Aim :- Demonstrate the use of different file accessing mode different attribute and real method

step I :- create a file object using open method and use the write accessing mode followed op by writing some content onto the file and then closing the file

Step II :- now open the file in read mode and then use read() readline() readlines() and store the output in variable and literally display the content of variable

Step III : now use the file object for finding the name of file the file mode in which it is opened whether the file is still open or close and finally the output of the softspa attribute

Step IV: now open the fileoject in write mode write some cont er content close subsequently then again open the file obje in ~~no~~ mode that is the update mode and write conte

Step V) : Open file object in read mode display the update written content and close() open again in mode with parameter passed and display the output subsequetly

step VI: now open file object in append mode open write method write content close the file object again open the file object in read ~~mode~~ mode and display the append output

```
def choice==5
    print("modulo division=", mod(x,y))
else
    print("wrong selection by user")
```

output

enter first number = 5

enter second number = 6

1 addition

2 subtraction

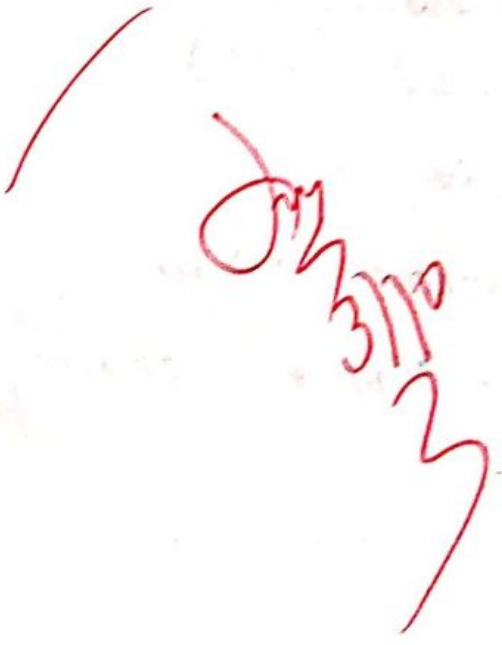
3 multiplication

4) division

5) modulo division

enter selection = 1

addition = 11



```
def choice==5
    print("modulo division=", mod(x,y))
else
    print("wrong selection by user")
```

output

enter first number = 5

enter second number = 6

1 addition

2 subtraction

3 multiplication

4) division

5) modulo division

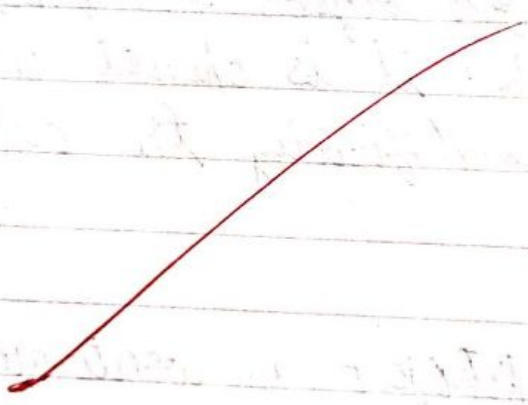
enter selection = 1

addition= 11

Step vii) now open file obj in append mode open write method write content close the file object again open the file object in read mode and display the append mode

step viii) open the file object in read mode declare a variable and perform fileobject bot tell method and store the object consequetly in variable

step viii) using the seek method with the argument with opening the file object in read mode and closing argument



```
abc.txt","w")
file object = write("computer science subject "+"\n")
file object = write("dbms\n Python\n DS\n")
file object close()

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

>>> The output of read method : computer science subject
DBms
Python
DS

```
# Readlines()
file obj = open ("abc.txt","r")
str2 = file obj.readline()
print = ("The output of readline method :", str2)
    file obj.close()
```

>>> The object of readline method= computer science object

```
# readlines()
file obj = open ("abc.txt","r")
str3 = file obj.readlines()
print ("The object of readlines methods :", str3)
    file obj.close()
```

>>> The output of readlins method : computer science subject
DBms
Python
DS

file attribute

```
a = file obj . name
print ("name of file(name attribute): ", a)
>>> (name of file (name attribute ), abc.txt
b = obj file obj . closed
print ("(close) attribute =", b)
>>> (close) attribute =", b)
c = file obj . mode
print=("file mode")()
>>> ("file obj ,"(r))
d = file obj . softspace
print = (softspace =", d)
>>> ("softspace |", 0)
```

wt mode

```
file obj = open(("abc.txt"), ("wt"))
file . write (("yash"))
file obj . close
```

rt mode

```
file obj = open("abc.txt"), ("rt"))
str1 = file . obj . read (r)
print (("output of rt"), str1)
file file object . close ()
>>> ('output of rt') , ("yash")
```

write mode

```
fileobj = open("abc.txt"), ("w"))
file obj . write (("dbms"))
fileobj . close()
```

Read mode

```
file obj = open(("abc.txt"), ("r"))
Str 2 = file obj . read ()
print(("output of read mode", str2)
>>> ("output of read mode, ('yash')
```

```
# append mode
file obj = open("abc.txt", "a")
file obj.write("data structures")
file.close()
file object = open("abc.txt", "r")
str 3 = file obj.read()
print("output of append mode", str 3)
  file obj.close()
>>> ("output of append mode:", "yash", "data structures")
# tell()
file obj = open("abc.txt", "r")
pos = file obj.tell()
 print("tell(): ", pos)
 file obj.close()
>>> (tell(); pos)
 # seek()
 file obj = open("abc.txt", "rw")
 str 4 = file obj.seek(0,0)
  str 8 = file obj.read(10)
 print("The beginning of line is=", str 8)
```

55

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

Use: raise stop iteration

```
>>> Y= counter()
>>> Z = iter(Y)
>>> Z.next()
    1
>>> Z.next()
    3
>>> Z.next()
    5
```

aim :- demonstrate the use of iterable and iterator

Theory: - in python iteration is an object which implement iterator class which has 2 method namely -- iter() -- and the -- next() -- list, tuple, dictionary and the set all represent a iterable object

a) write a program using iterable object for displaying the odd number in range 1 to 10

algorithm

Step I: define a iter() with argument and initializing the value and return that value

Step II: define the next() with an argument and compare the upper limit by using a conditional statement

Step III: now create an object of the given class and to pass this object in the iter method

Q2 write a program using an iterable for calculating the power of a given number for instance number entered is 2 then value calculated should be $1, 2^1, 2^2, 2^3, 2^4$

algorithm

step I: define iter() with argument and initialize value and return the value

step II: now define next() with an argument and compare the upper limit by using conditional statement

step III: now create an object of the given class and pass this object in the iter method

```
class hower:
    def __iter__(self):
        self.h=0
        return self
    def next(self):
        if self.h<=10
            num=self.h
            self.h+=1
            ho=2** num
            print("2**",self.h-1,"=",ho)
            return ho
        else:
            raise stop iteration
```

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

```
class fact:
    def __iter__(self):
        self.f=1
        return self

    def next(self):
        if self.f<=10:
            num=self.f
            self.f+=1
            fac=1
            for i in range(1,num+1):
                fac=fac*i
            print(self.f-1,"!=",fac)

        use:
            raise stop iteration
```

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

13 write a program using iterable concept to find factorial of number in range 1 to 10

algorithm

Step I :- define iter() with algorithm and iterable the value and return the value

step II :- define next() with an argument and compare the upper limit by using a conditional statement

StepIII :- now or create an object of the given class and pass this object in this the iter method

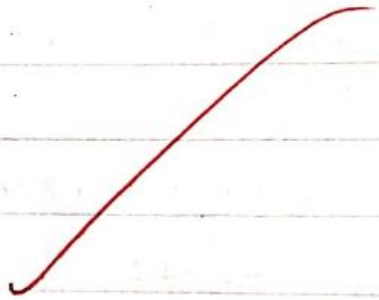
write a program using iterable concept to display multiple of 2 in range 1 to 10

algorithm

Step 1 : define a iter() with an argument and initialize the value and return the value

StepII : define a next() with an argument and compare the two upper limit by using a conditional statement

Step III : Now create an object of the given class and pass this object in the User method



