PROGRAM 6 :-

```
fileobj = open ("practical1.txt", "w")
fileobj.write ("python is a non procedural language \n
                It has A1 features \n")
fileobj.close()
fileobj = open ("practical1.txt", "r")
str1 = fileobj.read()
print ("the output of read method is", str1)
file obj.close()
a = fileobj.name
b = fileobj.closed
c = fileobj.mode
d = fileobj.softspace
print a, b, c, d
```

Output :

```
('the output of read method is', 'python is a non
  procedural language \n', 'It has A1 features\n')
  practical1.txt True r 0
```

```
#readline()
fileobj = open ("abc.txt", "r")
str2 = fileobj.readline()
Print ("the output of read method is:", str2)
    fileobj.close()
```

>>> The output of read method is : I am dnamika /n my roll no is 1754 /n Now I am doing FYBsc.CS /n It is my first practical /n

PRACTICAL: 01

Aim: To study different modes in file operation.

Algorithm:-

→ Objective: demonstrate the use of different file accessing mode, different attributes of the file object and differentiate among the different read method.

I Create a file object by using the open method and use the write access mode followed by writing some content on to the file and their closing the file.

I Open the file in read mode and use the read method or read line method or read lines method and finally display the content of that variable

II Use the file object for finding the name of the file, mode in which the file is open or close and finally the output of the softspace display attribute.

IV Now open the fileobject in write mode write some another content close subsequently then again open the file object in 'w+' mode that is the update mode and write contents

V Open file object in read mode, display the update written contents and close, open again in 'r+' mode with parameter passed and display the output subsequently.

VI Now open file object in append mode, open write mode write content close the file object, open again in read mode and display the append output.

```
# readlines()
fileobj = open("abc.txt","r")
str 3 = fileobj.readlines()
Print("the output of read method is:", str 3)
fileobj.close()
>>> the output of read method is : ["I am anamika/n
    my roll no is 1754/n now I am doing fyBSc is
    /n its my first practical /n"]

a = fileobj.name
Print("the output of name is :", a)
fileobj.close()
c = fileobj.mode
Print("the output of mode is :", c)
fileobj.close()
d = fileobj.softspace
Print("the output of softspace is :", d)
   fileobj.close()
>>>> the output of name is : abc.txt
>>> the output of closed is : True
>>> the output of mode is : r
>>> the output of softspace is : 0

# w+ mode
file obj = open("abc.txt", "w+")
fileobj.write("Anamika")
   fileobj.close()

# write mode
fileobj = open("practical1.txt", "w")
fileobj.write("programming with python")
   fileobj.close()
```

```
# r+ mode
fileobj = open("practical.txt", "r+")
str1 = fileobj.read(6)
print("output of r+", str1)
fileobj.close()
>>> ('output of r+', 'Anamika')

# read mode
fileobj = open("practical.txt", "r")
str2 = fileobj.read()
print("output of read mode", str2)
    fileobj.close
>>> ('output of read mode!',
    'programming with python')
# append mode
fileobj = open("practical1.txt", "a")
fileobj.write("data structure")
fileobj.close()
fileobj = open("practical1.txt", "r")
str3 = fileobj.read()
print("output of append mode:", str3)
fileobj.close()
>>> ('output of append mode:', 'Anamika, 'data
                                structure')
```

VII Open the file object in read mode, declare a variable and perform fileobject dot tell method and store the output consequently in variable.

VIII Use the seek method with the arguments with opening the fileobj in read mode and using subsequently.

IX Open file object with readmode also use the readlines method and store the output subsequently in and print the same for counting the length use the for condition statement and display the length

```
# tell ()
fileobj - open ("practical1.txt","r")
pos = fileobj.tell()
print ("tell () : ", pos)
fileobj.close
>>> ('tell () : ', pos)

# seek ()
fileobj = open ("practical1.txt","r")
str4 = fileobj.seek (0,0)
str8 = fileobj.read (10)
print ("The beginning of the file : ", str8).

# finding length of different lines exist within
    lines
fileobj = open ("practical1.txt","r")
str9 = fileobj.headlines()
print ("output :" str9)
    for line in str9:
                print(len(line))
    fileobj.close()
>>> ('output :', ['college database'])
```

last
03/21

Code:-

```
mytuple = ("Anamika", "Anushka", "Annu", "Kirti")
myiter = iter (mytuple)
print (next (myiter))
print (next (myiter))
print (next (myiter))
print (next (myiter))
```

Output:-

Anamika
Anushka
Annu
Kirti

Code:

```
mytuple = ("Anamika", "Anushka", "Annu", "Kirti")
for a in mytuple:
    print (a)
```

Output:

Anamika
Anushka
Annu
Kirti

Aim: To display elements of a tuple using iterator method.

