PRATICAL - 1

Objective :- Demonstrate the use of different file accessing modes different attributes read method.

Step1:- Create a file object using open method and use the write access mode followed by writing some contents onto the file and than closing the file

Step2:- Now open the file in read mode and than use read(), readline() and readlines() and store the output in variable and finally display the contents of variable

Step3:- Now use the file object for finding the name of the file, the file mode in which its opened whether the file is still open or close and finally the output of the softspace attribute

```
fileobj = open("abc.txt", "w")  # file open (write mode)
file obj. write("computer science subjects" + "\n")
file obj. write("DBMS \n Python \n DS \n")  # file write
file obj. close()  # file close.

fileobj = open("abc.txt", "r")  # read mode
# read().
str1 = file obj.read()
print("The output of read method:", str1)
file obj. close()
>>> ('The outputs of read method :', 'computer science
    Subjects \n DBMS \n Python \n DS \n')
# readline().
file obj = open("abc.txt", "r")
str2 = file obj.readline()
print("The output of readline method:", str2)
file obj. close()
>>> ('The output of readline method:', 'computer science
    subjects \n')
# readlines()
fileobj = open("abc.txt", "r")
str3 = file obj.readlines()
print("The output of readlines method:", str3)
file obj.close()
>>> ('The output of readlines method:', ['computer
    science subjects \n', 'DBMS \n', 'python \n', 'DS \n'])
# file attributes.
  as fileobj.name.
print("name of file (name attribute):", a)
>>> (name of file (name attribute)', 'abc.txt')
b = file obj.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 ("abc.txt", "w+")
fileobj.write ("Loukik sir")
fileobj.close()
```

```
# rt mode
fileobj = open ("abc.txt", "rt")
s1 = fileobj.read(6)
print ("output of rt", s1)
fileobj.close()
>>>('output of rt', 'Loukik')
```

```
# append mode
fileobj = open ("abc.txt", "a")
fileobj.write ("data structure")
fileobj.close()
fileobj = open ("abc.txt", "r")
s3 = fileobj.read()
print ("output of append mode:", s3)
fileobj.close()
>>> ('output of append mode:', 'Loukik sir data structure')
```

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

```
# read mode
fileobj = open ("abc.txt", "r")
g = fileobj.read()
print ("output of read mode:",
>>>('output of read mode:',
 'Loukik sir')
```

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

Step 5:- open file obj in read mode. display the update written contents and close open again in 'rt' mode with parameter passed and display the output subsequently.

Step 6:- Now open fileobj in append mode open write method write contents close the fileobj again open the fileobj in read mode and display the appending output

Step7:- open the fileobj in read mode declare a variable and perform fileobject dot tellmode and store the output consequently in variable.

Step 8:- use the seek method with the arguments with opening the file obj in read mode and closing subsequently.

Step 9:- open fileobj 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 conditional statement and display the length.

```
# tell()
fileobj = open("abc.txt", "r")
pos = fileobj.tell()
print("tell(): ", pos)
fileobj.close()
>>> ('tell(): ', 0)

# seek()
fileobj = open("abc.txt", "r")
st = fileobj.seek(0,0)
print("seek(0,0) is: ", set)
fileobj.close()
>>> ('seek(0,0) is: ', None)
fileobj = open("abc.txt", "r")
st1 = fileobj.seek(0,1)
print("seek(0,1) is: ", st1)
fileobj.close()
>>> ('seek(0,1) is: ', None)
fileobj = open("abc.txt", "r")
st2 = fileobj.seek(0,2)
print("seek(0,2) is: ", st2)
fileobj.close()
>>> ('seek(0,2) is: ', st2)
fileobj.close()
>>> ('seek(0,2) is: ', None)
# finding length of different lines exist within lines
fileobj = open("abc.txt", "r")
stat = fileobj.readlines()
print("output: ", stat)
for line in stat:
    print(len(stat))
fileobj.close()
>>> output: ['cookie kit sir data structure']
```

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))
```

```
>>> banona
    orange
    apple
```

for loop

```
mytuple1 = ("Kevin", "Stuart", "bob")
for x is mytuple1:
    print(x)
```

```
>>> Kelrin
    Stuart
    bob
```

Square and cube

```
def square(x):
    y = x * x
    return y
def cube(x):
    z = x * x * x
    return z
```

PRATICAL- 2

Objective:- Iterators

Step1:- Create a tuple with elements that we need to iterate using the iter and next method. the number of time we use the iter and next method we will get the next iterating element in the tuple. Display the same.

Step2:- The similar output can be obtained by using for conditional statement. An Iterable variable is to be declared in for loop which will iterate

Step3:- Define a function name square with a parameter which will obtain output of square value of the given number. In similar fashion declare cube to get the value raised 3. and return the same

Step4:- Call the declared function using function call.

Step5:- Using for conditional statement specifying the range use the list type casting with map method declare a 'lambda' i.e annonymus function. and print the same

Step6:- Delare a listnum variable and declare some elements then use the map method with help of lambda function give two argument display the output

Step7:- define a function even with a parameter then using conditional statements do cheek whether the number is even and odd and return respectively.

Step 8:- Define a class and within that define the iter() method which will initilize the first element within the container object.

Step 9:- Now use the next() and define the logic for displaying odd value

```
for r in range(5):
    values = list(map(lambda x: x(r), funct))
    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 t=2
    def __next__(self):
        num=self.num
        self.numt=0.
        return num
```

```
myobj = odd()
myiter = iter(myobj)
x = int(input("Enter a number: "))
for i in myiter:
    if(i<x):
        print(i)
```

>>> Enter a number: 15

1
3
5
7
9
11
13

Step 10:- Define an object of a class

Step 11:- Accept an number from the user till which we want to display the odd number.

Last
13/12/19

PRATICAL - 3

TOPIC:- Exceptions

Step1:- In the try block open an file in the open mode with write mode. write some content in file

Step 2:- In except (IOError) block use the error in IOError use the appropriate message to display.

Step3:- else display the operation is successful.

Step4:- In try block accept an input from the user.

Step5:- In except block use valueerror and print the same message.

step 6:- else display the operation is successful!

IOError:

```
try:
    fo= open ("abc.txt", "w")
    fo.write ("python is an intendent language")
except IOError:
    print("Enter appropriate mode for file operation!")
else:
    print ("operation is successful!")
```

>>> operation is successful!

R output:

>>> Enter appropriate mode for file operation!

value Error:

```
try:
  x= int (input ("Enter a statement:"))
except value Error:
    print ("ARITHMETIC ERROR!")
else:
  print ("operation is successful!")
```

>>> Enter a statement: abc
ARITHMETIC ERROR!

>>> Enter a statement: 123
Operation is successful!

Type error, zero division error

```
a=
b= input ("Enter:")
try:
    print (a/b)
except TypeError:
    print ("Incompatible value")
except ZeroDivisionError:
    print ("Denominator is zero so
        operation cannot be performed")
```

```
>>> Enter:t
Incompatible value
>>> Enter:2
0.5
>>> Enter:0
Denominator is zero so operation cannot be
performed
```

multiline exception

```
a, b=1, 0
try:
    print (1/0)
    print ('10'+10)
except (TypeError, ZeroDivisionError):
    print ("Invalid Input!")
```

```
>>> 1.0
>>> 1010
>>> Invalid Input!
```

step 7 : Accept an integer value from the user . In the try use the division method

step 8 :- for the exception to be raised use the except Keyword (and) i.e TypeError print Incompatible values

step 9 : use except with error of zero. ~~divisi~~ divison Error, and print the message according that is if entered number is zero not able to perform operation !

step 10 ; Declare state variable and values

step 11 ; For multiline exception use the error types by seperting them with a comma.

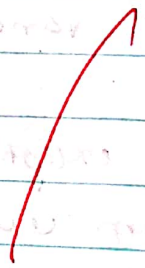
step 12 :- use try block open a file in write mode and subsequently enter values in the file

step 13 :- Use the IOError and display appropriate message.

step 14 : define a function with empty list and calculate the length of the list.

Step 15: define another function y.
initialize or declare some elements
in list and calculate the length of
the same and display the same.

Step 16: In try block accept input from the user
and if the user enters character values
raise an Error that is saying up
Enter integer values



1. using except keyword

```
try:
    a= open("abc.txt","w")
    a.write("python")
except IOError:
        print("Error!")

else:
    print successful

def x():
    l=[]
    print(len(l))

def y():
    li=[2,4,4,1]
    print((len(li))
print(x())
print(y())
```

output : successful
0
None
4
None

raise keyword:

```
try:
    a= int(input("Enter a number:"))
    raise value Error.
except value Error:
    print("Enter integer value!")
```

>>> Enter : xyz
Enter integer value

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'\dt'
String : " hello 123, howdy789, 45 howru"
output= re.findall (pattern, string)
print (output)
>>> ['123', '1789', '45']
```

split()

```
import re
pattern: r'\dt'
string= 'hello 123, howdy 789, 45 howru'
output = re.split (pattern, string)
print (output)
>>> ["hello", "howdy", 'howru']
```

PRATICAL - 4

Topics:- Regular expression.

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

step 2:- 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 the blank space with no-space use sub() with 3 arguments. and print the string without spaces

Step 5:- Import re module declare a sequence use search method for finding subsequently use the group() with dot operator as search() gives memory location using group() it will show up the matched string

Step6: import re module declare list with numbers.
use the conditional statement here we have used up the for conditional 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 creiteria 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 seprating the host & domain name. use the findall() and print the output respectively

```
# no-space:
import re
string = 'abc defghi'
pattern = r'\sti'
replace = ''
V1 = re.sub(pattern, replace, string)
print(V1)
>>> abcdefghi

# group()
import re
sequence = 'python is an interesting language'
V = re.search('\APython', sequence)
print(V)
V1 = V.group()
print(V1)
>>> <_sre.SRE_Match object at 0x0281DF00>
    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 or len(value)==10):
    print("Criteria matched for cell numbers!")
    else:
    print("Criteria failed!")
```

```
>>> criteria matched for cell number
criteria matched for cell number
criteria failed!
criteria matched for cell number
```

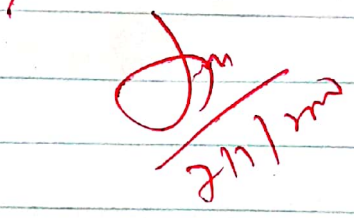
```
# vowels
import re
str1 = 'plant is life overall'
output = re.findall (r'\b[aeiouAEIOU]\w+', str1)
print (output)
>>> ['is', 'overall']
```

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

```
# counting of first 2 letters:
import re
s = 'mr.a, ms.b, ms.c, mr.t'
p = r'(ms|mr)\w+'
o = re.findall (p, s)
print (o)

n = 0
b = 0
for v in o:
```

Step9: import re module enter a string use pattern to display only two elements of the particular string use findall c) declare two variables with initial value as zero use for condition and subsequently use the if condition check whether condition satisfy odd up the or else increment value. And display the values subsequently.

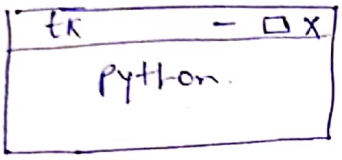


```
if (r == 'ms'):
    f = f+1
else:
    m = m+1
print ("No of males is:", m)
print ("No. of females is:", f)
>>> ['mr', 'ms', 'ms', 'mr']
('No of males is:', 2)
('No of females is:', 2)
```

