☆☆ INDEX ☆☆

No.	Title	Page No.	Date	Staff Member's Signature
1	Demonstrate the use of different file accessing mode, different attributes read method →	25-27	2-12-19	
2	To study about Iterat-ors →	28-30	9-12-19	
3	To demonstrate the use of exception handling →	31-32	16-12-19	
4	To demonstrate the use of Regular expression →	33-36	06-1-20	
5	To study about the GUI components →	37-38	12-1-20	
	B) To study about the GUI components →	39-43	03-01-20	
	c) To study about the GUI components →	44-47	03-01-20	

☆☆ INDEX ☆☆

No.	Title	Page No.	Date	Staff Member's Signature
6	To demonstrate the use of Spinbox, Paned Window and Canvas Widget ⟶	48-50	10-02-20	
7	To demonstrate the use of database Connectivity ⟶	51-53	17-02-20	

PRACTICAL – 1

AIM : Demonstrate the use of different file accessing mode, different attributes, read method.

STEP 1 : 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.

STEP 2 : 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.

STEP 3 : Now use the fileobject for finding the name of the file, the file mode in which it is opened whether the file is still open or close and finally the output of the softspace attribute.

STEP 4 : Now open the fileobj in write mode. Write some another content close. subsqu then again open the fileobj in 'w+' made that is the update mode and write contents.

STEP 5 : Open fileobj in readmode, display the update written contents and close. Open again in 'r+' mode with paramet -er passed out and display the output subsequently.

STEP 6 : Now open fileobj in append mode open write method write content close the fileobj again. Open the fileobj in read mode and display the appen output.

STEP 7 : Open the fileobj in read mode. Declare a variable and perform fileobj dot tell method and store the output consequently in variable.

STEP 8 : Use the seek method with the argum-ents with opening the fileobj in read mode and closing subsequently.

STEP 9 : Open fileobj with read mode also use the readlines method and store the outp-ut consequently in and print the same for counting the length use the for condi-tion statement and display the length.

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

fileobj = open("abc.txt", "r") # read mode
# Read
str_1 = fileobj.Read()
print("The output of Read method :", str_1)
fileobj.close()
>>> ('The output of Read method :', 'Computer Science
    subjects in DBMS in python in DS \n')

# Readline()
fileobj = open("abc.txt", "r")
str_2 = fileobj.Readline()
print("The output of Readline method :", str_2)
fileobj.close()
>>> ('The output of Readline method:', 'computer science
    subject \n')

# Readlines()
fileobj = open("abc.txt", "r")
str_3 = fileobj.Readlines()
print("The output of Readlines method:", str_3)
fileobj.close()
>>>('The output of Readlines method:', ['computer
  science subjects \n', 'DBMS\n', 'Python \n', 'DS\n'])
```

```
#file attributes
a=fileobj.name
print("name of file (name attribute):", a)
>>>('name of file (name attribute):', 'abc.txt')
b=fileobj.closed
print ("(close) attribute :", b)
>>>('(close) attribute :', 'True')
c=fileobj.mode
print ("file mode", c)
>>>('file mode', 'r')
d=fileobj.softspace
print ("softspace", d)
>>>('softspace :', 0)
```

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

```
# r+ mode
fileobj=open("abc.txt", "r+")
str_1=fileobj.read(r)
print("output of r+", str_1)
fileobj.close()
>>>('output of r+', 'Khushi')
```

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

```
# read mode
fileobj=open("abc.txt", "r")
str_2=fileobj.read()
print("output of read
 mode!", str_2)
>>>('output of read mode!',
 'Khushi')
```

as

```
# Append Mode
fileobj = open("abc.txt", "a")
fileobj.write("data structure")
fileobj.close()
fileobj = open("abc.txt", "r")
str_3 = fileobj.read()
print("Output of append mode :", str_3)
fileobj.close()
>>>('output of append mode :', 'Khushi', 'data struct

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

# Seek()
fileobj = open("abc.txt", "r")
str_4 = fileobj.seek(0,0)
str_5 = fileobj.read(10)
print("The beginning of the file :", str_5)

# Finding length of different lines exist within lines
fileobj = open("abc.txt", "r")
str_6 = fileobj.readlines()
print("output :" str_6)
For line in str_6:
        Print (len(line))
    fileobj.close()
>>>('output:', ['college database'])
```

PRACTICAL-2

AIM: To study about Iterators.

STEP 1: 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.

STEP 2: The similar output can be obtained by using for conditional statement. An iterable variable is to be declared in for loop which will iterate.

STEP 3: 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.

STEP 4: Call the declared function using function call.

STEP 5 : Using for conditional statement specifying the range use the list typecasting with map method declare a 'lambda' i.e- annonymus function and print the same.

STEP 6 : Declare a listnum variable and declare some elements then use the map method with help of lambda function give two argument display the output.

STEP 7 : Define a function even with a parameter then using conditional statements do check whether the number is even and odd and return respectively

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

STEP 9 : Now use the next() and define the logic for displaying odd value.

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.

#itey() and next()

```
mytuple 1 = ("banana", "orange", "apple")
myitey 1 = itey(mytuple)
print(next(myitey 1))
myitey 2 = itey(mytuple)
print(next(myitey 1))
myitel 3 = itey(mytuple)
print(next(myitey 1))
```

>>> banana
orange
apple

for loop

