PERFORMANCE

Term	Remarks	Staff Member's Signature
I		
II	Good Completed	

Exam Seat No. _____

THAKUR COLLEGE OF SCIENCE & COMMERCE

NAAC Accredited with Grade "A" (3rd Cycle)

THAKUR TRUSTS

ISO 9001 : 2015 Certified

Degree College

Computer Journal

CERTIFICATE

SEMESTER II **UID No.** _____

Class F.Y.B.Sc **Roll No.** 1755 **Year** 2019-20

This is to certify that the work entered in this journal is the work of Mst. / Ms. ADARSH YADAV

who has worked for the year 2019-20 in the Computer Laboratory.

Teacher In-Charge

Head of Department

Date : _____

Examiner

INDEX

INDEX

No.	Title	Page No.	Date	Staff Member's Signature
7	Project on Student Record Management System using GUI and database.	56	2/3/20	

Practical → 01

Aim: Demonstrate the use of the different file accessing mode, different attribute of the file object and differentiate among the different read methods.

Algorithm:

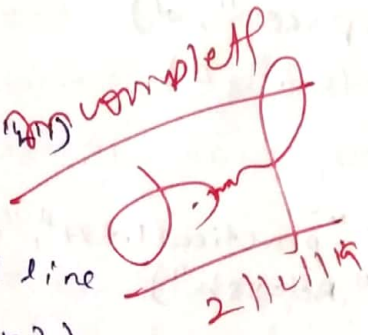
Step 1: Create a file object by using the open method and use the write access mode followed by writing some contents to the file and closing the file.

Step 2: Now open the file in the read mode and use the read ~~mode~~ method or read lines method and store the output in the variable and finally display the content in the variable

Step 3: Use the file object for finding the main of the file, mode in which the file is being opened whether the file is still open or close and finally the output of the softspace attribute.

Code :

```
fileobj = open ("practical1.txt","w")
fileobj.write ("Python is non procedural language
                \n It supports AI feature")
file.close ()
fileobj = open("practical1.txt","r")          # read ()
str1 =fileobj.read()
print ("The output of read method is ", str1)
fileobj.close ()
~~fileobj.close()~~
fileobj = open ("practical1.txt",
                                              # readline ()
str2 = fileobj.readline ()
print ("The output of readline
        method is ", str2)
fileobj.close ()
fileobj = open ("practical1.txt","r")         # readlines ()
str3 = fileobj.readlines ()
print ("The output of readline method is", str3)
a = fileobj.name
b = fileobj.closed
c = fileobj.mode
d = fileobj.softspace
print (a,b,c,d)
fileobj.close ()
```



Output:

```
('The output of read method is', 'Python is non procedural
  language \n It supports AI featured)
('The output of readline method is', 'Python is non-Procedural language
                                                    \n')
('The output of readlines method is', ['Python is non procedural
  language \n', ' It supports AI feature'])
('practical1.txt', False, 'r', 0)
```

```
# file attributes
a = fileobj.name
print ("name of file (file attribute) :", a)
>>> ('name of file ( name attribute ) ', 'practical1.txt')
b = fileobj.closed
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 ("practical1.txt", "w+")
fileobj.write ("Adarsh")
fileobj.close ()
```

```
# r+ mode
fileobj = open ("practical1.txt", "r+")
str1 = fileobj.read (6)
print ("output of r+ ", str1)
fileobj.close ()
>>> ('output of r+', 'Adarsh')
```

```
# write mode
fileobj = open ("practical1.txt", "w")
fileobj.write ("Programming with
                .python")
fileobj.close ()
```

```
# read mode
fileobj = open ("practical1.txt", "r")
str2 = fileobj.read ()
print ("Output of read mode!",
fileobj.close                    str2)
>>> ('output of read mode!',
    'programming with python')
```

Step 4: Now open the fileobject in write mode, write some another content close subsequently then again open the fileobject in 'w+' mode that is the update mode and write contents.

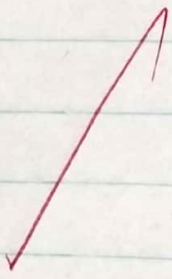
Step 5: Open file object in readmode, display the update written contents and close. Open again in 'r+' mode with parameter passed and display the output subsequently.

Step 6: Now open fileobject in append mode.. open write method. write content close the fileobject again. Open the fileobject in readmode and display the append output.

Step 7: Open the fileobject in readmode, declare a variable and perform fileobject dot. tell method and store the output consequently in variable.

Step 8: Use the seek method with the arguments with opening the fileobject in readmode and closing subsequently.

Step 9: Open fileobject with readmode also use the readlines methods and store the output consequently in and print the same for counting the length use the for condition statement and display the length.



```
# append mode
fileobj = open ("practical1.txt", "a")
fileobj.write ("data structure")
fileobj.close()
fileobj = open ("practical1.txt", "r")
str3 = fileobj.read()
print ("output of append mode :", str3)
fileobj.close()
>>>('output of append mode :', 'Adarsh', 'data structure')

# tell ()
fileobj = open ("practical1.txt", "r")
pos = fileobj.tell()
print ("tell (): ", pos)
fileobj.close
>>> ('tell (): ', pos)

# seek ()
fileobj = open ("practical1.txt", "r")
str4 = fileobj.seek (0, 0)
str8 = fileobj.read (10)
print ("The beginning of the file : ", str8)

# Finding length of different lines exist within line
fileobj = open ("practical1.txt", "r")
str9 = fileobj.readlines ()
print ("output :" str9)
For line in str9 :
        print (len (line))

fileobj.close ()
]>>> ('output :', ['college database'])
```

Code :

```
mytuple = ("Adarsh", "Ashitosh", "Yash", "Dinesh")
myiter = iter(mytuple)
print(next(myiter))
print(next(myiter))
print(next(myiter))
print(next(myiter))
```

output :

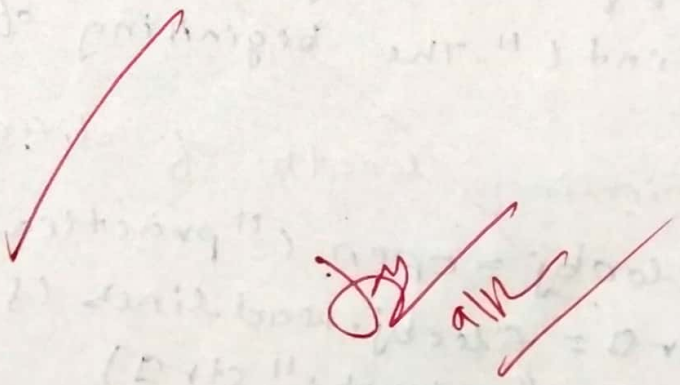
Adarsh
Ashitosh
Yash
Dinesh

Code :

```
mytuple = ("Adarsh", "Ashitosh", "Yash", "Dinesh")
For a in mytuple :
        print(a)
```

output :

Adarsh
Ashitosh
Yash
Dinesh



Practical - 02

Aim:

a) To display elements of a tuple using iterator method

Algorithm:

Step 1: Form a tuple with certain elements inserted in it.

Step 2: Use iter method with tuple and assign it to a variable.

Step 3: Use the next method with variable and print elements.

b) To use iter method with for loop.

Algorithm:

Step 1: Form a tuple with certain elements inserte in it.

Step 2: Use the for conditional statement to access each element of tuple.

Step 3: Print the elements of tuple.

c) Write a program using the iterable method for displaying the set of odd numbers.

Algorithm:

Step 1: Define a class which will contain various function.

Step 2: Define the iter method with an argument and return the value of argument.

Step 3: Define a function which increments the value of argument by two.

Step 4: Create an object which inherits the properties of class and take the user input.

Step 5: Use the for conditional statement followed by if conditional statement and print the answer.