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", front = "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)
```

Topic :- A] GUI Components.

Step 1:- use the tkinter libraries for importing the features of the 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 mentoined events.

2

Step 1:- Use the tkinter library for importing the feature of the text widgets.

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

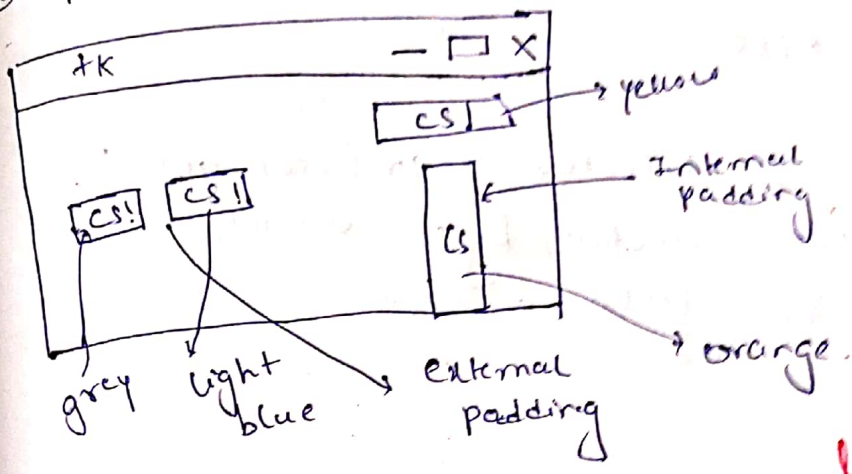
Step 4:- use the main loop() for the triggering of the corresponding events

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 color (bg)
4) The foreground fg and then use the pack(). with a relevant padding attribute.

```
l4 = Label (root, text = "CS", bg = "orange"
     fg = "black", font = "10")
l4.pack (side = TOP, pady = 50)
root.mainloop ()
```

output :



lab

2111

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()
```

PRATICAL- 5(B)

AIM:- GUI Components

step1:- Import the relevant methods from the tk inter 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 the given selection made from multiple option available.

step4:- Now define the parentwindow and define the option with control variable.

step5:- Use the listbox() and insert options on the parent window along with the pack() with specifying anchor attribute.

step6:- Create an object from radio button which will take following arguments { parentwindow object, text variable which will take the value option no 1, 2, 3.... variable argument, corresponding value of f. trigger the function declared.

Step7: Now call the pack for radio object so created and specify the argument using anchor attribute.

Step8: finally make use of the mainloop() along with parent object.

~~It 2~~

Step1: Import relevant methods from the Tkinter library.

Step2: Create a parent object corresponding to the parentwindow.

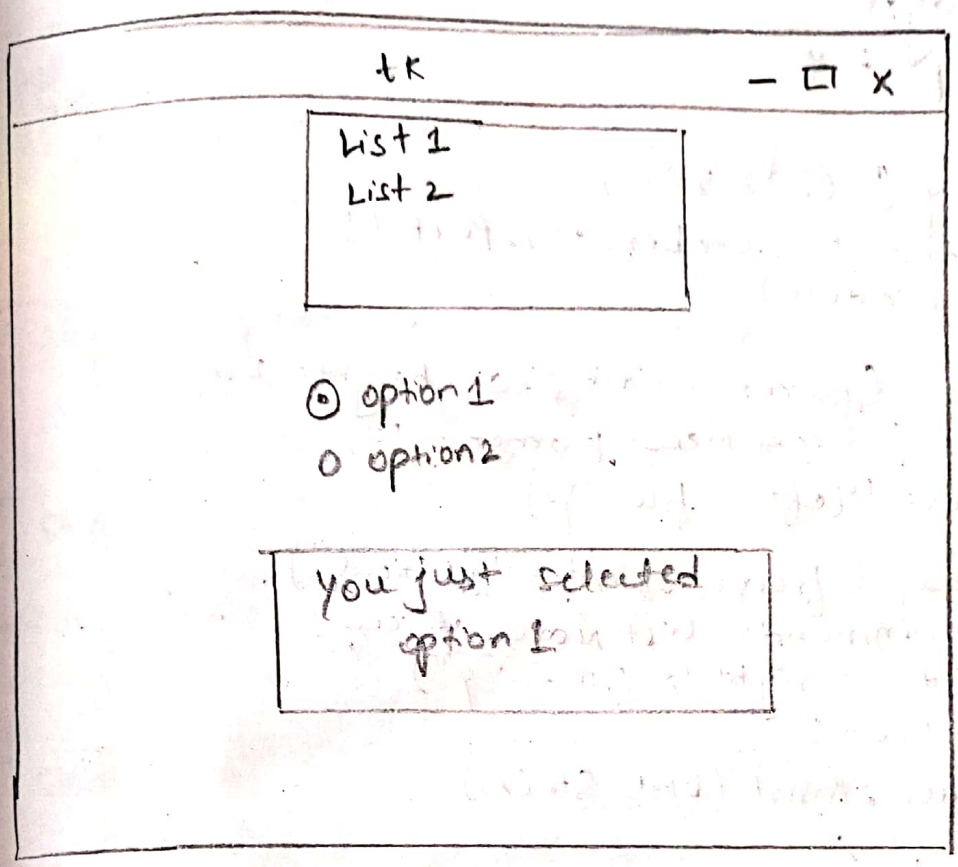
Step3: use the geometry() for laying of the window.

Step4: Create an object and the use the Scrollbar

Step5: use the pack() along with the Scrollbar object with side and fill attributes

Step6: use the mainloop with the parent object

output

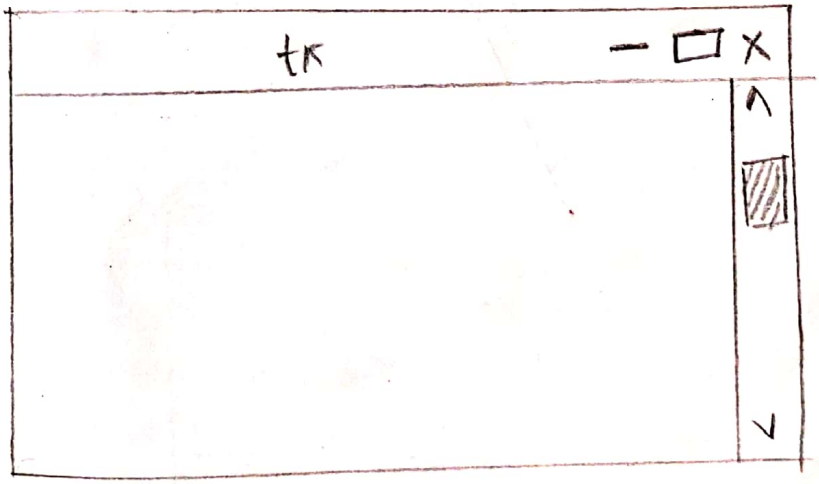


#2:

Scrollbar()

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

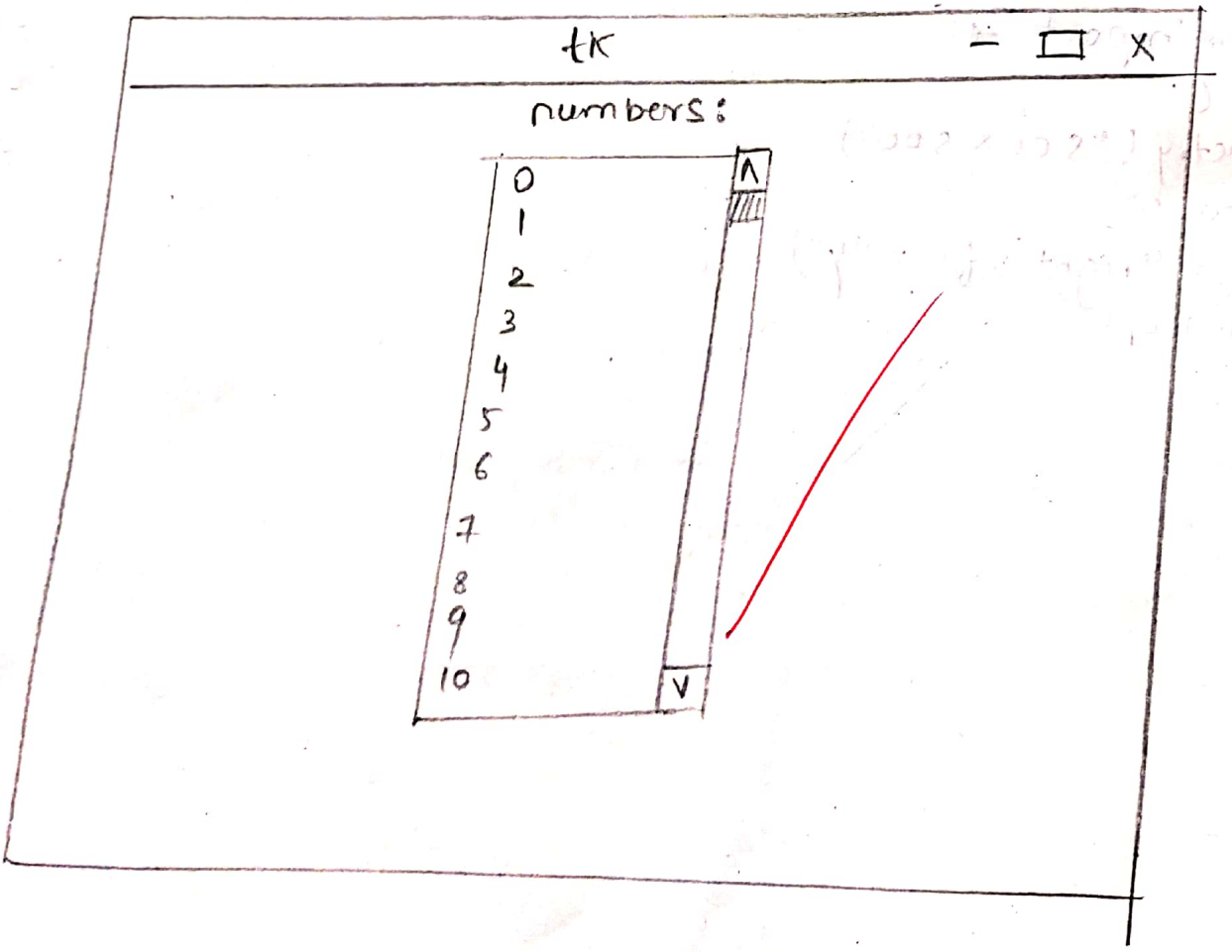
output:



#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:



#3:

step1:- Import the relevant libraries from the Tkinter method.

step2:- Create an corresponding object of the parent window.

step3:- use the geometry manager with pixel size (680 x 500) or any other suitable pixel value.

step4:- Use the label widget along with the parent object created and subsequently use the pack method.

step5:- Use the frame widget along with the parent object created and use the pack method.

Step6:- Use the listbox method along with the attributes like width, height font. Do create a listbox methods object. Use pack () for the same.

Step7:- Use the Scrollbar () with an object use the attribute of vertical then configure the same with object created from the Scrollbar () and use pack ()

Step8:- Trigger the event using mainloop.

4:-

Step1:- Import relevant methods from tkinter library

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

Step3:- Now define the frame object from the method and place it onto 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:- 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.