To display elements of a tuple using iterator method.

Algorithm:

Form a tuple with certain elements inserted in it.

Use iter method with tuple and assign it to a variable.

Use the next method with variable and print elements.

To use iter method with for loop.

Algorithm:

Form a tuple with certain elements inserted in it.

Use the for conditional statement to access each element of tuple.

Print the elements of tuple

c) WAP using the iterable method for displaying the set of odd numbers.

Algorithm:

I define a class which will contain various

II define the iter method with an argument and return the value of argument

III define a function which increments the value of argument by two

IV Create an object which inherits the properties of class and take the user input

V Use the for conditional statement followed by if conditional statement and print the answer.

Program:

```
class odd:
    def __iter__(self):
        self.num = 1
        return self
    def next(self):
        num = self.num
        self.num += 2
        return num
myobj = odd()
myiter = iter(myobj)
x = int(input("Enter range number :"))
for i in range myiter:
    if i < x:
        print(i)
```

Output:

Enter a number : 16

1
3
5
7
9
11
13
15

Program:

```
class myclass:
    def __iter__(self):
        self.a = 1
        return self
    def __next__(self):
        if self.a <= 20:
            x = self.a
            self.a += 1
            return x
        else:
            raise stop iteration
myobj = myclass()
myiter = iter(myobj)
for x in myiter:
    print(x)
```

Output:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

To print first 20 numbers using iter method.

Algorithm :

define a class which will contain various functions

define iter method with an argument and return the value of argument.

define next method which increments the value of argument by 1 and prints it

Create a object which inherits the property of class and take the user input.

Use the loop to print to the value of the variable.

e) To find if the number is odd or even from given list using map method.

Algorithm :

I declare a list num variable and declare some elements

II define a function even which consist various conditional statement

III Further use if conditional statement to check the modulo number of each element of list and return the message accordingly.

IV Use the map method to print the value in list format.

program:

```
num = [0,4,5,7,9,11,13,15,20,9]
num = list (map( lambda x:x%2,num))
print (num)
def even(x):
    if x%.2 == 0
        return "even"
    else:
        return "odd"
list (map( even, list num))
```

output:

[0,4,5,7,9,11,13,15,20,9]

[even, even, odd, odd, odd, odd, odd, odd, even, odd]

Program:

```
def square (x):
    return (x** 2)
def cube (x):
    return (x * * 3)
funct 1 = [square, cube]
for : in range (4):
        value = list (map(lambda x: x(2), func 1)]
print (list (value))
```

output:

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

f) To find square and cube of number simultaneously using map method.

Algorithm:

I define a function which returns the square of given number.

II define an another function which returns cube of given number.

III store the output simultaneously in the list

IV Use the for loop followed by the map function and print the value of the result.

9 To find square of a number without using map method.

Algorithm

I define a list which contains certain values.

II define an empty list.

III Use for loop followed by append method to append the result into the empty list.

IV Print the value of list.

Program:

```
list 1 = [1,2,3,4,5]
empty = [ ]
for ; in list 1 :
        empty . append (1 * * 2)
print (empty)
```

Output :

[1,4,9,16, 25]

Program:

```
try:
    fileobj = open("abc.txt","w")
    fileobj.write("python is an intended language")
except IOerror:
        print("It is an environmental error")

else:
    print("operation is successful")
```

Output:

Operation is successful.

a) Aim : To write a program using exception block related to environment error.

Algorithm :

I Open a previous file using any mode of file operation under try block.

II define the except block to print the message for the error.

III define the else block to print the message accordingly if no error is found.

b) Aim: WAP to demonstrate the use value error in given program

Algorithm:

I Accept the user input of integer datatype in the try block.