Program:

```
class odd :
    def __iter__(self):
        self.num = 1
        return self
    def next(self):
        num = self.num
        self.num += 2
        return num
myobj = odd()
myiter = iter(myobj)
x = int(input("Enter range number :"))
for i in myiter:
    if i<x:
        print(i)
```

Output:

```
Enter a number : 16
1
3
5
7
9
11
13
15
```

Program:

```
class myclass:
    def __iter__(self):
        self.a = 1
        return self
    def __next__(self):
        if self.a <= 20:
            x = self.a
            self.a += 1
            return x
        else:
            raise StopIteration

myobj = myclass()
myiter = iter(myobj)
for x in myiter:
    print(x)
```

Output:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

d) To print first 20 numbers using iter method:

Algorithm:

Step 1: Define a class which will contain various functions.

Step 2: Define iter method with an argument and return the value of argument.

Step 3: Define next method which increments the value of argument by 1 and prints it.

Step 4: Create a object which inherits the propert of class and take the user input.

Step 5: Use the loop to print to print the value of the variable.

ASO

e) To find if the number is odd or even from given list using map method.

Algorithm:

Step 1: Declare a list num variable and declare some elements.

Step 2: Define a function even which consist various conditional statement.

Step 3: Further use if conditional statement to check the modulo number of each element of list and return the message accordingly.

Step 4: Use the map method to print the value in list format.

Program:

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

output:

[0, 4, 5, 7, 9, 11, 13, 15, 20, 9]

[even, even, odd, odd, odd, odd, odd, odd, even, odd]

Program:

```
def square (x):
    return (x**2)
def cube (x):
    return (x**3)
funct1 = [square, cube]
For i in range (4):
    value = list (map (lambda x: x(2), funct1))]
print (list (value))
```

Output:

```
[0, 0]
[1, 1]
[4, 8]
[9, 27]
[16, 64]
```

f) To find square and cube of number simultaneously using map method.

Algorithm:

Step 1: Define a function which returns the square of given number.

Step 2: Define an another function which returns cube of given number.

Step 3: Store the output simultaneously in the list.

Step 4: Use the for loop followed by the map function and print the value of result.

9) To find square of a number without using map method.

Algorithm:

Step 1: Define a list which contains certain values.

Step 2: Define an empty list.

Step 3: Use for loop followed by append method to append the result into the empty list.

Step 4: Print the value of list.

Program:

```
list1 = [1, 2, 3, 4, 5]
empty = [ ]
for i in list1:
    empty.append (i**2)
print (empty)
```

output:

[1, 4, 9, 16, 25]

Program:

```
try:
    fileobj = open ("abc.txt", "w")
    fileobj.write ("Python is an intended language")

except IOError:
        print ("It is an Environmental Error!")

else:
        print ("operation is successful")
```

Output:

```
operation is successful
```

PRACTICAL - 03

Aim: To write a program using exception block related to environment error.

Algorithm:

Step 1: Open a previous file using any mode of file operation under try block.

Step 2: Define the except block to print the message for the error.

Step 3: Define the else block to print the message accordingly if no error is found.

Aim: Write a program to demonstrate the use of value error in given program.

Algorithm:

Step 1: Accept the user input of integer datatype in the try block.

Step 2: Define the except block with value error as a keyword and display appropriate message.

Step 3: Define else block if none of the error is encounter and display message accordingly.

Program:

```
try:
    x = int(input("Enter a number:"))
except ValueError:
        print("This is Value Error!")
except IOError:
        print("This is environmental error!")
else:
        print("operation successful!")
```

Output:

```
Enter a number: 20
operation successful!

Enter a number: 2+3
This is Value Error!
```

31

080

Program :

```
import re
sequence = input (" Enter name and roll no. of students
                                    without space :")
pattern = r'\d+'
output = re.findall (pattern , sequence)
print (output)
```

Output :

```
Enter name and roll no. of students without space :
Adarsh1755 Neeraj1741 Sachin1744
['1755', '1741', '1744']
```

Aim: Regular Expression

1) To segregate alphabetical value from numerical value in the given string.

Algorithm:

Step1: Import the regular expression library.

Step2: Initialize a variable which takes the user input that contains name & roll no. of student without space.

Step3: Define the pattern as per the requirement of program.

Step4: Use the findall method and print the output as per the input of pattern.

2) To extract consecutive two characters which are available at the word boundries.

Step 1: Import the regular expression library.

Step 2: Take a sentence from the user and assign it to a variable.

Step 3: Define a pattern that finds the consecutive character at word boundaries.

Step 4: Use the findall method and print the output.

Program :

```
import re
sentence = input("Enter a sentence :")
pattern = r'\b\w{2}|\w{2}\b'
output = re.findall(pattern, senten
print(output)
```

Output :

Enter a ~~sentence~~ sentence : Hello there, my name is Neeraj
['He', 'lo', 'th', 're', 'my', 'na', 'me', 'is', 'Ne', 'aj']

Program:

```
import re
x = input ("Enter a sentence : ")
y = input ("Enter a word to be found from the
                                    string :")
output = re.findall (y,x)
if (output):
    print ("match found")
else:
    print ("match not found")
```

Output:

i) Enter a sentence : My name is Adarsh
Enter a word to be found from the string : name
Match found

ii) Enter a sentence : My name is Adarsh
Enter a word to be found from the string : sachin
Match not found.

3) To find if a word is present in a given string.

Algorithm:

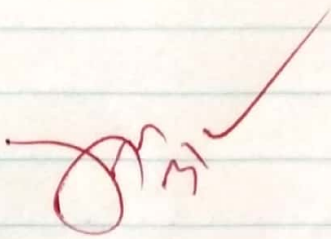
Step 1: Import regular ~~reg~~ expression library.

Step 2: Define a variable which take a sentence from the user.

Step 3: Define another variable which takes the word to be find.

Step 4: Use the findall method to generate the output

Step 5: Use the if conditional statement and display the message accordingly.



4) To find a word starting from vowel.

Algorithm:

Step 1: Import the regular expression library

Step 2: Take few lines as input from the user.

Step 3: Define a pattern which finds the vowels in the string.

Step 4: Use the findall method to find the word ~~string~~ starting with vowel.

Step 5: Print the output.

Program:

```
import re
string = input("Enter few lines:")
output = re.findall(r'\b[a,e,i,o,u]\w+|\b[A,E,I,O,U]\w+',string)
print(output)
```

output:

Enter few lines: My name is Neeraj. I am student.

['is', 'am']

Program:

```
import re
string = input ("Enter few names:")
pattern = re '\s+'
replace = ' '
output = re.sub(pattern, replace, string)
print (output)
```

Output:

Enter few names : Adarsh Yadav

AdarshYadav

5) Aim: To remove white space in a given string.

Algorithm:

Step 1: Import regular expression library.

Step 2: Take the string input from user which contains whitespace.

Step 3: Define the pattern as per the requirement.

Step 4: Assign a variable with empty string.

Step 5: Use the sub method to substitute the whitespace with string and print the output.

6) To find numbers starting with either 8 or 9 and with length 10 from a given list.

Algorithm:

Step 1: Import regular expression library.

Step 2: Define a list variable which contains the input numbers.

Step 3: Define the pattern required to extract numbers which satisfy the condition.

Step 4: Use the for loop followed by the if conditional statement and print the message accordingly.

program:

```
import re
list1 = ['8767522106', '9821971119', '7029523410',
         '9825904817']
pattern = r'\b[8-9]+\d{9}'

for i in list1:
    if re.match(pattern, i) and len(i)==10:
        print("Number satisfies the condition")

    else:
        print("Number does not satisfies the
                                    condition")
```

Output:

Number satisfies the condition
Number satisfies the condition
Number does not satisfies the condition
Number does not satisfies the condition

Program :

```
import re
email = ['xyz@tcsc.edu', 'abc@gmail.com', 'JKL@hotmail.com']
pattern = r'[@]'
for i in email:
    output = re.split(pattern, i)
    print(output)
```

Output :

```
['xyz', 'tcsc.edu']
['abc', 'gmail.com']
['jkl', 'hotmail.com']
```

7) To segregate the hostname and domain name from the email address

Algorithm:

Step 1: Import the regular expression library.

Step 2: Assign a variable which contains the email address in list datatypes.

Step 3: Define a pattern required to split the two parts.

Step 4: Use the for loop followed by re.split method to segregate the hostname and domain name and print the output.

T&O

PRACTICAL - OS (A)

TOPIC : GUI COMPONENTS : LABEL WIDGET, TEXT METHOD.

