: 1

Program

```
fileobj = open ("abc.txt","w")
fileobj.write ("Python \n Is \n Interted \n language")
fileobj.close()
fileobj= open ("abc.txt","r")
Str1 = fileobj.read()
Print "the output of read method \n", Str1)
fileobj.close()
a = fileobj.name
b = fileobj.closed
c = fileobj.mode
d = file obj.softspace
Print a, b, c, d.
```

Output

```
The output of read method \n Python
Is
Interted
language
abc.txt True r 0
```

Practical no. 1

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

Algorithm

step 1:- Read the file object by using the open method & use the Right access mode. followed by writing some content onto the file & then closing the file.

```
fileobj = open ("abc.txt", "w")
fileobj.write ("....." + "\n")
fileobj.write ("....\n....\n...\n")
fileobj.close()
```

step 2:- Now open the file in the read mode & use the read method or read line method or read lines method and store the output in the variable & finally display the content of that variable.

```
fileobj = open ("abc.txt", "r")
    Str1 = fileobj.read()
    Print "the output of read method in",
           str1
```

Step 3 :- Now use the file object for finding the name of the file, mode in which the file is open, whether the file is still open or close & finally the output of the softspace. display attribute

a = fileobj.name
b = fileobj.closed
c = fileobj.mode
d = fileobj.softspace
print a, b, c, d.

```
mytuple1 = ("banana", "oranges", "apples")
myiter1 = iter(mytuple1)
Print(next(myiter1))
for p in mytuple1:
    Print(p).
```

<u>output</u>.

```
for p in mytupl:
    Print(i)
```

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

-10-19.

Practical No. 2

Aim: The uses of iteration and iterable Method.

1) Create a tuple and use iter.
2) Next Method: to display first value.
3) For using the individual value for given obj can use for loop.

Sm 9/11/19

16/12/19

Practical No. 3

Exception

(1) Algorithm

step 1 :- Use the try block to define the normal course of action. for Eg :- Define the file obj and open the file in the write mode & write same content onto the file

step 2 :- Use the except block with the I/O error as an environment error and convey the appropriate message to the user else display the message that the operation is carried out successfully.

Program

```
try:
    fo = open ("abc.txt", "r")
    fo.write ("hi, this is saksh")
except IOError:
    Print ("Error occured order Environment error")
else:
    Print ("operation successful)
```

Output:

Environmental Error

Program :

```
try:
    a = int (input ("Enter the number"))
except. ValueError:
    print ("Arithmetic error")
else:
    print ("Succesful")
try:
    fo = open ("abc1.txt", "r")
    fo. write ("hello, there how are you?")
except IOError:
    print ("Environment error")
else:
    Print ("Succesfull)
```

output

Enter : 14

Arithmetic error.

Program for demonstrating the use of value error in the given program statement:-

Step 1: Accept the value from the user and if it is a valid value display the entered value & terminate the condition by using the break statement.

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

Step 3: You can define the multiple exception using the except statement for finding the different category of errors.

Practical-4

Regular expression

Step 1: Import re module declare Pattern and declare sequence use match method with declare arguments if arguments matched then Print the same otherwise Print Pattern NOT FOUND!

Step 2: Import re module declare Pattern with literal and meta character. Declare string value, use the find all (). with arguments and Print the same.

Step 3: Import re module declare Pattern with meta character use the split () and Print the output.

```
# 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 (segration)
import re
Pattern = r'\d+'
string = 'hello123, howdy 789, 45 howru'
output = re.findall(Pattern, string)
Print(output)
>>> ['123', '789', '45']

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

```
# no-space:
  import re
  String = 'abc def ghi'
  Pattern = r'\s+'
  replace = ''
  V1 = re.sub (Pattern, replace, string)
  Print (V1)
>>> abcdefghi

# group()
import re
Sequence = 'Python is an Interesting language'
  V = re.search ('\Ap\thon', Sequence)
Print(V)
  V1 = V.group()
  Print (V1)
>>> <_sre.SRE_Match object at 0x0281DF00>
    Python

# verifying the given set of Phone.numbers -
  Import re
  lst1 = ['8004567891', '9145673210', '7865432981',
          '9876543201']
  for value in lst1:
    if re.match (r'[8-9]{1}[0-9]{9}',
      value or len (value) == 10):
      Print ("Criteria matched for cell number!")
    else:
      Print ("Criteria failed!")
```

Step 4 : import re module declare string & accordingly declare Pattern replace the blank space with no space use sub () with 3 arguments & 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 ie search () gives memory location using group () it will show up the matched string.

Step 6 : import re module declare list with numbers use the conditional statement here we have used up the for condition statement. Use if condition for checking first number is either 8 or 9 & next number are in range of 0 to 9 & check whether the entered numbers are equal to 10. if criteria matches print all number matches otherwise Print failed.

step 7: import re module declare a string use the module with findall() for finding the vowels in the string & declare the same.

step 8: import re module declare the host & domain name declare pattern for seprating the host & domain name use the findall() & print the output respectively.

step 9: import re module enter a string use pattern to display only two elements of the particular string use findall() declare two variables with initial value as 2010 use for condition & subsequently use the if condition check whether condition satisfy add up the or else increment value & display the values subsequently.

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

~~vala~~ vowels

```
import re
str1 = 'Plant is life overall'
output = re.findall (r'\b [qsPOU AEIOU] \w+', str1)
Print (output)
>>> ['is', 'overall']
```

host & domain

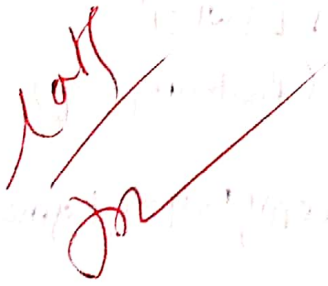
```
import re
seq = 'abc.tcsc@edu.com, xyz@gmail.com'
Pattern = r'[\w1.-] + [\w1.-]'
output = re.findall (Pattern, seq)
Print (output)
>>> ['abc.tcsc', 'edu.com', 'xyz', 'gmail.com']
```

counting of first 2 letters.

```
import re
S = 'mr.a, ms.b, ms.c, mr.t'
P = r'[/ms/mr/] +
O = re.findall (P, s).
Print (O)
m = 0
f = 0
for v in O:
```

```
if (v=='ms'):
    f=f+1
else
    m=m+1
Print("NO. of males Ps:",m)
Print("NO. of females Ps:",f)
```

```
>>> ['mr','ms','ms','mr']
('No. of males is:',2)
('No. of females Ps:',2)
```



Practical No. 5(A)

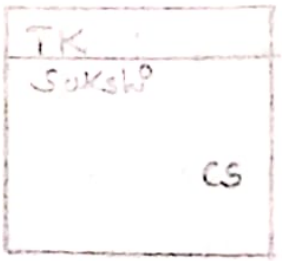
GUI

```
# from tkinter import *
root = TK()
L1 = label (root, text = "Sakshi", bg = "red",
            fg = "10")
L.Pack (Pady = 20)
L1 = Label (root, text = "CS", bg = "grey",
            fg = "black", font )
L1.Pack (side = LEFT, Padx = 20)
L2 = Label (root, text = "CS", bg = "blue", fg = "b
```

#] Algorithm

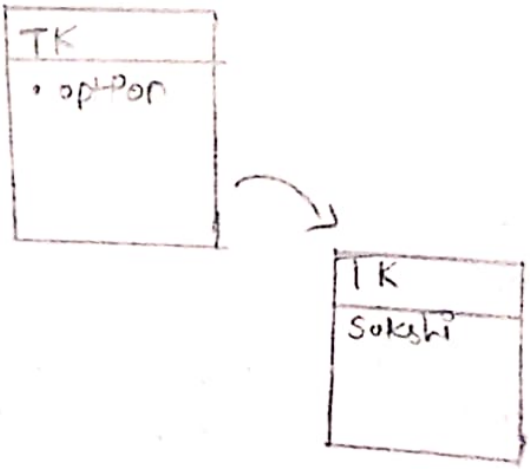
1. Use the TKinter library for importing the features of the text widget.
2. Create a variable from text method & position it on to the parent window.
3. Use the Pack method along with the object created from the text method & use the Parameter.
4. Use the main loop method for triggering of the corresponding events.

```
# from tkinter import *
root = Tk()
L1 = Label (root, text = "Sakshi", bg = "red", fg = "10")
L1 Pack (Pady = 20)
L1 = Label (root, text = "CS", bg = "grey", fg = "black", font
L1.Pack (side = LEFT, Padx = 20)
L2 = label (root. text = "CS", bg = "blue". fg = "black", font
L2. Pack (side = LEFT, Pady = 20)
root. mainloop()
```



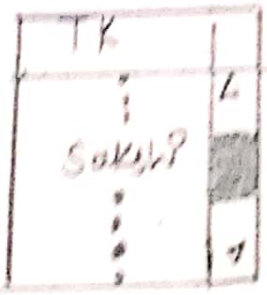
Radio Button

```
from tkinter import *
Root = Tk()
def set():
    Print ("Sakshi")
r1 = Radio Button (root, text = "option", command
r1.Pack()
root.mainloop()
```



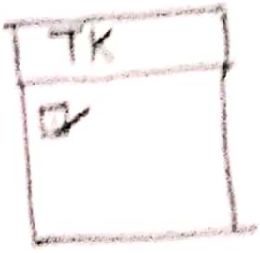
Scroll Bar

```
from tkinter import *
root = tk()
root = "____"
s = Scrollbar(root)
t = Text(root, height = 10, width = 10)
s.pack(side = RIGHT, fill = "y")
t.pack(side = LEFT, fill = "y")
s.config(command = t.yview)
t.config(yscrollcommand = s.set)
t.insert(END, para)
root.mainloop()
```



Listbox

```
from tkinter import *
window = tk()
window.geometry("680x500")
Label(window, text = "numbers:").pack()
frame = Frame(window)
frame.pack()
lstnodes = Listbox(frame.pack(), ("times of 10,)
lstnodes.pack(side = LEFT, fill = "y")
for x in range(100):
    lstnodes.insert(END, str(x))
window.mainloop()
```



5. Now repeat the above steps with the label method which takes the arguments.

6. Repeat the steps using Radio Button

List Box

1. Above mentioned all the steps repeat for the Listbox using Listbox method.

Practical 5 (B)

ScrollBar

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

2. Write a para into the new created object.

3. Use the scrollbar method with text & config attribute & also use the pack method.
Use text & insert to write the text in the scroll bar & make use of it.

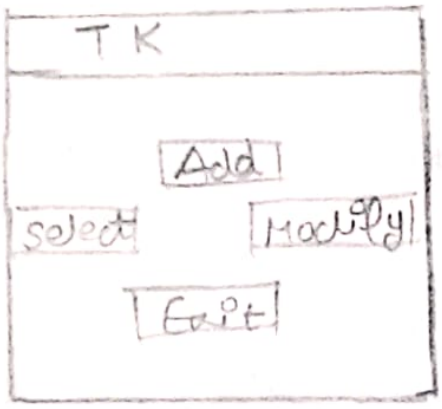
4. Finally make use of the main loop method.

12] ALGORITHM

1. Import the relevant method from tkinter library
2. Define the object corresponding to the Parent window & define the size of the Parent window in terms of pixels.
3. Use the Frame method & create leftframe & rightframe object with text attribute.
4. Create the required buttons by using the Button method with the text attribute then use the Pack method with side, Padx & Pady attribute respectively.
5. Repeat the steps for the desired no. of buttons & then finally use the mainloops method.

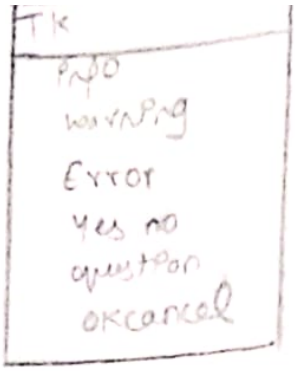
Frame & Button

```
from tkinter import *
window = Tk()
window.geometry("650x500")
frame = Frame(window)
frame.pack()
leftframe = Frame(window)
leftframe.pack(side = LEFT)
rightframe = Frame(window)
Frame.pack(side = RIGHT)
b1 = Button(frame, text = "select")
b1.pack(side = LEFT, ~~text = "select"~~) padx=50, pady=10)
b2 = Button(frame, text = "Modify")
b2.pack(side = RIGHT, padx = 60, pady = 0)
b3 = Button(frame, text = "Add")
b3.pack(side = Bottom, pady = 70)
b4 = Button(frame, text = "Exit")
b4.pack(side = Top, pady = 10)
window.mainloop()
```



Message box

```
from tkinter import *
# import messagebox
root = Tk()
def info():
    messagebox.showinfo("Sakshi", "Python")
def warning():
    messagebox.showwarning("Sakshi", "Python")
def error():
    messagebox.showerror("Sakshi", "Python")
def askyesno():
    messagebox.askyesno("Sakshi", "Python")
def question():
    message.askquestion("Sakshi", "Python")
def okcancel():
    message.askokcancel("Sakshi", "Python")
b1 = Button(root, text="Info", command=info)
b1.pack()
b2 = Button(root, text="warning", command="warning")
b2.pack()
b3 = Button(root, text="error", command=error)
b3.pack()
b4 = Button(root, text="yes no", command=askyesno)
b4.pack()
b5 = Button(root, text="question", command=question)
b5.pack()
b6 = Button(root, text="okcancel") command=okcancel)
b6.pack()
```



Practical 5 (c)

#3] Message Box

1. Import the relevant method from the tKinter Library

2. Define a function & use the message box along with the different methods available.

3. Create an object from the button method & place it onto the Parent window with the title of the Button specified & the corresponding event called for the triggering.

4. Use the Pack method to display the button widget & finally use the mainloop method.

5. If the user want to hide the Parent window & only the Information window should be visible corresponding to the & options given above the withdraw method is used.

#4 Traversing

1. Define a function & create a object of the given window by using the three methods namely config, title & minsize.

2. Now create a button object by using the button method with text & command attribute which will trigger the corresponding events.

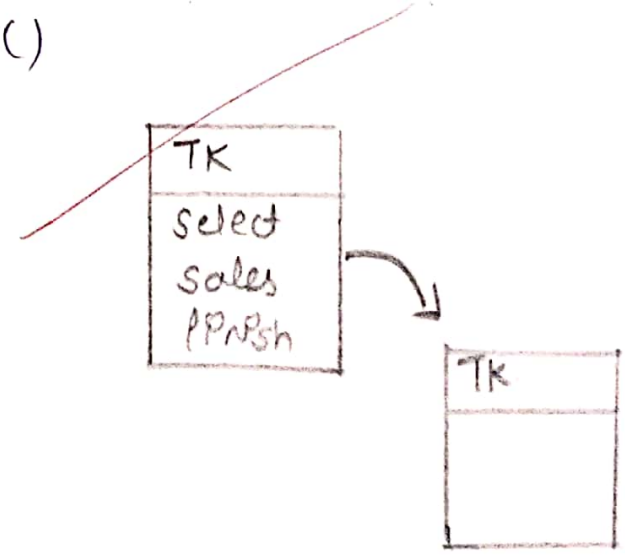
3. Use the Pack method with first button object.

4. Now repeat the same steps for the next button object.

5. Finally use the mainloop method for triggering the corresponding events.

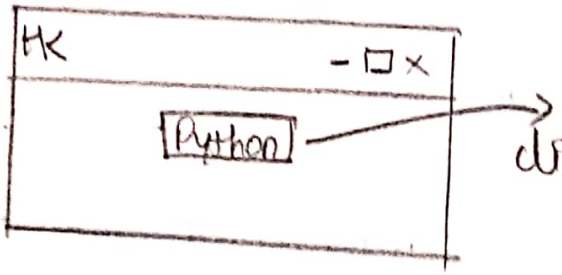
Traversing

```
from tkinter import *
root = TK()
def main()
    root = TK()
    root.config()
    root.title()
def sales():
    Root = TK()
    Root.config()
    root.title()
def finish():
    quit()
b1 = Button (root, text = "select", command = main)
b1.Pack()
b2 = Button (root, text = "Sales", command = sales)
b2.Pack()
b3 = Button (root, text = "finish", command = finish)
b3.Pack()
Root.mainloop()
```

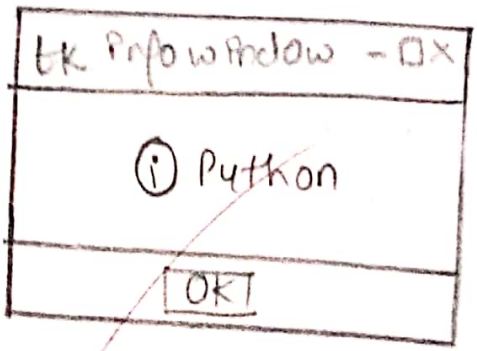


```
from tkinter import *
import tkMessageBox
root = Tk()
def function():
    tkMessageBox.showinfo("info window",
    "Python")
    b1 = Button(root, text="Python", command
        = function)
    b1.pack()
root.mainloop()
```

output:



click than this window is pop up



Practical 5(B)

Aim :- Gui Components.

Step 1: Import the relevant methods from tkinter library

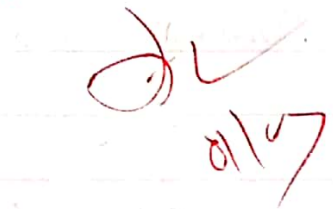
Step 2: Import tkMessagebox

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

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

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

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



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

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

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

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

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

Step 6 : Finally called the mainloop() for event driven Programming.

Multiple window
Different button (Relief)

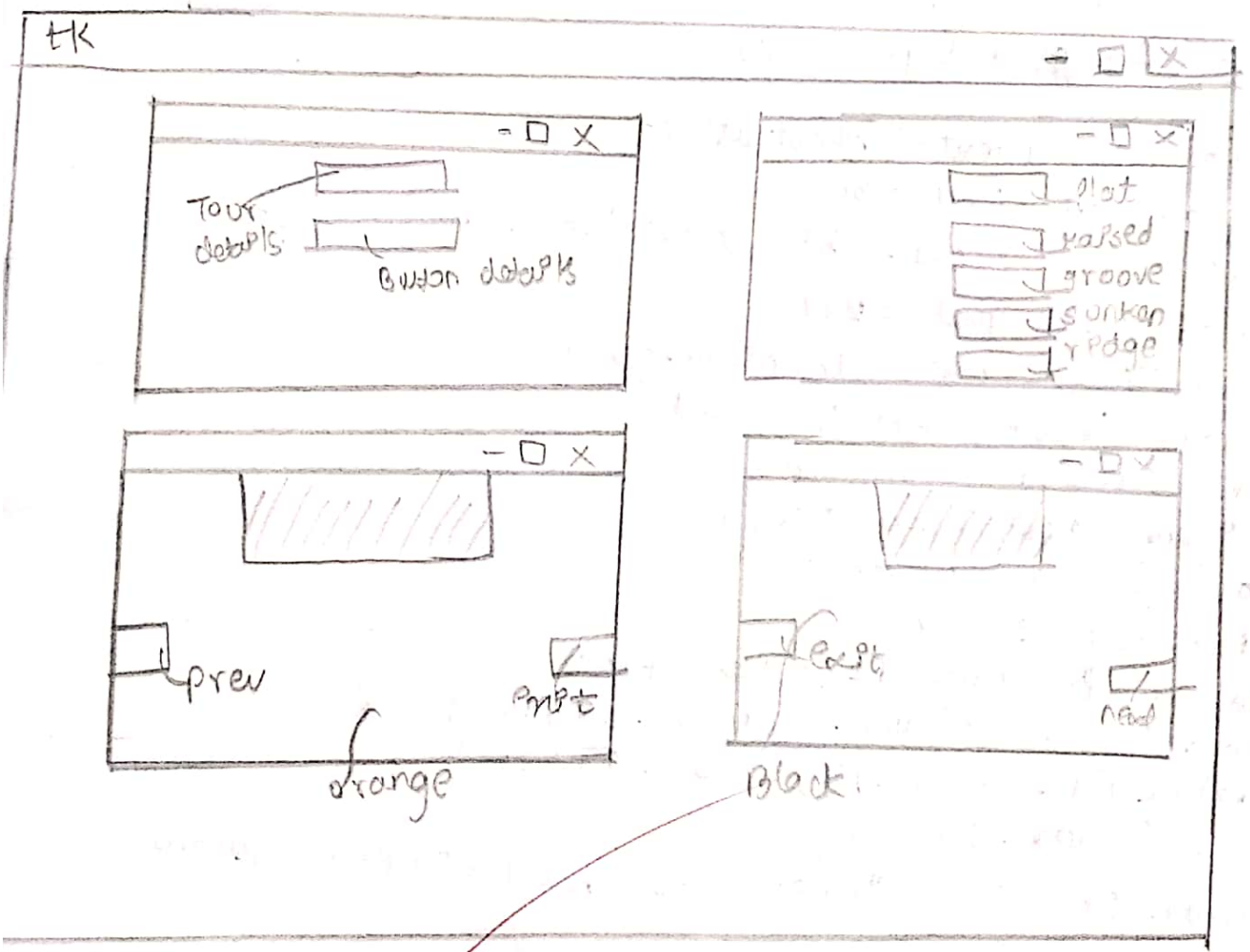

```
from Tkinter import *
root = Tk()
root.minsize (300,300)

def main():
    top = Tk()
    top.config (bg = "black")
    top.title ("HOME")
    top.minsize (300,300)
    l = label (top, text = "SAN FRANCISCO
    \n Places of Interest : \n Golden Gate
Bridge \n Lombard Street \n Chinatown \n coit Tower")
    L.Pack()
    b1 = Button (top, text = "next", command = second)
    b1.Pack (side = RIGHT)
    b2 = Button (top, text = "exit", command = terminate)
    b2.Pack (side = LEFT)
    top.mainloop()
```

```
def second():
    top2 = Tk()
    top2.config(bg = "orange")
    top2.title("About us!")
    top2.minsize(300,300)
    l = Label(top2, text = "Created by: Satoshi Kamble
        \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 = top2.quit)
    b2.pack(side = RIGHT)
    top2.mainloop()

def button3():
    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 = "groove 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 DETAILS", command = button)
b6.pack()
root.mainloop()
```



```
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," 10: Name: Apple")
    list2.insert (2," Products: iphone")
    list1.insert (3," language : swift")
    list1.insert (4," OS. IOS")
    list1.grid (ipadx = 30)
def abouts():
    list2 = label (text =" About us" )
    list2.grid (ipadx = 30)
    list3 = label (text =" Steve Jobs Theatre March 2020")
    list3.grid (ipadx = 24)
p1 = 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)
p2 = p1.subsample (5,4)
l1 = Label (f1, image = p2, relief = FLAT)
l1.grid (row =1, column = 0, padx = 20, pady =15)
l2 = Label (f2, image = p1, relief = SUNKEN)
l2.grid (padx = 25, pady =10)
b1 = Button (f1, text =" Information", relief = SUNKEN, command = info)
b1.grid (row1, column =0)
```

Practical - 5(c)

Aim: Gui components.

Step 1: Import relevant methods from the tkinter library

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

Step 3: define a function finish with the messagebox widget which will display a message i.e a warning message & subsequently 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 & insert the same & finally use the grid() with ipadx attribute.

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

Step 6: Use PhotoImage widget with file & filename with gif attribute.

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

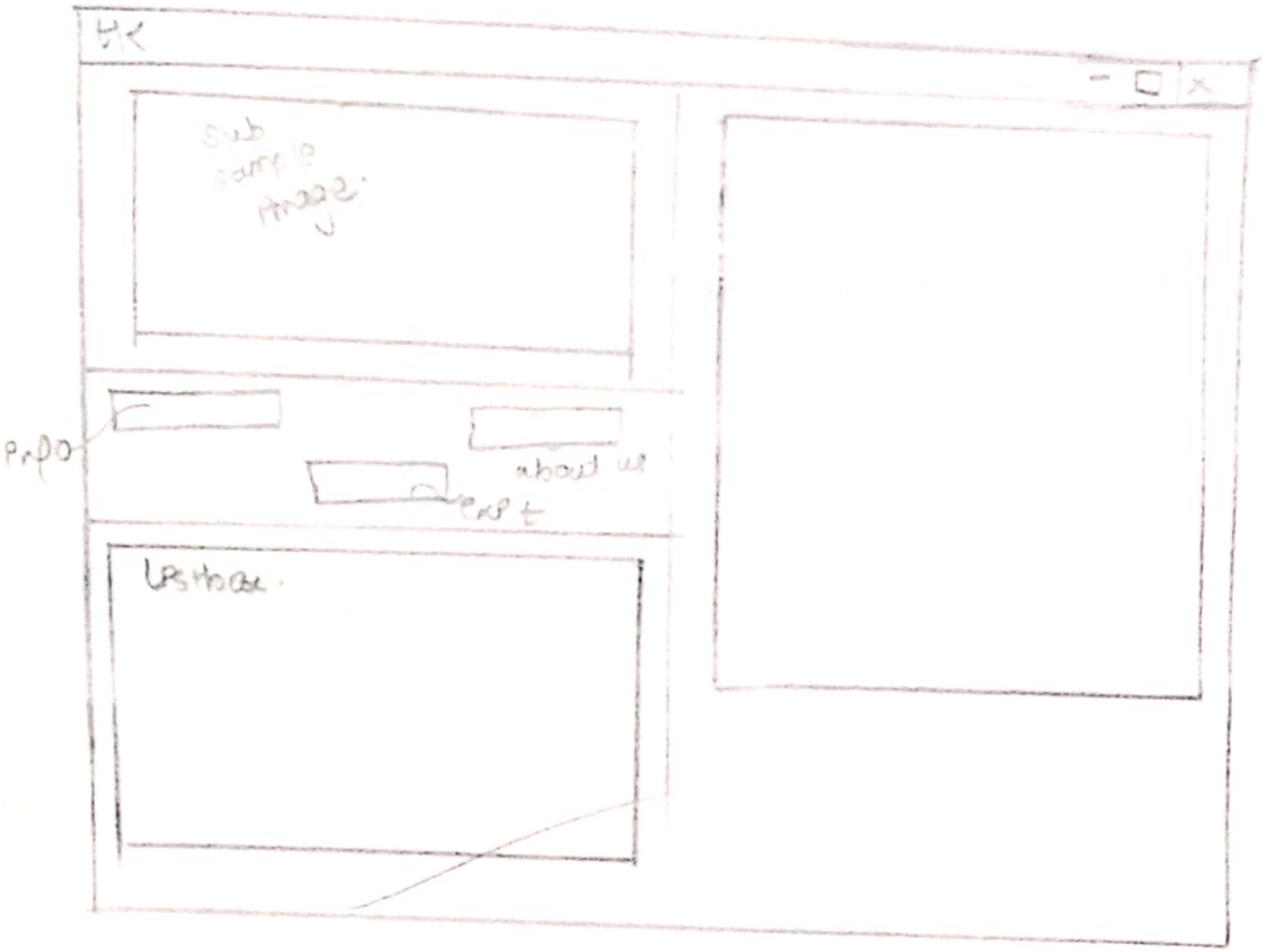
Step 8. Similarly creat another frame object - as declared by step 7.

Step 9: create another object & use the subsample (5, 4)

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

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

```
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 ()
```



:8

Source code:

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

Output:

S

9

Practical - 5 (E)

Aim: To make use of spinbox widget, Paned window & canvas widget.

Spinbox widget:

Step 1: Create an object from the ~~Tc~~ Tk method & subsequently create an object from the spinbox method.

Step 2: Make the object so created onto the Parent window & 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.

Scanned with CamScanner

Paned Window

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

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

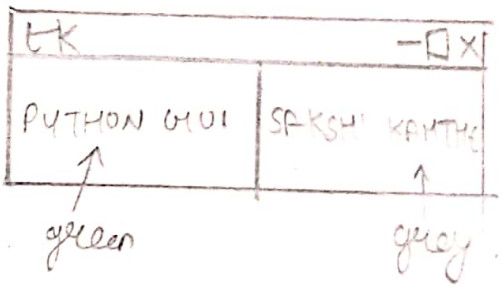
step 3: Similarly create another label object and place it onto the 2nd Paned window object & add onto the 2nd Paned

step 4: Create a second Paned window object and add it onto the 1st Paned window with orientation specified

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 = Panel window (P, orient = VERTICAL, bg = "blue")
P.add (P1)
L2 = label (P1, text = "SAKSHI KAMTHE", bg = "grey")
P1.add (L2)
root.mainloop()
```

output.

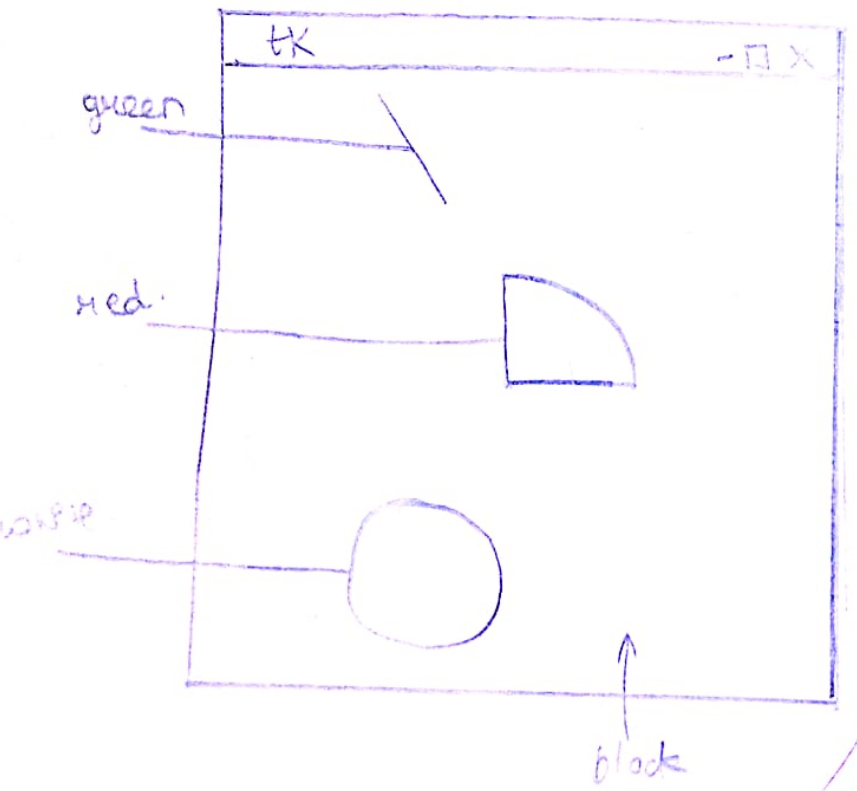


Pr

Source code

```
from tkinter import *
root = Tk()
c1 = Canvas (root, height = 400, width = 400, bg = "blue")
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:

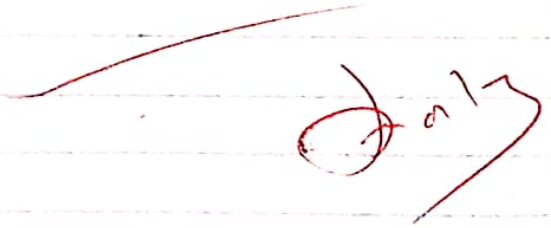


* Canvas widget:

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

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

step 3→ Now call the pack method & the mainloop method.



Pr

Practical-6.

Aim: Database

Algorithm:

Step1: Import db library & 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 site & 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.

g
p
P
l
a

Program:

```
import dbm
db = dbm.open ("database", flag = 'c')
if db ["www"] != None:
        print ("good")

else:
    print ("Not good")
```

output:

good

Program:

```
import os, sqlite3
connection = sqlite3.connect ("student.db")
cl.execute ('Create table Student (Name, RNo, DOB)')
cl.execute ('Insert Into Student values
            ("Adarsh", 1755, 21-01-2000)')
cl.execute ('Insert Into Student values.
            ("Sakshi", 1770, 16-08-2001)')
cl.execute ('insert Into Student values.
            ("Prank", 1762, 02-02-2000)')
Connection commit()
cl.execute ('select from student')
cl.fetchall()
cl.execute ('Drop table Student')
```

2) Algorithm

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

Step 2: Now create connection objects using sqlite library & 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 & respective datatype.

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

Step 6: Use the commit method

Step 7: Use the execute statement along with the cursor objects from accessing the value the data base using selecting from where clause.

Step 8. Finally use the fetchall method for display the value for the table using the cursor objects.

Step 9: Use the execute method & the drop table.

output:

[('Adarsh', 1755, 21-01-2000), ('Saksh', 1770, 16-08-2001), ('Pinank', 1762, 02-02-2000)]

Program:

```
from tkinter import *
root = TK()
root.geometry ('500 x 500')
root.title ("Registration Form")
L = Label (root, text = "Registration Form",
           width = 20, font = ("bold", 20))
L.pack()
L1 = Label (root, text = "Full Name", width = 20,
            font = ("bold", 10))
L1.pack()
E1 = Entry (root)
E1.pack()
L2 = label (root, text = "Email", width = 20,
            font = ("bold", 10))
L2.pack()
E2 = Entry (root)
E2.pack()
L3 = Label (root, text = "Gender", width = 20,
            font = ("bold", 10))
```

Project

Registration form

Small Project based on Gui widgets.

→ 1st import the package

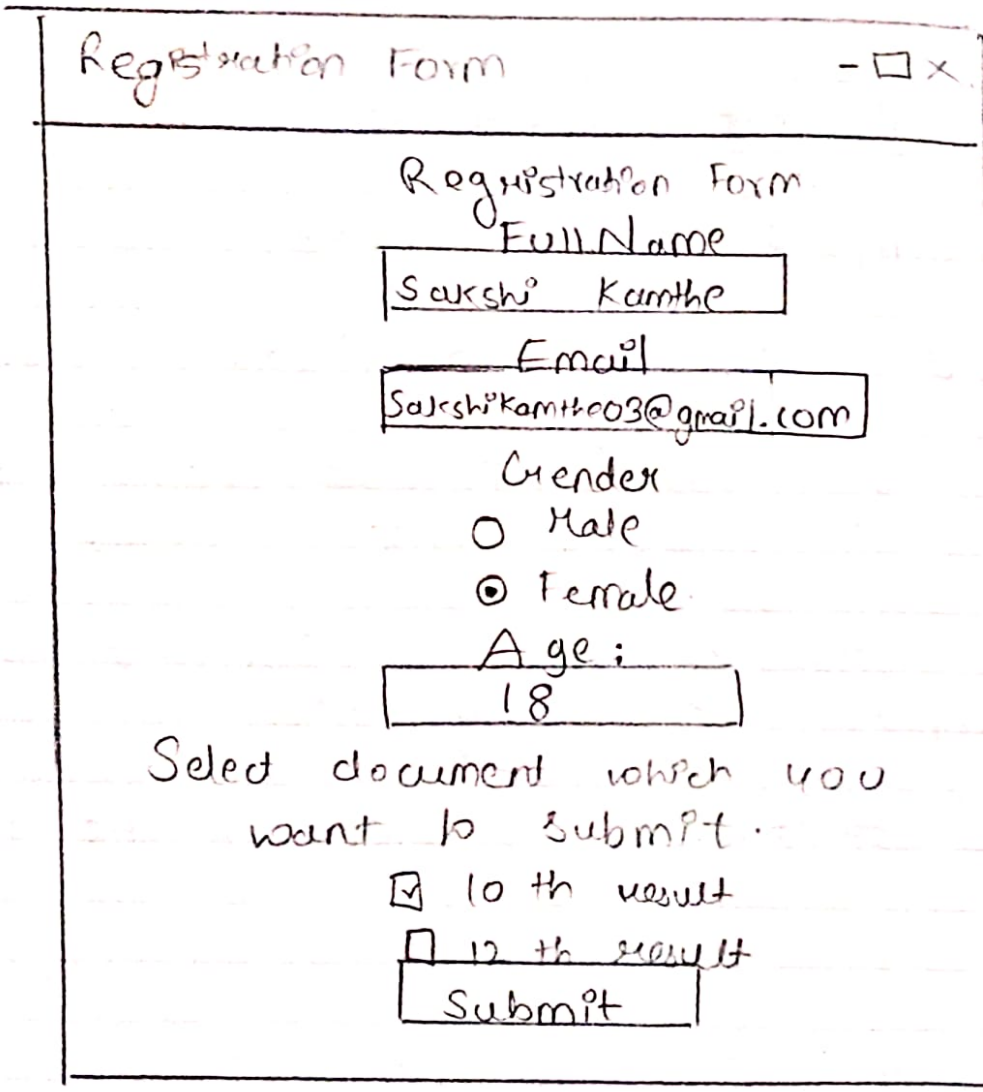
→ Second, we have to draw a window

→ Now we have to add some Level as well as some entry boxes.

→ We have to add a Submit button to submit the information as users.

* The 5 Widgets is used in the Project are:

i) Label
ii) Entry
iii) Radiobutton
iv) Button.
v) checkbutton.



```
L3.Pack()
var = IntVar()
R1 = Radiobutton (root, text = "Male", Padx = 5,
                  Variable = var, value = 1)

R1.Pack()
R2 = Radiobutton (root, text = "Female", Padx=20,
     variable = var, value = 2)

R2.Pack()
L4 = Label (root, text = "Age:", width = 20,
     font = ("bold", 10))
L4.Pack()
E2 = Entry (root)
E2.Pack()
Label (root, text = "Select document which you
       want to Submit").Pack()
C1 = checkbutton (root, text = "10th result")
C1.Pack()
C2 = checkbutton (root, text = "12th result")
C2.Pack()
B1 = Button (root, text = "Submit")
B1.Pack()
root.mainloop()
```

```
# Connection of sqlite3 with Python
import sqlite3
conn = sqlite3.connect("test.db")
Print ("Opened database successfully");

# creation of table.
conn.execute("CREATE TABLE COMPANY_11
   CID INT PRIMARY KEY NOT NULL,
   NAME TEXT NOT NULL,
   AGE INT NOT NULL,
   ADDRESS CHAR(50),
   SALARY REAL);"")
Print ("Table created successfully");
conn.close()

# insert into table values.
conn = sqlite3.connect('test.db')
Print ("Opened database successfully");
conn.execute("INSERT INTO COMPANY_11 (ID, NAME, AGE, ADDRESS,
                                      SALARY)
VALUES (1712, 'TANVI', 20, 'California', 2200000.00)");
conn.execute("INSERT INTO COMPANY_11 (ID, NAME, AGE, ADDRESS,
                                      SALARY)
VALUES (1770, 'Sakshi', 80, 'Texas', 115000.00)");
conn.execute("INSERT INTO COMPANY_11 (ID, NAME, AGE, ADDRESS,
                                      SALARY)
VALUES (1769, 'Adarsh', 30, 'Newyork', 116000.00)");
```

Project

Database. Connectivity.

→ I made this Project in order to show that how Python is has Platform Portability with the database operation.

→ This Project deals with database management of employee's database.

→ Here by we have entered the data values, secondly we have displayed values updated the values & deleted & trun values.

→ This Project says how can we do the database operations with in Python.

conn.execute ("WSERT NTO COMPANY-11 (ID, NAME, AGE, ADDRESS
SALARY)\
VALUES (1763, 'PINANK', 27, 'Norway', 119000.00)");
conn.commit()
Print ("Records created successfully");
conn.close()

# displaying of the values using the select clause
conn = sqlite3.connect ('test.db')
cursor = conn.execute ("SELECT id, name, address, salary
from COMPANY_11")
for row in cursor:
Print ("ID = ", row[0])
Print ("NAME =", row[1])
Print ("ADDRESS =", row[2])
Print ("SALARY =", row[3])
Print ("\n operation done successfully");
conn.close()

# UPDATE select from where clause.
conn = sqlite3. conned ('test, db')
Print ("UPATION OF VALUES");
conn.execute (" UPATE COMPANY-11 set SALARY = 22000.0
where ID = 1712")
conn.commit.
Print ("Total number of row updated:", conn.total_ch

```
cursor = conn.execute ("SELECT id, name, address, salary from
                        COMPANY_11")

for row in cursor:
   print ("ID =", row[0])
   print ("NAME =", row[1])
   print ("ADDRESS =", row[2])
   print ("SALARY =", row[3])
   print ("\n Operation done successfully");
conn.close()
```

#DELETION using where clause

```
conn = sqlite3.connect ("Test.db")
print ("DELETION");
conn.execute ("DELETE from COMPANY_11 where ID = 1770;")
conn.commit
print ("Total number of rows deleted:", conn.total_changes)
cursor = conn.execute ("SELECT id, name, address, salary
                        from COMPANY_11")

for row in cursor:
   print "ID =", row[0]
   print "NAME =", row[1]
   print "ADDRESS =", row[2]
   print "SALARY =" row[3], "\n"
   print "Operation done successfully";
conn.close()
```

TRUNCATE DDL Statement.

```
conn = sqlite3.connect('test.db')
cursor = conn.execute("DROP TABLE company-11");
print("\n VALUES truncated!")
conn.close.
```

OUTPUT:

```
Opened database successfully.
Table created successfully
Opened database successfully
Records created successfully
('ID =', 1712)
('NAME =', U'Tanvi')
('ADDRESS =', U'California')
('SALARY =', 220000.0)
operation done successfully.
('ID =', 1770)
('NAME =', USakshi')
('ADDRESS =', U Texas')
('SALARY =', 115000.0)
```

operation done successfully
('ID = ', 1769)
('NAME =', u'PINANK')
('ADDRESS =', u'Newyork')
('SALARY =', 116000.0)

operation done successfully
('ID = 1763)
('NAME =', u'Adarsh')
('ADDRESS =', u'Norway')
('SALARY =', 119000.80)

operation done successfully
UPDATION OF VALUES
('Total number of rows updated :', 1)
('ID =', 1712)
('NAME =', u'Tanvi')
('ADDRESS =', u'California')
('SALARY =', 22000.0)

operation done successfully
('ID =', 1770)
('NAME =', u'Sakshi')
('ADDRESS =', u'Texas')
('SALARY =', 115000.0)

operation done successfully
('ID =', 1769)
('NAME =', u'PINANK')
('ADDRESS =', u'

operation done successfully.
('ID = ', 1763)
('NAME = ', 'Adarsh')
('ADDRESS = ', 'Norway')
('SALARY = ', 119000.0)

operation done successfully
DELETION
('Total number of rows deleted = ', 1)
ID = 1712
NAME = Tanvi
ADDRESS = California
SALARY = 220000.0.

operation done successfully
ID = 1770.
NAME = Sakshi
ADDRESS = Texas
SALARY = 115000.0

operation done successfully.
ID = 1769
NAME = Pinank.
ADDRESS = Newyork.
SALARY = 116000.0 ₹.

operation done successfully.

VALUES truncated!