SEMESTER- II

Practical - 1.

OBJECTIVE : Demonstrate the use of different file accessing modes 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 content of variable.

Step 3: Now use the fileobject for finding the name of the file, the file mode in which its opened whether the file is still open or close and finally the output of the soft space attribute.

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

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

Step 6: Now open file obj in append mode open write method write contents close the file obj again open the file obj in read mode and display the appending output.

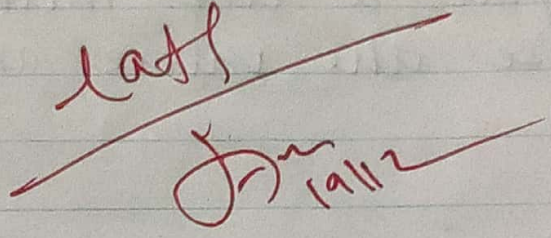
Step 7: Open the file obj in readmode declare a variable and perform file obj dot tell method and store the output consequently in variable

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

```
file obj = open("abc.txt", "w")
File obj .write(" Computer Science Subj "+"\n")
File obj ..write(" DBMS in PYTHON in D \n")
Dile obj . close ( )

File obj = open ("abc.txt", "r")
str = fileobj . read ( )
Print (" The output of read method :", str )
 file obj ..close ( )
```

step 9: Open file obj with read mode also use the readlines method and store the output consequently in eg print the same for counting the length use the conditional statment and display the length.



ESD

PRACTICAL - 2.

AIM : Demonstrate the use of iterators & iterable objects.

Program 1 :

Algorithm

Define a variable of list datatype containing

step 1 : Define a iter method with an arg & initialise the value and return that value.

step 2 : Define the next method with an arg & compare the upper limit by using a conditional statment.

step 3 : Now create an object of the given class & pass this object in the iter method.

```
class Myrange:
    def __iter__(self):
        self.a = 1
        return self
    def __next__(self):
        if self.a <= 10:
            x = self.a
            self.a += 1
            return x
        else:
            raise stop STOP ITERATION

Myclass = myrange()
Mydata = iter(myclass)
for x in mydata:
    Print x..
```

Output:

1
2
3
4
5
6
7
8
9
10

```
class odd:
    def __iter__(m):
        m.value = 1
        return m
    def __next__(m):
        if m.value < 15:
            m.value += 2
            return m.value
        else:
            raise stopIIERATION
y = iter(odd())
while true:
    Print(next(y))
```

output:

1
3
5
7
9
11
13
15

2. WAP using iterators to display odd no. till 15

Algorithm :

step 1 : Create a class with in that define a iter method with argument and initialize the value and return that value.

step 2 : Define the next method with an argument.

step 3 : Use if conditional statement to check wether the variable in 1st step is smaller than 16.

step 4 : Increase the variable value by 2 and return it use the stop Iteration

step 5 : Create an object of given data pass the object in iter method

step 6 : Use while conditional loop to pass the next method.

3. Factorial of number till 10

Algorithm:

step 1: Create a class within class define a iter method using an argument, initialize a value and return it.

step 2: Define a next method with an argument.

step 3: Create a variable facto with value 1 and use for conditional loop to calculate factorial till value of variable declared in step 1.

step 4: Print the Facto's value and increase value of variable in step 1 by 1.

Step 5: Use if conditional statment to raise STOP ITERATION. if range of variable in step 1 is bigger than 10.

Step 6: Create a object this class and pass it to iter method and use while loop to use next method.

```
class Fact:
    def __iter__(m):
        m.value = 1
        return m
    def __next__(m):
        Facto = 1
        For i in range(1, m.value + 1):
            Facto = Facto * (i)
        Print(" The Factorial of", m.value," is",Fa
        m.value += 1
        if (m.value == 7):
            raise STOP ITERATION.
    y = iter(Fact())
        while True:
            next(y)
```

Output

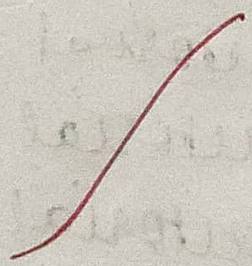
The Factorial of 1 is 1
The Factorial of 2 is 2
The Factorial of 3 is 6
The Factorial of 4 is 24
The Factorial of 5 is 120
The Factorial of 6 is 720.

```
class power:
    def __iter__(Pow):
        Pow.num = 1
        Pow.num = int(input("enter number:"
        return Pow
    def __next__(Pow):
        if Pow.num<=6
        nu = Pow.nums ** Pow.num
        Pow.num += 1
        return nu
    else:
        raise STOP iteration
y = iter(Power())
while true:
    Print(next(y))
```

Output

Enter Number : 2

2
4
8
16
32
64



4. Power of number taken from user.

Algorithm.

step 1: Create a class and define a iter method with an argument, initialize a value & return it also take a input from user.

step 2: Define a next method with an argument

step 3: If the initialized value or input variable is bigger than 5 then raise STOP ITERATION or else return the multiplication of argument by itself

step 4: Create a object of given class and pass it to iter method and use while loop to use next method.