#1) Step1 : Use the tkinter library for importing the features of the text widget.

Step2 : Create an object using the Tk()

Step3 : Create an ~~object~~ variable using the widget Label and use the text method.

Step4 : Use the mainloop() for triggering of the corresponding above mention events.

#2)

Step1 : Use the tkinter library for importing the features of the text widget.

Step2 : Create a variable from the text method and position it on the parent window.

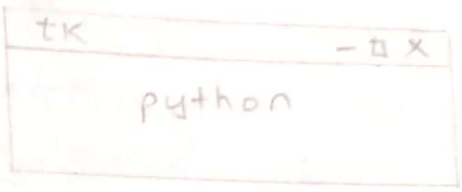
Step3 : Use the pack() with the object created from the ~~text~~ text() and use the parameter.

1) side = LEFT , padx = 20
2) side = LEFT , pady = 30
3) side = TOP , ipadx = 40
4) side = TOP , ipady = 50

creation of parent window

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

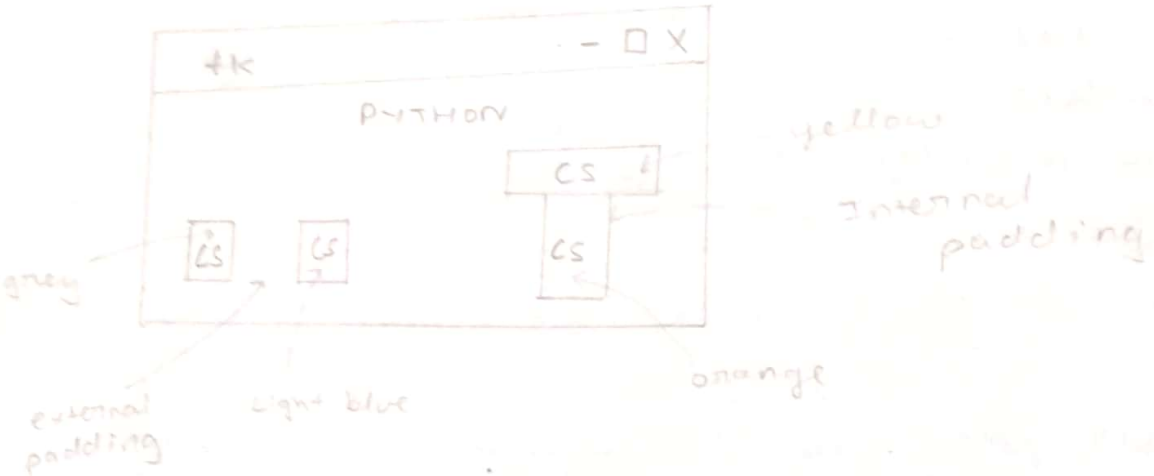
Output:



#2:

```
from tkinter import *
root Tk()
l = label (root , text = "python")
l.pack()
l1 = label ( root, text = "cs", bg = "grey", fg = "black", font = "10
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 = "10")
l3 .pack (side = TOP, ipadx = 40)
l4 = Label (root, text = "cs", bg = "orange", fg = "black",
                                                    font = "10")
l4. pack ( side = TOP , ipady = 50)
root.mainloop()
```

output:



Radiobutton

```
from tkinter import *
root = TK()
root.geometry ("500x500")
def select():
    selection = "You just selected" + str(var.get())
    l1 = Label(text = selection, bg = "white", fg = "green")
    l1.pack (side = TOP)
var = StringVar()
l1 = Listbox()
l1.insert (1, "List 1")
l2 = Listbox()
l2.insert (2, "List 2")
l1.pack (anchor = N)
l2.pack (anchor = S)
```

Step4: Use the mainloop() for the triggering of the corresponding events.

Step5: 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 color (bg)
4) The foreground (fg) and then use pack() with a relevant padding attributes.

PRACTICAL - 5(B) : ~~MessageBox~~, Radiobutton, ~~Multiple~~ ~~Window~~ Scrollbar and Frame Widg

Step1: Import the relevant method from the tkinter library. create an object with the parent window.

Step2: Use the parent window object along with the geometry() declaring specific pixel size of the parent window.

Step3: Now define a function which tells the user ~~about~~ 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.

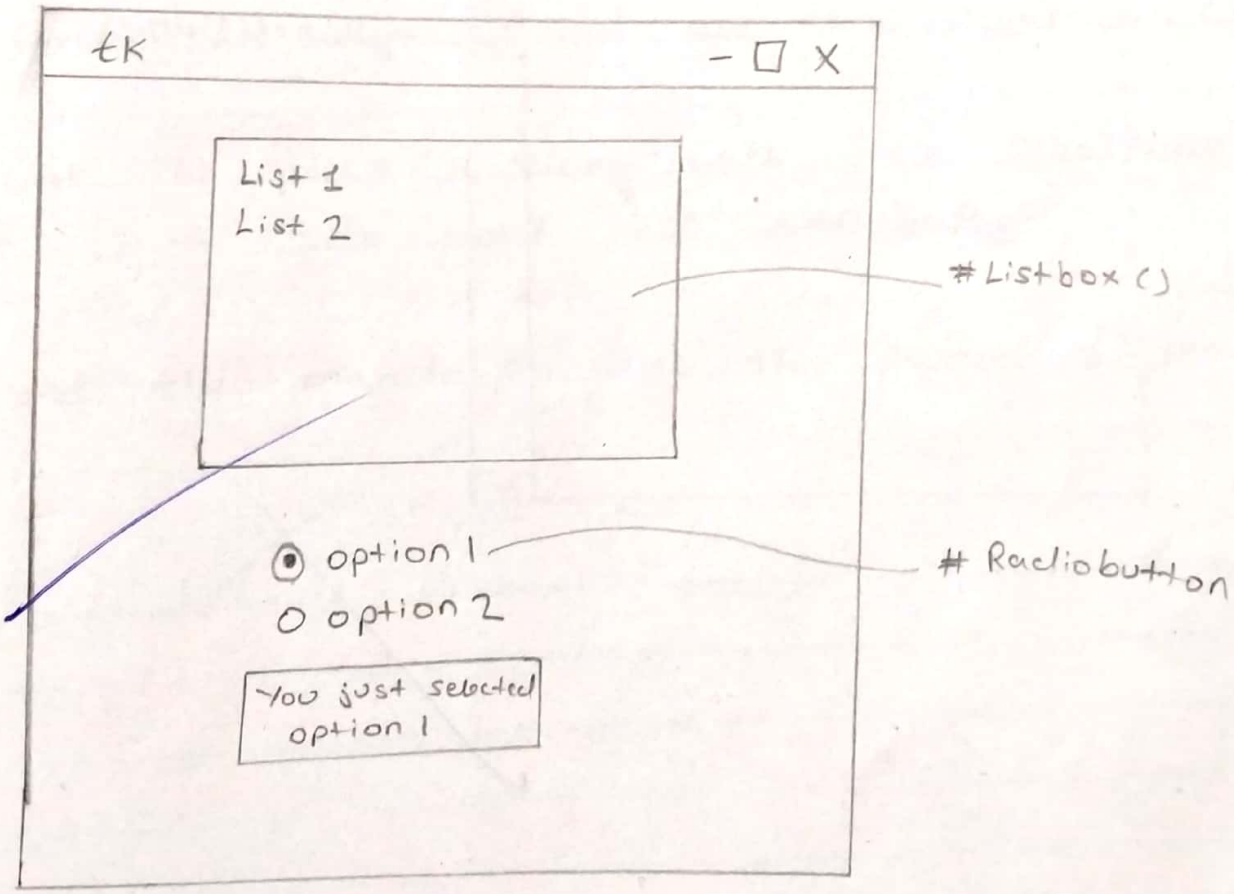
Step 6: Create an object from radio button which will take following arguments parent window object, text variable which will take the values option no 1, 2, 3, ... variable argument, corresponding value and trigger the function declared

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

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

```
r1 = Radiobutton (root, text = "option 1", variable = var,
     value = "option 1", command = select)
r1.pack(anchor = N)
r2 = Radio button (root, text= "option 2", variable = var,
     value = "option 2", command = select)
r2.pack(anchor = N)
root.mainloop ()
```

output :