```
mytuple 1 = ("Kevin", "stuart", "bob")
for x in mytuple1:
    print(x)
```

>>> Kevin
Stuart
bob

Square and cube

```
def square(x):
    Y = x * x
    return Y
def cube(x):
    Z = x*x*x
    return Z
Funct 1 = [square, cube]
```

&s

```
for r in range(5):
    value = list(map(lambda x : x(r), funct 1))
    print(value)
```

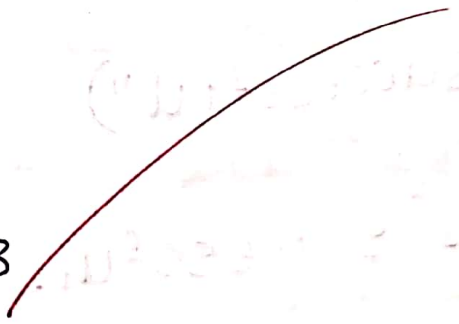
```
>>> [0, 0]
    [1, 1]
    [4, 8]
    [9, 27]
    [16, 64]
```

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

```
#Odd numbers
class odd:
    def __iter__(self):
        self.num = 1
        return self
    def __next__(self):
        num = self.num
        num = self.num + 2
    def __next__(self):
        num = self.num
        self.num += 2
        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
```



PRACTICAL – 3

AIM : To Demonstrate the exception handling.

Q. Write a program using the exception block related to the environment error.

STEP 1 : Use the try block to define the normal force of action

For eg. → Define file object and open the file in write mode and write some content into the file.

STEP 2 : Use the except block with the input output error environment error and convey the app. message to the user, else display the message that the operation is carried out successfully.

```
#try:
    fileobj = open("abc.txt", "w")
    fileobj.write("Python is an indented language")
except IOError:
        print("It is an environmental Error!")
else:
        print("Operation is successful")
```

>>> Output is — Operation is successful.

Q. Write a program for demonstrating the use of value error in the given prog -m statement.

STEP 1 : Accept the value from the use and if it is a valid value display the enter value and terminate the condition by using break statemen

STEP 2 : Define the except block with val error as a keyword and display t appropiate message.

STEP 3 : We can define the multiple exce -tion using the except statement for finding the different category of errors.

```
#try:
    fileobj=open("abc.txt", "w")
    fileobj.write("all Indians are my brothers and
                   sisters")
except ValueError:
    print("sry! invalid number")
except IOError:
    print("This is environmental error!")
else:
    print ("Operation Successful!")
```

```
>>> Enter a no.→T
    Sry! invalid number
>>> Enter a no.→5544
    Operation Successful.
```

PRACTICAL – 4

AIM : To demonstrate the use of regular expression.

STEP 1 : Import re module declare pattern and declare sequence use match method with declare 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.

STEP 3 : Import re module declare pattern with meta character use the split() and print the output.

STEP 4 : 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 corelation 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 and next number are in range of 0 to 9 and check whether the entered numbers are equal to 10. if criteria matches print the 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 and declare the same.

STEP 8 : Import re module declare the host and domain name declare pattern for seperating the host and domain name. Use the findall() and print the output respectively.

STEP 9 : Import re module enter a string use pattern to display only two elements of the particular string. Use the findall() declare two variables with initial value as zero use for condition and subsquently use the if condition check whether condition satisfy add up the or else increment value. And display the values subsquently.

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")
```

>>> The output is → matched pattern found!

numerical values (segregation)

```
import re
pattern = r '\d+'
String = 'hello123, howdy789, 45howru'
output = re.findall (pattern, string)
print (output)
```

>>> ['123', '789', '45']

Split()

```
import re
pattern = r '\d+'
string = 'hello123, howdy789, 45howru'
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 ('\A python', sequence)
print (v)
v1 = v.group ()
print (v1)
```

>>> <_sre.SRE_Match object at 0x0281DF00>
python

```
# verifying the given set of phone.numbers
import re
list 1 = ['8004567891', '9145673210', '7865432981',
          '9876543201']
for value in list 1 :
    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 !")
```

```
>>> Criteria matched for cell numbers
    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']
```

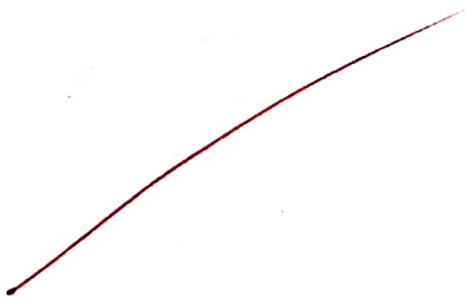
```
# host and domain
  import re
  seq='abc.tcsc@edu.com, xyz@gmail.com'
  pattern = r'[\w\.-]+[\w\.-]'
  output=re.findall(pattern, seq)
  print(output)
```

```
>>> ['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 is :", m)
print ("No. of females is :", f)
```

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



PRACTICAL – 5

AIM : To study about the GUI components.

STEP 1 : Use the tkinter library 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 abore mention events.

#2

STEP 1 : Use the tkinter library for importing the features of 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 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 mainloop() for the trigger-ing of the corresponding events.

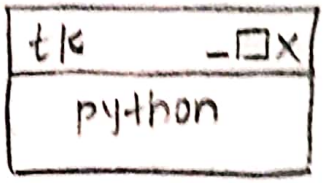
STEP 5 : Now repeat above steps with the label() with takes the following arguments -

i) Name of the parent window
ii) Text attribute which defines the string.
iii) The background color (bg)
iv) The foreground fg and then a attribute i.e - use the pack() with a relevant padding attributes.