II define the except block with value error as a keyword and display appropriate

III define else block if none of the error is encounter and display message accordingly.

program:

```
try:
x = int(input("enter a number:"))
except value error:
        print("This is value error!")
except IOerror:
        print("This is environmental error!")
else:
        print("operation successful!")
```

output:

```
enter a number: 20
operation successful!
enter a number: 2+3
This is value error!
```

```
# match()
import re
pattern = r"FYCS"
sequence = "FYCS represents computer science stream"
if re.match(pattern, sequence):
    print("matched pattern found!")
else:
    print("NOT FOUND!")
```

>>> matched pattern found

```
# numerical values (segregation)
import re
pattern = r'\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']

Practical: 04

Topic: Regular Expression

Import remodule declare pattern and declare sequence use match method with declare arguments if arguments matched than print the same otherwise print pattern NOT FOUND!

Import re module declare pattern with literal and mita character. declare string value. Use the findall () with arguments and print the same

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

Import remodule declare string and accordingly declare pattern replace the blank space with no_space. Use rule() with 3 arguments and print the string without spaces.

Import remodule declare a sequence use search method for finding subsequently use the group () with dot operator as search() gives

memory location using group() it will show up the matched string.

VI Import remodule 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 all no matches otherwise print failed.

VII Import re module declare a string use the module with findall() for finding the vowels in the string and declare the same.

```
# no_space :
import re
string = 'abc def ghi'
patton = 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
list1 = ['8004567891', '9145673210', '7865432981',
         '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!")
```

```
>>> criteria matched for cell numbu
    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:
```

VIII Import re module declare the host and domain name declare pattern for separating the host and domain name. Use the findall() and print the output respectively

IX Import remodule enter a string use pattern to display only two elements of the two particular string Use findall() declare two variables with initial value as zero use for condition and subsequently use the if condition check whether condition satisfy add up the or else increment value and display the value subsequently

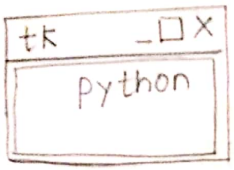
```
if (v == 'ms'):
    f = f+1
else:
    m = m+1
print ("No of males is :",m)
print ("No of females is :", f)
```

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

creation of parent window

```
from Tkinter import *
    root = TK()
    l = label (root , text = "python")
     l . pack ()
     root . main loop ()
```

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, ipad x = 40)
```

Practical 05

Aim GUI Components

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

Create an object using the tk()

Create a variable using the widget label and use the text method.

Use the mainloop() for triggering of the corresponding above mention events

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

Create a variable from the text method and position it on the parent window.

III Use the pack() along with the object created from the text() and use the paramater

1) side = LEFT, padx - 20
2) side = LEFT, pady - 30
3) side = TOP, ipadx = 40
4) side - TOP, ipady = 50

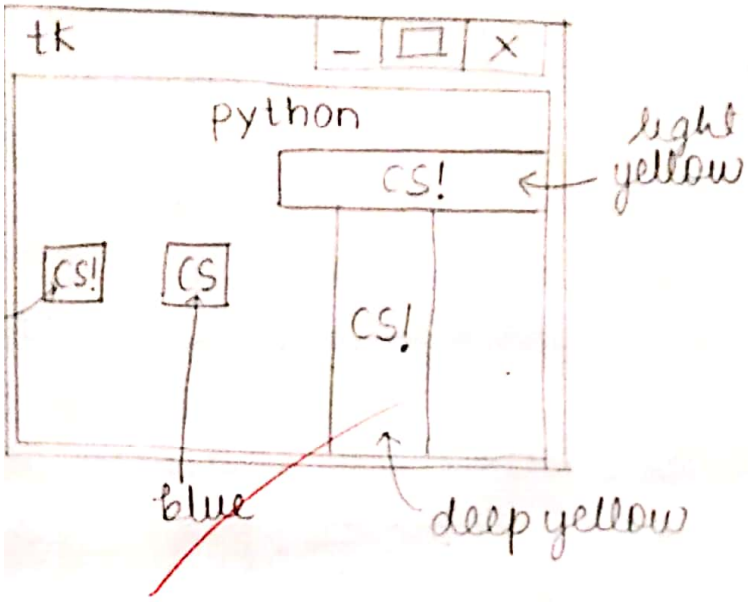
IV Use the mainloop() for the triggering of the corresponding events :-

V Now repeat above steps with the label() which takes the following arguments :-

1) Name of the parent window
2) Text attribute which defines the string.
3) The background color (bg)
4) The foreground fg and then use the pack() with a relevant padding attributes

```
(root, text = "CS!", bg = "orange",
fg = "black", font = "10")

(side = TOP, ipady = 50)
mainloop()
```



Program:

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

Output :-

a) SPINBOX

I Create an object from the tk method and subsequently create an object from the spinbox method.