```
class mult
    def __iter__(self):
        self.n=1
        return self

    def next(self):
        if self.n<=10
            num = self.n
            self.n+ =1
            table = 2*num
            print("2 *", num, "=", table)
        else:
            raise stop iteration
```

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

```
def accept_age():
    age = int(input("Enter your age : "))
    if age>30 or age<15:
        raise value error
    else:
        print("your age is", age)
valid = false
while not valid
    try:
        age = accept_age()
        valid = True
    except value error
        print("your age is not in range")
```

>>> Enter your age : 40
your age is not in range

>>> Enter your age : 18
your age is 18

aim:- demonstrate the use of exception handling

Theory: an exception is an event which occurs during execution of program which distrupt the normal flow of program this exception represent an object which represent an error

a) write a program to check the range of age of the student in given class and if age does not fall in given range use value error exception otherwise return the valid no

algorithm

step 1:- define a function which will accept the age of the student from standard input

step 2:- use if condition to check whether the input age fall in range and so return the age else use value error exception

step 3:- define the while loop to check whether the boolean expression holds true use the try block to accept the age of student and terminate the looping condition

step 4:- use except with value error and print the message not a valid range

02 write a program to check whether the number in given class and if the number is a floating point use value error as exception for the given input

algorithm

Step I:- use try block and accept the using input and convert it into integer datatype and subsequently terminate the block

Step II use the except block with exception as value error and display appropriate message in suspicious code is part of try block

```
while True:
    try:
        a=int(input("Enter a number:"))
        print("valid number")
        break
    except value error
        print("not a valid number ! try again")
```

```
>>> Enter a number\5-6
    not a valid number! try again
>>> Enter a number\5
    valid number
```

```
def divide(a,b):
    ans = a/b
    return ans
while True:
    try:
        a= int(input("Enter 1st number"))
        b= int(input("Enter 2nd number"))
        ans= divide(a,b)
        print("division of",a,"and",b,"is",ans)
        #break
    except ZeroDivisionError:
            print("Error!")
```

```
>>> Enter 1st number 1
    Enter 2nd number: 1
    division of 1 and 1 is 0
```

a) write a program to demonstrate use of zero division error

algorithm

step I :- use the block and accept the input using input() and then convert it into integer data type

step II :- define a function with 2 parameter to divide the number given by user

step III :- define while T/loop to check whether the boolean expression holds true

step IV : use except with zero division error and to print the message

24/12/24

Practical - 7

aim :- demonstrate the use of regular expression

theory :- regular expression represent the sequence of character which is mainly used for finding & replacing the given pattern in a string and common useage of regular expression involves following function

Searching a given string
finding a string
breaking a string into smaller
substring
Replacing part of string

Q1 write a regular expression extracting numeric and alphabetic values from a given string

algorithm

step 1 :- now apply string and pattern in find all() and display the output

step 2 :- id is used for matching all decimal digits where D is used to match non decimal digit

Q2 write a regular expression for finding the match string at the biging of given sequence

Code 1:

```
import re
String = "hello 123# abc#567"
Result=re.findall ("\d+",String)
Result = re.findall ("\D+",String)
print (Result)
print (Result1)
```

output:

```
>>> ['123#', '#567']
>>> ['hello', 'abc']
```

Code 2:

```
inhort re
String = "hython is an important language"
Result = re.search ("^A hython", String)
print (Result)
if result:
    print ("match found").
else:
    print ("match not found")
```

output

```
<re match object ; span
```

match found

08

Code 3 :

```
import re
li = ["9876543210", "8765432109", "765432098", "6543210987"]
for element in li
    result = re.match("[8-9]{1}[0-9]{9}", element)
    if result
    print("correct mobile number")
    print(result.group(1))
else:
    print("incorrect mobile number")
```

output

correct mobile no
9876543210
correct mobile no
8765432109
incorrect mobile no
incorrect mobile no

algorithm

step1 :- import re module and apply a string

step2 :- Use search() with "A" python and string as two parameter

step 3 now ab display the output

step 4 now use of conditional statement for user to know whether the match is found or not

3) write a regular expression to check whether the given mobile number start with 8 or 9 and the total length of digit should be atmost 10

step1 : import re module and apply a string of mobile no 8

step2 : now use for conditional statement to find if the number start with 8 or 9 and the toltotal number should length of 10 use match() inside for statement to find the match is given string

step 3 :- use if conditional statement to know whether we have a match or not if we have use group() to display the output and if we don't display incorrect mobile no

4) write a regular expression for extracting a word from given string along with space character in between the word and subsequently extract the word without space character

algorithm

step1: import re module and apply a string

step2: use findall() to extract a word from given string

step3: use "\w*" to extract word along with space and use "\w+" to extract word without space

step4: now display the output

Q5 write a regular expression for extracting first and last word from a string

algorithm

step1: import re module and apply a string

step2: use findall() in which use "^\w.*" as one parameter to find first word of string then use "\w+$" as parameter to find last word of string

step3: now display the result

code

```
import re
string = "python is important"
result1 = re.findall ("\w*", string)
result2 = re.findall ("\w+", string)
print (result1)
print (result2)
# output
>>> ['python', '', 'is', '', 'important']
>>> ['python', 'is', 'important']
```

codes

```
import re
string = "string is important"
result = re.findall ("^\w+", string)
result1 = re.findall ("\w+$", string)
print (result)
print (result1)
#output
>>> ['python']
>>> ['important']
```

Code6

```
import re
string = "emit 20/12-12-2019"
result = re.findall ("\d{2}-\d{2}-\d{4}", string)
print(result)
output result1 = re.findall ("\w+g", string)
print(result1)
```

Output:- ['28-12-2019']

Code 7