Creation of parent window:

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

OUTPUT :

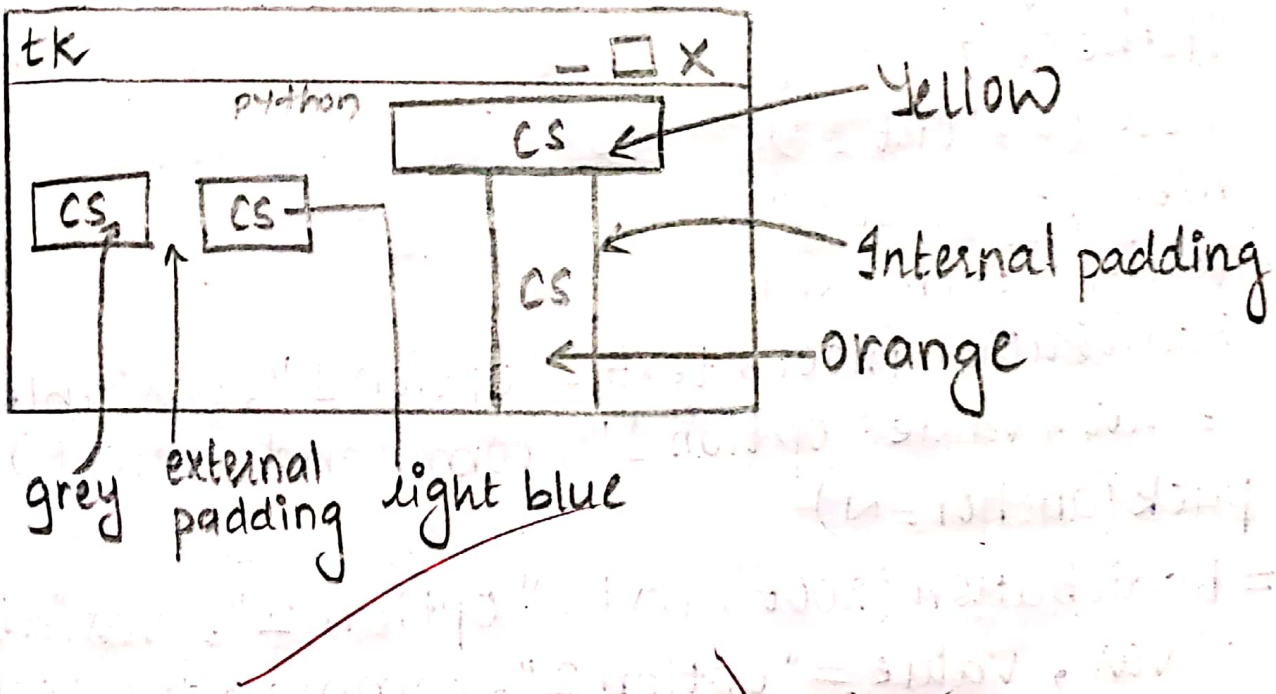


2

```
from tkinter import *
root TK()
l = label(root, text = "Python")
l.pack()
l1 = Label(root, text = "CS", bg = "grey", fg = "black",
        font = "10")
l1.pack(side = LEFT, padx = 20)
l2 = Label(root, text = "CS", bg = "light blue",
            fg = "black", font = "20")
l2.pack(side = LEFT, pady = "30")
l3 = Label(root, text = "CS", bg = "yellow", fg = "black",
            font = "10")
```

```
l3. pack(side = TOP, ipadx = 40)
l4 = Label(root, text = "CS", bg = "orange",
           fg = "black", font = "10")
l4. pack(side = TOP, ipady = 50)
root. mainloop()
```

OUTPUT :



PRACTICAL-5(B)

AIM : To study about the GUI components.

#1

STEP 1 : Import the relevant methods from the tkinter 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 about the given selection mode from multiple option available.

STEP 4 : Now define the parentwindow and define the option with control variable.

STEP 5 : Use the listbox() and insert options on the parent window along with the pack() with specifying anchor attribute.

STEP 6 : Create an object from radio button which will take following arguments and parentwindow object, text variable which will take the values options no. 1, 2, 3, ... variable argument, corresponding value and triger the function declared.

STEP 7 : Now call the pack() for radio obj so created and specify the argume using anchor attribute.

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

~~#~~ 2

STEP 1 : Import relevant methods from the tkinter library.

STEP 2 : Create a parent object corresponding to the parentwindow.

STEP 3 : Use the geometry() for laying of th window.

STEP 4 : Create an object and use the scrollb ().

STEP 5 : Use the pack() along with the scrol - bar() object with side and fill attribute

STEP 6 : Use the mainloop with the parent object.

#3

STEP 1 : Import the relevant librabies from the tkinter method.

STEP 2 : Create an corresponding object of the parent window

STEP 3 : Use the geometry manager with pixel size (680 x 500) or any other suitable pixel value.

STEP 4 : Use the label widget along with the parent object created and simultaneously use the pack method.

STEP 5 : Use the frame widget along with the parent object created and use the pack method.

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

STEP 7 : Use the scrollbar() with an object use the attribute of vertical then configure the same with object created from the scroll-bar() and use pack().

STEP 8 : Trigger the events using mainloop.

#4

STEP 1 : Import relevant methods from tkinter library.

STEP 2 : Define the object corresponding to parent window and define the size of parent window in terms of no. of pixels.

STEP 3 : Now define the frame object from the method and place it on to the parent window.

STEP 4 : Create another frame object termed as the left frame and put it on the parent window on its LEFT side.

STEP 5 : Similarly define the RIGHT frame and subsq. define the button object placed onto the given frame with the attribut as text, active background and foregroun

STEP 6 : Now use the pack() along with the side attribute.

STEP 7 : Similarly create the button object corresponding to the MODIFY operation put in the frame object on side = "right".

STEP 8 : Create another button object and place it on to the RIGHT frame and label the button as ADD.

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

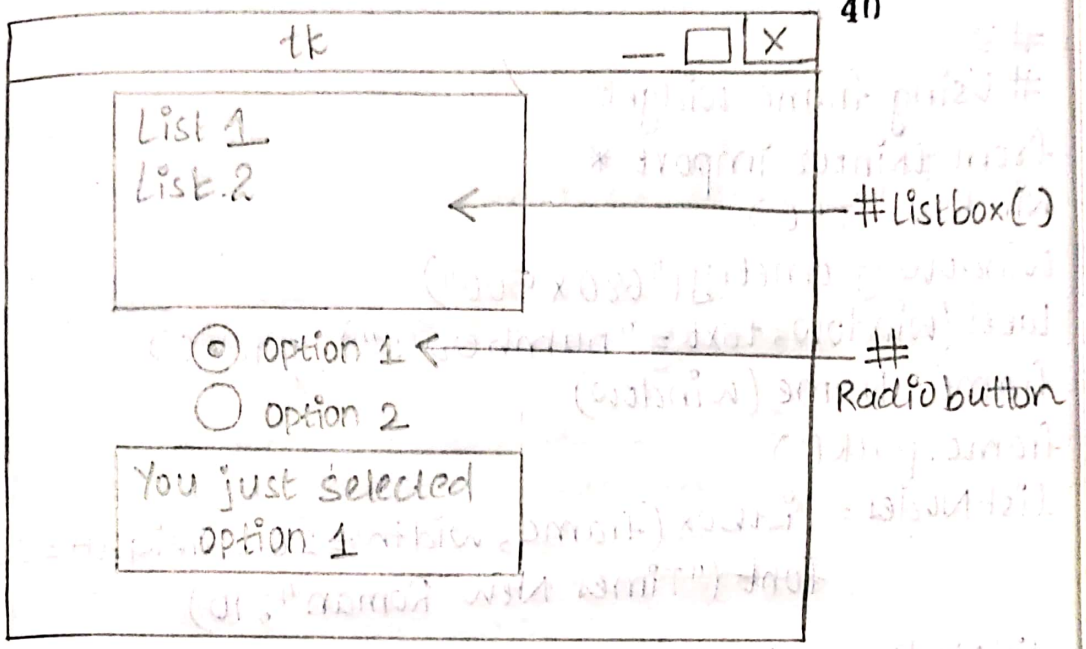
STEP 10 : Use the pack() simultaneously for all the objects and finally use the mainloop.

1

Radio button

```
from tkinter import *
root = Tk()
root.geometry("500x500")
def select():
    selection="you just selected" + str(var.get())
    t1=Label(root, 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()
```

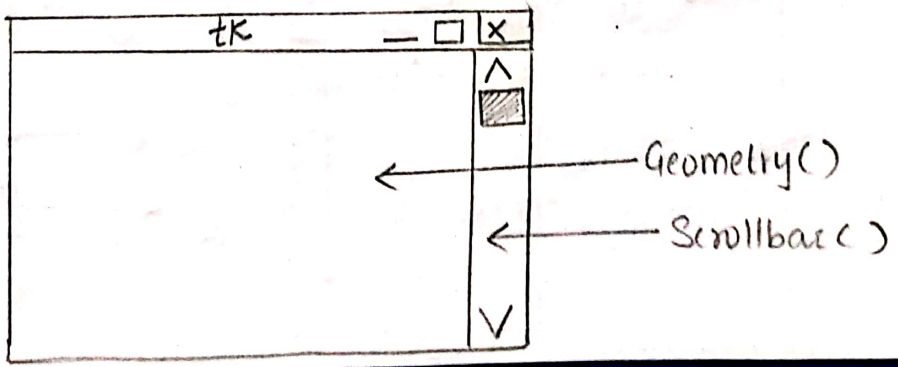
OUTPUT :



#2:

```
Scrollbar()
from tkinter import *
root = TK()
S = Scrollbar(root)
S.pack(side="right", fill="y")
root.mainloop()
```

OUTPUT :

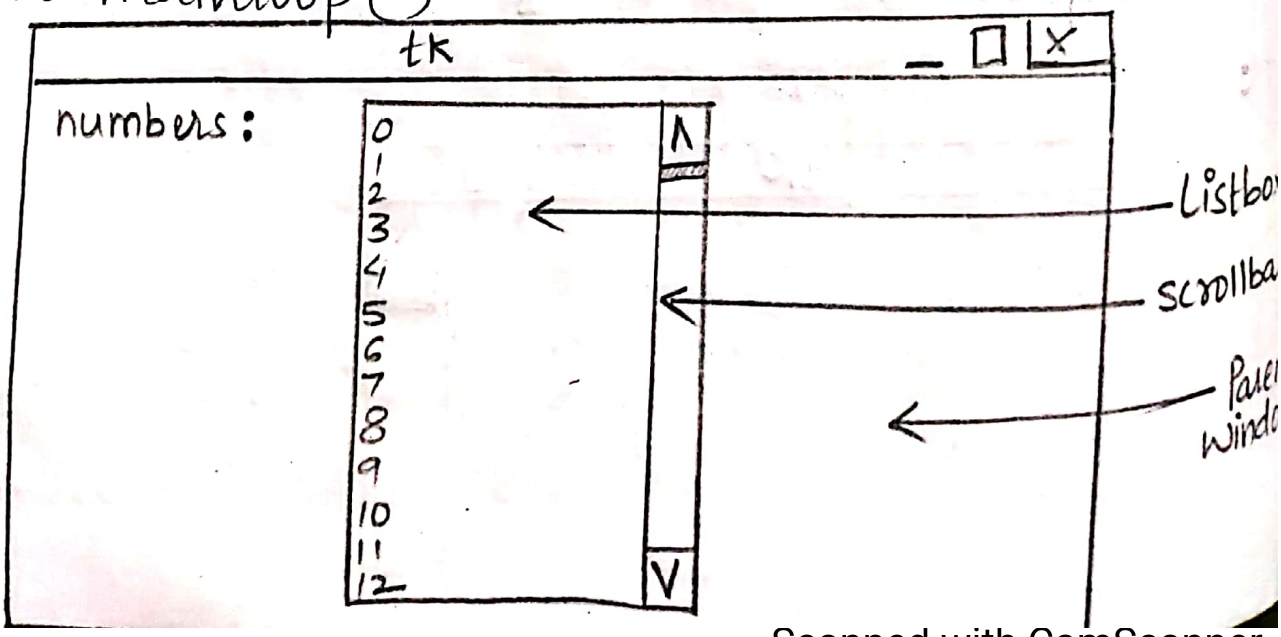


```
#3
#Using frame widget
from tkinter import *
window = TK()
window.geometry("680x500")
Label(window, text="numbers:").pack()
frame = Frame(window)
frame.pack()
listNodes = Listbox(frame, width=20, height=2
            font("Times New Roman", 10)

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()
```

→ end space

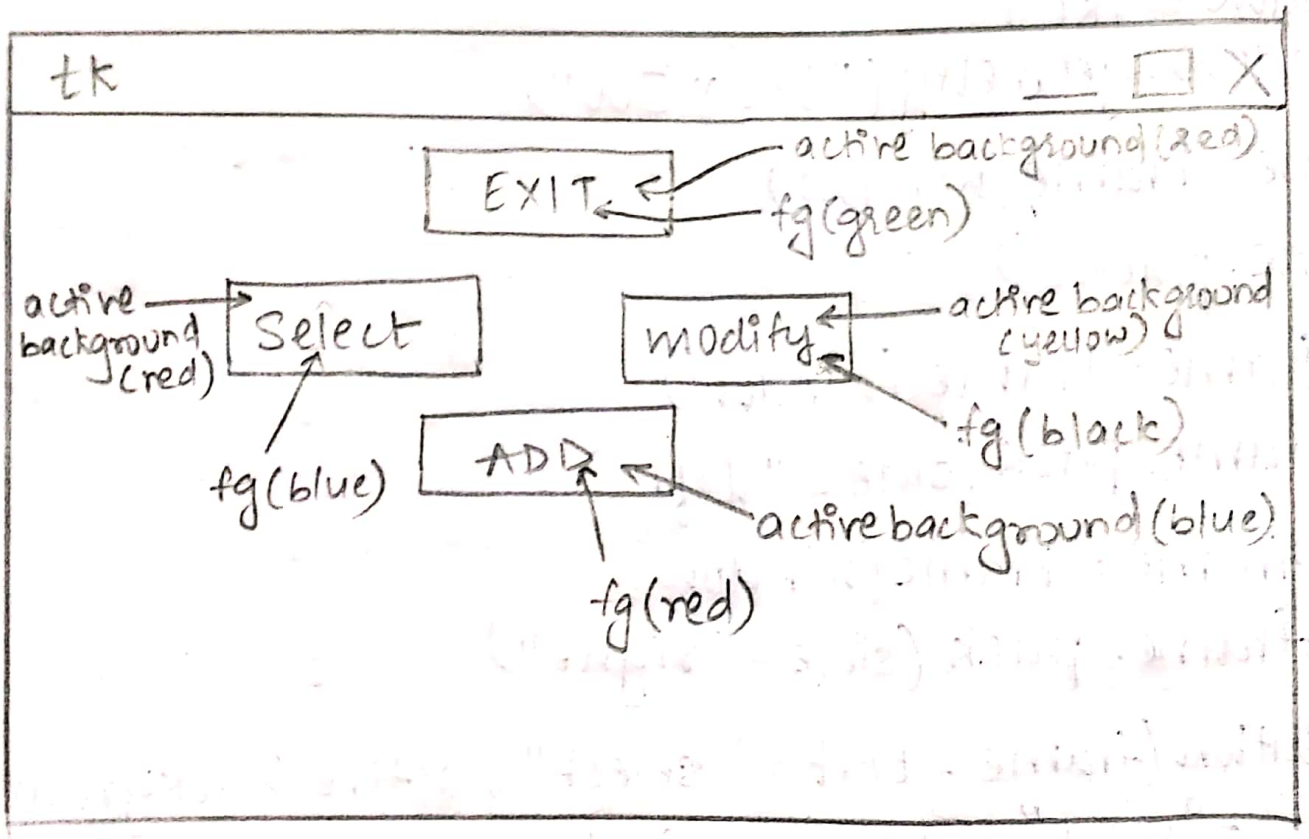
OUTPUT:



#4

```
from tkinter import *
window = Tk()
window.geometry("680x500")
frame= Frame(window)
frame.pack()
leftframe = Frame(window)
leftframe.pack(side="left")
rightframe = Frame(window)
rightframe.pack(side="right")
b1 = Button(frame, text="select", activebackground
    ="red", fg="blue")
b2 = Button(frame, text="modify", activebackground
    ="yellow", fg="black")
b3 = Button(frame, text="ADD", activebackground
    ="blue", fg="red")
b4 = Button(frame, text="EXIT", activebackground
    ="red", fg="green")
b1.pack(side="LEFT", padx=20)
b2.pack(side="right", padx=30)
b3.pack(side="bottom", pady=20)
b4.pack(side="top")
window.mainloop()
```

OUTPUT:



PRACTICAL – 5(c)

AIM : To study about the GUI components

#1

STEP 1 : Import the relevant methods from tkinter library.

STEP 2 : Import tkMessagebox

STEP 3 : Define a parent window object alon with the parent window.

STEP 4 : Define a function which will use tkmessagebox with info window attribut

STEP 5 : Declare a button which parent wind object along with the command attrib

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

#2

STEP 1: Import the relevant methods from the Tkinter library along with parent window object declared.

STEP 2: Use parentwindow object along with minimize function for window size.

STEP 3: Define a function main, declare parent window object and use config(), title(), minisize(), Label() as well as button() and use pack() and mainloop() simultaneously.

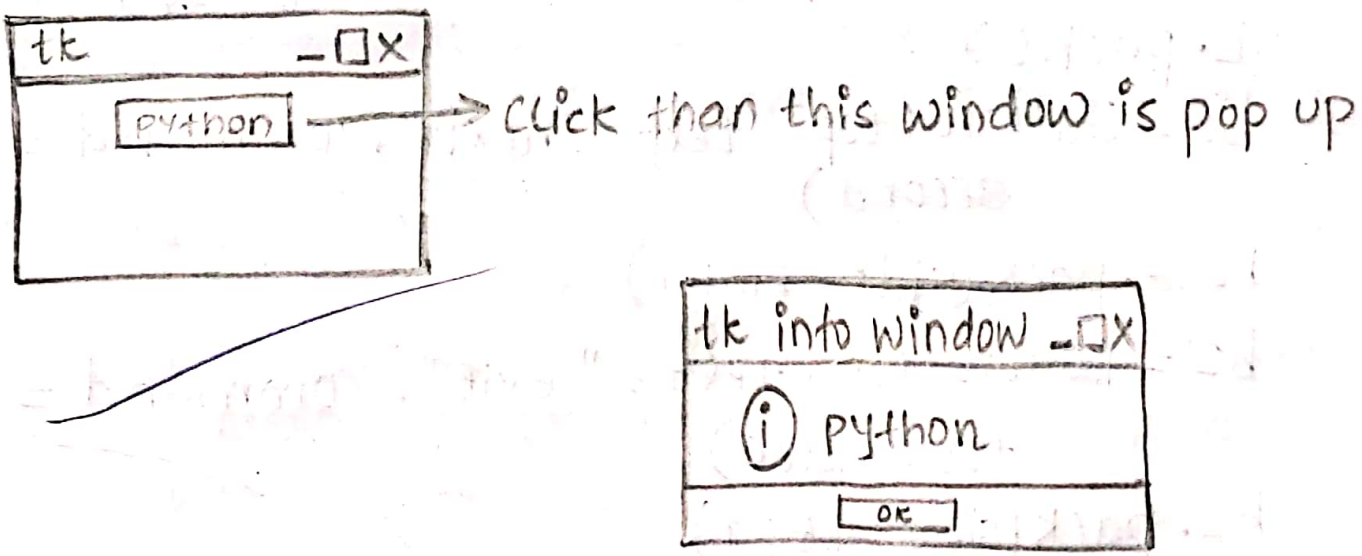
STEP 4: Similarly, define the function second and use the attribute accordingly.

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.

```
#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 :



```
#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()
```

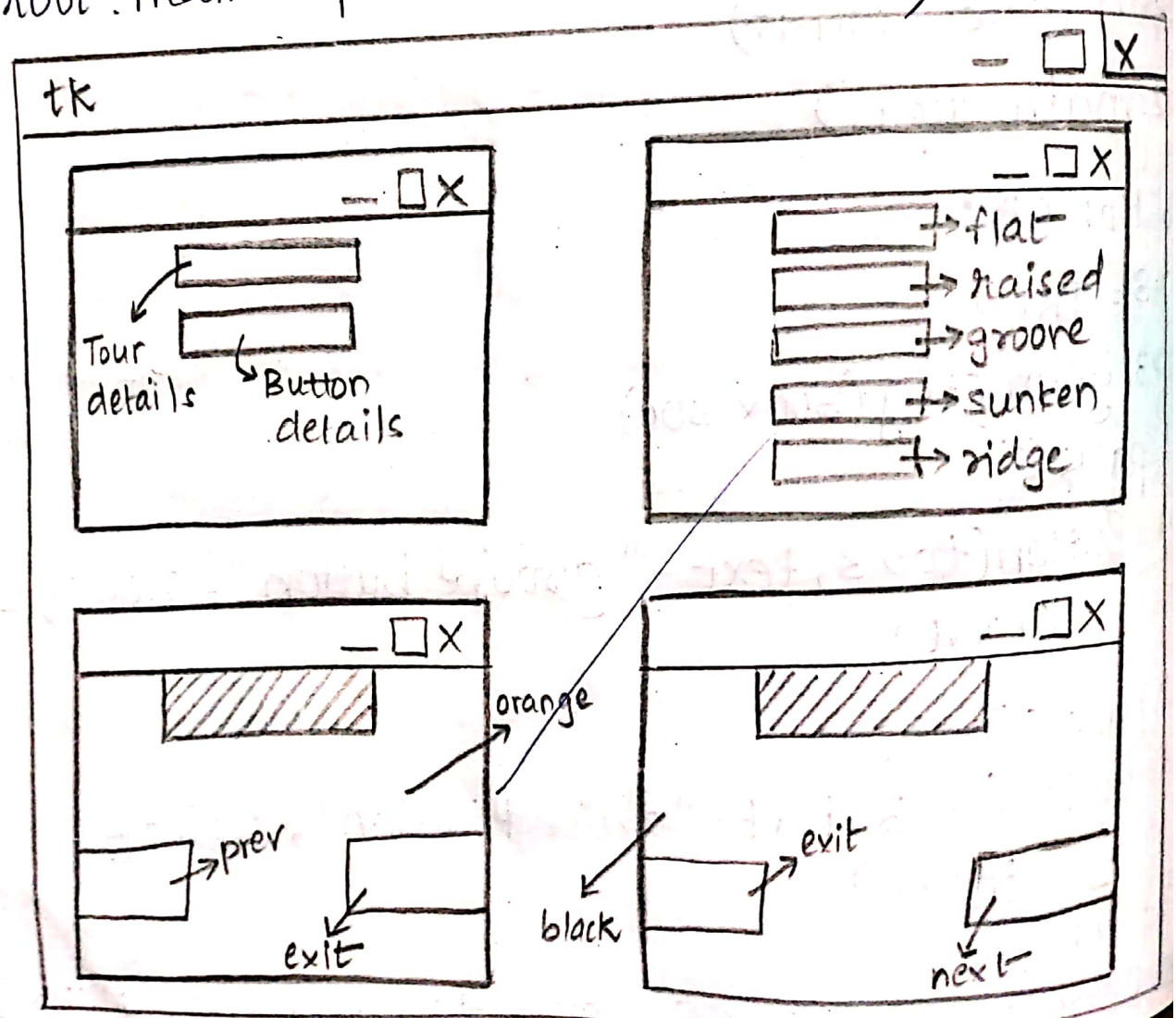
```
def second():
    top2 = TK()
    top2.config(bg = "orange")
    top2.title("About us!")
    top2.minsize(300, 300)
    L=Label(top2, text="created by: Khushi Singh \n
      for more details contact to our official account")
    l.pack()
    b3=Button(top2, text = "prev", command = main)
    b3.pack(side = LEFT)
    b2 = Button(top2, text = "exit", command =
            terminate)
    b2.pack(side = RIGHT)
    top2.mainloop()

def button():
    top3 = TK()
    top3.geometry(300x300)
    b1.pack()
    b2 = Button(top3, text = "groove button", relief =
            GROOVE)

    b2.pack()
    b3 = Button(top3, text = "raised button", relief =
            RAISED)
    b3.pack()
    b4=Button(top3, text = "Sunken button", relief=SUNKEN)
```

```
b4.pack()
b5=Button(top3,text="ridge button",relief=R
top3.mainloop()

def terminate():
    quit()
    b5=Button(root,text="TOUR DETAILS",command=
    b5.pack()
    b6=Button(root,text="BUTTON DETAILS",comman
                    button
    b6.pack()
root.mainloop()
```



PRACTICAL – 6

AIM : To demonstrate the use of Spinbox Paned Window and Canvas Widget.

#1

STEP 1 : Create an object from the tk meth and subsequently create an object fro the spinbox method.

STEP 2 : Make the object so created onto th parent window and trigger the corres -ding events.

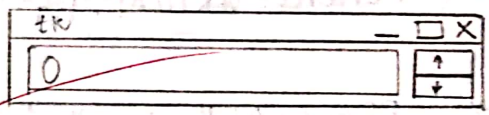
STEP 3 : Use the pack method to provide th direction using anchor method.

STEP 4 : Use the mainloop method to termi

SOURCE CODE :


```
from tkinter import *
root = Tk()
S1 = Spinbox(root, from_ = 0, to_ = 10)
S1.pack(anchor = S)  capital S.
root.mainloop()
```

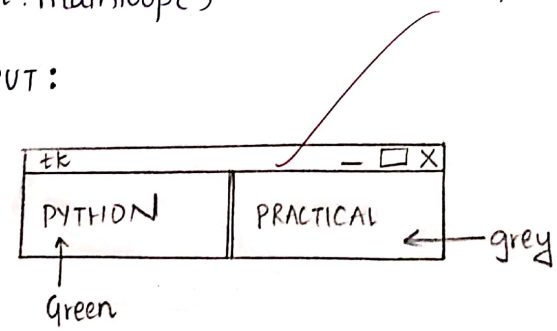
OUTPUT :



SOURCE CODE :

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

OUTPUT :



#2

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

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

STEP 3 : Similarly create a second paned window object and add it onto the 1st paned window with orientation specified.

STEP 4 : Now create another label object and place it onto the 2nd paned window object and add it onto the 2nd pane.

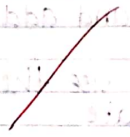
STEP 5 : Now use the mainloop method to terminate.

#3

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

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

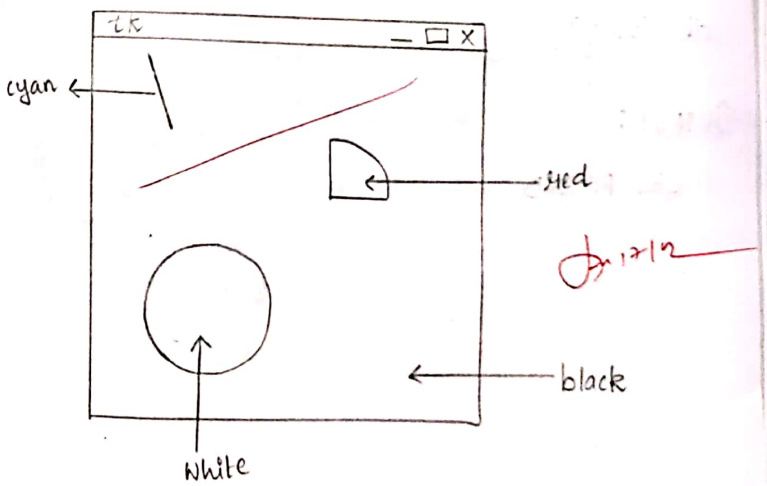
STEP 3 : Now call the pack method and main-loop method.



SOURCE CODE :

```
from tkinter import *
root = Tk()
C1 = Canvas(root, height="400", width="400", bg="black")
oval = C1.create_oval(20,80,150,250, fill="violet")
line = C1.create_line(30,410,50,60, fill="cyan")
arc = C1.create_arc(20,140,150,60, fill="red")
C1.pack()
root.mainloop()
```

OUTPUT :



SOURCE CODE :

```
import dbm
db= dbm.open("database", 'c')
db["https://www.google.com"]=("employee.
                                db")
if db["https://www.google.com"]!= None:
    print("URL FOUND")
else:
    print("URL NOT FOUND")
db.close()
```

OUTPUT :

URL FOUND

PRACTICAL – 7

AIM : To demonstrate the use of Database Connectivity.

#1

STEP 1 : Import the dbm library and use the open method for creating the database by specifying the name of the database along with the corresponding flag.

STEP 2 : Use the object so created for accessing the given website and the corresponding regular names for the website.

STEP 3 : Check whether the given URL address with the regular name of the page is not equal to null than display the message that the particular URL address is found else not found.

STEP 4 : Finally use the close method to terminate the database library.

#2

STEP 1 : Input the corresponding libraries for making the database connection which are os and sqlite3.

STEP 2 : Now create the connection object using sqlite3 library and the connect method for creating the new database.

STEP 3 : Now create the cursor object using the cursor method from the connection object created in the above step.

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

STEP 5 : Now with the cursor object use the insert statement for entering the values corresponding to different fields considering the datatype.

STEP 6 : Use the commit method to complete the transaction using the connection object.

SOURCE CODE :

```
import os, sqlite3
connection = sqlite3.connect("employee")
cursor1 = connection.cursor()
cursor1.execute('create table dcs
                (emp_name char[10], emp_id
                 int[10])')
<sqlite3.Cursor object at 0x02EED0A0>
cursor1.execute('insert into dcs values
                ("Khushi", 1234)')
<sqlite3.Cursor object at 0x02EED0AD>
connection.commit()
cursor1.execute('select emp_name from dcs')
<sqlite3.Cursor Object at 0x02EED0AD>
cursor1.fetchall()
```

OUTPUT :

[('Khushi')]

#

```
import os, sqlite3
connection = sqlite3.connect("employee.db")
C1 = connection.cursor()
C1.execute('create table employee(Name, Id_no,
                                  DOB)
<sqlite3.Cursor object at 0x02FC2BE0>
C1.execute('insert into employee values("sheetal",
                    1111, 13-5-2001)')
<sqlite3.Cursor object at 0x02FC2BE0>
C1.execute('insert into employee values("Khus
                    1112, 12-2-2001)')
<sqlite3.Cursor object at 0x02FC2BE0>
C1.execute('insert into employee values("Salon
                    1113, 24-11-2001)')
<sqlite3.Cursor object at 0x02FC2BE0>
```


STEP 7 : Use the execute statement along with the cursor object for accessing the values from the database using the select / from / where clause.

STEP 8 : Finally use the fetchall() for displaying the values from the table using the cursor object.

```
connection.commit()
c1.execute('select * from employee')
<sqlite3.Cursor object at 0x02FC2BE0>
c1.fetchall()
c1.execute('Drop table employee')
<sqlite3.Cursor object at 0x02FC2BE0>
```

OUTPUT :

```
[('Khushi', 1112, 12-2-2001),
('Sheetal', 1111, 13-5-2001),
('Saloni', 1113, 24-11-2001)]
```

→ fetchall output

```
>>>
```

↳ After Droping the table, the table is cleared out so only blank space will appear.