II Make the object so created onto the parent window and trigger the corresponding events

III Use the pack method to provide the direction using anchor method

IV Use the mainloop method to terminate.

b CANVAS WIDGET

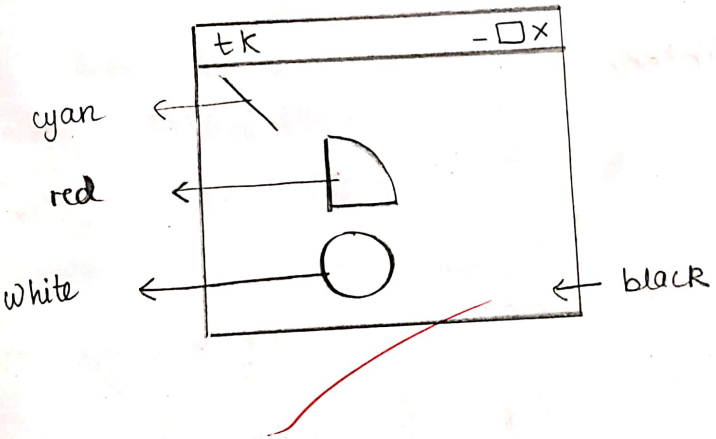
I Use the tkinter method and create an object from the canvas method and use the attribute height, weight, bg color and the parent window object

II Use the method create oval, create line and create arc along with the canvas object so created and use the co-ordinates value. Also use the fill attribute to assign various colors.

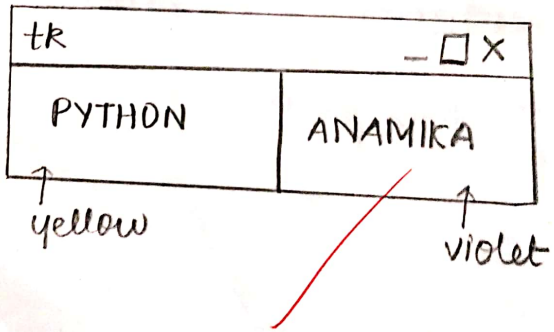
III Now call the pack method and mainloop method.

source code:

```
from tkinter import *
root = TK()
c1 = canvas (root, height = 400, width = 400, bg = "black")
oval = c1.create_oval (20,140, 150, 250, fill = "white")
line = c1.create_oval (30,40,50,60, fill = "cyan")
arc = c1 . create .oval (20,140,150,60, fill = "red")
c1. pack ()
root.mainloop()
```



```
from tkinter import *
root = Tk()
P = paned window (bg = "pink")
P.pack(fill = BOTH, expand = 10)
L1 = Label (p, text = "PYTHON"), bg = "yellow")
P.add(L1)
P1 = paned window (P, orient = VERTICAL, bg = "green")
P.add(P1)
L2 = label(P1, text = "ANAMIKA", bg = "Violet")
P.add(L2)
root.mainloop()
```



c Paned Window

I Create an object from paned window and use the pack method with the attribute fill and expand

II 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.

III Similarly create a second paned window object and add it on the 1st paned window with orientation specified

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

V Now use the mainloop method to terminate it

I Import db library and use the open method for creating the database by specify name of the data base along with the corresponding flag.

II Use the objects for accessing the given web size and the corresponding regular for the web.

III Check whether the given URL addresses with the regular of the pages is not equal to none than display the message from URL addres else not found.

code:

```
import dbm
db = dbm.open("data base", flag, "c")
if db ["WOW"] != None:
    print ("good")
else:
    print ("not good").
```

output:

good.

b)

Program :-

```
import os, sqlite3
connection = sqlite3.connect("student.db")
c1 = connection.cursor()
c1.execute('create table student(name, RNO, DOB)')
c1.execute('insert into student values("Anamika", 1745, 23-09-2000)')
c1.execute('insert into student values("Shraddha", 1746, 24-08-2001)')
c1.execute('insert into student values("Kiti", 1747, 18-2-1999)')
connection.commit()
c1.execute('select * from student')
c1.fetchall()
c1.execute('Drop table student')
```

I

II

III

IV

V

VI

b)

I Import the corresponding library taking of database connection.