Jan 12/12/19

5. Algorithm :

Step 1 : Define a Variable of list data type containing names of fruit.

Step 2 : Now iterate over that variable and print each element using next method or some conditional loops statment

```
i = ["Apple", "Banana"]
myiter = iter(i)
For K in i :
    Print(K)
```

output

Apple
Banana.

850

IO error :

```
try :
    fo = open ("abc.txt", "w")
    fo = write ("python is an indentent langu
except IO error :
    Print (" Enter appropriate mode for file operatio
else :
    Print (" operation is successful")
```

>>> operation is successful

R Output :

>>> Enter appropriate mode for file operation

Value Error :

```
try :
    x = try (input (" Enter a statment : "))
except value error :
    Print (" Airthmatic error")
else :
    Print (" operation is successful")
```

>>> Enter a statment : abc

Airthmatic error

>>> Enter a statment : 123

operation is successful.

PRACTICAL NO : 03

AIM : Exceptions

Algorithm

Step 1 : In the try block open file in the open mode with write mode, write some contents in the file

Step 2 : In except (I/O error) block use the error in I/O error use the appropriate message to display

Step 3 : Else display the operation is successful

Step 4 : In try block, accept input from user

Step 5 : In except block use `Value Error' and print the same message.

Step 6 : Else display the operation is successful

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

Step 8: For the exception to be raised use the except keyword (and) ie. type error print, in compatible values.

Step 9: Use except with error zero division error Eg print the message accordingly that is entered number is zero or not able to perform operation.

Step 10: Declare static variables Eg values.

Step 11: For multiple exception use the error types by seperating them with comma.

type error, zero division error:

```
a = 1
b = input ("enter:")
try:
    Print (a/b)
except type error:
    Print ("Incompatible value")
except zero division error:
    Print ("Denominator is zero so operation could not be performed")
```

output

```
>>> Enter 2.
    0.5
>>> 0
```

Denominator is zero so operation could not be performed

Multiline exception

```
a,b = 1,0
try:
    Print (1,0)
    Print ('10' + 10)
except (type error, zero division error):
    Print ("Invalid input")
```

output

```
>>> 1.0
>>> 1010
>>>>> Invalid input
```

using except value:

```
try:
    a = open("abc.txt", "w")
    a.write("python")
except IO error:
    print("error")
else:
    print("successful")
def x():
    l = []
    print(len())

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

print(x())
print(y())
```

- output: Successful
  0
  None
  4
  None

raise keyword

```
try:
    a = int(input("enter a number:"))
    raise value error
except value error:
    print("enter integer value")
```

output

>>> enter: xy2.

enter integer value.

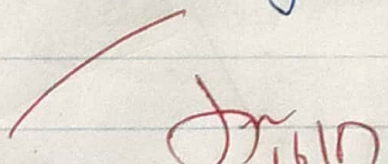
Step 12: Use try block, open a file in write mode & subsequently enter value in file.

Step 13: Use the IO error and display appropriate message

Step 14: Define a function with empty list & calculate the length of same and display the same.

Step 15: Define another function y, initialize elements in the list.

Step 16: In the try block accept input from the user and if the user enters character value, raise an error that is saying up enter integer values.

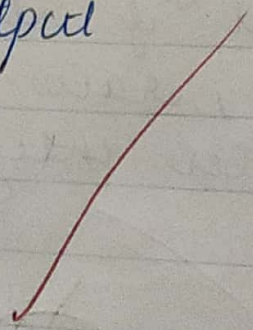


PRACTICAL - 04
Regular Expression

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

Step 2: Import re module declare pattern with literal & meta character. Declare string values. Use the findall () with argument & 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 (segreation)

```
import re
pattern = r'\d+'
string = " hello 123, howdy 789, 45 hour 4"
output = re.findall (pattern, string)
Print (output)
```

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

split ()

```
import re
pattern = r'/d+'
string = ' hello123, howdy 789, 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('\APython', sequence)
Print(V)
V1 = V.group()
print(V1)
>>> <- sre.SRE_Match object at 0x0281DF00>
    PYTHON
```

```
# Verifying the given set of phone numbers
import re
list = ['8004547891', '9184654321', '7865432819',
        '9876543210']
for value in list1:
  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 and accordingly declare pattern replace the blank space with no space. use sub() with 3 arguments and print the string without spaces.

Step 5 : import re module declare a sequence use search method for finding subsequently use the group() with dot operator as search() gives memory location using group().

Step 6 : Import re module declare list with numbers. Use the conditional statment here we have used up the for Condition statment. 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 entered numbers are equal to 10.

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 and the domain name. Declare pattern for seperating the host and 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 variables with initial value as zero.

>>> Criteria matched for cell number
Criteria failed !

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

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

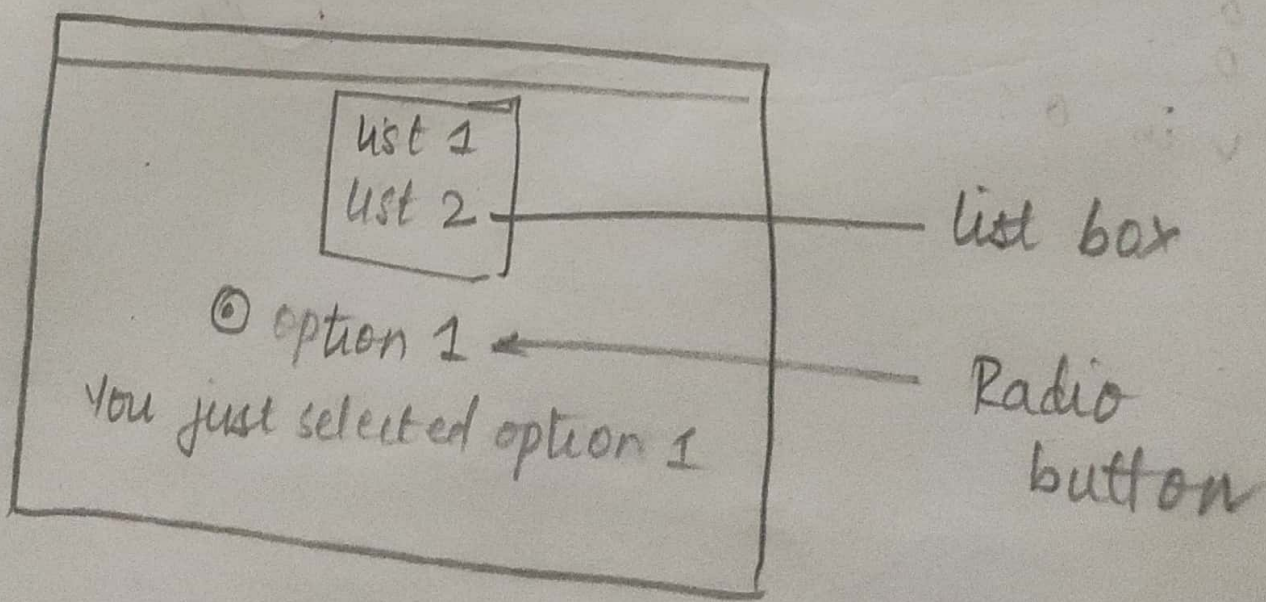
Counting of first 2 letters.

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:

Radio button

```
from tkinter import *
root = tk()
root= geometry("500x500")
def select():
    selection = "You just selected" + str(var.get())
    l = label(text = selection, bg = "green")
    l.pack(side = Top)
var = string var()
l = listbox()
l1. insert(1, "list 1")
l2. insert(2, "list 2")
l.pack(anchor = N)
r1 = Radiobutton(root, text = "option 1", variable =
                 var, value = "option1", command = sel
r1. pack(anchor = N)
root. mainloop()
```

Output



PRACTICAL - 05

GUI Components.

A B].

Step 1: Use the tkinter library for importing the features of text object.

step 2: Create an object using TK()

step 3: Create a variable using the widget Label, Now define a function which tells the user about the given selection mode.

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

step 5: Use the ~~mainloop~~ ~~listbox~~ listbox() and insert options on the parent window along with the pack() with specifying anchor attribute

Step 6: Create an object from radio button which will take following arguments (parent window object, text variable) value option no 1, 2, 3 ---- variable argument, Corresponding value & trigger the function declared.

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

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

#2.

Step 1: Import relevant methods from the tkinter library.

Step 2: Create a parent object corresponding to the parent window.

step 3: Use geometry() for layering of windows

Step 4: Create an object and use scrollbar()

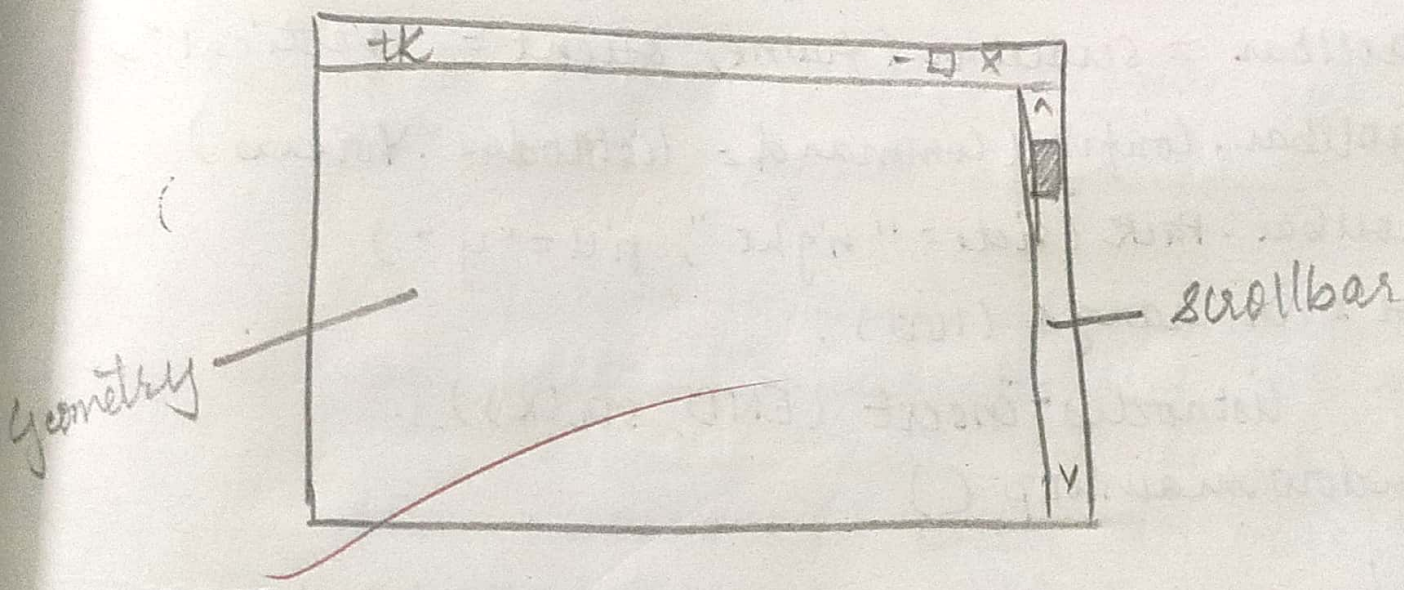
Step 5: Use pack() method along with scrollbar object with side & fill attribute.

Step 6: Use mainloop() with parent object

Scrollbar()

```
from tkinter import *
root = tk()
root.geometry("500x500")
S. scrollbar()
S.pack(side = RIGHT, fill=y)
root.mainloop
```

output

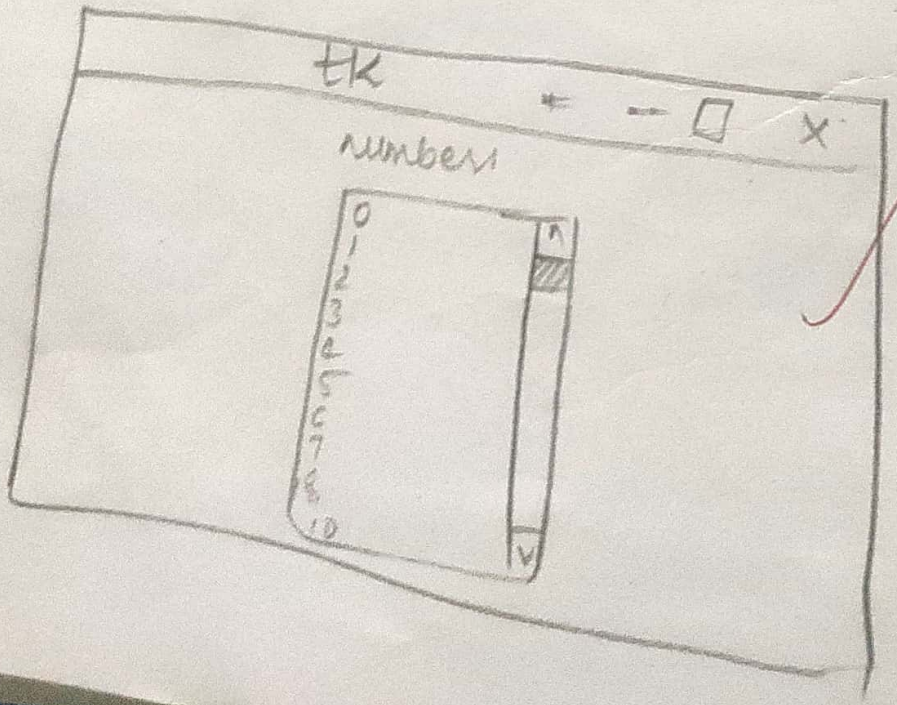


#3

using frame widget

```
from tkinter import *
window = Tk()
window.geometry("480x500")
label(window, text="numbers:").pack()
frame = Frame(window)
frame.pack()
listNodes = listbox(frame, width=20, height=20,
                    font=("Times New Roman", 10))
listnodes.pack(side="left", fill='y')
scrollbar = scrollbar(frame, orient="vertical")
scrollbar.config(command=listnodes.yview)
scrollbar.pack(side="right", fill="y")
for x in range(100):
    listnodes.insert(END, str(x))
window.mainloop()
```

Output :



3

Step 1 : Import the relevant library from tkinter

step 2: Create an corresponding object from parent window.

step 3: Use geometry manager with pixel size.

step 4: Use label widget along with parent object created & subsequently use pack()

step 5: Use frame widget along with parent object.

Step 6: Use listbox method along with attribute like width, height font.

Step 7: Use scrollbar() with an object. Use the attribute of vertical, then Configure the same with object oriented created from scrollbar() & use pack()

Step 8: Trigger the events using mainloop().

(B)

AIM - GUI Components

A)

Step 1: Use the tkinter library from import the feature of text widget.

Step 2: Create an object using tk()

Step 3: Create a variable using widget label & text method.

Step 4: Use mainloop() for triggering of Conditional above mention event.

B)

step 1: Use tkinter library for importing features of text widget.

step 2 - Create a variable and position it on the parent window from text method

step 3 - Use pack() method along with the object oriented created from text()

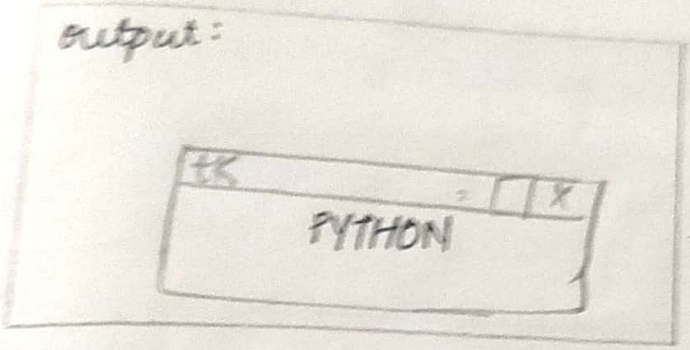
and use the parameter

Step 4 - Use mainloop() for triggering of the corresponding events.

creation of parent window

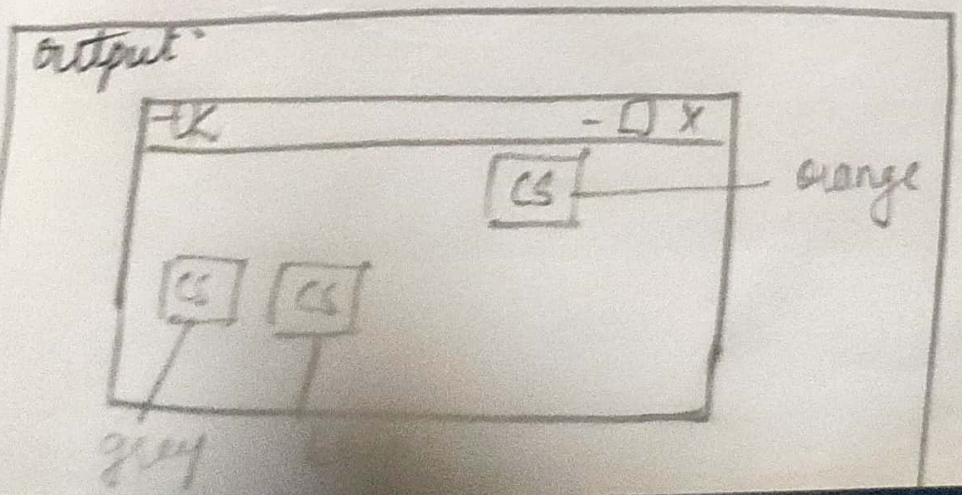
```
from tkinter import *
root = Tk()
l = Label(root, text="python")
l.pack()
root.mainloop()
```

output:




```
from tkinter import *
root = Tk()
l = Label(root, text="python")
l.pack()
l1 = Label(root, text="CS", bg="grey", fg="blaue", font=10)
l1.pack()(side=LEFT, pady=20)
l2 = Label(root, text="CS", bg="blue", fg="pink", font=20)
l2.Pack(side=LEFT, Pady=30)
root.mainloop.
```

output:



Step 5 - Now repeat above step with the label() which takes the following argument

1) Name of parent window
2) Text attribute which defines the string
3) The background colour (bg)
4) The foreground (fg).

```
# Multiple window
# Difference between (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, next="SAN FRANSISCO \n places of int
    \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=terminat
    b2.pack(side=LEFT)
    top.mainloop()

def second():
    top 2 = Tk()
    top 2 Config(bg="orange")
    top 2.title("About us!")
    top 2.minsize(300,300)
```

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

step 2: Use parentwindow 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 as button() & pack() & mainloop().

step 4: Declare another function button along with parent object and declare button with attributes like FLAT, RIDGE, GROOVE, RAISED, SUNKEN along with relief widget.

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

```
l = Label (top2, text = " Created by:
    Unnati Gangar \n for more details
    Contact to our official site ")
l.pack ()
bs = Button (top2, text = "prev", command = main)
b3.pack = Button (top2, text = " exist ")
          command = terminate)
b2. pack (side = RIGHT)
top 2. main loop ()

def button ():
    top 3 = TK ()
    top 3 geometry ("250 x 250")
    b1 = Button top 3, text = "flat button", relief = FLAT )
    b1. pack ()
    b2 = Button (top 3, text = raised button", relief = RAISED)
    b2. pack ()
    b 3 = Button top 3, text = groove button", relief = GROOVE)
    b 3. pack ()
    b4 = Button (top 3, text = " Sunken button", relief = SUNKEN
    b4. pack ()
b5 = Button (top 3, text = " ridge button", relief = RIDGE)
b5. pack ()
top 3. mainloop ()
def terminate ():
    b 5. quit ()
    b5 = Button (root, text = " Names", command = main)
    b5. pack ()
    b6 = Button (root, text = " Button details", command = Bu
    b6. pack ()
root. mainloop ()
```

```
from tkinter import *
root = Tk()
root.title("Python")
root.maxsize(1000, 900)
root.config(bg="black")
leftframe = Frame(root, bg="pink", height="500",
                  width="200")
leftframe.grid(row=0, column=0)
label(leftframe, text="photo", height=2, width
    grid(row=0, column=0)
rightframe = Frame(root, bg="light green", height=
    400, width="250")
rightframe.grid(row=0, column=0)
image1 = PhotoImage(file="dance.gif")
image1.subsample(1, 2)
image2 = PhotoImage(file="dance.gif")
image2.subsample(3, 4)
label(leftframe, image=image1).grid(row=0,
    column=0, padx=10, pady=10)
toolbar = Frame(leftframe, width=200, height=4
    bg="white").grid(row=2, column=0)
label(toolbar, text="Personal Info", height=2,
    width=20, relief=RAISED).grid(row=0, colum
def name();
    Print("Name:")
def hob();
    Print("Hobby:")
```

(D)

AIM: Displaying the image 870 cc

Algorithm

Step 1: Create an object corresponding to the parent window and use the following 3 methods - title, maxsize, config

Step 2: Create a leftframe object from the frame method and place it onto the parent window with the height, width and the bg specified. Subsequently use the grid method with the row, column padx, pady specified. And

Step 3: Now create a ~~lef~~ rightframe object from the frame method.

Step 4: Create a label object from the label method and place it onto the leftframe with text attribute denoting the orginal image with relief attribute used as RAISED value and subsequently use grid method with row, column value specified as (0,0) with some external padding values

SS.

Step 5: Now use the photoimage method to the file attribute specified.

Step 6: Use the sub sample method with object of the image and give the x, y co-ordinate values

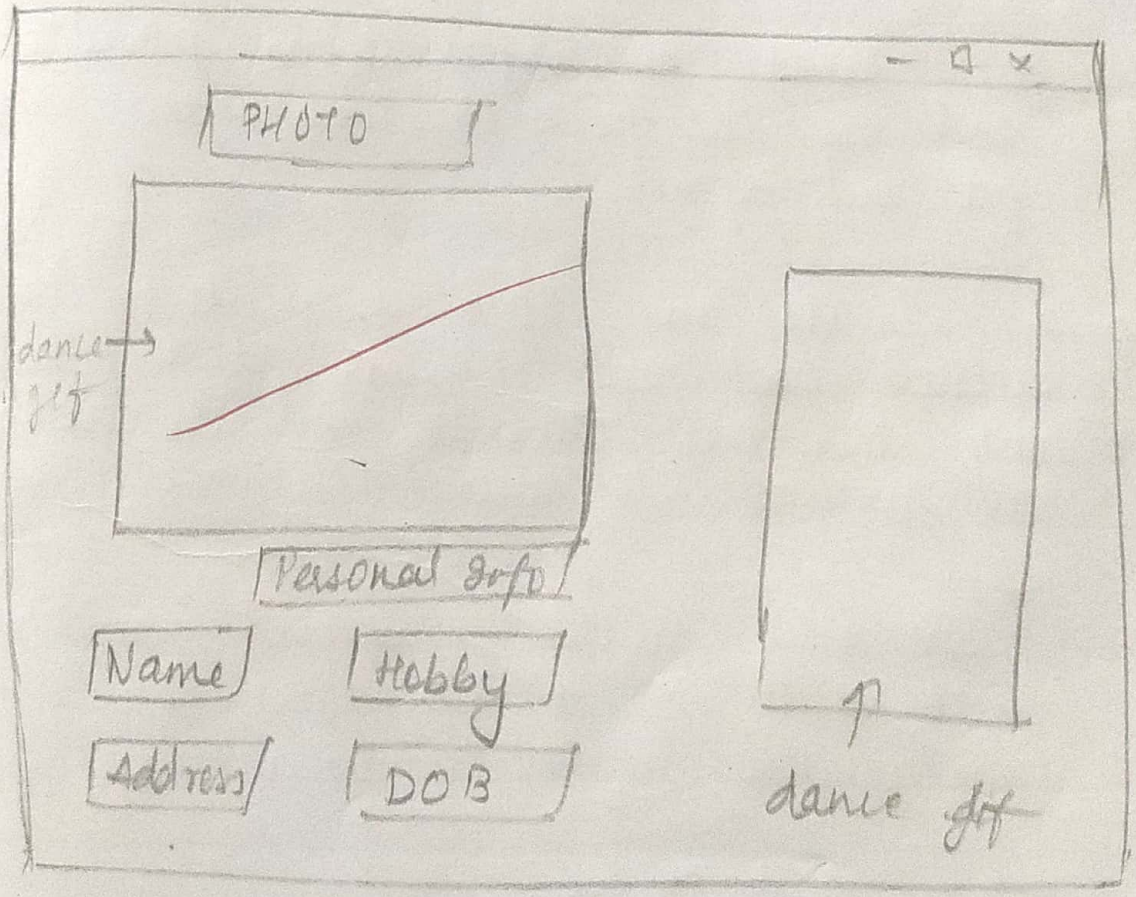
Step 7: Use the label method to position it onto the left frame by placing the image after the sampling.

Step 8: Create another label object positioning it onto the rightframe and specifying the image & background attributes with row and column specify at as (0,0)

Step 9: Now create a toolbar object from the frame method and position it onto the leftframe with the height & width specified & position it on to the second row.

Step 10: Now define the various function for different toolbar options provided in the leftframe.

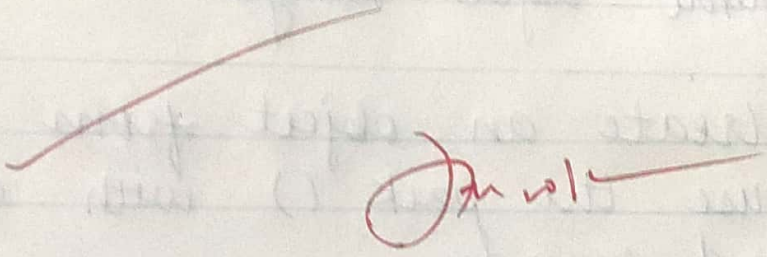
```
def add ();
    Print (" Address : Mumbai " )
def dob ( ):
    Print (" DOB :                " )
Button ( toolbar, text = " Name ", height = 1, width = 10,
    command = name ). grid (row = 0, column = 0)
Button ( toolbar, text = " Hobby ", height = 1, width = 10,
    Command = hob ). grid (row = 2, column = 0 )
Button ( toolbar, text = " DOB ", height = 1, width = 10,
    Command = dob ). grid (row = 2, column = 1 )
root. mainloop ( )
```



step 11: From the label method position the text on the toolbar use the relief attribute & corresponding grid value and incorporate the internal padding as well.

step 12: Create the label method position it onto the toolbar with the next title as person information & position it on same row but next column.

step 13: Now make use of mainloop method.



(E)

* Spinbox Widget :

Step 1 : Create an object from the tk() and subsequently create an object from the spinbox.

step 2 : Make the object so created onto the parent window by tigger the corresponding events.

(F)

* Paned window ~~object~~ widget :

Step 1 : Create an object from paned window() and use the pack() with attribute fill and expand

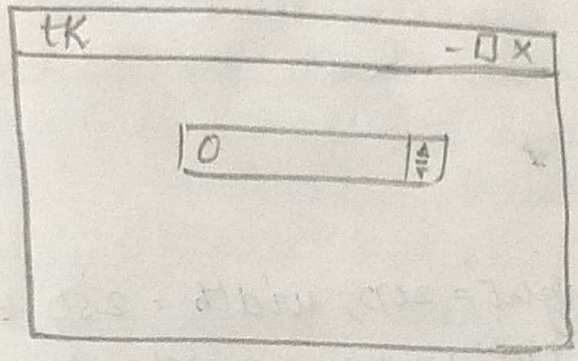
Step 2 : Create an object from the label method and pull it onto the panedwindow with the text attribute and use the add method to embed the new object.

step 3 : Similarly create a second panedwindow object and add it onto the first panedwindow with orientation specified

step 4 : Now create another label object and place it onto the second panedwindow object and add it onto the senondwindow. Trigger main

```
from tkinter import *
master = Tk()
s = spinbox(master, from_ = 0, to = 10)
s.pack()
master.mainloop()
```

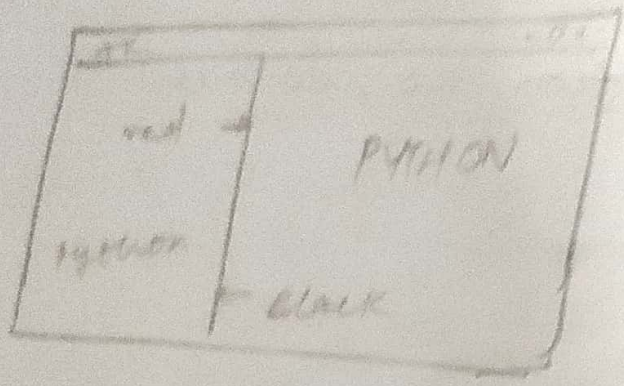
Output:




```
from tkinter import *
root = Tk()
P = Paned window (bg = "red")
P.pack (fill = BOTH, expand = 10)
l = Label (P, text = "python")
P.add = (l)
P1 = Paned window (P, orient = VERTICAL, bg = "black"
P.add (p1)
l1 = label (P1; text = "PYTHON.")
P1.add (l1)
root.mainloop()
```

888

output




```
from tkinter import *
root = Tk()
c = Canvas(root, height = 200, width = 250, bg = 'pink')
arc = c.create_arc(10, 20, 30, 40, start = 0,
        extent = 100, fill = "red")
line = c.create_line(50, 60, 70, 80, fill = "black")
oval = c.create_oval(90, 100, 110, 120, fill = "grey")
c.pack()
root.mainloop()
```

(01)

* Canvas Widget

step 1: Create an object from the Canvas method and use the attribute height, width bg and parent window objects.

step 2: Use the method create line, create oval and create are along with the Canvas object so created and use the co-ordinate values.

step 3: Similarly use the other method & call the past method & use the main loop.

Jm 201

SPD

PRACTICAL NO : 06 :

AIM : Database Connectivity

#1 :

Step 1 : Import the (DBM) dbm library & use the open() for creating the database by specifing the name of the database along with the corresponding flag.

Step 2 : Use the object so created for accessing the given website by corresponding regular name for the website.

Step 3 : Check whether the given url address matches with the regular name of the page is not equal to none than display the message that particular found / matched or else not found / Unmatched.

Step 4 : Use the close() to terminate database library.

#1

```
>>> import dbm
>>> db = dbm.open("database", flag='c', mode=438)
>>> db["name"] = "name"
>>> if db["name"] = "name"
>>> if db["name"] != None:
        print("database not empty // match!")
    else:
        Print("database empty!" // NOT match")
>>> match
>>> db.close()
```

```
import os, sqlite3
conn = sqlite3.connect("employer.db")
cur = conn.cursor()
cur.execute('create table dos(Name char,
    Roll no int')
cur.execute("insert into dos values('Unnati',
    ('Jay', 1897)")
conn.commit()
cur.execute('select * from dos')
Print(cur.fetchall())
Conn close()
```

Output -

[('Unnati', 1797), ('Jay', 1897')]

#2:

Step 1: Import Corresponding library to make database connection, os & SQL lite - 3

Step 2: Now Create the Connection object using SQL lite - 3 library & the Connect() for creating new database.

Step 3: Now create cursor object using the cursor() & from the Connection object created.

Step 4: Now use the execute() for creating the table with the column name & respective database type.

Step 5: Now with cursor - object use the insert statment for entering the values Corresponding to different fields, Corresponding the datatype

Step 6: Use the Commit() to Complete the transaction using the Connection object

Step 7: Use the execute statment along with cursor - object for accessing the values from the database using the select from where clause.

Step 8: Finally use the fetch() or fetchall() for displaying the values from the table using the cursor-object

Step 9: execute() & drop the table syntax for terminating the database & finally use the close().

27/12