```
import re
string "abc@test.edu"
result1 = re.findall ("^[\w+]", string)
result2 = re.findall ("[\w+]", "[\w+]", string)
result3 = re.findall ("[\w\.-]+", string)
print(result1)
print(result2)
print(result3)
```

Output

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

05 Write a regular expression for extracting the date in format dd-mm-yyyy by using the findall() where the string has following format amit 201 20-12-2019

algorithm

step 1:- import re module and display string

step 2:- use findall method and use "\d {2,3}-\d {2,3}-\d {4,5} as an parameter

step 3:- now display the output

6 write a re for extracting the
1) username from email id
2) hostname from email id
3) both username and hostname from email id

algorithm

step 1 : import re module and apply a string
step 2:- use findall() to find username hostname and both of email id
step 3:- use "^\w+" for username use "@\w+.\w+" for hostname and use "[\w.-]+" for both as parameter in findall
step 4 : display the output

07/01/2020

Practical - 5

topic :- gui components

step 1:- use the tkinter library for importing the feature 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 coo corresponding ../abovementioned events

#2

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

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

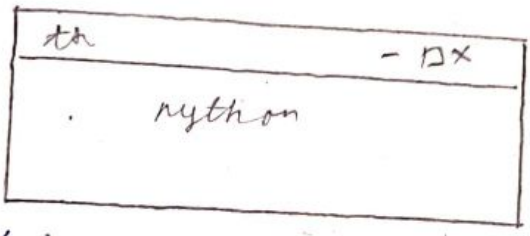
step 3 :- use the pack() along with the object from the text() and use the parameter

1) side = left , padx = 20
2) side = left , pady = 30
3) side = TOP , grade = 90
4) ...

creating of parent window

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

output

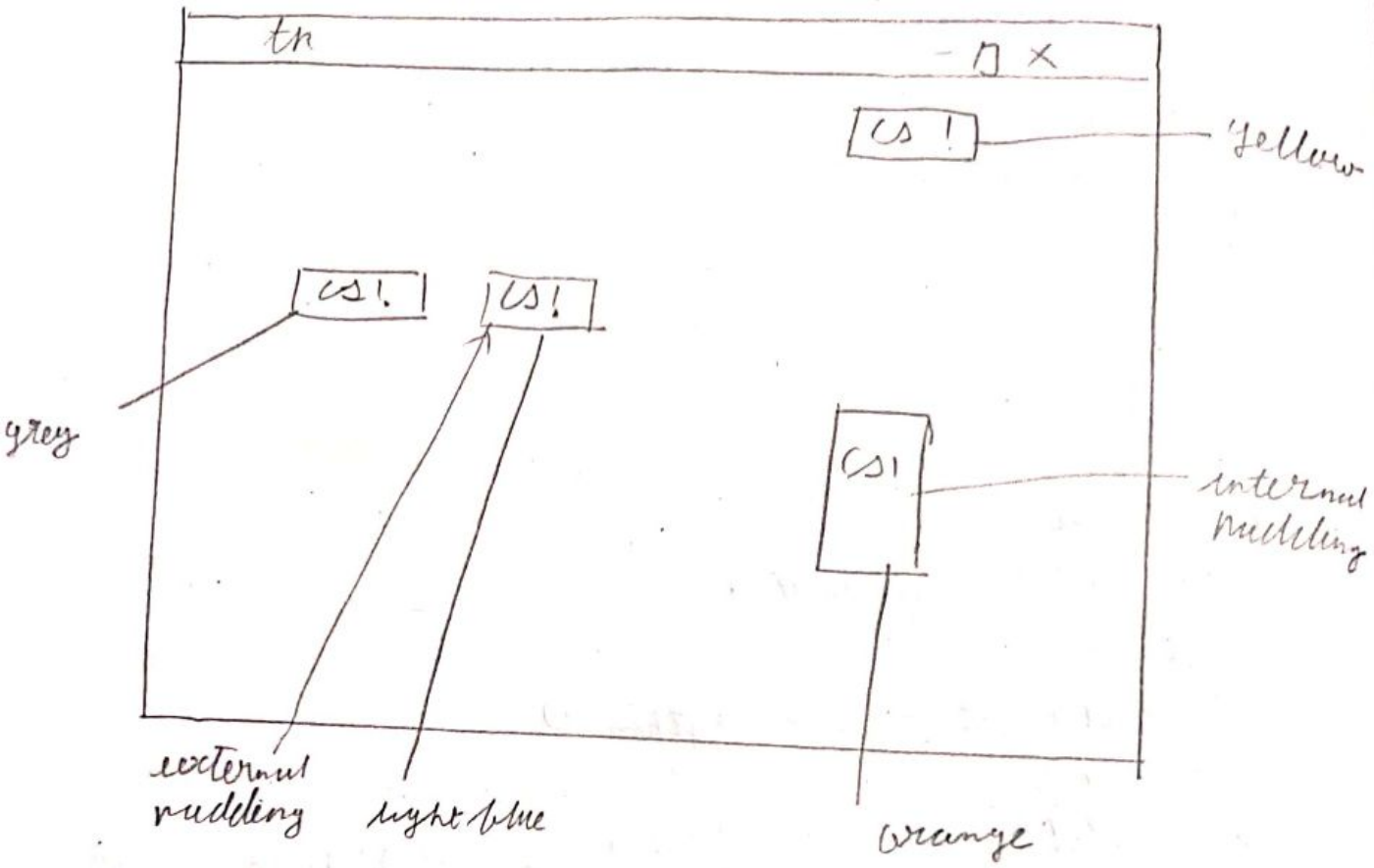


label attribute

```
from Tkinter import *
root Tk()
l = label (root, text = "Python")
l.pack()
l1 = label (root, text = "CS!"), bg = "gray"), fg = ("black"), font = "10"
l1.pack (side = LEFT, padx = 20)
l2 = label (root, text = "CS"), bg = "light blue", fg = "black", font =
l2.pack (side = LEFT, pady = 30)
l3 = label (root, text = "CS"), bg = 'yellow" fg = "black", font =
l3 = pack (side = TOP, ipadx = 40)
l4 = label (root, text = "CS", bg = "orenge") fg = ("black") font "10
l4.pack (side = TOP, ipady = 50)
root.mainloop()
```

output

34 output



Step 7: Use the mainloop() for the triggering of the corresponding events

Step 8: Now Repeat d following step with the label which takes the following argument

1) name of the parent window
2) text attribute which define the string
3) the background colour (bg)
4) the foreground fg and then use the pack() with relevent padding attribute

25. # radio button