II Now create connection objects using sqllite library and connecting method for create the new database

III Now create the cursor objects using cursor method from the connection objects create in steps

IV Now use the executing method for creating the table with the column name and respective data type.

V Now with the cursor objects use insert statements for entering the values co-ordinating into the different field considering the data types.

VI Use the commit method to complete the transition to use the connection objects

VII Use the execute statement along with the cursor objects for accessing the value the database using selecting from when clause.

VIII Finally use the fetch all method from display the value for the table using the cursor objects

IX Use the execute method and the drop table syntax for terminating the database finally use the close method

Output :

```
[ ("Anamika", 1745, 23-09-2000),
  ("Shraddha", 1746, 24-08-2001),
  ("Kirti", 1747, 18-02-1999)]
```

PROJECT 1:

TOPIC: COMPOUND INTEREST RATE CALCULATOR USING GUI

```
from tkinter import*
import tkinter.messagebox

class Interest:
  def __init__(self,root):
    self.root=root
    self.root.title("interest rate calculator")
    self.root.geometry("1350x800+0+0")
    self.root.configure(background='gray')

#================Frame==============================================================
====================

    MainFrame = Frame(self.root, bd=20, width=1350, height=700, padx=10, pady=10, bg="powder
blue", relief=RIDGE)
    MainFrame.grid()

    LeftFrame = Frame(MainFrame, bd=10, width=600, height=600, padx=10, pady=42,  relief=RIDGE)
    LeftFrame.pack(side=LEFT)

    RightFrame=Frame(MainFrame, bd=10, width=560, height=600, padx=10, pady=10, relief=RIDGE)
    RightFrame.pack(side=RIGHT)

#====================Frame==============================================================
=========================

    LeftFrame0 = Frame(LeftFrame, bd=5, width=712, height=143, padx=5, pady=5, bg="powder blue",
relief=RIDGE)
    LeftFrame0.grid(row=0,column=0)
    LeftFrame1 = Frame(LeftFrame, bd=5, width=712, height=179, padx=5, pady=12, relief=RIDGE)
    LeftFrame1.grid(row=4,column=0)
    LeftFrame2 = Frame(LeftFrame, bd=5, width=712, height=167, padx=5, pady=10, relief=RIDGE)
    LeftFrame2.grid(row=2,column=0)
```

```
    RightFrame1 = Frame(RightFrame, bd=20, width=1350, height=700, padx=10, pady=10, bg="powder
blue", relief=RIDGE)
    RightFrame1.grid(row=1,column=0)

#=============================================================================================
=======================

    var1=StringVar()
    var2=StringVar()

#=============================================Functions=============================
================================

    def Reset():
        var1.set("")
        var2.set("")
        self.txtRateTable.delete("1.0",END)


    def iExit():
        iExit=tKinter.messagebox.askyesno("Interest Rate", "confirm if you want to exit")
        if iExit >0:
            root.destroy()
            return
    def balance(initBalance, rate, NumYears):
        finalBalance = initBalance*((1+rate/100)**NumYears
        return finalBalance
    def table():
        self.txtRateTable.delete("1.0",END)
        initBalance = float(var1.get())
        NumYears=int (var2.get())
```

```
        for rate in range(-6,9,3):
            b= balance(initBalance, rate, NumYears)
            print("Rate: %2d%%, Balance: Rs.2f" % (rate, b))
            iRate = str (rate)
            iB = str (b)
            self.txtRateTable.insert(END,"\nRate:" +(iRate)+ "%"+ "Balance:" + ('Rs%.2f' % float(iB)) +
"\n\n")


        #======================Entry and
Labels================================================================================
        self.lblTitle=Label(LeftFrame0, text= "Interest Rate Calculator", padx=17, pady=4,
bd=1,font=('arial',40,'bold'),
                            bg="powder blue", width= 20)
        self.lblTitle.pack()


        self.lblBalance=Label(LeftFrame2, text= "please enter the initial
balance:",font=('arial',20,'bold'),bd=2,justify=LEFT )
        self.lblBalance.grid(row=0, column=0, padx=15)


        self.txtBalance=Entry(LeftFrame2, textvariable= var1, font=('arial',20,'bold'), bd=5, width=14,
justify=RIGHT)
        self.txtBalance.grid(row=0, column=1, padx=3, pady=10)


        self.lblYear=Label(LeftFrame2, text= "please enter the number of
years:",font=('arial',20,'bold'),bd=2, justify=LEFT)
        self.lblYear.grid(row=1, column=0, padx=15)


        self.txtYear=Entry(LeftFrame2, textvariable= var1, font=('arial',20,'bold'), bd=5, width=14,
justify=RIGHT)
        self.txtYear.grid(row=1, column=1, padx=3, pady=10)
```

```
#=============================================================================================
==============================

    self.lblRateTable=Label(RightFrame1,font=('arial',20,'bold'),text="\t Pay Back").grid(row=0,
column=0)

    self.txtRateTable= Text(RightFrame1, height=15, width=52, bd=16,font=('arial',12,'bold') )

    self.txtRateTable.grid(row=1, column=0, columnspan=3)



#=================================================Button=============================
==============================

    self.btnRate = Button(LeftFrame1, text="Interest Rate", padx=5, pady=2, bd=4,
width=12,font=('arial',20,'bold'),height=1, bg="powder blue", command= table)

    self.btnRate.grid(row=3, column=0)


    self.btnReset = Button(LeftFrame1, padx=5, pady=2, bd=4,
width=12,font=('arial',20,'bold'),height=1, fg="black", text="Reset",bg="powder blue",
command=Reset).grid(row=3, column=1)

    Self.btnExit = Button(LeftFrame1, padx=5, pady=2, bd=4, width=12,font=('arial',20,'bold'),height=1,
fg="black", text="Exit",bg="powder blue", command=iExit).grid(row=3, column=2)

if __name__=='__main__':

  root=Tk()

  application = Interest(root)

  root.mainloop()
```

