)
b1=Button(frame, text="select", active background
          ='red', fg="black")
b1.pack()
b2=Button(frame, text="modify", active background
      ="blue", fg=purple)
   b2.pack()
b3=Button(frame, text="Add" active background=
      "yellow", fg="red")
b3.pack()
b4=Button(frame, text="EXIT" active background=
      "green", fg="black")
   b4.pack()
top=mainloop()
```

step6:- pack method for the corresponding radio objects so created and specify the attribute as an anchor attribute.

step7:- Now define the label object from the corresponding method and place it on parent window. subsequently use pack method for this window and make use of the main loop method.

step8:- Import the relevant method from tkinter library.

step9:- Define the object corresponding to the object window and define the size of parent window in terms of no. of pixels.

step10:- Now define the frame object from the method and place it on to the parent window.

step11:- create another frame object the left frame and put it onto parent window on its left side.

step12:- similarly it define the right frame and subsequently define the button object placed onto the given frame with the attribute as text active bg and fg.

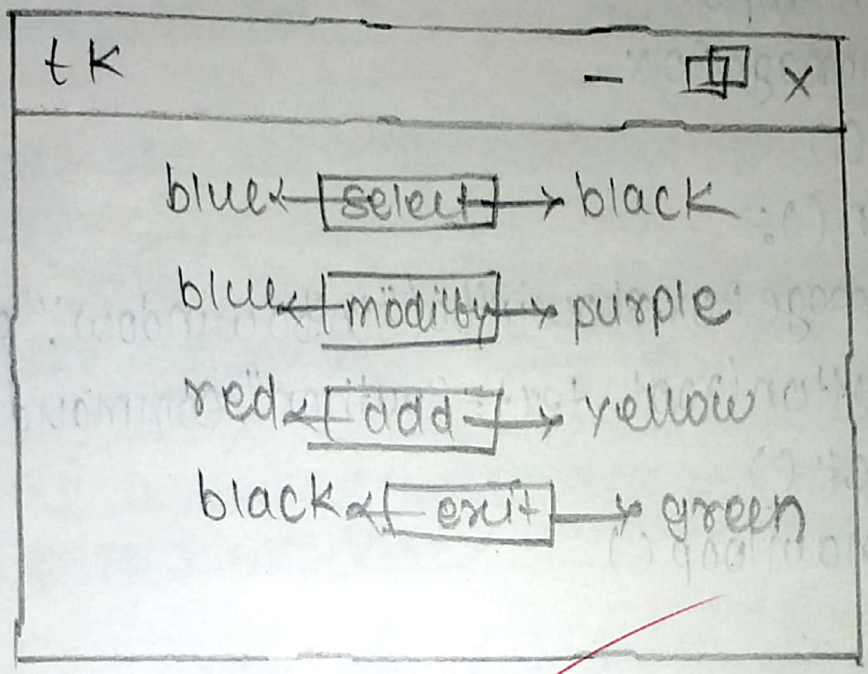
Step13:. Now use the pack method along with the side attribute.

Step14:. Similarly create the button object corresponding to modify option and put it onto the frame object with side equal to right attribute set.

Step15:. Add another button and put it on right frame object and term it as exit.

Step16:. use the pack method for all the object and finally use the main loop method.

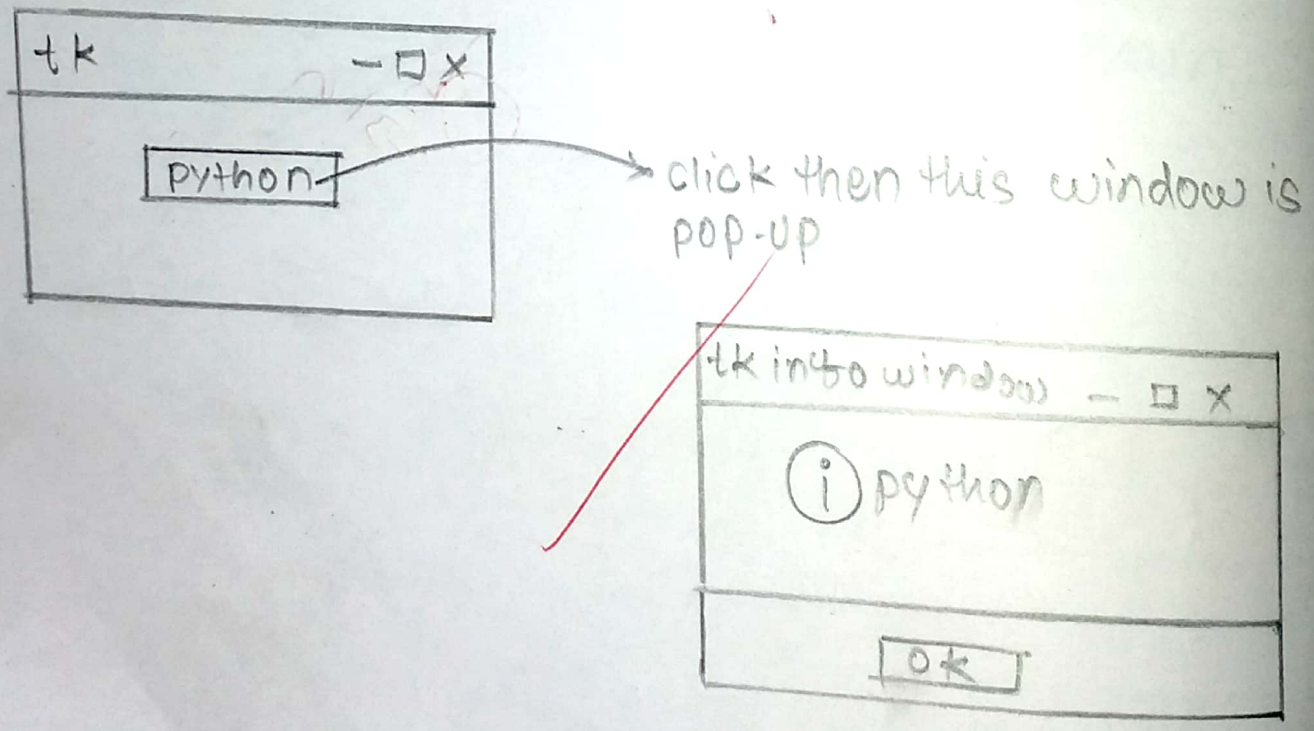
output:-



01