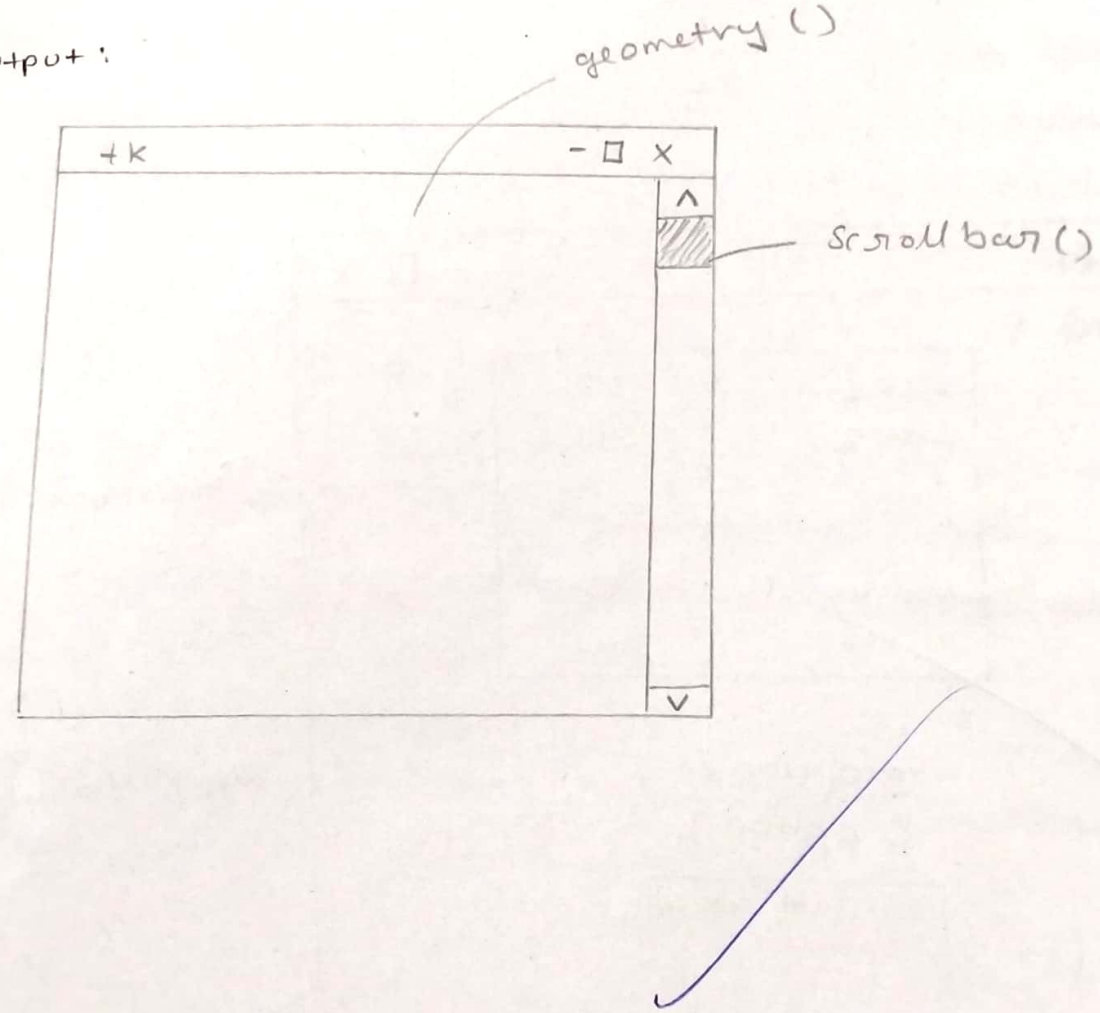


#2

Scrollbar()

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

output:



#2 :

Step 1: Import relevant method 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 attribute.

Step 6: use the mainloop with the parent object.

#3

Step 1: Import the relevant libraries from the tkinter method.

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

Step 3: Use the geometry manager with pixel size (680×500) 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 ~~the~~ use the pack method.

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

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

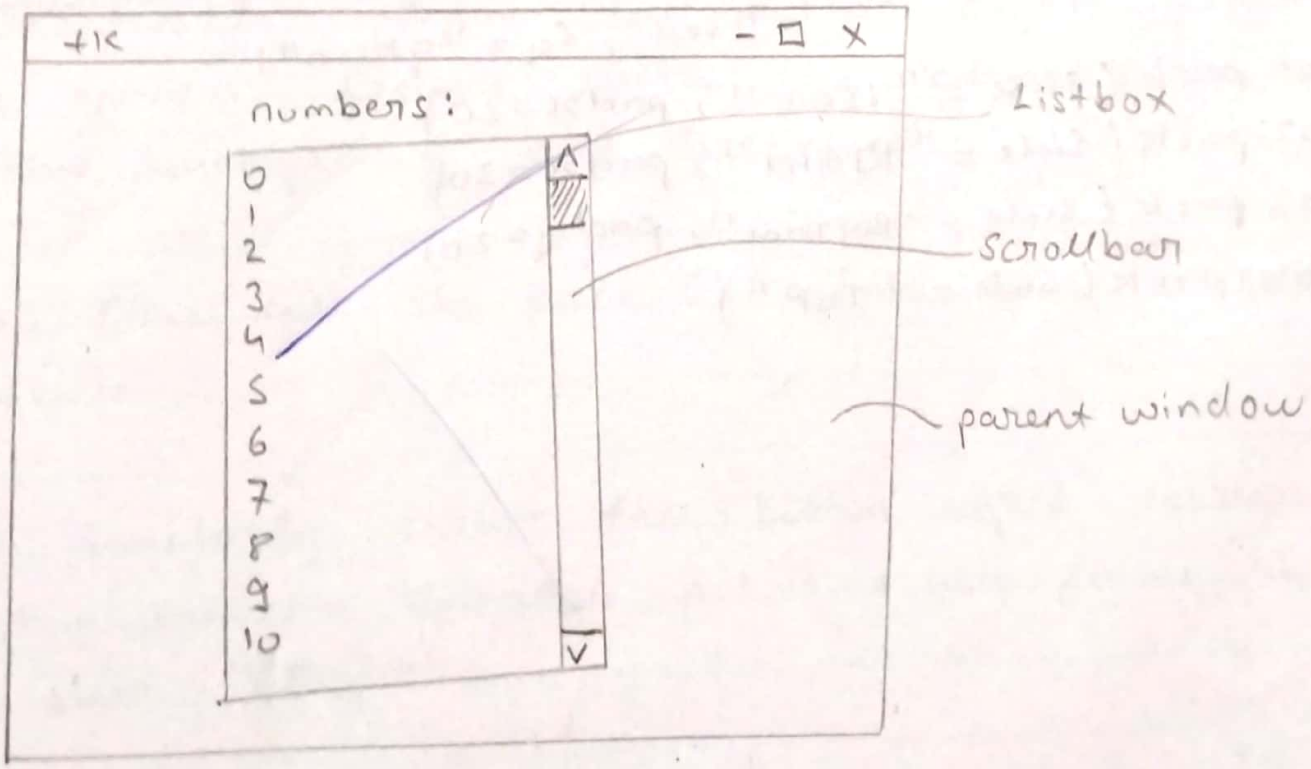
Step 8: Trigger the events using mainloop method.

#3

using frame widget

```
from tkinter import *
window = Tk()
window.geometry("680x500")
Label(window, text="numbers:").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 :



SAO

#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
                    = "blue", fg = "red")
b3 = Button(frame, text = "ADD", activebackground
                    = "yellow", fg = "black")
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")
```

Step1: Import relevant method from tkinter library.

Step2: Define the object corresponding to parent window and define the size of parent window in terms of no. of pixels.

Step3: Now define the frame object from the method and place it on to the parent window.

Step4: Create another frame object termed as the left frame and put it on the parent window on its LEFT side.

Step5: Similary define the RIGHT frame and subsequently define the button object placed onto the given frame with the attribute as text, active background and foreground.