```
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, "list1")
l2.insert(2, "list2")
l1.pack(anchor = N)
r1 = radiobutton(root, text = "option1", variable
                 = var, value = "option1", command = select)
r1.pack(anchor = N)
r2 = radiobutton(root, text = "option2" variable =
                 var, value = "option2", command = select)
r2 = pack(anchor = N)
root.mainloop()
```

win :- gui component

Step 1 :- import the relevant method from the thinter library Create an object with the parent window

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

Step 3 :- now define a function which tells the user the given selection made from multiple option available

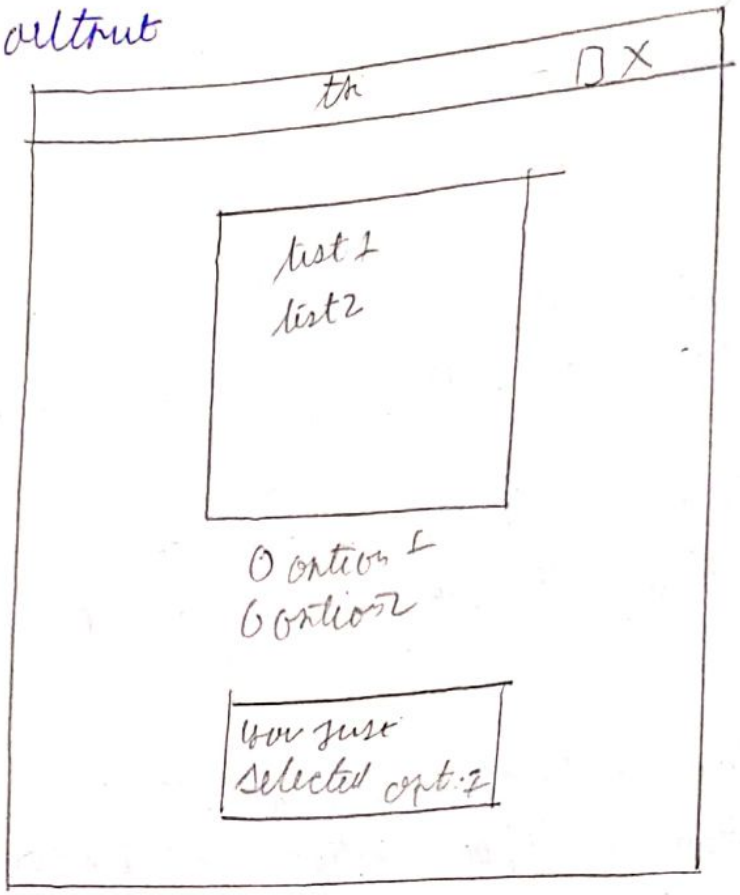
Step 4 :- now define the parent window and define the option with control variable

Step 5 :- Use the listbox() and to insert option on the parent window along with the pack() with specifying anchor attribute

Step 6 :- Create an Object from window radio button which will take following argument parent window object text variable which will take the values option 0,1,2,3... variable argument corresponding values and trigger the function declared

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

output

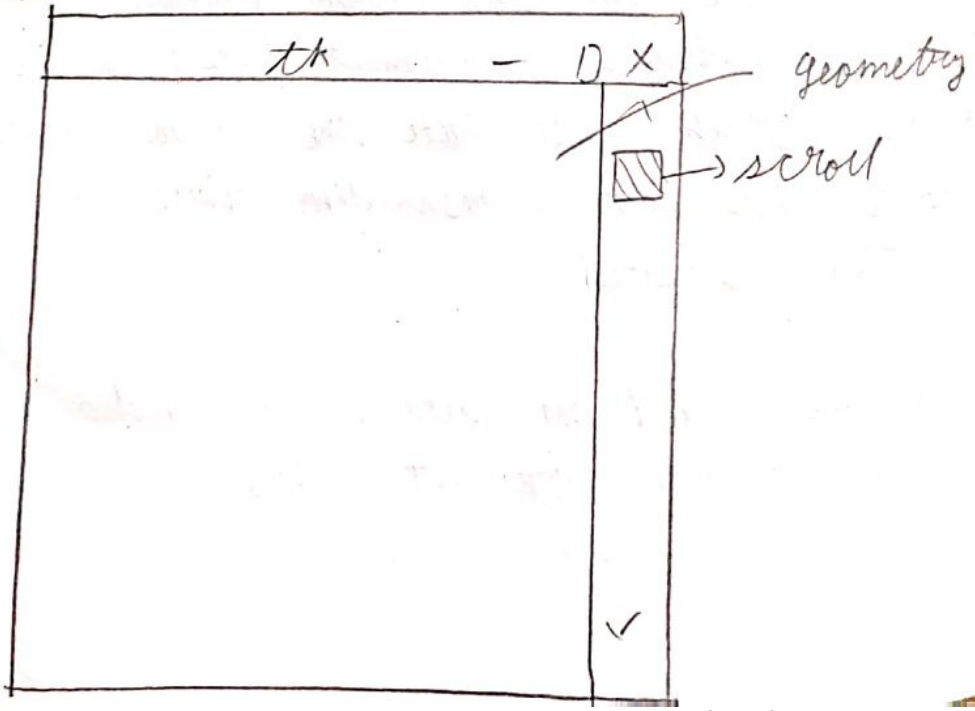


#2:

scrollbar()

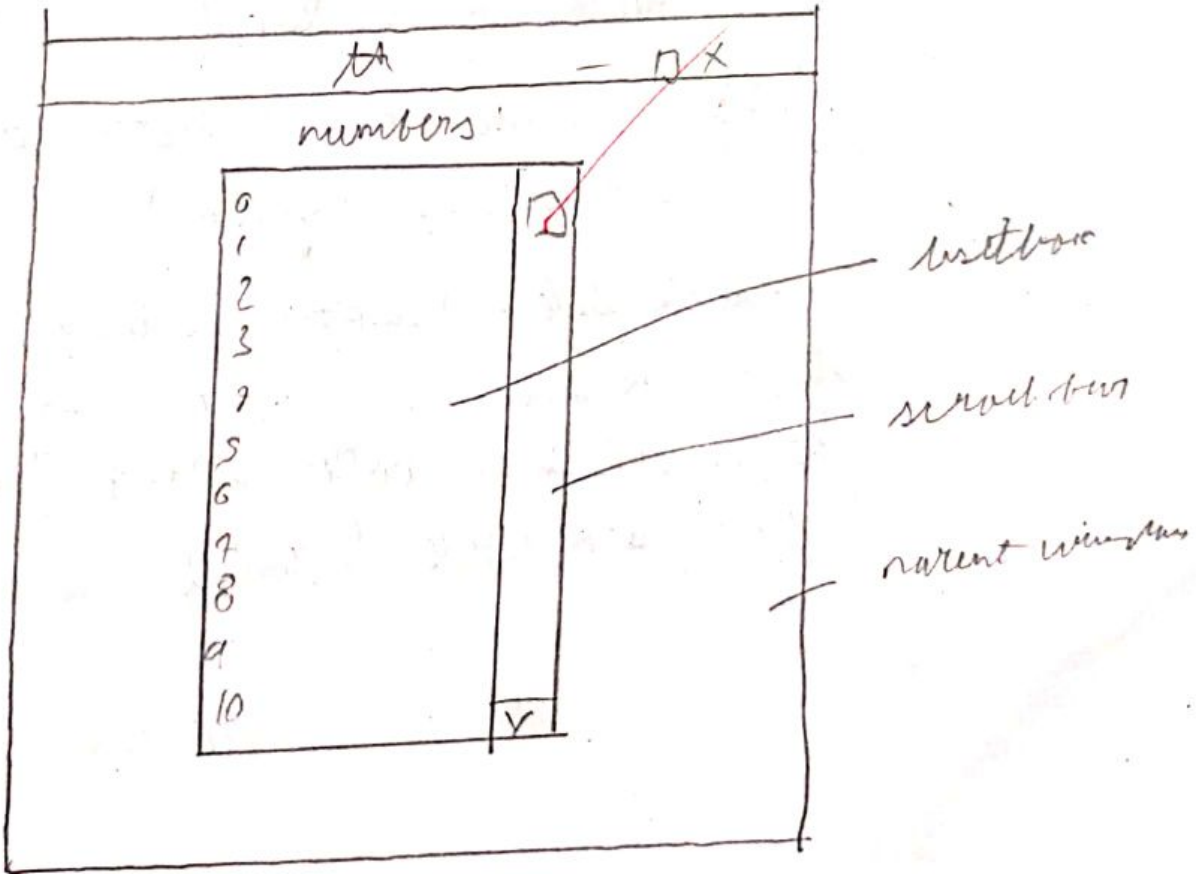
```
from tkinter import*
    root=TK()
    root = geometry ("500x500")
    s= scrollbar()
    s.pack (side="right", fill="y")
root.mainloop()
```

output



```
# using frame widget
from tkinter import *
window = Tk()
window.geometry("680x500")
label(window, text="numbers").pack()
frame = Frame(window)
frame.pack()
listnode = 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 = listnode.yview)
scrollbar.pack(side = "Right", fill = y)
    for x in range(100)
        listnode.insert(End, str(x))
window.mainloop()
```

output



#P.