Interest Rate Calculator

please enter the initial balance:

please enter the number of years:

Interest Rate	Reset	Exit

Pay Back

PROJECT 2 : STUDENT DATABASE MANAGEMENT SYSTEM

```
#Frontend

from tkinter import*
import tkinter.messagebox
import stddatabase

class Student:
    def __init__(self,root):
        self.root = root
        self.root.title("Student Database Management System")
        self.root.geometry("1350x750+0+0")
        self.root.config(bg="pink")
        StdID = StringVar()
        Firstname = StringVar()
        Surname = StringVar()
        DoB = StringVar()
        Age = StringVar()
        Gender = StringVar()
        Address = StringVar()
        Mobile = StringVar()
        #==================Function================
        def iExit():
            iExit = tkinter.messagebox.askyesno("Student Database Management System","Confirm if you
want to exit")
            if iExit > 0:
                root.destroy()
                return
        def clearData():
            self.txtStdID.delete(0,END)
```

```
        self.txtfna.delete(0,END)
        self.txtSna.delete(0,END)
        self.txtDoB.delete(0,END)
        self.txtAge.delete(0,END)
        self.txtGender.delete(0,END)
        self.txtAdr.delete(0,END)
        self.txtMobile.delete(0,END)
    def addData():
      if(len(StdID.get())!=0

stddatabase.addStdRec(StdID.get(),Firstname.get(),Surname.get(),DoB.get(),Age.get(),Gender.get(),Addr
ess.get(),Mobile.get()):
          studentlist.delete(0,END)

studentlist.insert(END,(StdID.get(),Firstname.get(),Surname.get(),DoB.get(),Age.get(),Gender.get(),Addr
ess.get(),Mobile.get()))
    def DisplayData():
      studentlist.delete(0,END)
      for row in stddatabase.viewData()
      studentlist.insert(END,row,str(""))
  def StudentRec(event):
    global sd
    searchStd = studentlist.curselection()[0]
    sd = studentlist.get(searchStd)
    self.txtStdID.delete(0,END)
    self.txtStdID.insert(END,sd[1])
    self.txtfna.delete(0,END)
    self.txtfna.insert(END,sd[2])
    self.txtSna.delete(0,END)
    self.txtSna.insert(END,sd[3])
```

```
        self.txtDoB.delete(0,END)
        self.txtDoB.insert(END,sd[4])
        self.txtAge.delete(0,END)
        self.txtAge.insert(END,sd[5])
        self.txtGender.delete(0,END)
        self.txtGender.insert(END,sd[6])
        self.txtAdr.delete(0,END)
        self.txtAdr.insert(END,sd[7])
        self.txtMobile.delete(0,END)
        self.txtMobile.insert(END,sd[8])
    def DeleteData():
      if(len(StdID.get())!=0:
        stddatabase.deleteRec(sd[0])
        clearData()
        DisplayData()
    #==================Frames==============
    MainFrame = Frame(self.root, bg="pink")
    MainFrame.grid()
    TitFrame = Frame(MainFrame,bd=2, padx=54, pady=8, bg="yellow", relief = RIDGE)
    TitFrame.pack(side=TOP)
    self.lblTit = Label(TitFrame ,font=('arial', 47,'bold'),text="Student Database Management
System",bg="yellow")
    self.lblTit.grid()
    ButtonFrame = Frame(MainFrame,bd=2,width=1350, height=70, padx=18, pady=10, bg="yellow",
relief = RIDGE)
    ButtonFrame.pack(side=BOTTOM)
    DataFrame = Frame(MainFrame,bd=1,width=1300, height=400, padx=20, pady=20, bg="pink", relief
= RIDGE)
    DataFrame.pack(side=BOTTOM)
```

```
        DataFrameLEFT = LabelFrame(DataFrame,bd=1,width=1300, height=400, padx=20,