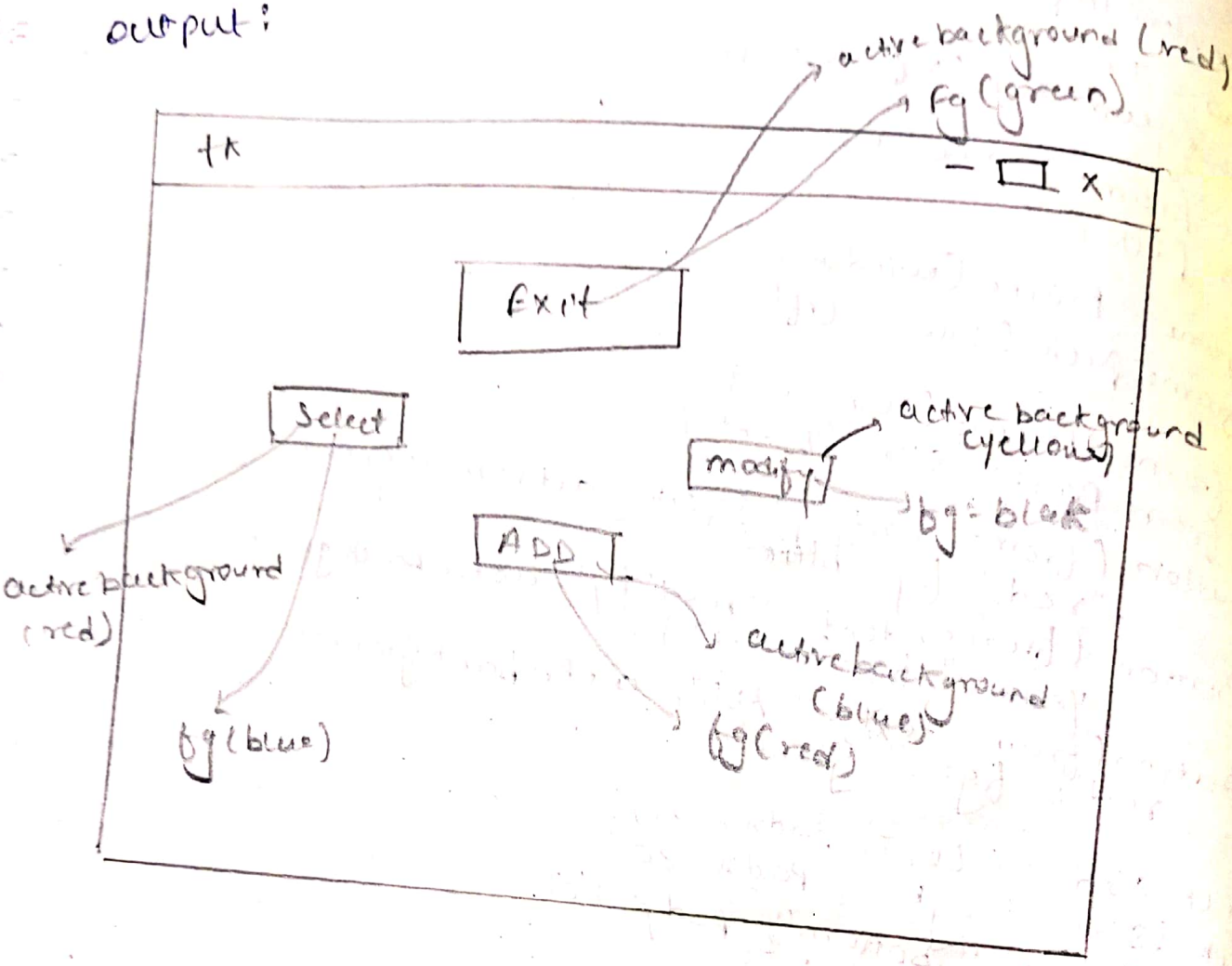
Step6:- Now use the pack() along with the side attribute.

Step7:- Similarly create the button object corresponding to the modify operation put it into frame object on side: "right"

```
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
     ="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")
```

28/07

output:



step 8:- Create another button object & place it on to the right frame & label the button as ADD

step 9:- Add another button & puts it on the top of frame and label it as Exit.

step 10:- use the pack() simultaneously for all the objects & finally use the mainloop()

PRATICAL - 5(C)

AIM:- Gui components

Step 1:- Import the relevant methods from Tkinter library.

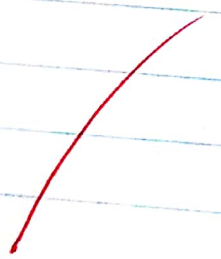
step 2:- Import tk messagebox.

step 3:- Define a parent window object along with the parent window.

step 4:- define a function which will use Tkmessagebox with showinfo method along with info window attribute.

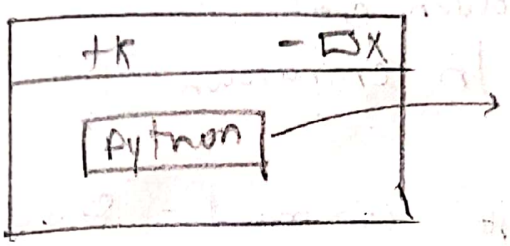
step 5:- Declare a button which parent window object along with the command attribute.

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

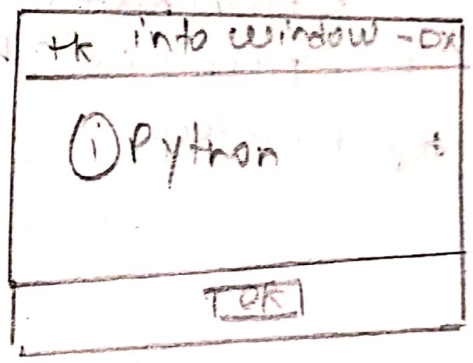


```
# message box
from tkinter import *
import tk messageBox
root = Tk()
  def function():
      tk messageBox.showinfo("info window", "Python")
      b1 = Button(root, text="python", command
                  = function)
      b1.pack()
      root.mainloop()
```

output:

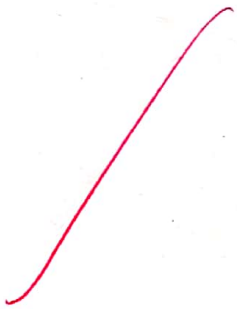


click then this window is pop up



multiple window
Different button (Relief())

```
from Tkinter import *
root = Tk()
root.minimize (300, 300)
def main():
    top = Tk()
    top.title ("HOME")
    top.minsize (300, 300)
L = Label (top, text = "SAN FRANSISCO
In places of intrest: \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()
```