```
#message box
from Tkinter import *
import tkmessageBox
    root=Tk()
def function():
        tkMessageBox.showinfo("infowindow","python")
        b1=Button(root,text="python",command=function)
        b1.pack()
        root.mainloop()
```

Output:



Practical 5(c)

Aim:- GUI components.

step1:- Import the relayout method from tkinter library.

step2:- Import tk messagebox.

step3:- Define a parent window object along with the parent window.

step4:- define a function which will use tkmessagebox with showinfo method along with info window attribute.

step5:- Declare a button with parent window object along with the command attribute.

step6:- place the button widget onto the parent window and finally call mainloop() for triggering of the events called above.

Step1:- Import the relevant methods from the tkinter library along with parent window object declared.

Step2:- use parent window object along with minsize function for window size.

Step3:- Define a function main, declare parent window object and use config(), title(), minisize(), label() as well as button() and use pack() and mainloop() simultaneously.

Step4:- Similarly define the function second and use the attributes accordingly.

Step5:- Declare another function button along with parent object and declare button with attributes like FLAT, RIDGE, GROOVE, RAISED, SUNKEN along with the relief widget.

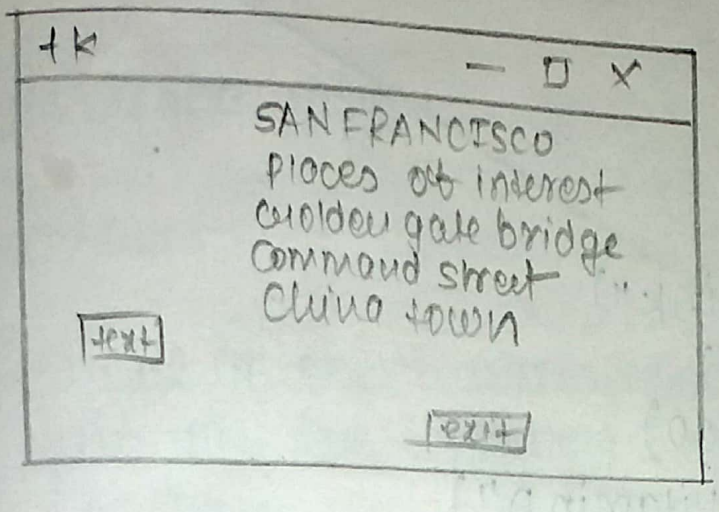
Step6:- Finally called the mainloop() for event driven programming.

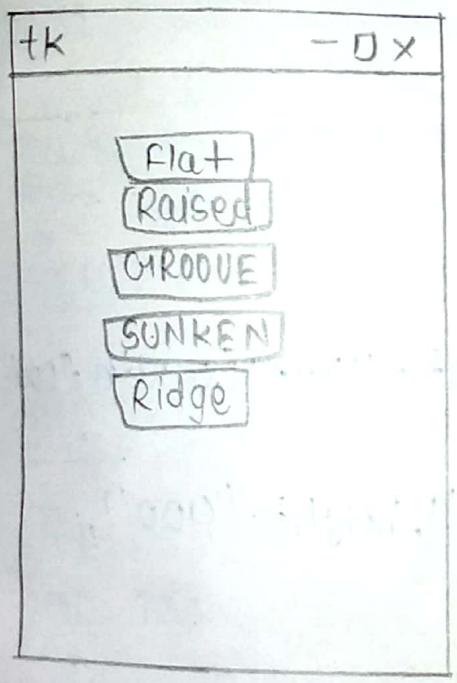
#multiple window
#Differeut button(Relief())

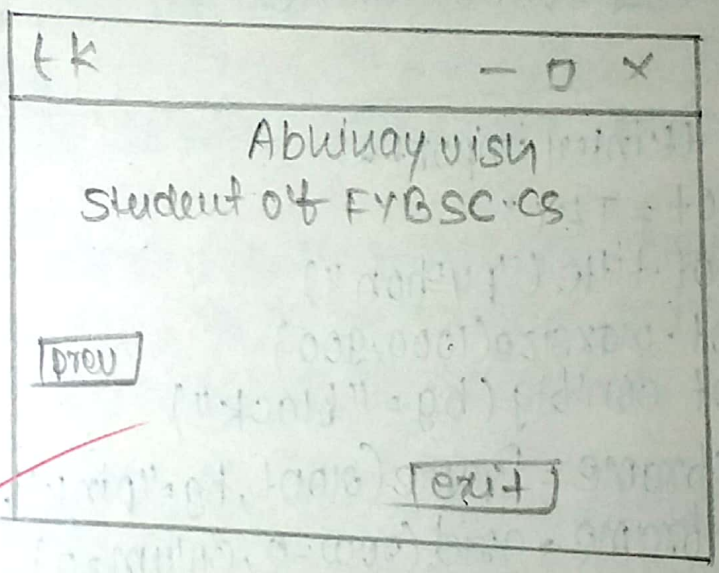
```
from Tkinter import
  root=Tk()
  root.minsize(300,300)

  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 commau
               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=terminat
      b2.pack(side=LEFT)
      top.mainloop()
```

```
def second():
    top2=Tk()
    top2.config(bg="orange")
    top2.title("About us!")
    top2.minsize(300,300)
    L=label(top2,text="created by: Abhinay vishin for more detail contact to our official account")
    L.pack()
    b3=Button(top2,text="prev",command=main)
    b3.pack(side=LEFT)
    b2=Button(top2,text="exit",commaud=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 loction",relief=RAISED)
    b3.pack()
    b4=Button(top3,text="SUNKEN button",relief=SUNKEN
    b4.pack()
    b5=Button(top3,text="Ridge button",relief=RIDGE)
    top3.mainloop()
    def terminate():
        quit()
    b5=Button(root,text="TOUR DETAILS"commaud=mai
    b5.pack
    b6=Button(root,text="BUTTON DETAILS",commaud=but
    b6.pack()
    root.mainloop()
```

output:-







```
from tkinter import *
  root = Tk()
  def main():
      r100 = Tk()
      r100.config(bg = "pink")
      r100.title("main")
      r100.minsize(200,200)
      d = label(r100, text = "vitamin D")
      d.pack()
      d1 = Label(r00, text =
```

```
from tkinter import *
  r100t = Tk()
  r100t.title("python")
  r100t.maxsize(1000,900)
  r100t.config(bg = "black")
  leftframe = frame(r100t, bg="pink", height = "400", width = "20
  leftframe.grid(row=0, colum=0)
  right frame = frame(r100t, bg: "light green", height = "400",
                width = "250")
  right frame.grid(row=0, column=0)
  Label(leftframe, text = "photo", height = 2, width=20)
        grid(row=0, column=0)
  image1 = PhotoImage(file = "dance.gif")
  image1.subsample(3,4)
  image2 = PhotoImage(file = "dance.gif")
  image2.subsample(3,4)
```

* Displaying the image:-

Algorithm:-

Step1:- create an object corresponding to the parent window and use the following 3 methods. Title.maxsize.config

Step2:- create a leftframe object from the frame method and place it onto the parent window with the height, width and the bg specified. subsequently use the grid method with the row, column, padx, pady specified.

Step3:- Now create a rightframe object from the frame method with the width, height specified and the row and the column value should be specified.

Step4:- create a label object from the label method and place it onto the leftframe with text attribute denoting the original image with relief attribute used as raised value and subsequently use grid method with row, column value specified as (0,0) with some external padding values.

Step5:- Now, use the photoimage method with the file attribute specified.

Step6:- use the sub-sample method with the object of the image and give the x,y co-ordinate values.

Step7:. use the label method and position it onto the leftframe and placing the image after the sampling and use the grid method for the positioning in the first row.

Step8:. create another label object positioning it onto the rightframe and specifying the image and background attribute with row and column attribute specify it as (0,0)

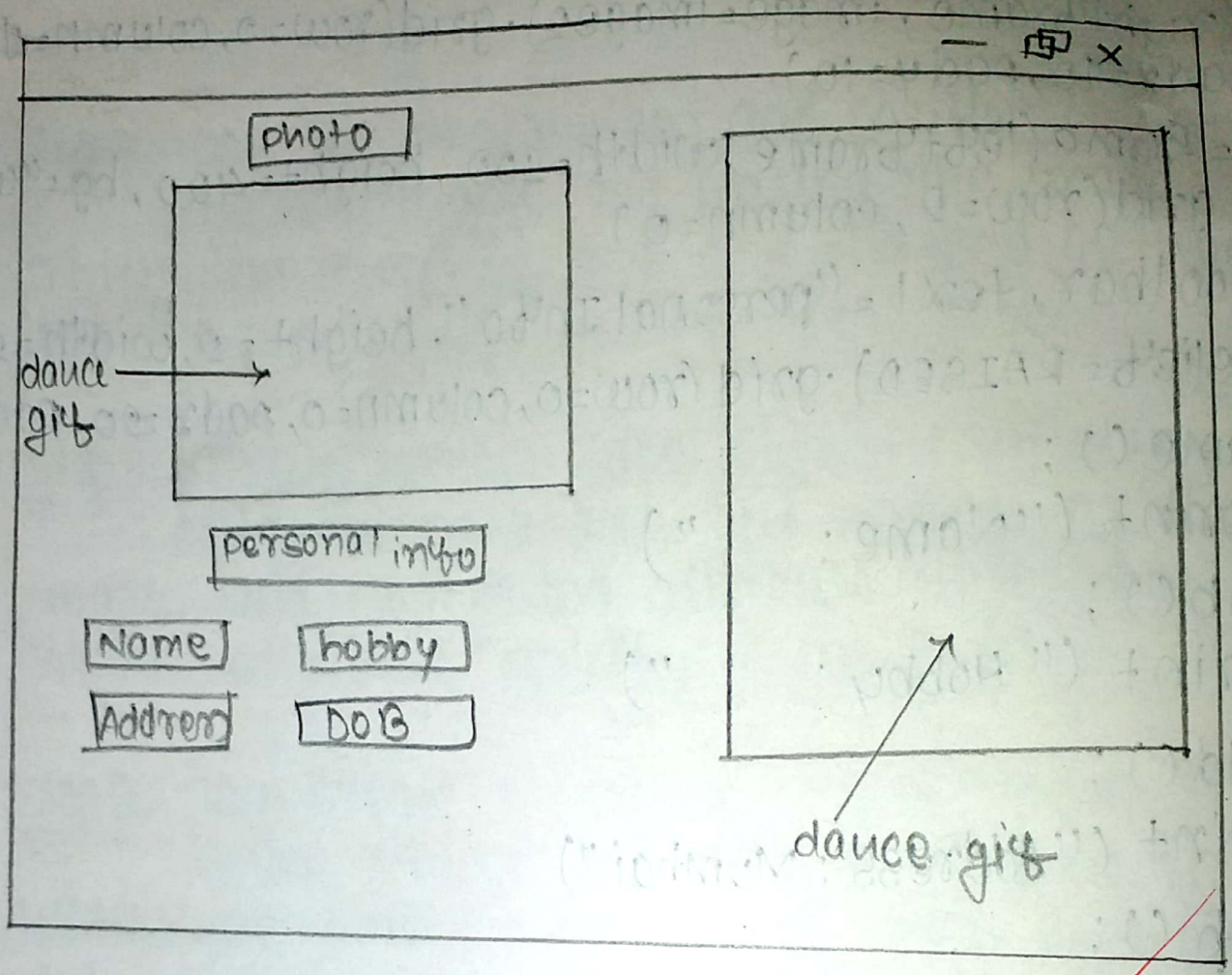
Step9:. Now create a toolbar object from the frame method and position it onto the leftframe with the height and width specified. and position it onto the second row.

Step10:. Now, define the various function for different toolbar options provided in the leftframe.

Step11:. From the label method position the text on the toolbar use the relief attribute and corresponding grid value and incorporate the internal padding as well.

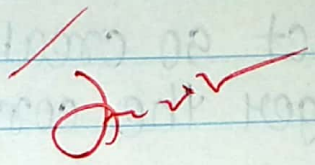
```
Label(leftframe, image=image1).grid(row=0, column=0,
    padx=20, pady=10)
Label(rightframe, image=image2).grid(row=0, column=1,
    padx=10, pady=10)
toobar=frame(leftframe, width=200, height=400, bg="white").
    grid(row=2, column=0)
Label(toolbar, text="personal Info", height=2, width=20,
    relief=RAISED).grid(row=0, column=0, padx=20, pady=10)