```
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 = "black")
b2 = button(frame, text = "modify", activekey = yellow
      bg = "blue")
b3 = button(frame, text = "label", activebackground =
      "blue", fg = "red")
b4 = button(frame, text = "exit", activebackground
      = "red", bg = "green")
b1.pack(side = "LEFT", padx = 20)
b2.pack(side = "right", padx = 30)
b3.pack(side = "bottom", pady = 20)
b4.pack(side = "top")
```

step 8 :- finally make use of the mainloop() along with parent object

#2:

step 1 : import relevant method from the tkinter library

step 2 : create a parent window corresponding to the parent window

step 3 : use the geometry() for sizing 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 a pixel size (600x500) or any other suitable pixel value

step 4 :- use the label widget along with the parent window created and subsequently use the pack method

steps : use the frame widget along with the parent object created and else the pack method

step6: use the listbox method along with the attribute like width height font to create a listbox method 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 function scrollbar() and use pack()

step8 trigger the event using mainloop

##

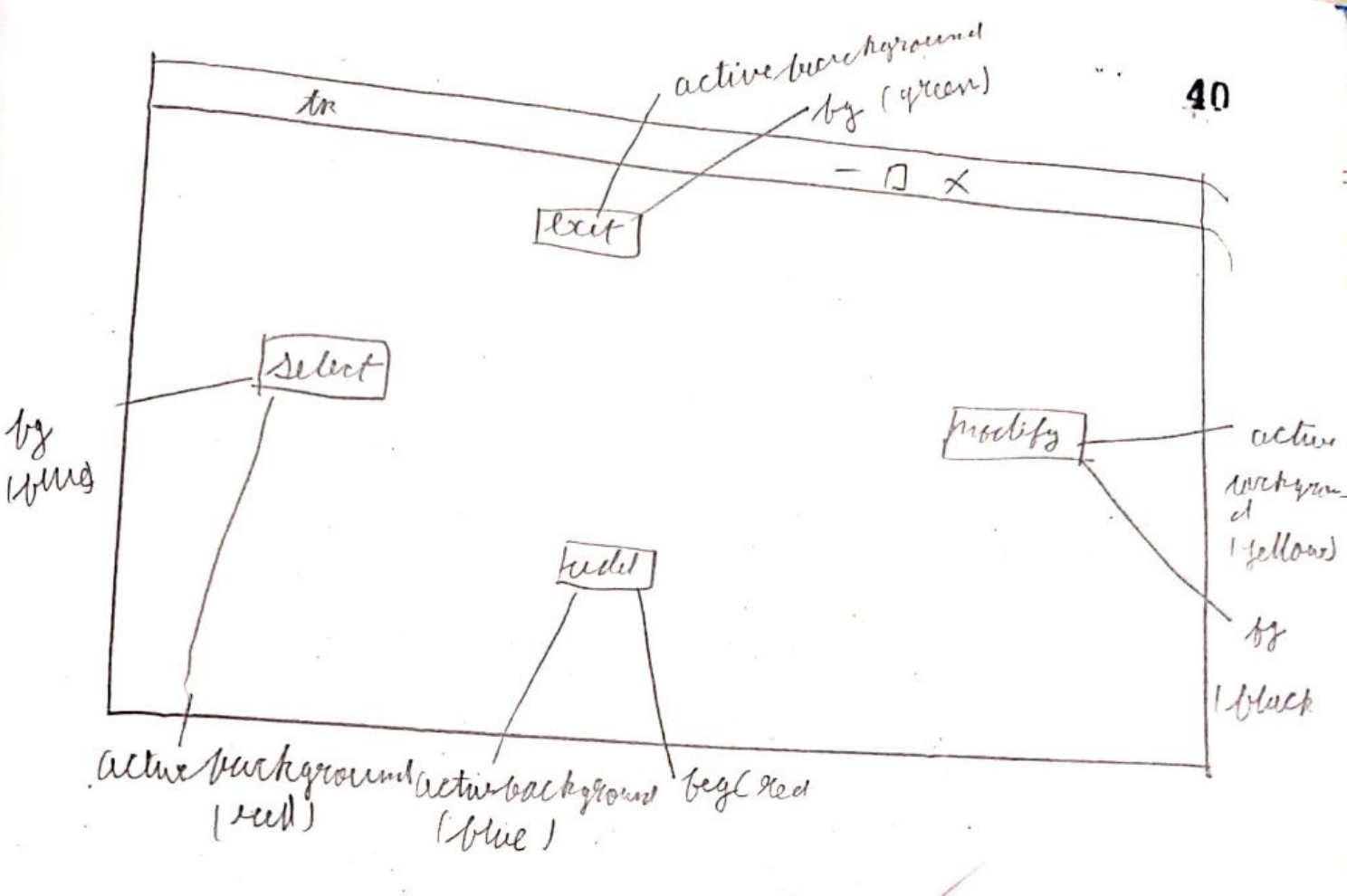
step1 :- import relevant method from tkinter library

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 the parent window

step4 :- Create another frame object termed as the left frame and put it on the parent window on its LEFT side

steps: 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 the MODIFY operation put it frame object on side = "right"

step8 : create another button object and place it on the Right frame and label the button as add

step9 : add another button and put it on the top of frame and label it as exit

step10 use the pack() simultaneously for all the object and finally use the mainloop()

practical - 5(i)

aim :- gui component

step 1: import the relevant method from tkinter library.

step 2: impact the messagebox

step 3: define a parent window object along with the parent window

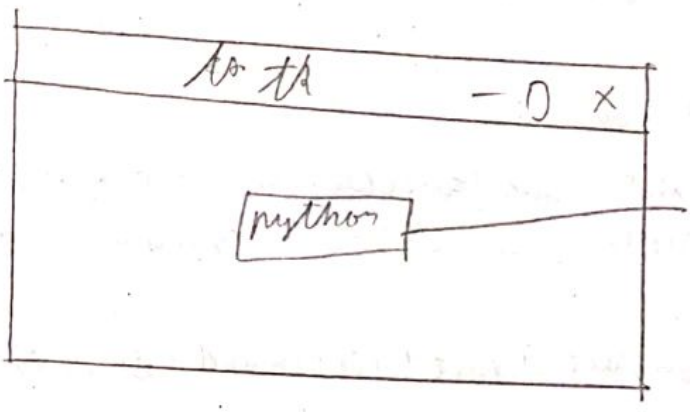
step 4: define a function which will use tk messa box with showin method along with in window attribute.

step 5: declare a button with parent window obje along with the command attribute.

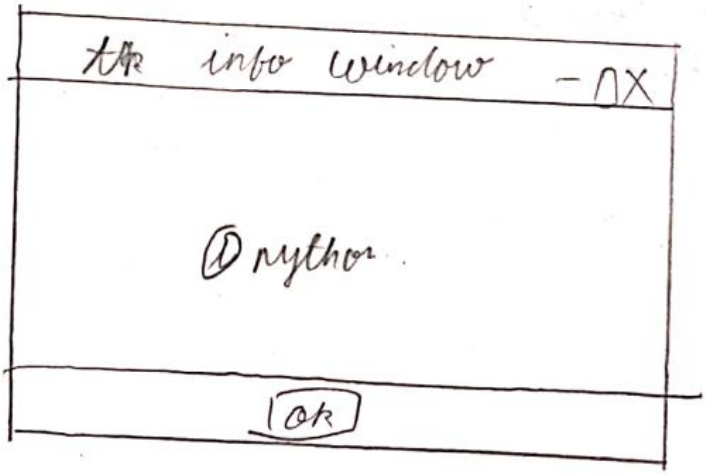
step 6: place the button widget onto parent window finally call mainloop() for triggering of the event called

```
# 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:



Click then this window will pop up



SA

```
# multiple window
# different button (radio)
from tkinter import *
root = TK()
root.minsize(300,300)
def main():
    ton = TK()
    top.config(bg = "black")
    top.title("home")
    top.minsize(300,300)
    l = label(top, text = "san francisco\n place of interest: golden bridge\n command street\ chinatown\n coit tower")
    l.pack()
    b1 = button(ton text = "next", command = second)
    b1.pack(side = Right)
    b2 = button(Top) text = "exit", command = terminate)
    b2.pack(side = LEFT)
        ton.mainloop
```

step 1: import the relevent method from the thinter library along with parent window object declare

step 2: use parent window object declare along with minisize function for window size

step 3:- define a function main declare parent window object and use config() title() minize() label() as well as button and use pack() and mainloop() simultaneously

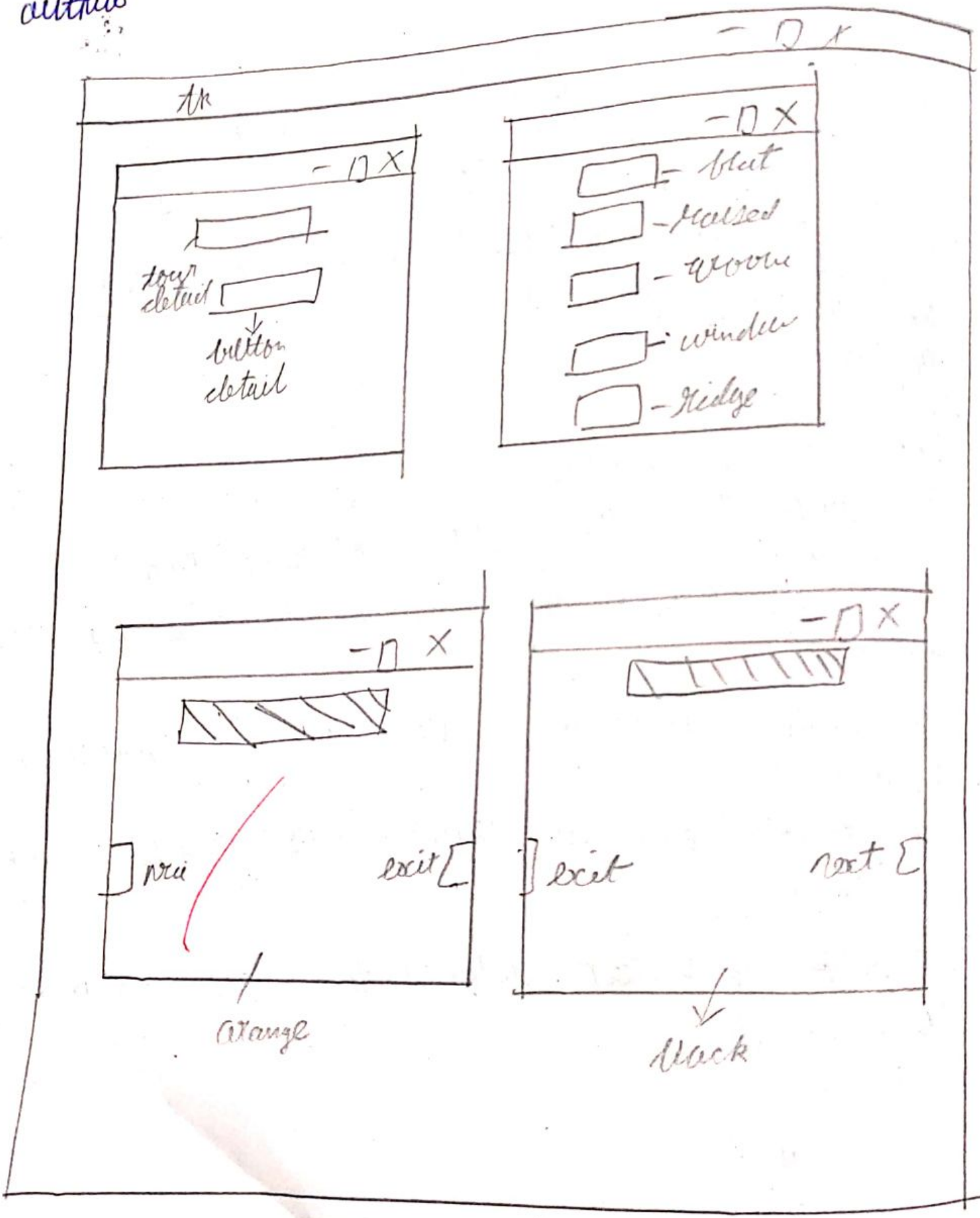
step 4:- similarly define the function second and the attribute accordingly

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

step 6:- finally called the mainloop() for event driven programming

```
def second()
    top2 = Tk()
    top2.config(bg = "orange")
    top2.title("about us!")
    top2.minisize(300,300)
    l = label(top2, text = "created by
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", 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 = "raised button", relief = RAISED)
    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)
top3.mainloop()
def terminate():
    quit()
b5 = button(root text = "TOUR DETAILS", command = main)
b5.pack()
b6 = button(root text = "button detail command = button
b6.pack()
root.mainloop()
```

output



practical 5 (a)

aim : gui components

step 1 :- import relevent method from the thinter library

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

step 3: define a function finish with the messagebox widget which will display a message i.e a warning message 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()

step 6 : use photoimage widget with file and filename with gif attribute

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

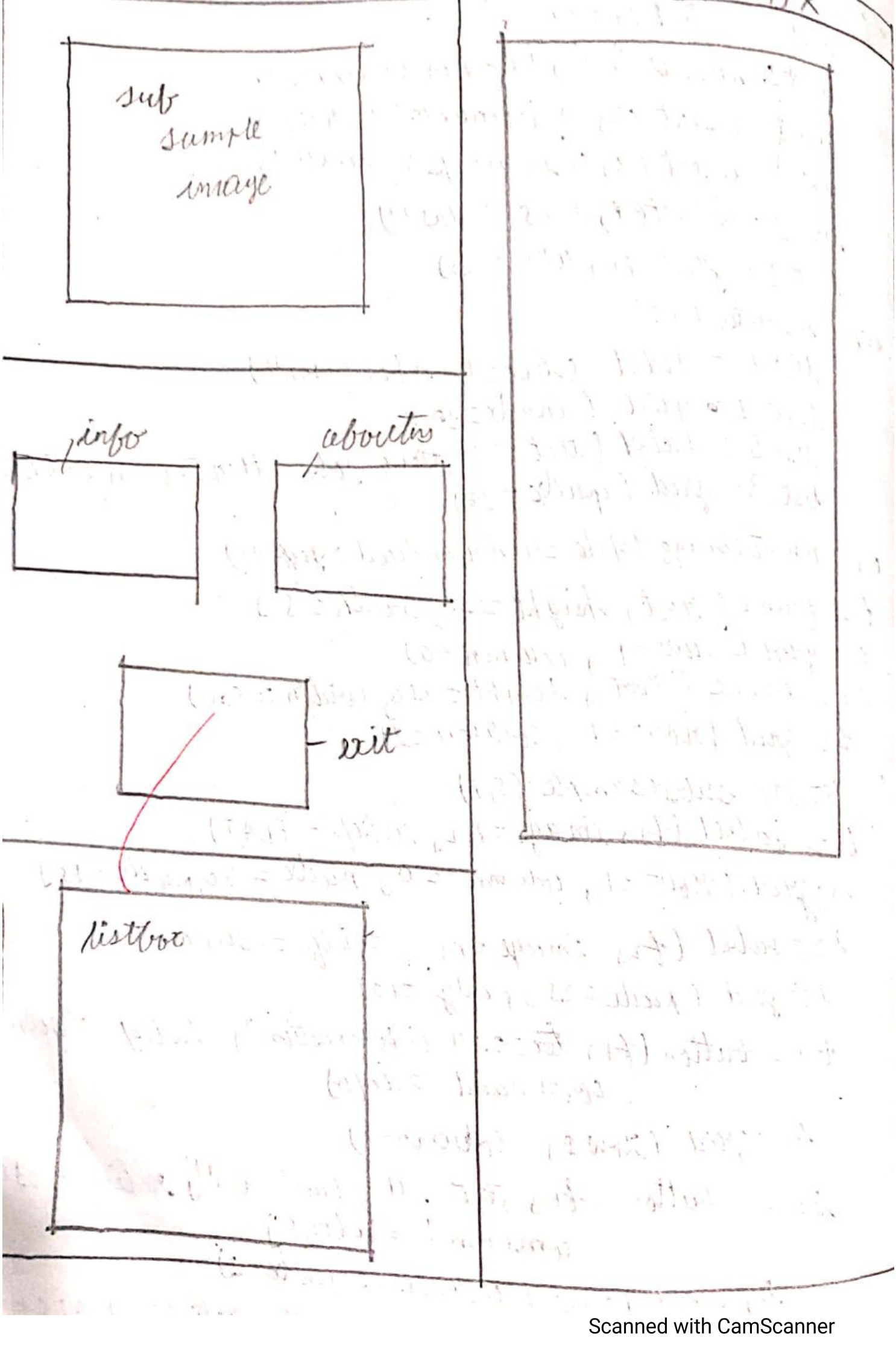
step 8 : similarly create another frame object as declared by step

step 9 : create another object and use the subsequ pack()

step 10 : use label widget et along with the frame object relief attribute and subsequently use the grid()

step 11 : now create button dealing with different of frames

```
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, "co. name: apple")
    list1.insert(2, "product: iphone")
    list1.insert(3, "language: swift")
    list1.insert(4, "OS : ios")
    list1.grid(padx=30)
def aboutus():
    list2 = label(text="about us")
    list2.grid(padx=30)
    list3 = label(text="steve jobs theatre march 2020")
    list3.grid(padx=20)
m1 = photoimage(file="download.gif")
f1 = frame(root, height=35, width=5)
f1.grid(row=1, column=0)
f2 = frame(root, height=250, width=500)
f2.grid(row=1, column=1)
m2 = m1.subsample(5,4)
l1 = label(f1, image=m2, relief=FLAT)
l1.grid(row=1, column=0, padx=20, pady=15)
l2 = label(f2, image=m1, relief=sunken)
l2.grid(padx=25, pady=20)
b1 = button(f1, text="information", relief=sunken
            command=info)
b1.grid(row=1, column=0)
b2 = button(f1, text="about us", relief=sunken
            command=about)
b2.grid(row=1, column=2 padx=2)
b3 = button(f1, text="exit", relief=RAISED, command
            =finish)
b3.grid(row=2, column=1, padx=15)
root.mainloop()
```



Aim: Demonstrate the use of gui by creating a human face and converting celsius into farenite

1) write a program to draw human face using gui

algorithm

step 1: import relevant method from tkinter library

step 2: create an object corresponding to the parent window from tk()

step 3: create an object from canvas() and place it onto parent window along with height and width

step 4: now use pack() for positioning of widget onto the parent window

step 5: now create an object face and use object create_oval() with coordinate 50, 50, 350, 350 and outline = 'black', fill = "yellow" as attribute to create face

step 6: now create eye1 object and again use object create_oval() with appropriate co-ordinate along with fill as attribute create left eye

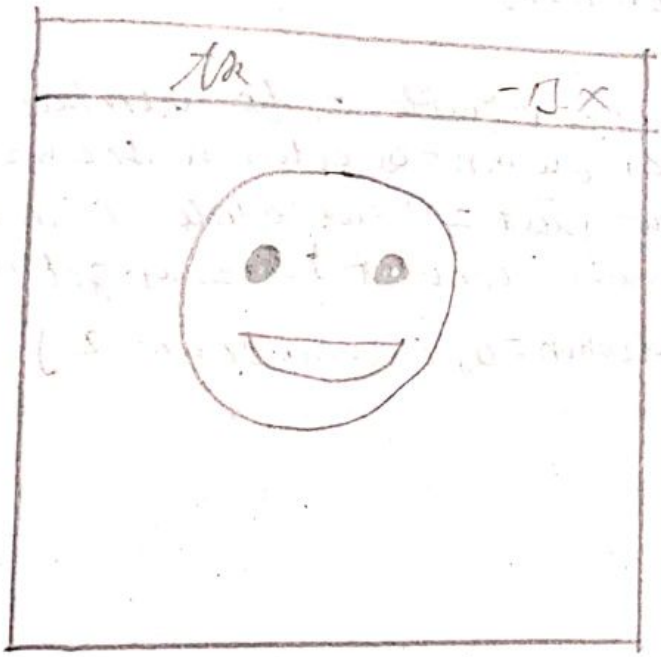
step 7: now object the same step 6 to create right eye

3.5A

Step 8: Create an object mouth and use Object.cr
-arc() with appropriate co-ordinate start
, extent = -180 and fill="red", width = 5 as attribu
to create mouth

Step 9: finally use the mainloop()

```
# code
from tkinter import*
root = tk()
c = canvas (root, width =500, height =500)
c.pack()
face = c.create_oval (50,50,350,350, outline ="black", fill="yellow")
eye1 = c.create_oval (125,125, 75,175, fill ="black")
eye2 = c.create_oval (225,125,275,175, fill="black")
mouth = c.create_arc (175,225,275,275, start=0, extent =-180
                      width =5, fill="red")
```



code :

```
from tkinter import *
window = tk()
fahrenheit = double_var()
fahrenheit.set(32.0)
def convert(celsius):
    fahrenheit.set((9.0/5.0) * celsius + 32)
l1 = label(window, text = "temperature in celsius")
l1 = grid(row=0, column=0)
e = entry(window, textvariable = celsius)
e.grid(row=0, column=1)
celsius = intvar()
l2 = label(window, textvariable = fahrenheit)
l2.grid(row=2, column=0, columnspan=2)
b = button(window, text = "calculate", command
           = lambda: convert(celsius.get()))
b.grid(row=1, column=0, columnspan=2)
mainloop()
window
```

02 write a program to convert celsius into farenheit using gui

algorithm:

Step1: import all the relevant method in tkinter library

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

Step 3: now initialize fahrenheit as double var() and set it to 320

Step 4: now define a function convert with argument celsius and to convert celsius into farenheit using .set()

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

Step 6: now use grid() for position the object onto the parent window initialize celsius as integer using intvar()

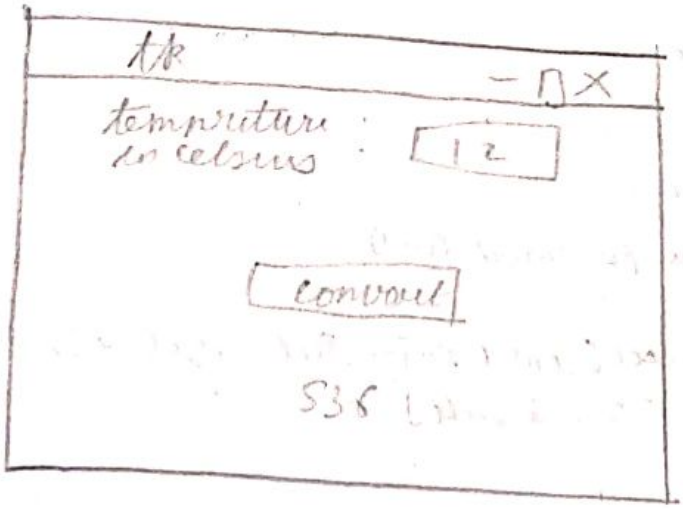
Step 7: create another object and use entery widget to enter the input and place it onto the parent window

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

Step9: now again use label() along with text variable attribute to display output and use grid() for positioning

Step10: finally use mainloop()

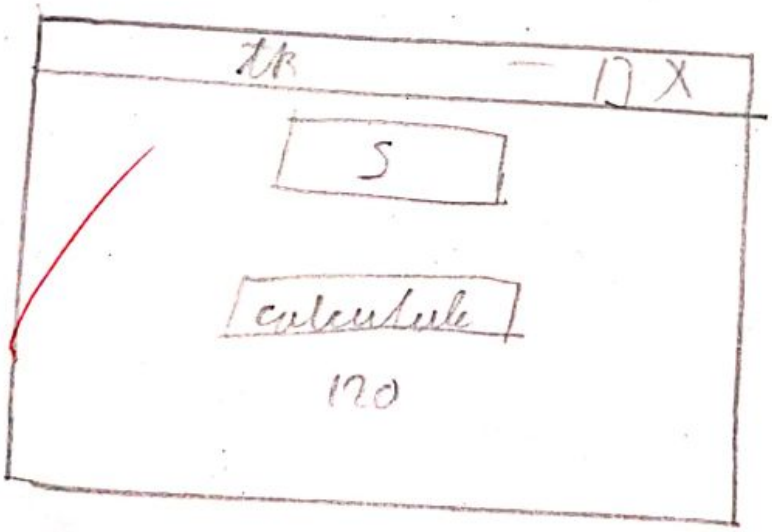
output c



#1 code

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

output



aim : Write a program to find factorial of number and use arthmeetic operation on two number using gui

Q1 Write a program to find factorial of number using gui algorithm :

step 1 :- import relevant methods from tkinter library

step 2 :- now ~~declare~~ define a function factorial to calculate factorial using recursive function

step 3 : define another function calculate to call factorial function

step 4 : now create an object with entry() and use pack() for positioning on parent window

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

step 6 : now again create an object with label() to show show output

step 7 : finally use the mainloop()

Q2 Write a program to perform arithmetic operation on a number using gui

Algorithm

step 1: import relevant method from tkinter library

step 2: now create an object corresponding to parent window

step 3: now define a function calculate to carry out arithmetic operation on a number

step 4: now create object with label() as num1 and num2 and use grid() to place it onto parent window

step 5: create object with entry() to take input from user()

step 6: now initialize v as integer using integer()

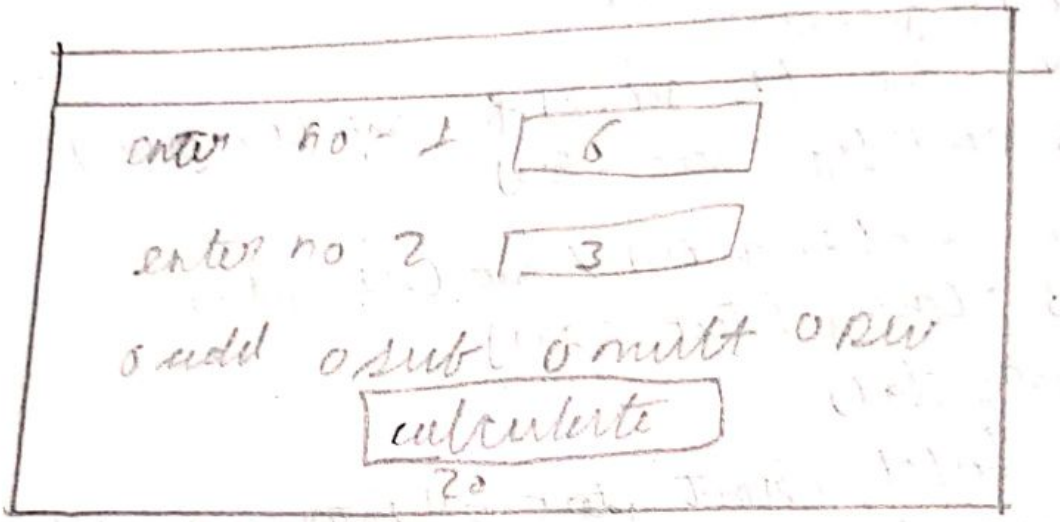
step 7: now create 4 object with radiobutton() to choose any one of arithmetic operation and use grid() for positioning onto parent window

step 8: now create a object with button() along with command attribute to carry out the arithmetic operation of users choice

```
# from tkinter import *
def calculate():
    if int(v.get()) == 1:
        res = int(e1.get()) + int(e2.get())
        l3.config(text = res)
    elif int(v.get()) == 2:
        res = int(e1.get()) - int(e2.get())
        l3.config(text = res)
    elif int(v.get()) == 3:
        res = int(e1.get()) * int(e2.get())
        l3.config(text = res)
    else:
        res = int(e1.get()) / int(e2.get())
        l3.config(text = res)
root = Tk()
l1 = Label(root, text = "Enter a no:")
l1.grid(row = 0, column = 0)
e1 = Entry(root)
e1.grid(row = 0, column = 0)
l2 = Label(root, text = "Enter a no:")
l2.grid(row = 1, column = 1)
e2 = Entry(root)
e2.grid(row = 1, column = 1)
    v = IntVar()
r1 = Radiobutton(root, text = "sub", variable = v, value = 1)
r1.grid(row = 2, column = 0)
r2 = Radiobutton(root, text = "sub", variable = v, value = 2)
r2.grid(row = 2, column = 1)
r3 = Radiobutton(root, text = "mult" variable = v, value =
r3.grid(row = 2 column = 2)
r4 = Radiobutton(root, text = "div" variable = v, valu
r4.grid(row = 2) column = 3)
B = Button(root, text = "calculate" command = calculate)
B.grid(row = 3, column = 1 columnspan = 2)
```

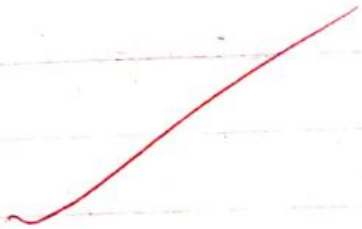
l3 = label(root)
l3.grid(row=4, column=2)
root.mainloop()

output :



step 9: now create a object with label() to show output

step 10: finally use the mainloop()



Practical 8

aim :- demonstrate the use of socket module and server direct program

a1 write a program to demonstrate use of socket module and server direct program

algorithm

1) import the socket module to import relevant method

2) define a function as server program to get hostname

3) now get value for port variable to initialize port no above 1024

4) use socket() to get instance

5) now use bind() function to bind host address and port together to configure how many direct to source can list simultaneously

now use accept() to accept new connection

now print the address

use while loop as true to receive data structure

```
# code 1
import socket
def server
    host = socket.gethostname()
    port = 5000
    server = socket.socket()
    server.socket.bind((host,port))
server.socket.listen(2)
conn, address
print("connection from:" + str(address))
while true:
    data = conn.recv(1024).decode()
    if not data:
        break
    print("from connected user" + str(data))
    data = input("->")
    conn.send(data.encode())
```

now run the program and now right write client programs)

output 1

connection from: (127.0.0.1, 578.22)
from connected user the
Hello
from connected user: how are you
- Ok Good
from connected user user: ok, awesome!
ok then bye

A2: # code 2

```
import socket
def direct_program():
    host = socket.gethostname()
    port = 5000
    direct_socket = socket.socket()
direct_socket.connect((host, port))
message = input("_>")
while message.lower().strip() != 'bye':
    direct_socket.send(message.encode())
    data = direct_socket.recv(1024).decode()
print("Received from server: " + data)
message = input("_>")
direct_socket.close()
```

output

```
$ python 3.6 socket_client.py
A: Hi
Received from server: Hello
→ how are you
Received from server: good
awesome
Received bye from server - Ok then bye
→ bye
```

now close direct program

algorithm

1) import socket module to import methods that are relevant

2) define a function direct program get the hostname and give port a value 5000

3) now again initiate by using socket.socket

4) use convert() to convert the server

5) now take the input (i+p)

use while conditional loop to send a message

now use decode to recieves response

now show the data

again take input

close the program by using close()

Practical 9

Aim : Demonstrate the use of database connectivity algorithm

1) import sqlite 3 module to import relational database

2) Now initialize a variable connect to connect to database by using connect() to a new database using extension db

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

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

5) use fetch all() to show the output

use commit to save all changes

use close() to terminate the program

```
>>> import sqlite3
>>> conn = sqlite3.connect("student1.db")
>>> cur = conn.cursor()
>>> cur.execute("create table student (roll-no int(5)
primary key, name
varchar(50) not well address
varchar(50) not well class varchar
(50), dob date))
sqlite3.Cursor object at 0x0322GBB60)
>>> cur.execute("insert into student values (101, "yash"
, "borivali", "byes", "13-4-2001")
sqlite3 cursor object at 0x03?22EB60
cur.execute("insert into student values
(102, "kiran", "kandivali", "byes", 3-4-20
sqlite3.cursor object at 0x03 22CB60
cur.fetchall()
[(101, 'yash', 'borivali', 'byes', 3-4-20
(102, 'kiran', 'kandivali', 'byes', 13-9-20
cur.execute("update student set
dob = 13-4-2002 where roll-no =
sqlite3.cursor object at 0x3322E
cur.execute("select * from student
where address = "kandivali")
sqlite3 cursor object at 0x0322EBE0
cur.fetchall()
[(102, 'kiran', 'kandivali', byes, 13-09-2002
cur.execute("commit")
sqlite3.cursor object at 0x03222EB60
cur.close()
```