Step6: Now use the pack() along with the side attribute.

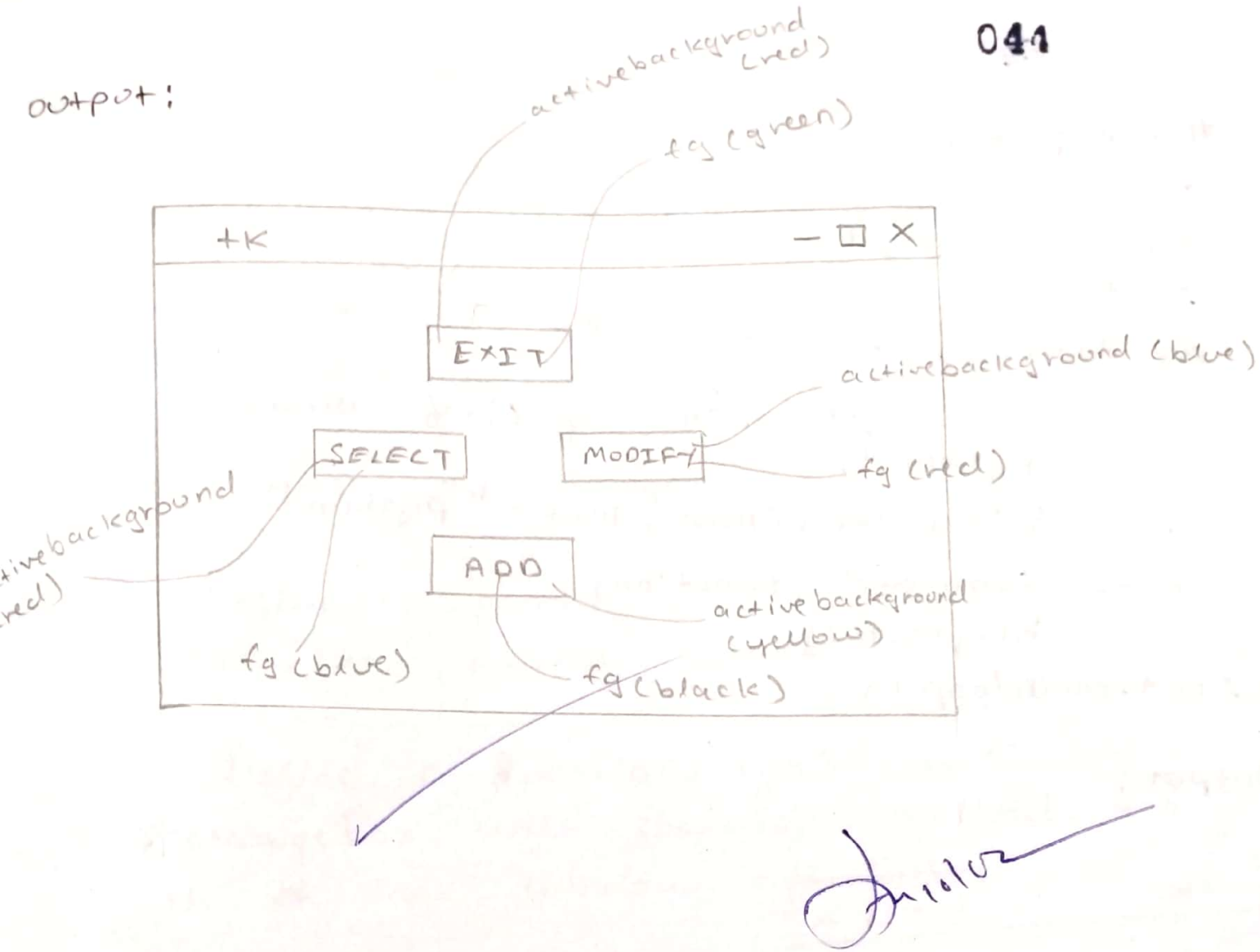
Step7: Similarly create the button object corresponding to the MODIFY operation put it into frame object on side = "right".

Step 8 : Create another button object and place it onto the RIGHT frame and label the button as ADD.

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

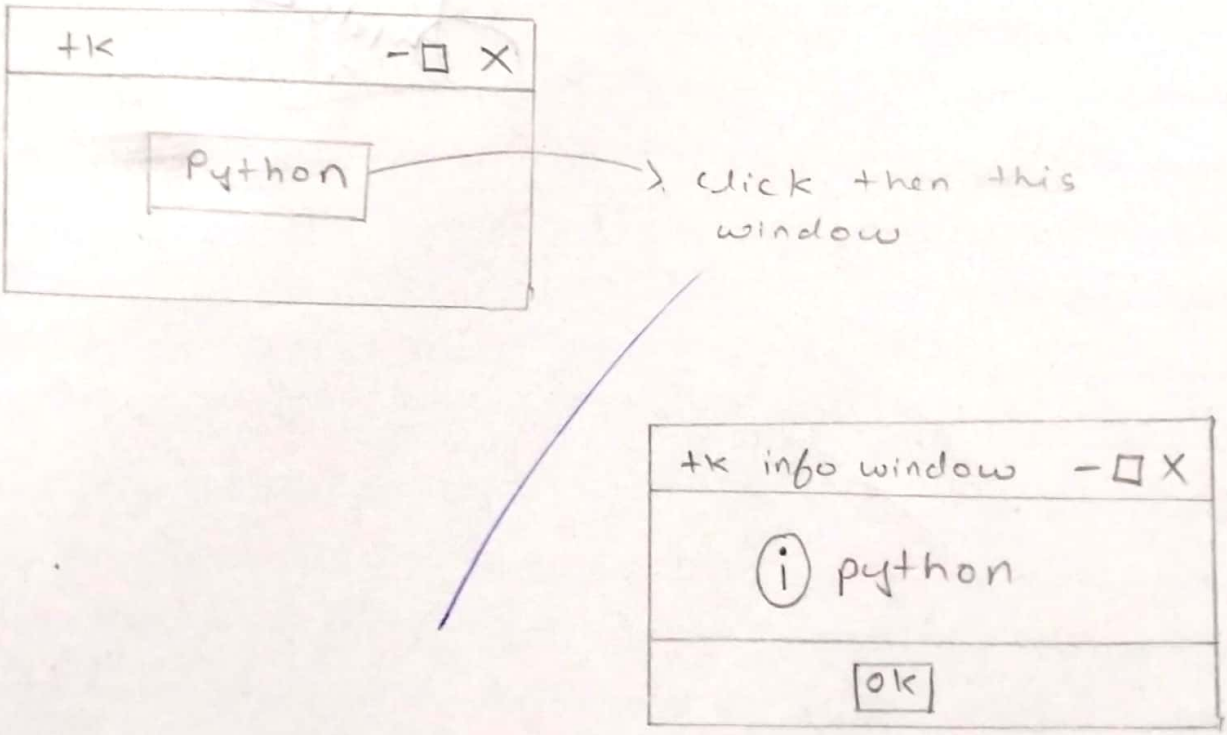
Step 10 : Use the pack() simultaneously for all the objects and finally use the mainloop().

output:



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

Output:



PRACTICAL → 5 (c)

Aim: GUI Components : Message Box and MULTIPLE WINDOW

#1 Step 1: Import the relevant methods from tkinter library.

Step 2: Import tk Messagebox.

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

Step 4: Define a function which will use tk Messagebox 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.

#2

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

Step2: Use parentwindow object along with minsize function for window size.

Step3: Define a function main, declare present window object and use config(), title(), minsize(), 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.