def name();
    print("Name:    ")
def hob();
    print("Hobby:    ")
def add();
    print("Address: Mumbai")
def dob();
    print("DOB: 24/08/1982")
Button(toolbar, text="Name", height=1, width=16, command=
    name).grid(row=1, column=0)
Button(toolbar, text="Hobby", height=1, width=16, command=
    hob).grid(row=1, column=1)
Button(toolbar, text="add", height=1, width=16, command=add
    grid(row=2, column=0)
Button(toolbar, text="DOB", height=1, width=16, command=dob)
    grid(row=2, column=1)
root.mainloop()
```

81



step12:. create the label method position it onto the toolbar with the next title as personal information and position it on same row but next column.

step13:. Now, make use of mainloop method.



f) Spin box:-

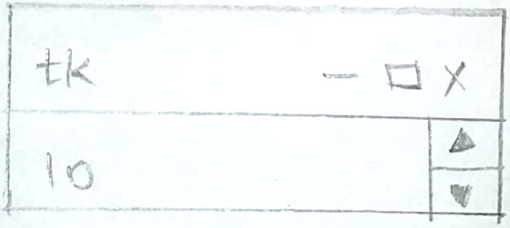
Step1:- create an object from the tkmethod and subsequently create an object from spinbox method..

Step2:- Make the object so created onto the parent window and trigger the corresponding event.

#Spin box

```
from tkinter ~~method~~ import *
root = Tk()
s = Spinbox(root, from = 0, to = 10)
s.pack(anchor = N)
root.mainloop()
```

output :-

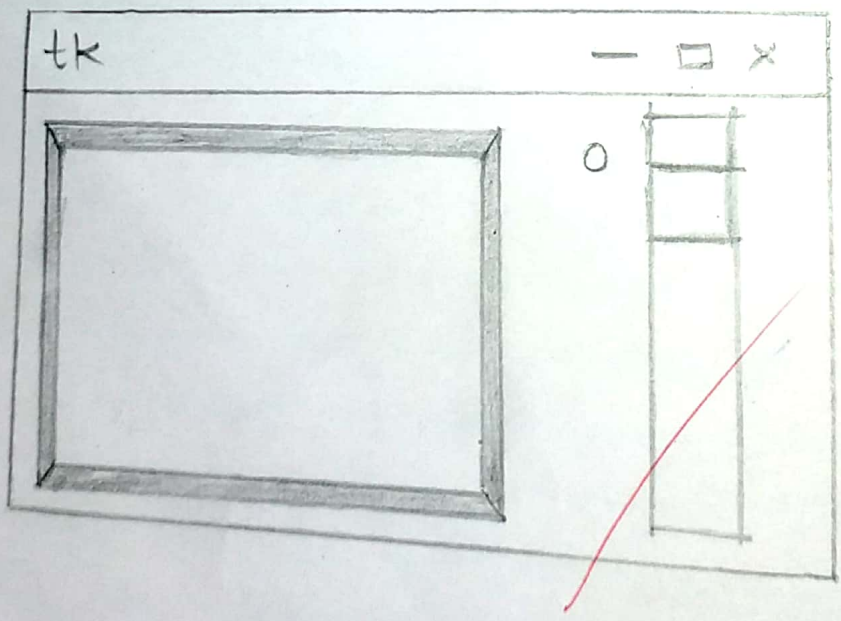


02

#Paned window