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

Step2:- use parentwindow object along with minimize function for window size

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

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

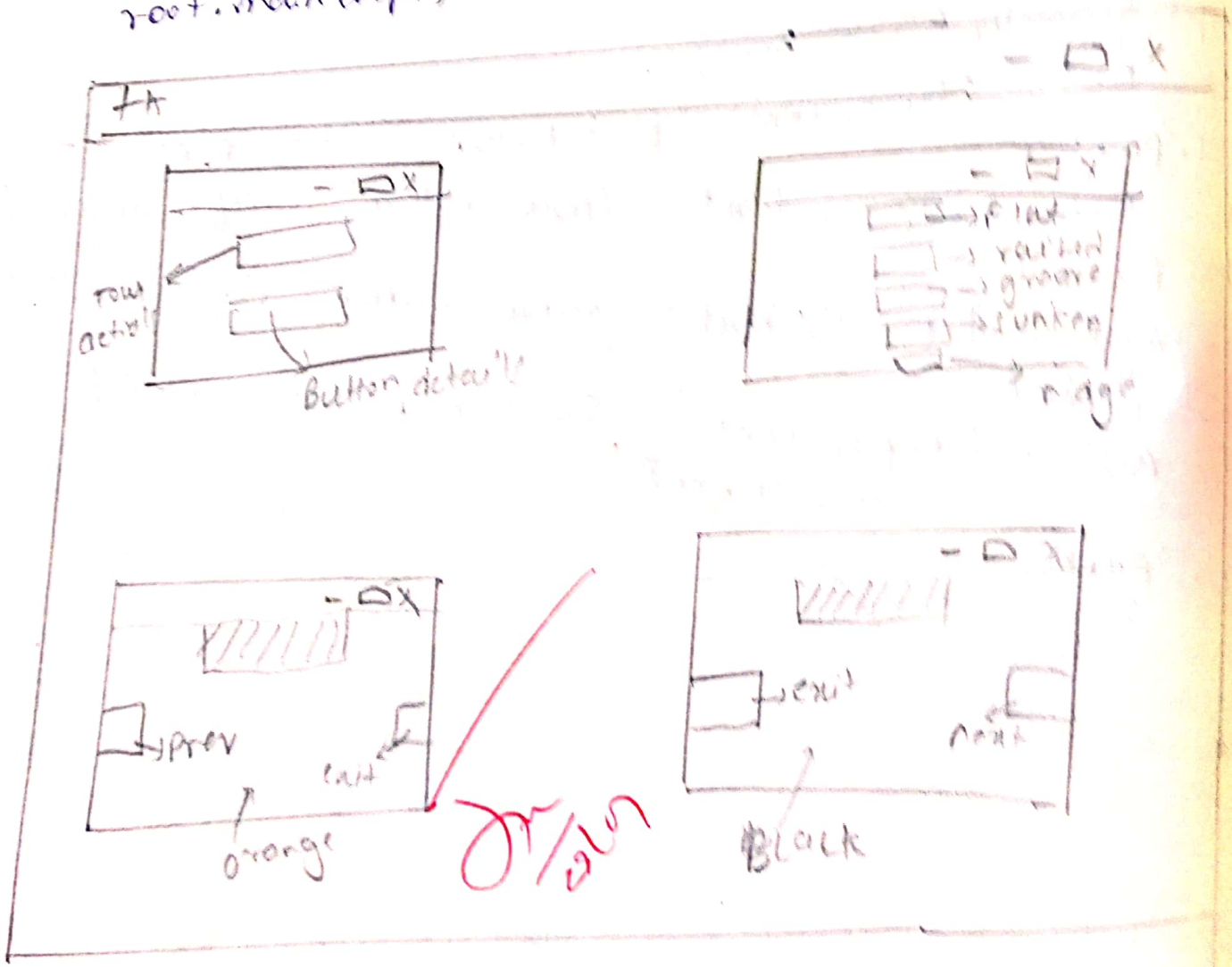
Step5:- Declare another function button along with parent object and declare button with attribute like FLAT, RIDGE, GROOVE, RAISED, SUNKEN along with relief widget

Step6:- 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: Sachit \n
        for more details contact to our official account")
    L.pack()
    b3 = Button(top 2, text = "previous", 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("300x 300")
    b1 = Button(top 3, text = "flat button", relief = FLAT)
    b1.pack()
    b2 = Button(top 3, text = "Groove bottom", relief = GROOVE)
    b2.pack()
    b3 = Button(top 3, text = "raised button", relief = RAISED)
    b3.pack()
    b4 = Button(top 3, text = "sunken button", relief
            = SUNKEN)
    b4.pack()
```

```
bs = Button ( top, text = "bridge button",
     relief = RIDGE )
tops.mainloop()
def terminate ():
    bs.quit()
bs = Button(root, text = "TOUR ARRIVE",
     command = main )
bs.Pack()
bc = Button (root, text = "BUTTON DEPART"
     command = button
bc.Pack()
root.mainloop()
```



PRATICAL - 5(B)

AIM: GUI component

step1: Import relevant methods from the tkinter library.

step2: create parent window object and use the config method along with background color attribute specified

step3: define a function finish with the messagebox widget which will display a message i.e a warning message and subsequently terminate the program

step4: 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 name and finally use the grid() with ipadx attribute.

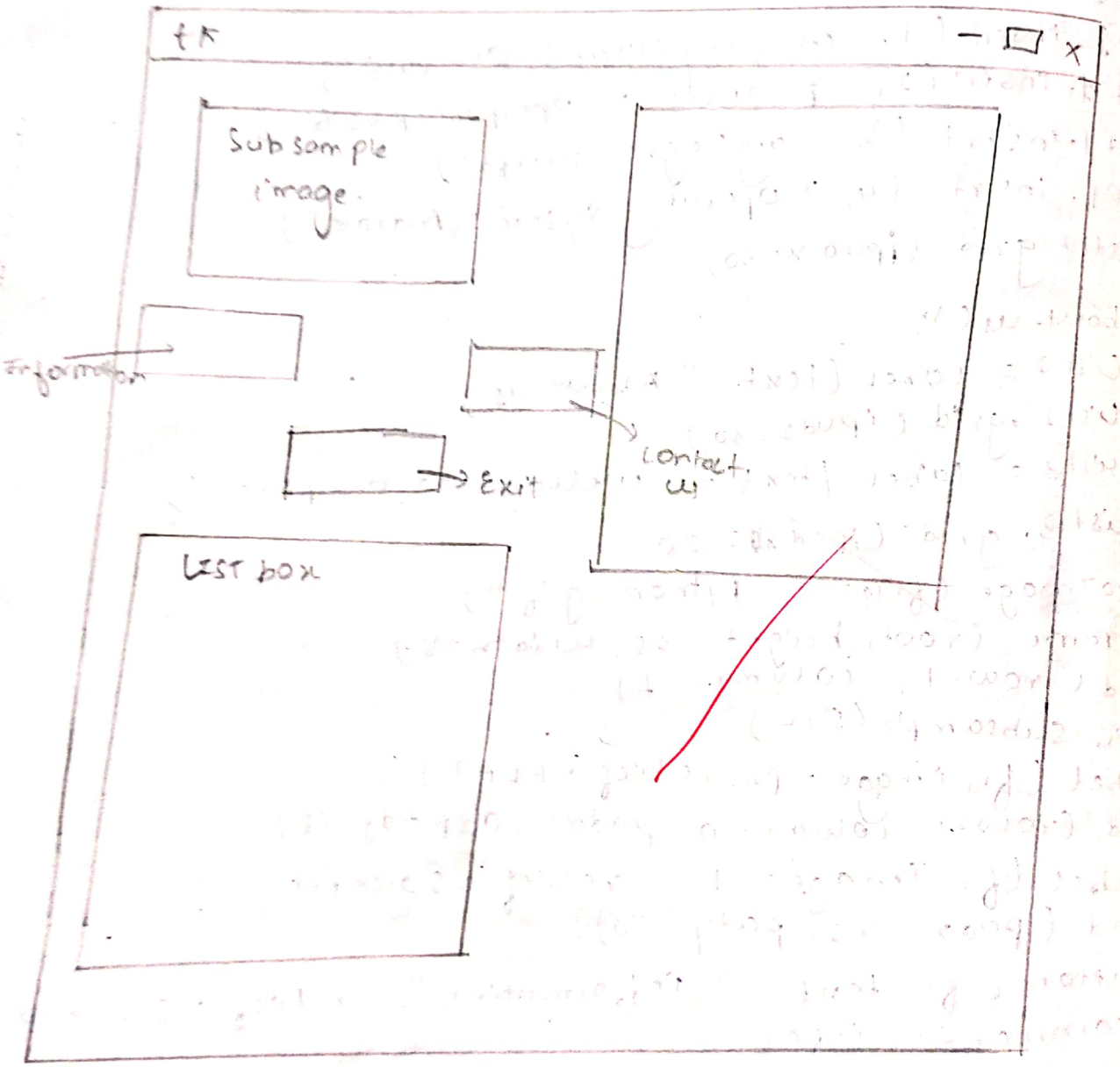
step5: Define a function about us with label widget and text attribute and subsequently use the grid()

```
from Tkinter import *
root = Tk()
root = config (bg = "grey")
def finish():
    messagebox.askokcancel("warning", This will
        end the program")
    quit()

def info():
    list1 = Listbox()
    list1.insert(1, "company name : oneplus")
    list1.insert(2, " products : oneplus 6T")
    list1.insert(3, "Language: swift")
    list1.insert(4, "Operating system : Android")
    list1.grid(ipadx=30)

def about_us():
    list2 = Label(text = "About us")
    list2.grid(ipadx=30)
    list3 = label(text = "welcome to oneplus")
    list3.grid(ipadx = 20)
p1 = PhotoImage(file = "Python.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, column=0)
```

```
b2 = Button (f1, text = " contact us", relief = SUNKEN,
             command = contact us)
b2.grid (row = 1, column = 2, padx = 5)
b3 = Button (f1, text = EXIT", relief = RAISED,
             command = FINISH)
b3.grid (row = 1, column = 1, ipadx = 15)
root.mainloop()
```



step6:- Use photoimage widget with file and filename with gif attribute.

step7:- Create a frame object along with the frame() along with parent window object height & width specified and subsequently use the grid() with row & column attribute specified.

step8:- Similarly create another frame object. as declared by step7.

step9:- Create another object & use the subsample (5,4)

step10:- use label widget along with the frame object, relief attribute and subsequently use the grid().

step11:- Now create button object dealing with different section of frame.

PRATICAL - 5(E)

Aim:- Demonstrate the use of Paned window and

Paned window

Step1:- Create an object from Paned window and use the pack() with the attribute fill and exp

Step2:- Create an object from the Label() and it onto the paned window with the text attribute and use the add() to embed the new object

Step3:- Similarly, Create a second Paned window object and add it on to the first Paned window with orientation specified

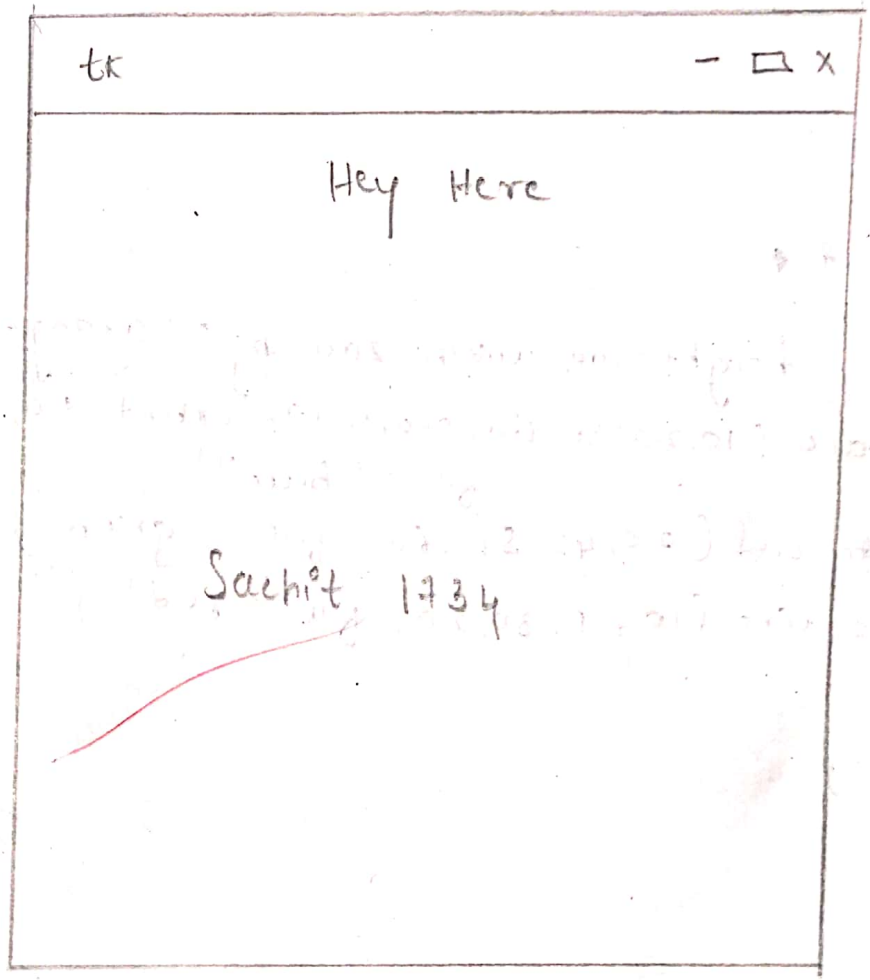
Step4:- Now, create another Label object and it onto the second paned window object and if it onto the second Paned window created

Step5:- Now use the mainloop() to trigger the

Paned windows()

```
from tkinter import *
root= TK()
p= PanedWindow(root, orient: VERTICAL)
p. Pack (fill: BOTH, expand = 10)
L1= Label (P, text = "Hey Here")
p. add(L1)
P1= PanedWindow (P, orient = HORIZONTAL).Pack (fill: BOTH, expa
p.add(P1)
L2= Label (P1, text, "Sachit 1734")
 P1. add (L2)
 mainLoop( )
```

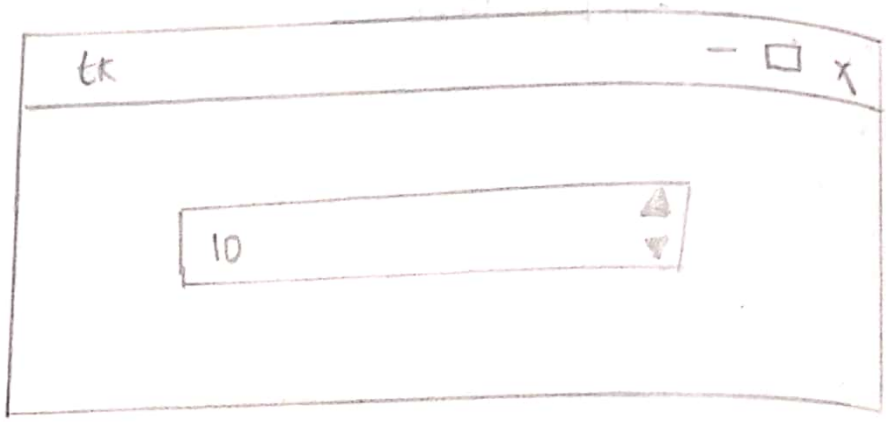
output



SpinBox

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

Output



canvas

```
from tkinter import *
root = Tk()
C = Canvas(root, height=100, width=200, bg="Orang
carc = C.create_arc(10,20,30,40, start=10, extent=50
                    fill="blue")
coval = C.create_oval(20,42,51,60, fill="green")
C.line = C.create_line(10,15,30,20, fill="red")
C.pack()
root.mainloop()
```

Spin Box

Step1:- Create an object from the Tk() and subsequently create an object from the spinbox()

Step2:- make the object so created onto the parent window and trigger the corresponding events.

Step3:- use the pack() to proride the directions using anchor attributes.

Step4:- use the mainloop() to terminate and trigger the events

Canvas:-

Step1:- Import relevant methods from tkinter library declare a parent window object.

Step2:- Create a canvas object along with the canvas method with attributes such as height, width, background color

Step3:- Use create_arc() along with canvas object declared along with start and extent coordinates.

Step4:- Similarly for oval and line. use the pack method and finally call the mainloop method.

CA

PRACTICAL 6 (A)

Aim:- Database connectivity.

Step I:- Import corresponding library to make database connection, os and sqlite3.

Step II :- Now Create the connection object using sqlite3 library and the connect() for creating new database.

Step III :- Now create cursor object using the cursor() and from the connection object created.

Step IV :- Now use the execute() for creating the table with the column name and respective datatype

Step V :- Now with cursor object use the insert statement for entering the values corresponding to different field, corresponding to the datatype

Step VI :- use the commit() to complete the transaction using the connection object.

Step VII :- use the execute statement along with cursor object accessing the values from database using the select from where

Step VIII :- Finally use fetch() or fetchall() and print the values the table using cursor object.

Step IX :- finally close() the cursor object

Source code:-

```
import os, sqlite3
con = sqlite3.connect("student db")
c = con.cursor
c.execute ("CREATE TABLE Student (roll INTEGER, name TEXT )")
c.execute ("INSERT INTO Student values (1734, "sachit")
           (1735, "Rashid"), (1732, "Gaurav")")
c.execute ("SELECT * from student WHERE roll = 1734")
print (c.fetchall())
con.commit()
c.close()
```

output

```
[(1734, 'sachit')]
```

No.

Source code:

```
>>> import dbm
>>> db= dbm.open("database","c")
>>> db["name"] = "name"
>>> if db["name"] != Name:
        print ("database empty")

    else:
        print("match found")

    Match found
>>> db.close.
```

PRACTICAL 2 - (B)

Aim: Database connectivity.

step I : Import the dbm library and use the open for creating the database by specifying the name of the database along the with the corresponding flag.

step II : Use the object so created for accessing the given website and corresponding regular name for the website

step III : Check whether the given URL address matches with the regular name of the page or is not equal to name none then display the message that particular found ! match or else not found !unmatched.

step IV : Use the close() to terminate the database library

☆☆ INDEX ☆☆

No.	Title	Page No.	Date	Staff Member's Signature
1)	Demonstrate the use of different file accessing mode differences attributes, read method	22	26/11/2019	
2)	Iterators	25	3/12/19	
3)	Exceptions	30	17/12/19	
4)	Regular Expression	31	24/12/19	
5)	A. GUI components	35	7/1/2020	
6)	GUI components (5b)	37	9/1/2020	
7)	GUI components (5C)	42	21/1/2020	
8)	GUI component (5D)	46	28/1/2020	
9)	GUI component (5E)	48	11/2/2020	
10)	Database connectivity (6A)	50	18/2/2020	
11)	Database connectivity 6(B)	51	25/2/2020	