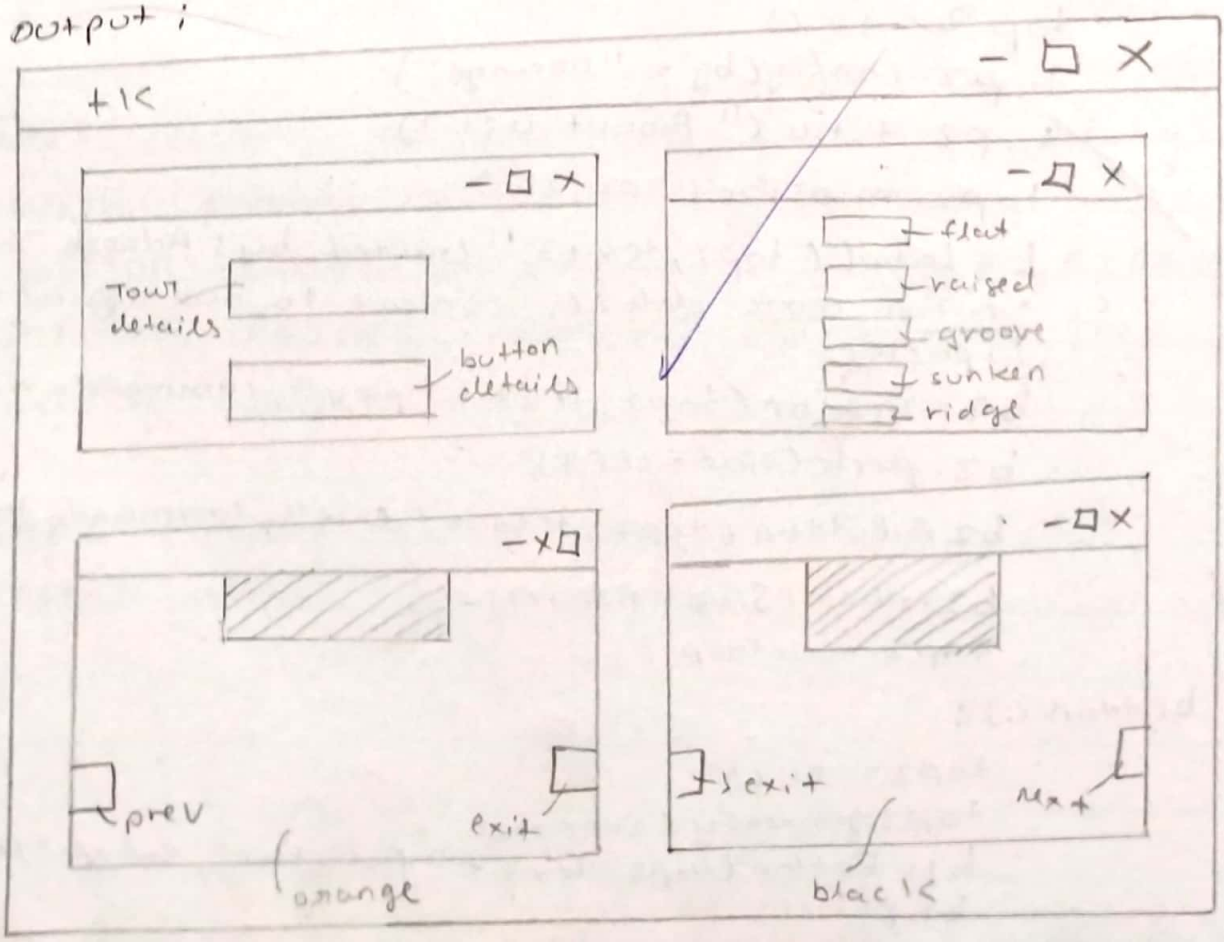
Step6: Finally called the mainloop() for event driven programming.

#multiple window
Different 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
    Interests \n Golden gate Bridge \n Lombared 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()
def second():
    top2 = Tk()
    top2.config(bg = "orange")
    top2.title("About us!")
    top2.minsize(300,300)
    L = Label(top2, text = "Created by: Adarsh Yadav
    \n For more details contact to our 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()
b5 = Button (top3, text = "ridge button", relief = RIDGE)
b5.pack()
top3.mainloop()
def terminate():
    top4 = Tk()
    top4.geometry("300*300")
    b6 = Button (root, text = "TOUR DETAILS", command = main)
    b6.pack()
    b7 = Button (root, text = "Button DETAILS",
                 command = button
    b7.pack()
    top4.mainloop()
root.mainloop()
```

output :



TAD

PRACTICAL - 5 (D)

Aim: GUI Components : Grid Method and PhotoImage

Step1: Import relevant methods from the tkinter library.

Step2: 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 and subsequently terminate the program.

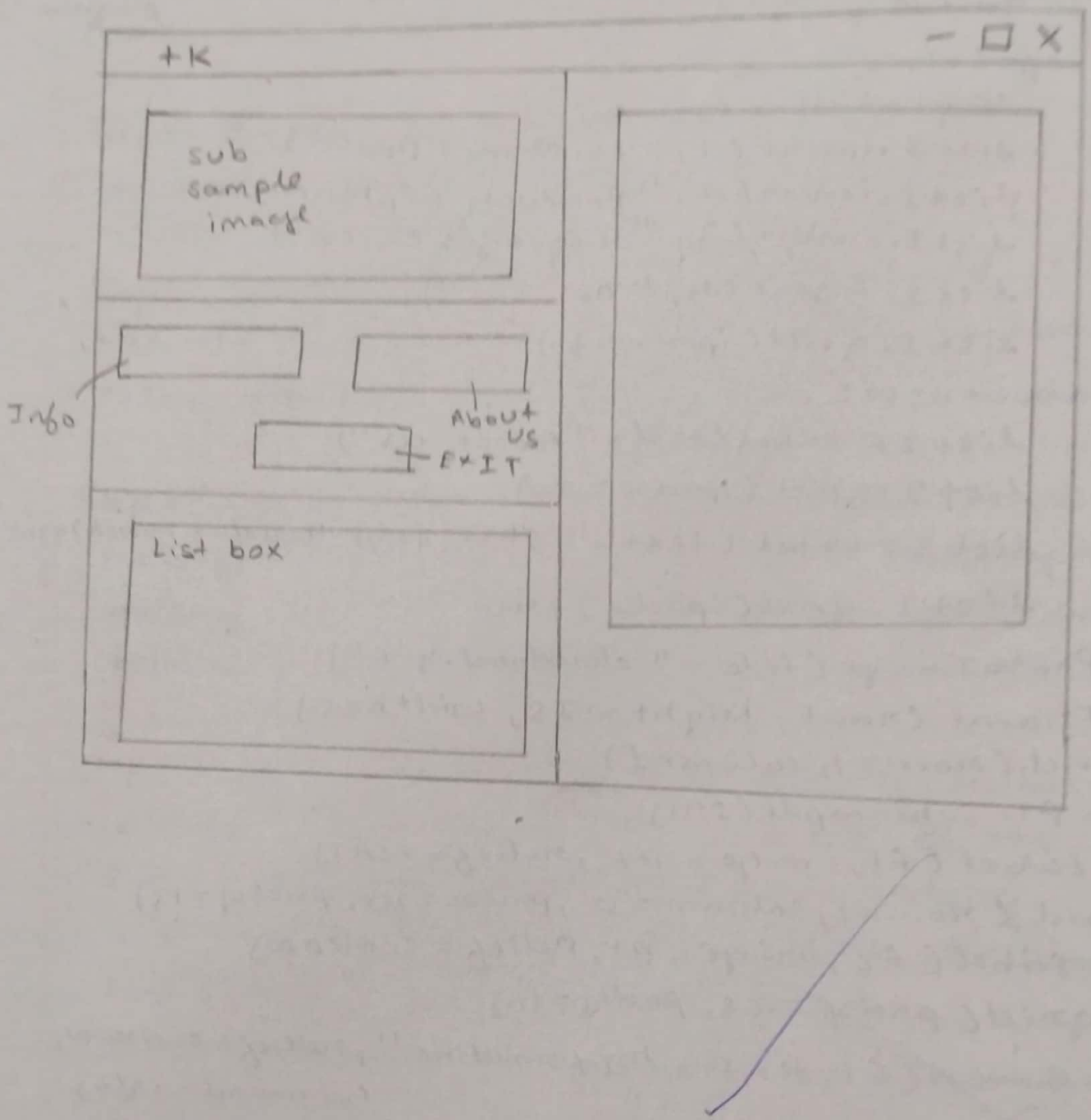
Step 4: Define a function info use a listbox widget along with the object of the same. Use the listbox object along with insert method and insert the same and finally use the grid() with ipadx attribute.

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

Code :


```
from tkinter import *
root = Tk()
root= config( bg = "gray")
def finish():
        messagebox. askokcancel ("warning", This will end the
        quit()                                              program")
def info():
        list1 = Listbox()
        list1.insert(1, "1. Name : Apple")
        list1.insert(2, "Products : iphone")
        list1. insert(3, "language: swift")
        list1. insert(4, "OS: IOS")
        list1. grid(ipadx=30)
def about us():
        list2 = Label(text = "About us")
        list2. grid (ipadx = 30)
        list3 = Label (text = "Steve Jobs thereafter March 2020")
        list3. grid(ipadx = 24)
P1 = PhotoImage (file = "download.gif")
f1 = Frame (root, height = 35, width=5)
f1. 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 (row = 1, col = 0)
b2 = Button (f1, text = "About us", relief = SUNKEN,
                                        command = about us)
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()
```

output :



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 & 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 widge along with the frame object, relief attribute and subsequently use the grid().

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

10/02

PRACTICAL - 5(E)

Aim: To make use of Spinbox widget, Paned window and Canvas widget.

Spinbox widget:

Step 1: Create an object from the Tk method and subsequently create an object from the spinbox method.

Step 2: Make the object so created onto the parent window and trigger the corresponding events.

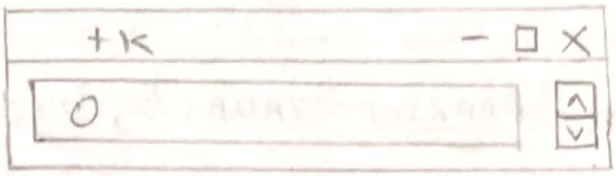
Step 3: Use the pack method to provide the direction using anchor method.

Step 4: Use the mainloop method to terminate.

Source code:

```
from tkinter import *
root = Tk()
s1 = Spinbox(root, from_=0, to_=10)
s1.pack(anchor = S)
root.mainloop()
```

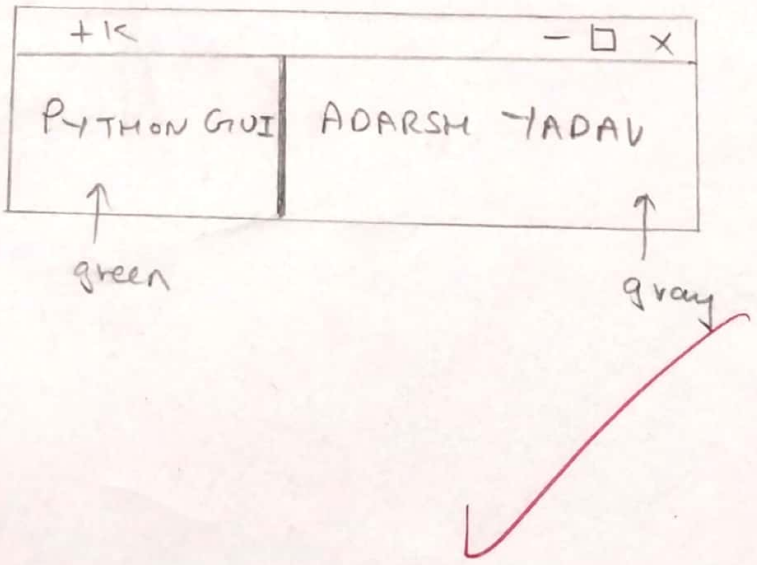
Output:



Source code:

```
from tkinter import *
root = Tk()
P = PanedWindow (bg = "red")
P.pack (fill = BOTH, expand = 10)
L1 = Label (P, text = "PYTHON GUI", bg = "green")
P.add (L1)
P1 = PanedWindow (P, orient = VERTICAL, bg = "blue")
P.add (P1)
L2 = Label (P1, text = "ADARSH YADAV", bg = "grey")
P1.add (L2)
root.mainloop()
```

Output:



Panned Window:

Step 1: Create an object from panned window and use the pack method with the attribute fill and expand.

Step 2: Create an object from the label method and put it onto the panned window with the text attribute and use the add method to embed the new object.

Step 3: Similarly create another label object and place it onto the 2nd paned window object and add onto the 2nd pane

Step 4: Create a second paned window object and add it onto the 1st paned window with orientation specified.

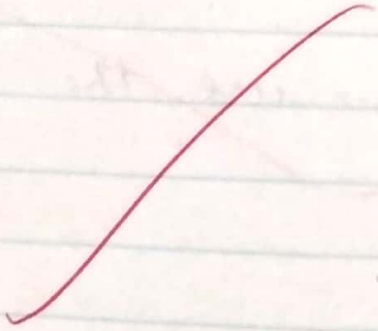
Step 5: Now use the mainloop method to terminate.

* Canvas Widget :

Step 1 : Use the tkinter method and create an object from canvas method and use the attribute height, weight, bg colour and the parent window object.

Step 2 : Use the method create oval, create line and create arc along with the canvas object so created and use the co-ordinate value. Also use fill attribute to assign various colours.

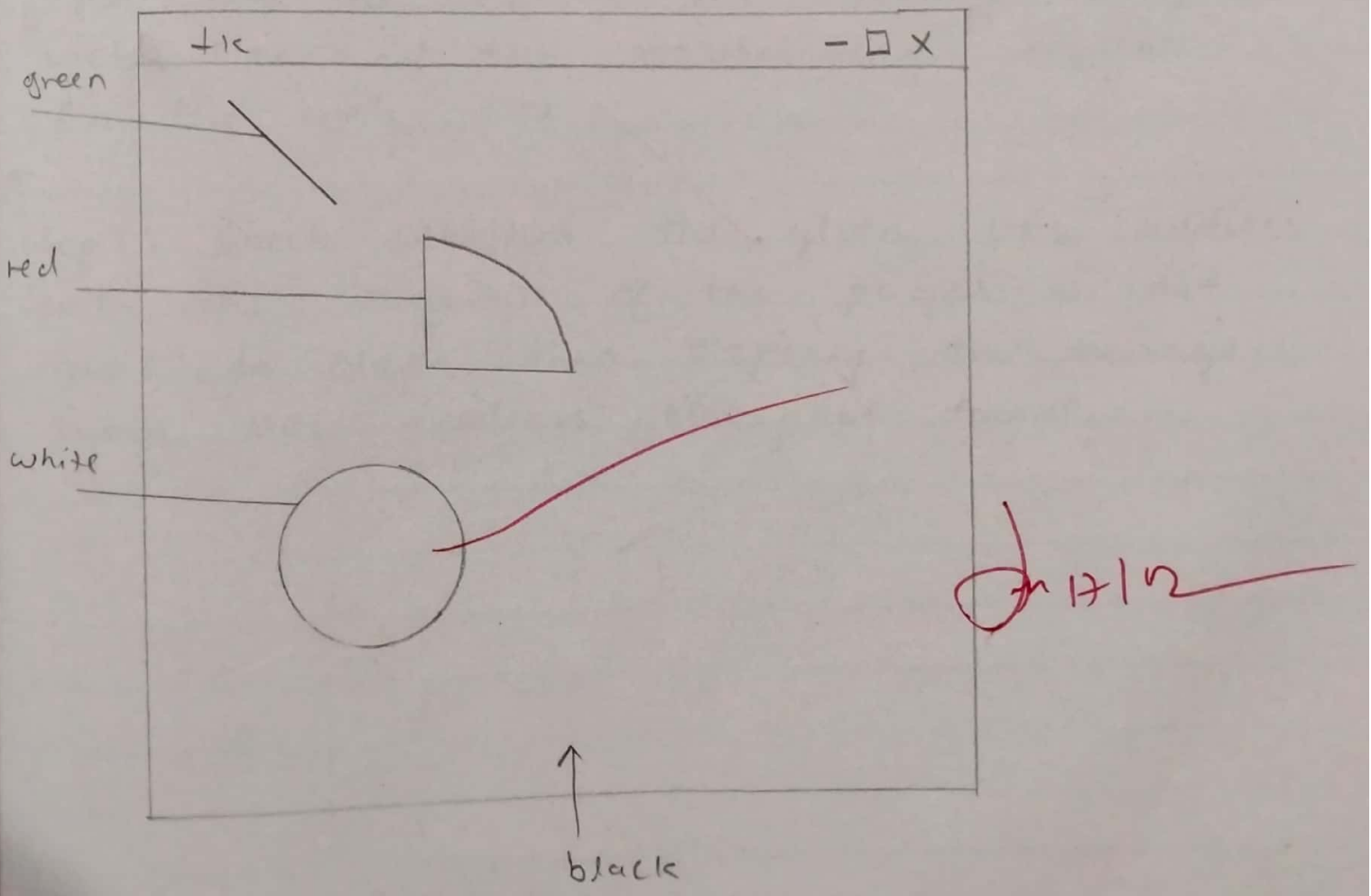
Step 3 : Now call the pack method and the mainloop method.



Source code:

```
from tkinter import *
root = Tk()
c1 = canvas(root, height = 400, width =400, bg = "black")
oval = c1.create_oval (20, 140, 150, 250 , fill= "white")
line = c1. create _oval (30,40, 50, 60, fill = "green")
arc = c1. create _oval ( 20,140,150, 60, fill = "red")
c1.pack()
root.mainloop()
```

Output:



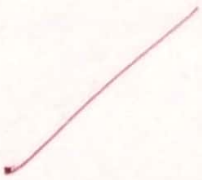
17/12

Program:

```
import dbm
db = dbm.open("database", flag = "c")
if db["www"] != None:
        print("good")
else:
        print("Not good")
```

output:

good



PRACTICAL - 06

Aim : Database.

Algorithm :

Step 1 : Import db library and use the open method for creating the database by specify name of the database along with the corresponding flag.

Step 2 : Use the objects for accessing to given web size and the corresponding regular for the web size.

Step 3 : Check whether the given URL address with the regular of the pages is not equal to None then Display the message from URL address else not found.

2) Algorithm:

Step 1: Import the corresponding library taking of database connection.

Step 2: Now create connection objects using Sqlite Library and connecting method for create the new database.

Step 3: Now create the cursor object using Cursor method from the connection objects create steps.

Step 4: Now use the executing method for creating the table with the column name and respective datatype.

Step 4: Now use the executing method for creating the table with the column name and respective datatype.

Step 5: Now with the cursor object use insert statement for entering

Step 6: use the commit method

Program:

```
import os, sqlite3
connection = sqlite3.connect ("student.db")
c1 = connection.cursor()
c1.execute ('Create table student (Name, RNO,
                                   DOB)')
c1.execute ('insert into student values
            ("Adarsh", 1755, 21-01-2000)')
c1.execute ('insert into student values
            ("Abhinav", 1772, 23-03-2000)')
c1.execute ('insert into student values
            ("Pinank", 1762, 02-02-2000)')
connection.commit()
c1.execute ('select * from student')
c1.fetchall()
c1.execute ('Drop table student')
```

Output:

[('Adams', 1755, 21-01-2000), ('Abinav', 1772, 23-03-2000), ('Pinank', 1762, 02-02-2000)]

step7: Use the execute statement along with the cursor objects from accessing the valve the data base using selecting from where clause.

step8: Finally use the fetchall method to display the value for the table using the cursor objects.

step9: use the execute method and the drop table.

Python Project

TOPIC: Student Record Management System using GUI and database.

Summary: I have created a project which takes the detail of the student and stores in database. It makes user to easily enter the detail instead of making changes or adding something in the source code. It can take multiple entries. It makes our work easier.

Student Record Management System

student name:	Adarsh
Roll no:	1755
Branch:	CS
Phone Number:	8169908906
Father Name:	Ramprasad
Address:	Kandivali

☑ Make sure all entries are right

Add Student

View ALL students

DELETE STUDENT

CLOSE

ADDED SUCCESSFULLY

ADDED SUCCESSFULLY

('Abhinav', 1772, 'cs', 9969142526, 'vinod', 'Miva Road')

('Adarsh', 1755, 'cs', 8169908906, 'Ramprasad', 'Kandivali')

SUCCESSFULLY DELETED THE USER

source code:

```
from tkinter import *
import sqlite3, sys
def connection():
    try:
        conn = sqlite3.connect("student.db")
    except:
        print("cannot connect to the database")
    return conn
def verifier():
    a=b=c=d=e=0
    if not student_name.get():
        t1.insert(END, "<> student name is required<>\n")
        a=1
    if not roll_no.get():
        t1.insert(END, "<> Roll no is required <>\n")
        b=1
    if not branch.get():
        t1.insert(END, "<> Branch is required <>\n")
        c=1
    if not phone.get():
        t1.insert(END, "<> Phone number is required<>\n")
        d=1
    if not father.get():
        t1.insert(END, "<> Father name is required <>\n")
    if not address.get():
        t1.insert(END, "<> Address is Required <>\n")
        f=1
```

```
if a==1 or b==1 or c==1 or d==1 or e==1
                                or f==1:
    return 1
else:
    return 0
def add_student():
        ret=verifier()
        if ret==0:
            conn=connection()
            cur=conn.cursor
            cur.execute("CREATE TABLE IF NOT
EXISTS STUDENTS( NAME TEXT, ROLL_NO INTEGE ,BRANCH TEXT,
PHONE_NO INTEGER, FATHER TEXT, ADDRESS TEXT)")
            cur.execute("Insert into STUDENTS
values(?,?,?,?,?,?)",(student_name.get(),int(rollno.get(
branch.get(),int(phone.get()),father.get(),address.get()))
            conn.commit()
            conn.close()
            t1.insert(END,"ADDED SUCCESSFULLY\n")
def view student():
    conn=connection()
    cur=conn.cursor()
    cur.execute("select * from STUDENTS")
    data=cur.fetchall()
    conn.close()
    for i in data:
        t1.insert(END,str(i)+"\n")
```

```
def delete_student():
        ret= verifier()
        if rete == 0:
            conn = connection()
            cur = conn.cursor()
            cur.execute("DELETE FROM STUDENTS
WHERE ROLL_NO = ?", (int(roll_no.get()),))
            conn.commit()
            conn.close()
            tl.insert(END, "SUCCESSFULLY DELETED
                                    THE USER\n"
def clse():
    sys.exit()

if __name__ == "__main__":
    root = Tk()
    root.title("Student Record Management System")
    student_name = StringVar()
    roll_no = StringVar()
    branch = StringVar()
    phone = StringVar()
    father = StringVar()
    address = StringVar()

    label1 = Label(root, text = "Student name:")
    label1.place(x=0, y=0)

    label2 = Label(root, text = "Roll no:")
    label3.place(x=0; y=30)
```

```
label3 = Label (root, text = "Branch :")
label3.place (x=0, y=60)

label4 = Label (root, text = "Phone Number:")
label4.place (x=0, y=90)

label5 = Label (root, text = "Father Name :")
label5.place (x=0, y=120)

label6 = Label (root, text = "Address :")
label6.place (x=0, y=150)

e1 = Entry (root, textvariable = student_name)
e1.place (x=100, y=0)

e2 = Entry (root, textvariable = roll_no)
e2.place (x=100, y=30)

e3 = Entry (root, textvariable = branch)
e3.place (x=100, y=60)

e4 = Entry (root, textvariable = phone)
e4.place (x=100, y=90)

e5 = Entry (root, textvariable = father)
e5.place (x=100, y=120)

e6 = Entry (root, textvariable = address)
e6.place (x=100, y=150)
```

```
c1 = Checkbutton (root, text = "make sure all entries are right")
c1.place (x=0, y = 180)
t1 = Text (root, width=80, height = 20)
t1.grid (row=10, column=1)

b1 = Button (root, text = "ADD STUDENT", command=
add_student, width = 40)
b1.grid (row=11, column=0)

b2 = Button (root, text = "VIEW ALL STUDENTS",
command = view_student, width = 40)
b2.grid (row=12, column=0)

b3 = Button (root, text = "DELETE STUDENT", command
= delete_student, width = 40)
b3.grid (row=13, column=0)

b4 = Button (root, text = "CLOSE", command = close,
width = 40)
b4.grid (row=15, column=0)

root.mainloop().
```