```
from tkinter import *
    root=Tk()
P=PanedWindow()
P.pack(fill=BOTH, expand=10)
e=Entry(P, bd=11, bg='Black')
P.add(e)
P1=Panedwindow(P, orient=VERTICAL)
P.add(P1)
top=Scale(P1, orient=VERTICAL)
P1.add(top)
root.mainloop()
```

output:



6) Paned window:-

step1:- create an object from the paned window method and use the pack method with the attribute fill and expand.

step2:- create an object from the label method and put it onto paned window with the attribute and use the add method to embed the new object.

step3:- Similarily, create a second paned window object and add it onto the first paned window ~~and~~ with orientation specified.

step4:- Now, create another label object and place it onto the second paned window object and add it onto the second pane.

step5:- Trigger main loop.

Practical 6

Aim: Database connectivity

#1:

Step1: Import the (DBM) dbm library & use the open() for creating the database by specifying the name of the database along with the corresponding flag.

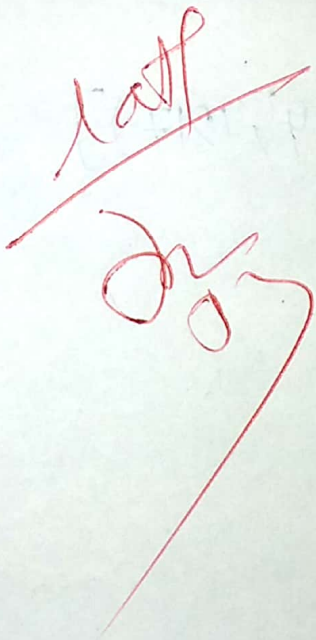
Step2: Use the object so created for accessing the given website & corresponding regular name for the website.

Step3: Check whether the given address matches with the regular name of the page is not equal to none than display the message that particular found/match or else not found/unmatched.

Step4: Use the close() to terminate database library.

#1.

```
>>> import dbm
>>> db = dbm.open("database", flag = 'c', mode = 438)
>>> db["name"] = "name"
>>> if db["name"] != None:
        print("database not empty // match!")
    else:
        print("database empty !" // NOT match")

>>> match
>>> db.close()
```



52

#2
```
import os, sqlite3
conn = sqlite3.connect("employee.db")
cur = conn.cursor()
cur.execute('create table dos (Name char, RollNo int
cur.execute("insert into dos values ('Tanvi, 1712),
    ('Tanmay', 1217)")
conn.commit()
cur.execute('select * from dos')
print (cur.fetchall())
con.close()
```

output:

```
[('Tanvi', 1712), ('Tanmay', 1217)]
```

#2.

step1: Import corresponding library to make database connection, OS & SQL lite-3.

step2: Now, create the connection object using SQL lite-3 library & the content connect() func creating new database.

step3: Now, create cursor object using the cursor() & from the connection object created.

step4: Now, use the execute() func creating the table with the column name & respective datatype.

step5: Now with cursor object use the insert statement for entering the values corresponding to different fields, corresponding to the datatype.

step6: use the commit() to complete the transaction using the connection object.

step7: use the execute statement ~~using~~ along with cursor_object for accessing the values from the database using the select from where clause.

step8: Finally, use the fetch() or fetchall() for displaying the values from table using the cursor_object.

Step 9:- Execute () & drop table syntax then terminating the database and finally use the close.