    bg="yellow",font=('arial', 47,'bold'), text="Student Info\n")

        DataFrameLEFT.pack(side=LEFT)

        DataFrameRIGHT = LabelFrame(DataFrame,bd=1,width=450, height=300, padx=31, pady=3,
    bg="yellow", relief = RIDGE,font=('arial', 20,'bold'),text="Student Details\n")

        DataFrameRIGHT.pack(side=RIGHT)


        #=========================Labels and Entry Widget============================

        self.lblStdID = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Student ID:",padx=2, pady=2,
    bg="yellow")

        self.lblStdID.grid(row=0, column=0, sticky=W)

        self.txtStdID = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=StdID,width=39)

        self.txtStdID.grid(row=0, column=1)


        self.lblfna = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Firstname",padx=2, pady=2,
    bg="yellow")

        self.lblfna.grid(row=1, column=0, sticky=W)

        self.txtfna = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=StdID,width=39)

        self.txtfna.grid(row=1, column=1)


        self.lblSna = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Surname:",padx=2, pady=2,
    bg="yellow")

        self.lblSna.grid(row=2, column=0, sticky=W)

        self.txtSna = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=StdID,width=39)

        self.txtSna.grid(row=2, column=1)


        self.lblDoB = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Date of Birth:",padx=2, pady=2,
    bg="yellow")

        self.lblDoB.grid(row=3, column=0, sticky=W)

        self.txtDoB = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=DoB,width=39)

        self.txtDoB.grid(row=3, column=1)
```

```
        self.lblAge = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Age:",padx=2, pady=2,
bg="yellow")
        self.lblAge.grid(row=4, column=0, sticky=W)
        self.txtAge = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=Age,width=39)
        self.txtAge.grid(row=4, column=1)

        self.lblGender = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Gender:",padx=2, pady=2,
bg="yellow")
        self.lblGender.grid(row=5, column=0, sticky=W)
        self.txtGender = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=Gender,width=39)
        self.txtGender.grid(row=5, column=1)

        self.lblAddress = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Address:",padx=2, pady=2,
bg="yellow")
        self.lblAddress.grid(row=6, column=0, sticky=W)
        self.txtAddress = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=Address,width=39)
        self.txtAddress.grid(row=6, column=1)

        self.lblMobile = Label(DataFrameLEFT ,font=('arial', 20,'bold'),text="Mobile:",padx=2, pady=2,
bg="yellow")
        self.lblMobile.grid(row=7, column=0, sticky=W)
        self.txtMobile = Entry(DataFrameLEFT ,font=('arial', 20,'bold'),textvariable=Mobile,width=39)
        self.txtMobile.grid(row=7, column=1)
        #========================ListBox and ScrollBar Widget============================
        scrollbar = Scrollbar(DataFrameRIGHT)
        scrollbar.grid(row=0, column=1, sticky='ns')

        studentlist = Listbox(DataFrameRIGHT, width=41, height=16, font=('arial', 12,'bold'),
yscrollcommand=scrollbar.set)
```

```
        studentlist.grid(row=0, column=0, padx=8)

        scrollbar.config(command = studentlist.yview)

        #========================Button  Widget==========================

        self.btnAddData = Button(ButtonFrame, text="Add New",font=('arial', 20,'bold'),height=1,
width=10, bd=4, command=addData)

        self.btnAddData.grid(row=0, column=0)


        self.btnDisplayData = Button(ButtonFrame, text="Display",font=('arial', 20,'bold'),height=1,
width=10, bd=4,command=DisplayData)

        self.btnDisplayData.grid(row=0, column=1)


        self.btnClearData = Button(ButtonFrame, text="Clear",font=('arial', 20,'bold'),height=1, width=1
bd=4, command=clearData)

        self.btnClearData.grid(row=0, column=2)


        self.btnDeleteData = Button(ButtonFrame, text="Delete",font=('arial', 20,'bold'),height=1,
width=10, bd=4, command=DeleteData)

        self.btnDeleteData.grid(row=0, column=3)


        self.btnSearchData = Button(ButtonFrame, text="Search",font=('arial', 20,'bold'),height=1,
width=10, bd=4, command=SearchDatabase)

        self.btnSearchData.grid(row=0, column=4)


        self.btnUpdateData = Button(ButtonFrame, text="Update",font=('arial', 20,'bold'),height=1,
width=10, bd=4,command=update)

        self.btnUpdateData.grid(row=0, column=5)


        self.btnExit = Button(ButtonFrame, text="Exit",font=('arial', 20,'bold'),height=1, width=10, bd=4,
command=iExit)

        self.btnExit.grid(row=0, column=6)

if __name__=='__main__':
```

```
root = Tk()
application = Student(root)
root.mainloop()


import sqlite3
#backend


def studentData():
    con=sqlite3.connect("student.db")
    cur.execute("CREATE TABLE IF NOT EXISTS student(id INTEGER PPRIMARY KEY, StdID text, Firstname
text, Surname text, DoB text, \
            Age text, Gender text, Address text, Mobile text)")
    con.commit()
    con.close()


def addStdRec(StdID, Firstname, Surname, DoB, Age, Gender, Address,Mobile):
    con=sqlite3.connect("student.db")
    cur=con.cursor(*)
    cur.execute("INSERT INTO student VALUES (NULL, ?,?,?,?,?,?,?,?,)",StdID, Firstname, Surname, DoB,
Age, Gender, Address,Mobile))
    con.commit()
    con.close()


def viewData():
    con=sqlite3.connect("student.db")
    cur=con.cursor()
    cur.execute("SELECT * FROM student")
    rows=cur.fetchall()
    con.close()
```

```
    return rows


def deleteRec(id):
    con=sqlite3.connect("student.db")
    cur=con.cursor()
    cur.execute("DELETE FROM student WHERE id=?",(id,))
    con.commit()
    con.close()


def searchData(StdID="", Firstname="", Surname="", DoB="", Age="", Gender="", Address="",
Mobile=""):
    con=sqlite3.connect("student.db")
    cur=con.cursor()
    cur.execute("SELECT * FROM student WHERE StdID=? OR Firstname=? OR Surname=? OR DoB=? OR
Age=? OR  Gender=? \
        OR Address=? OR Mobile=? ",(StdID, Firstname, Surname, DoB, Age, Gender, Address,Mobile))
    rows=cur.fetchall()
    con.close()
    return rows
def dataUpdate(id,StdID="", Firstname="", Surname="", DoB="", Age="", Gender="", Address="",
Mobile=""):
    con=sqlite3.connect("student.db")
    cur=con.cursor()
    cur.execute("UPDATE student SET StdID=?, Firstname=?,  Surname=?, DoB=?,  Age=?,
Gender=?,Address=?,Mobile=?, WHERE id=?",\
            (StdID, Firstname, Surname, DoB, Age, Gender, Address,Mobile, id))
    con.commit()
    con.close()
studentData()
```

Student Database Management System

Student Info

Student Details

Student ID:
Firstname
Surname:
Date of Birth:
Age:
Gender:
Address:
Mobile: