NAME: Saumik Dutta

Rollno:- 40

Programme:- MCA 2021-23 Batch

Course:- Python (Trimester-2)

Python Programming - LAB: Journal

# **Practical 1:**

## Write a Python Program to count the number of character(char frequency) in a string

program:-

ch='y'

while ch.lower()=='y':

str1 = input ("Enter the string: ")

d = dict()

for c in str1:

if c in d:

d[c] = d[c] + 1

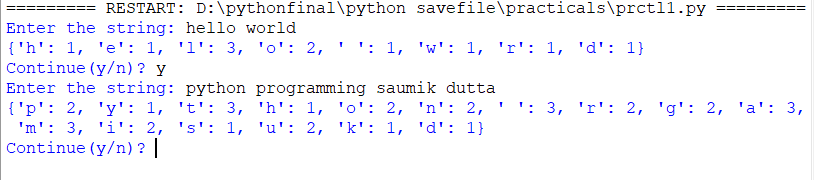
else:

d[c] = 1

print(d)

ch=input("Continue(y/n)? ")

output:-



# **Practical 2:**

## Write a python Program to get a string made of the first 2 and last 2 character from a given string. If the string length is less than 2, return “Empty” instead of empty string

Program:-

def prbstmt2():

ch='y'

while ch.lower()=='y':

str1=input("Enter a String: ")

if len(str1)>=2:

print("Expected Result: ",str1[0:2]+str1[-2:])

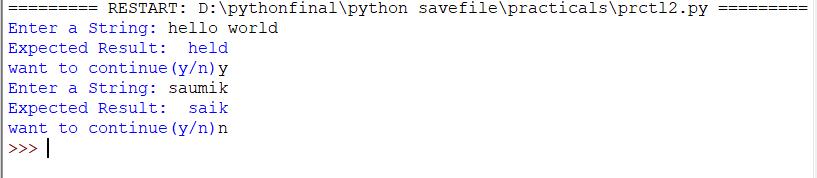
else:

print("Empty String(String len<2)")

ch=input("want to continue(y/n)")

prbstmt2()

output:-



# **Practical 3:**

Write a python program to add ‘ing’ at the end of a string (length at least 3). If the given string already ends with ‘ing’ then add ‘ly’ instead. If the string length of the given string is less than 3, leave it unchanged.

program:-

ch='y'

ing='ing'

ly='ly'

while ch.lower()=='y':

str2=input("Enter a String: ")

if len(str2)>=3:

if (str2.endswith('ing')):

print("Result: ",str2.replace('ing',ly))

else:

print("Result: ",str2+ing)

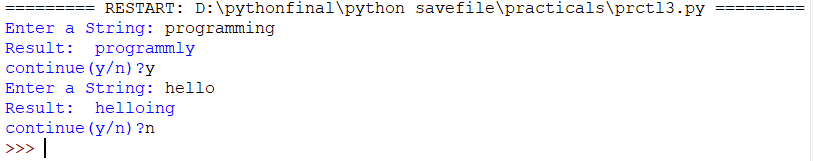
ch=input("continue(y/n)?")

else:

print("Result: ",str2)

ch=input("continue(y/n)?")

output:-



# **Practical 4:**

## Write a Python Program to get a string where all occurrences of its first character have been changed to ‘@/#/$’ except the first char itself.

program:-

ch='y'

while ch.lower()=='y':

str2=input("Enter a String :")

str3=input("Enter a character to replace: ")

if len(str2)>=3:

str4=str2[0]+str2[1:].replace(str2[0],str3)

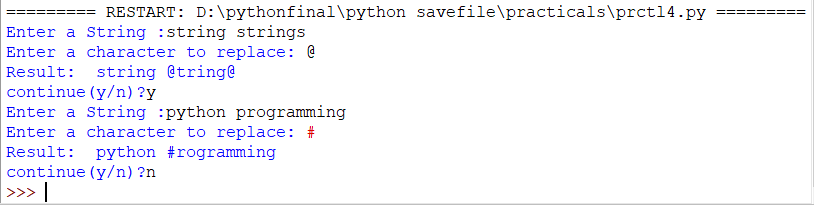
print("Result: ",str4)

else:

print("Empty String")

ch=input("continue(y/n)?")

output:-



# **Practical 5:**

## Write a python program to create list by concatenating a given list which range goes from 1 to n.

program:-

ch='y'

while ch.lower()=='y':

lst=[]

lst1=[]

m=int(input("Enter Number of elements: "))

n=int(input("Enter range: "))

str1=''

for i in range(0,m):

str1=input("Element : ")

lst.append(str1)

for x in range(1,n+1):

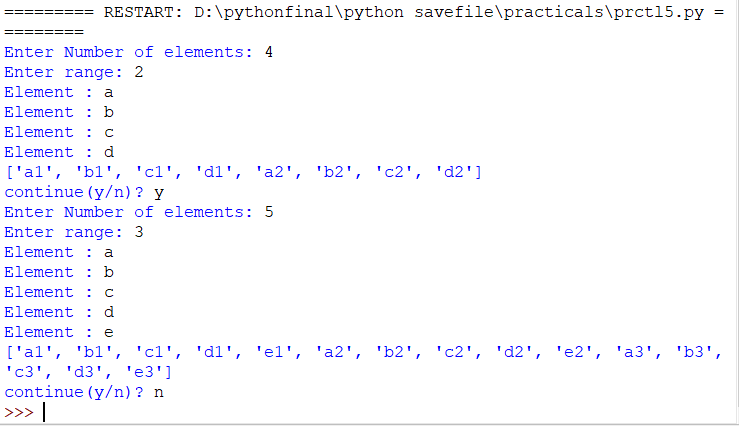
for y in range(0,m):

lst1.append(lst[y]+str(x))

print(lst1)

ch=input("continue(y/n)? ")

output:-



# **Practical 6:**

## Write a python program to convert a list of multiple integer into single string.

program:-

ch='y'

while ch.lower()=='y':

lst=[]

lst1=[]

n=int(input("Enter Number of elements: "))

str1=''

for i in range(0,n):

int1=int(input("Element: "))

lst.append(int1)

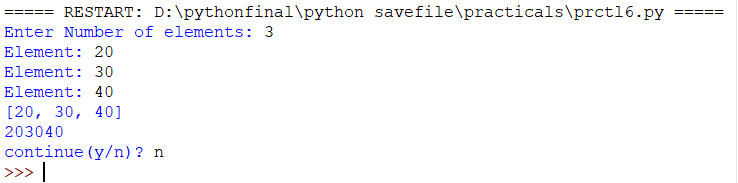
str1=str1+str(int1)

print(lst)

print(str1)

ch=input("continue(y/n)? ")

output:-



# **Practical 7:**

## Write a python program to transpose a matrix.

program:-

ch='y'

while ch.lower()=='y':

a=input("enter no of rows ")

b=input("enter no of columns ")

y=[]

z=[]

for e in range(0,int(a)):

x=[]

for f in range(0,int(b)):

x.append(int(input("element ")))

y.append(x)

for i in range(0,int(a)):

for j in range(0,int(b)):

print(y[i][j], end = " ")

print()

print("transposing matrix")

for i in range(0,int(b)):

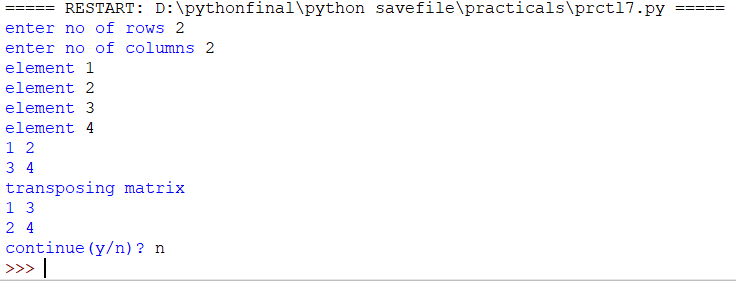
for j in range(0,int(a)):

print(y[j][i], end = " ")

print()

ch=input("continue(y/n)? ")

output:-



# **Practical 8:**

## Write a python program to Perform Matrix Multiplication.

program:-

ch='y'

while ch.lower()=='y':

print("Enter value of rows and cols for both matrix")

a=input("enter no of rows for a: ")

b=input("enter no of columns for a: ")

c=input("enter no of rows for b: ")

d=input("enter no of columns for b: ")

s=[]

t=[]

if(a==d):

print("Matrix a")

for e in range(0,int(a)):

x=[]

for f in range(0,int(b)):

x.append(int(input("element ")))

s.append(x)

print("Matrix b")

for e in range(0,int(c)):

y=[]

for f in range(0,int(d)):

y.append(int(input("element ")))

t.append(y)

print("matrix a")

for i in range(0,int(a)):

for j in range(0,int(b)):

print(s[i][j], end = " ")

print()

print("matrix b")

for i in range(0,int(c)):

for j in range(0,int(d)):

print(t[i][j], end = " ")

print()

print("result matrix c")

u=[[0 for i in range(int(d))] for i in range(int(a))]

for i in range(0,len(s)):

for j in range(0,len(t[0])):

for k in range(0,len(t)):

u[i][j]+=s[i][k] \* t[k][j]

for i in range(0,int(a)):

for j in range(0,int(d)):

print(u[i][j], end = " ")

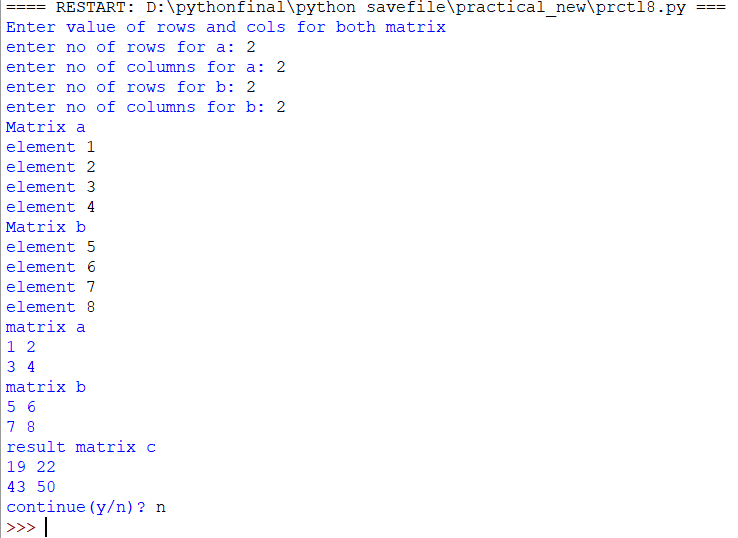
print()

else:

print("Matrix multiplication cannot be performed ,no of rows!= no of columns")

ch=input("continue(y/n)? ")

output:-



**Practical 9:**

A string with paranthesis is well bracketed if all paranthesis are matched i.e every opening bracket has a matching closing bracket and vice versa.Write a python program having function wellbrackered(s) that takes string s containing parenthesis and return True if s is well bracketed else False

Hint: Keep track of nesting depth of brackets.Intially the depth is 0,The depth increases with each opening bracket and decreases with each closing bracket.What are the constraints on the value of nesting depth to be wellbracket?  
Ex:wellbracketed(“22”))  
False  
Wellbracketed”(a+b)(a-b)”  
True

program:-

def wellbracketed(exprs):

a1=exprs.count("(")

a2=exprs.count(")")

t1= "(" in exprs

t2= ")" in exprs

if(t1==True or t2==True):

if(a1==a2):

print("True")

else:

print("False")

else:

print("Expression does not contain parenthesis")

ch='y'

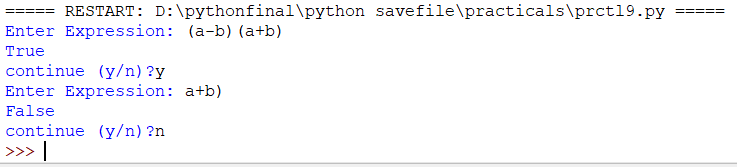
while ch.lower()=='y':

exprs=input("Enter Expression: ")

wellbracketed(exprs)

ch=input("continue (y/n)?")

output:-



**Practical 10:**

A list rotation consist of taking the last element and moving it to front.For instance, if we rotate the list[1,2,3,4,5] we get [5,1,2,3,4]

Write a python having function rotatelist(l,k) that takes list l and a positive integer k and return the list l after k roations.If k is not positive, your function should return l unchanged.Note that yout function should not change l itself and should return roatated list

Eg: rotatelist([1,2,3,4,5],3)

o/p:[3,4,5,1,2]

program:-

ch='y'

def rotatelist(lst,k):

lstn=[]

if k>0:

lstn = lst[-k:] + lst[:-k]

return lstn

else:

return lst

while ch.lower()=='y':

lst=[]

n=input("Enter size of list: ")

k=input("Enter No of rotations: ")

for i in range(0,int(n)):

a=input("Element: ")

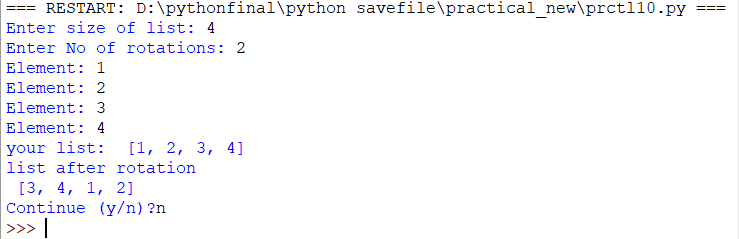
lst.append(int(a))

print("your list: ",lst)

print("list after rotation \n",rotatelist(lst,int(k)))

ch=input("Continue (y/n)?")

output:-



**Practical 11:**

Write a python function ascending(I) that returns True if each in its input list is atleast as big as the one before it

Eg:ascending([])  
True  
Ascending([3,3,4])

program:-

ch='y'

def ascending(lst1):

val=""

for i in range(0,len(lst)-1):

if(lst[i]>lst[i+1]):

return False

return True

while ch.lower()=='y':

lst=[]

n=input("Enter size of list: ")

for i in range(0,int(n)):

a=int(input("Element: "))

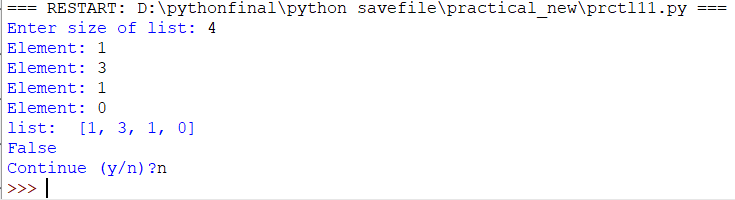
lst.append(a)

print("list: ",lst)

print(ascending(lst))

ch=input("Continue (y/n)?")

output:-



**Practical 12:**

Write a python program to create a dictionary with key as first character and

value as words starting with that character.

The program takes a string and creates a dictionary with key as first character and value as words starting with that character.

example:

Enter string: Python is my most favorite programming language in the entire world

#('e',':' ,['entire])

#('f',':' ,['favorite'])

#('i',':' ,['is','in'])

#('m',':' ,['most','my'])

#'l',':' ,['language'])

#('p',':' ,['programming','Python'])

#('t',':' ,['the'])

#('w',':' ,['world'])

program:-

ch='y'

while ch.lower()=='y':

s=input("Enter String: ")

s\_chr=s.split()

d={}#dictionary blank

for s\_chrs in s\_chr:

if(s\_chrs[0] not in d.keys()):

d[s\_chrs[0].lower()]=[]

d[s\_chrs[0].lower()].append(s\_chrs)

else:

if(s\_chrs[0] in d.keys()):

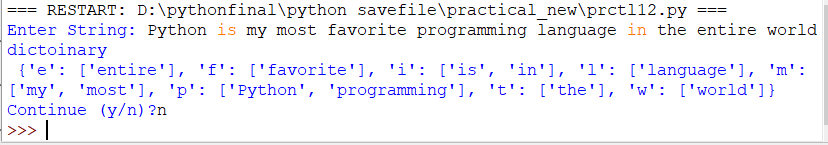
d[s\_chrs[0].lower()].append(s\_chrs)

dnew = dict( sorted(d.items(), key=lambda x: x[0].lower()) )

print("dictionary \n",dnew)

ch=input("Continue (y/n)?")

output:-



**Practical 13:**

Write a python program to read item information (id, name, price, qty) in file “item.dat”.Write a menu driven program to perform the following operations using Random access file.  
1) Search for specific item by name.  
2) Find costliest item.

3) Display all item and total cost.

program:-

ch="y"

while ch=="y":

print("Select appropriate option")

print("1. Search for specific item")

print("2. Find the costliest Item")

print("3. Display All items and total cost")

otp=int(input("Enter choice"))

def DisplayAll():

f1=open("D:\python savefile\practicals\item.dat","r")

print(f1.read())

f1=open("D:\python savefile\practicals\item.dat","r")

x=True

sum1=0

while x:

f2=f1.readline()

if not f2:

x=False

else:

l1=[]

l1=f2.split(",")

sum1+=int(l1[2])

print("Total price of all items: ",sum1)

f1.close()

def dall():

f1=open("D:\python savefile\practicals\item.dat","r")

a=[]

b=[]

x=[]

y=[]

r=True

s=input("enter item to be searched")

while r:

f2=f1.readline()

if not f2:

r=False

else:

l1=[]

l1=f2.split(",")

x.append(l1)

for i in range(0,len(x)):

for j in range(0,len(x[i])):

if s in x[i]:

print(x[i],end=" ")

break

def CostliestItems():

f1=open("D:\python savefile\practicals\item.dat","r")

x=True

y=[]

max1=0

while x:

f2=f1.readline()

if not f2:

x=False

else:

l1=[]

l1=f2.split(",")

y.append(l1)

if(int(l1[2])>max1):

max1=int(l1[2])

print("Costliest Item is : - ",max1)

for i in range(0,len(y)):

for j in range(0,len(y[i])):

if str(max1) in y[i]:

print(y[i],end=" ")

break

if(otp==1):

dall()

elif(otp==2):

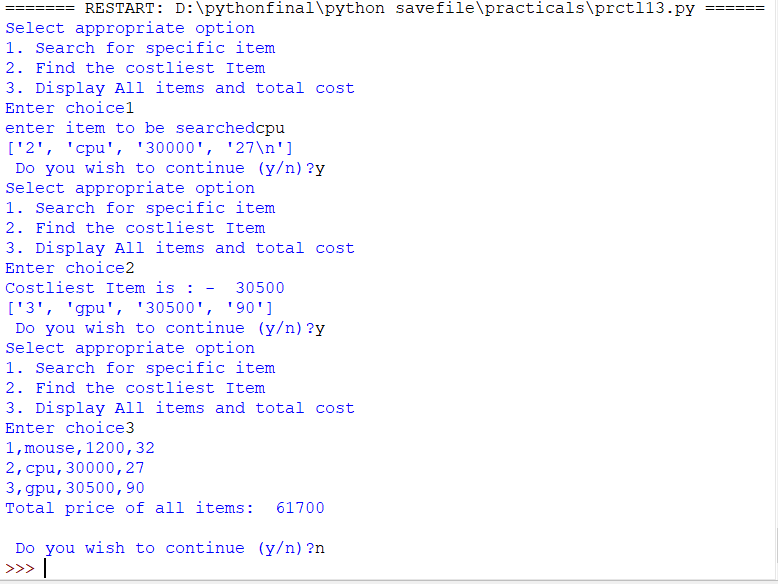
CostliestItems()

elif(otp==3):

DisplayAll()

ch=input("\n Do you wish to continue (y/n)?")

output:-



**Practical 14:**

Write a python program that finds the largest file in the given directory.

program:-

import os

ch='y'

while ch.lower()=='y':

directory=input("Enter Directory: ")

lis=os.listdir(directory)

fname={}

fsize=0

for i in lis:

fname[os.stat(i).st\_size]=i

for j in fname:

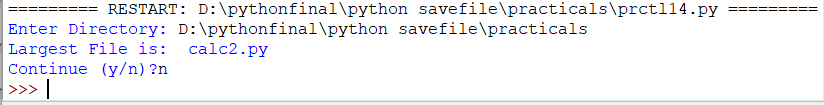
if(int(j)>fsize):

fsize=j

print("Largest File is: ",fname[fsize])

ch=input("Continue (y/n)?")

output:-



**Practical 15:**

Write a python program to display files under specific directory.

program:-

import os

ch='y'

while ch.lower()=='y':

directory=input("Enter the Directory: ")

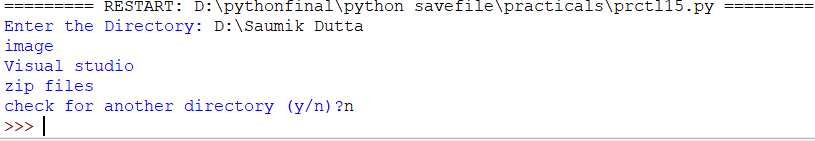
#print(os.listdir(directory),end=" ")

for i in os.listdir(directory):

print(i)

ch=input("check for another directory (y/n)?")

output:-



**Practical 16:**

Write a python GUI program to design scientific calculator.

program:-

from tkinter import \*

import tkinter.messagebox

import math

root = Tk()

root.geometry("650x400+300+300")

#root.iconbitmap(True, "images/icon.ico")

root.title("Scientific Calculator")

switch = None

# Button on press

def btn1\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '1')

def btn2\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '2')

def btn3\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '3')

def btn4\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '4')

def btn5\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '5')

def btn6\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '6')

def btn7\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '7')

def btn8\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '8')

def btn9\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '9')

def btn0\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, '0')

def key\_event(\*args):

if disp.get() == '0':

disp.delete(0, END)

def btnp\_clicked():

pos = len(disp.get())

disp.insert(pos, '+')

def btnm\_clicked():

pos = len(disp.get())

disp.insert(pos, '-')

def btnml\_clicked():

pos = len(disp.get())

disp.insert(pos, '\*')

def btnd\_clicked():

pos = len(disp.get())

disp.insert(pos, '/')

def btnc\_clicked(\*args):

disp.delete(0, END)

disp.insert(0, '0')

def sin\_clicked():

try:

ans = float(disp.get())

if switch is True:

ans = math.sin(math.radians(ans))

else:

ans = math.sin(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def cos\_clicked():

try:

ans = float(disp.get())

if switch is True:

ans = math.cos(math.radians(ans))

else:

ans = math.cos(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def tan\_clicked():

try:

ans = float(disp.get())

if switch is True:

ans = math.tan(math.radians(ans))

else:

ans = math.tan(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def arcsin\_clicked():

try:

ans = float(disp.get())

if switch is True:

ans = math.degrees(math.asin(ans))

else:

ans = math.asin(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def arccos\_clicked():

try:

ans = float(disp.get())

if switch is True:

ans = math.degrees(math.acos(ans))

else:

ans = math.acos(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def arctan\_clicked():

try:

ans = float(disp.get())

if switch is True:

ans = math.degrees(math.atan(ans))

else:

ans = math.atan(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def pow\_clicked():

pos = len(disp.get())

disp.insert(pos, '\*\*')

def round\_clicked():

try:

ans = float(disp.get())

ans = round(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def logarithm\_clicked():

try:

ans = float(disp.get())

ans = math.log10(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def fact\_clicked():

try:

ans = float(disp.get())

ans = math.factorial(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def sqr\_clicked():

try:

ans = float(disp.get())

ans = math.sqrt(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def dot\_clicked():

pos = len(disp.get())

disp.insert(pos, '.')

def pi\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, str(math.pi))

def e\_clicked():

if disp.get() == '0':

disp.delete(0, END)

pos = len(disp.get())

disp.insert(pos, str(math.e))

def bl\_clicked():

pos = len(disp.get())

disp.insert(pos, '(')

def br\_clicked():

pos = len(disp.get())

disp.insert(pos, ')')

def del\_clicked():

pos = len(disp.get())

display = str(disp.get())

if display == '':

disp.insert(0, '0')

elif display == ' ':

disp.insert(0, '0')

elif display == '0':

pass

else:

disp.delete(0, END)

disp.insert(0, display[0:pos-1])

def conv\_clicked():

global switch

if switch is None:

switch = True

conv\_btn['text'] = "Deg"

else:

switch = None

conv\_btn['text'] = "Rad"

def ln\_clicked():

try:

ans = float(disp.get())

ans = math.log(ans)

disp.delete(0, END)

disp.insert(0, str(ans))

except Exception:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

def mod\_clicked():

pos = len(disp.get())

disp.insert(pos, '%')

def btneq\_clicked(\*args):

try:

ans = disp.get()

ans = eval(ans)

disp.delete(0, END)

disp.insert(0, ans)

except:

tkinter.messagebox.showerror("Value Error", "Check your values and operators")

# Label

disp = Entry(root, font="Verdana 20", fg="black", bg="#abbab1", bd=0, justify=RIGHT, insertbackground="#abbab1", cursor="arrow")

disp.bind("<Return>", btneq\_clicked)

disp.bind("<Escape>", btnc\_clicked)

disp.bind("<Key-1>", key\_event)

disp.bind("<Key-2>", key\_event)

disp.bind("<Key-3>", key\_event)

disp.bind("<Key-4>", key\_event)

disp.bind("<Key-5>", key\_event)

disp.bind("<Key-6>", key\_event)

disp.bind("<Key-7>", key\_event)

disp.bind("<Key-8>", key\_event)

disp.bind("<Key-9>", key\_event)

disp.bind("<Key-0>", key\_event)

disp.bind("<Key-.>", key\_event)

disp.insert(0, '0')

disp.focus\_set()

disp.pack(expand=TRUE, fill=BOTH)

# Row 1 Buttons

btnrow1 = Frame(root, bg="#000000")

btnrow1.pack(expand=TRUE, fill=BOTH)

pi\_btn = Button(btnrow1, text="π", font="Segoe 18", relief=GROOVE, bd=0, command=pi\_clicked, fg="white", bg="#333333")

pi\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

fact\_btn = Button(btnrow1, text=" x! ", font="Segoe 18", relief=GROOVE, bd=0, command=fact\_clicked, fg="white", bg="#333333")

fact\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sin\_btn = Button(btnrow1, text="sin", font="Segoe 18", relief=GROOVE, bd=0, command=sin\_clicked, fg="white", bg="#333333")

sin\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

cos\_btn = Button(btnrow1, text="cos", font="Segoe 18", relief=GROOVE, bd=0, command=cos\_clicked, fg="white", bg="#333333")

cos\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

tan\_btn = Button(btnrow1, text="tan", font="Segoe 18", relief=GROOVE, bd=0, command=tan\_clicked, fg="white", bg="#333333")

tan\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn1 = Button(btnrow1, text="1", font="Segoe 23", relief=GROOVE, bd=0, command=btn1\_clicked, fg="white", bg="#333333")

btn1.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn2 = Button(btnrow1, text="2", font="Segoe 23", relief=GROOVE, bd=0, command=btn2\_clicked, fg="white", bg="#333333")

btn2.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn3 = Button(btnrow1, text="3", font="Segoe 23", relief=GROOVE, bd=0, command=btn3\_clicked, fg="white", bg="#333333")

btn3.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnp = Button(btnrow1, text="+", font="Segoe 23", relief=GROOVE, bd=0, command=btnp\_clicked, fg="white", bg="#333333")

btnp.pack(side=LEFT, expand=TRUE, fill=BOTH)

# Row 2 Buttons

btnrow2 = Frame(root)

btnrow2.pack(expand=TRUE, fill=BOTH)

e\_btn = Button(btnrow2, text="e", font="Segoe 18", relief=GROOVE, bd=0, command=e\_clicked, fg="white", bg="#333333")

e\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sqr\_btn = Button(btnrow2, text=" √x ", font="Segoe 18", relief=GROOVE, bd=0, command=sqr\_clicked, fg="white", bg="#333333")

sqr\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

sinh\_btn = Button(btnrow2, text="sin−1", font="Segoe 11 bold", relief=GROOVE, bd=0, command=arcsin\_clicked, fg="white", bg="#333333")

sinh\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

cosh\_btn = Button(btnrow2, text="cos-1", font="Segoe 11 bold", relief=GROOVE, bd=0, command=arccos\_clicked, fg="white", bg="#333333")

cosh\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

tanh\_btn = Button(btnrow2, text="tan-1", font="Segoe 11 bold", relief=GROOVE, bd=0, command=arctan\_clicked, fg="white", bg="#333333")

tanh\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn4 = Button(btnrow2, text="4", font="Segoe 23", relief=GROOVE, bd=0, command=btn4\_clicked, fg="white", bg="#333333")

btn4.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn5 = Button(btnrow2, text="5", font="Segoe 23", relief=GROOVE, bd=0, command=btn5\_clicked, fg="white", bg="#333333")

btn5.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn6 = Button(btnrow2, text="6", font="Segoe 23", relief=GROOVE, bd=0, command=btn6\_clicked, fg="white", bg="#333333")

btn6.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnm = Button(btnrow2, text="-", font="Segoe 23", relief=GROOVE, bd=0, command=btnm\_clicked, fg="white", bg="#333333")

btnm.pack(side=LEFT, expand=TRUE, fill=BOTH)

# Row 3 Buttons

btnrow3 = Frame(root)

btnrow3.pack(expand=TRUE, fill=BOTH)

conv\_btn = Button(btnrow3, text="Rad", font="Segoe 12 bold", relief=GROOVE, bd=0, command=conv\_clicked, fg="white", bg="#333333")

conv\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

round\_btn = Button(btnrow3, text="round", font="Segoe 10 bold", relief=GROOVE, bd=0, command=round\_clicked, fg="white", bg="#333333")

round\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

ln\_btn = Button(btnrow3, text="ln", font="Segoe 18", relief=GROOVE, bd=0, command=ln\_clicked, fg="white", bg="#333333")

ln\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

logarithm\_btn = Button(btnrow3, text="log", font="Segoe 17", relief=GROOVE, bd=0, command=logarithm\_clicked, fg="white", bg="#333333")

logarithm\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

pow\_btn = Button(btnrow3, text="x^y", font="Segoe 17", relief=GROOVE, bd=0, command=pow\_clicked, fg="white", bg="#333333")

pow\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn7 = Button(btnrow3, text="7", font="Segoe 23", relief=GROOVE, bd=0, command=btn7\_clicked, fg="white", bg="#333333")

btn7.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn8 = Button(btnrow3, text="8", font="Segoe 23", relief=GROOVE, bd=0, command=btn8\_clicked, fg="white", bg="#333333")

btn8.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn9 = Button(btnrow3, text="9", font="Segoe 23", relief=GROOVE, bd=0, command=btn9\_clicked, fg="white", bg="#333333")

btn9.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnml = Button(btnrow3, text="\*", font="Segoe 23", relief=GROOVE, bd=0, command=btnml\_clicked, fg="white", bg="#333333")

btnml.pack(side=LEFT, expand=TRUE, fill=BOTH)

# Row 4 Buttons

btnrow4 = Frame(root)

btnrow4.pack(expand=TRUE, fill=BOTH)

mod\_btn = Button(btnrow4, text="%", font="Segoe 21", relief=GROOVE, bd=0, command=mod\_clicked, fg="white", bg="#333333")

mod\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

bl\_btn = Button(btnrow4, text=" ( ", font="Segoe 21", relief=GROOVE, bd=0, command=bl\_clicked, fg="white", bg="#333333")

bl\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

br\_btn = Button(btnrow4, text=" ) ", font="Segoe 21", relief=GROOVE, bd=0, command=br\_clicked, fg="white", bg="#333333")

br\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

dot\_btn = Button(btnrow4, text=" • ", font="Segoe 21", relief=GROOVE, bd=0, command=dot\_clicked, fg="white", bg="#333333")

dot\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btnc = Button(btnrow4, text="C", font="Segoe 23", relief=GROOVE, bd=0, command=btnc\_clicked, fg="white", bg="#333333")

btnc.pack(side=LEFT, expand=TRUE, fill=BOTH)

del\_btn = Button(btnrow4, text="⌫", font="Segoe 20", relief=GROOVE, bd=0, command=del\_clicked, fg="white", bg="#333333")

del\_btn.pack(side=LEFT, expand=TRUE, fill=BOTH)

btn0 = Button(btnrow4, text="0", font="Segoe 23", relief=GROOVE, bd=0, command=btn0\_clicked, fg="white", bg="#333333")

btn0.pack(side=LEFT, expand=TRUE, fill=BOTH)

btneq = Button(btnrow4, text="=", font="Segoe 23", relief=GROOVE, bd=0, command=btneq\_clicked, fg="white", bg="#333333")

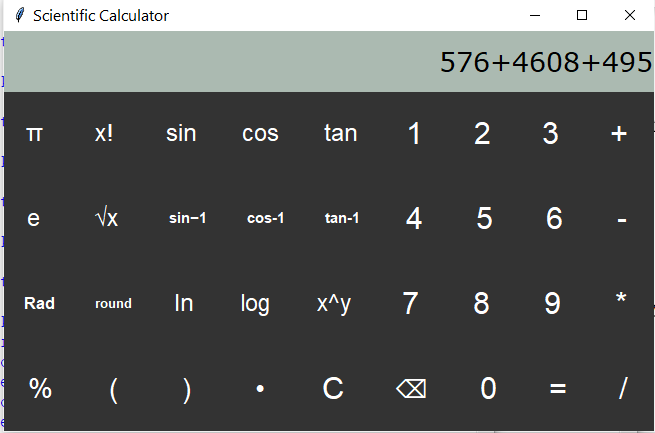
btneq.pack(side=LEFT, expand=TRUE, fill=BOTH)

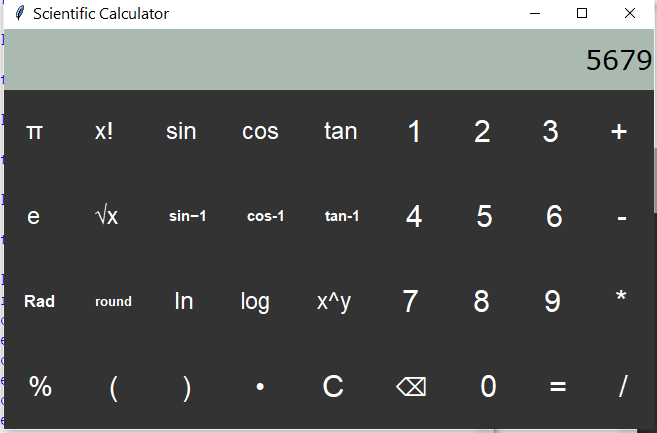
btnd = Button(btnrow4, text="/", font="Segoe 23", relief=GROOVE, bd=0, command=btnd\_clicked, fg="white", bg="#333333")

btnd.pack(side=LEFT, expand=TRUE, fill=BOTH)

root.mainloop()

output:-





**Practical 17:**

Write a python GUI program to design age calculator.

program:-

from tkinter import \*

import tkinter.messagebox

gui=Tk()

def agecalc():

res\_day=Label(gui,text="Day: ")

res\_month=Label(gui,text="Month: ")

res\_year=Label(gui,text="Year: ")

ress=Label(gui,text="YOur Age is :")

m30=[4,6,9,11]

m31=[1,3,5,7,8,10,12]

m28=[2]

mt=[1,2,3,4,5,6,7,8,9,10,11,12]

mnt =[31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

t1day=0

dobday=int(dob\_dd.get())

dobmnth=int(dob\_mm.get())

dobyyyy=int(dob\_yyyy.get())

pday=int(pres\_dd.get())

pmnth=int(pres\_mm.get())

pyear=int(pres\_yyyy.get())

if(dobyyyy>=pyear):

tkinter.messagebox.showerror("Value error","dob cannot be greater than presentday")

elif(dobyyyy<=pyear):

if((dobday<=31 and pday<=31)and (dobmnth in mt and pmnth in mt)):

rday=pday-dobday

rmnth=pmnth-dobmnth

ryear=pyear-dobyyyy

if((dobday<=mnt[dobmnth])and (pday<=mnt[pmnth])):

if(dobmnth>pmnth):

nmnth=12-(dobmnth-pmnth)

ryear=ryear-1

rsd=str(rday)

rsm=str(nmnth)

rsy=str(ryear)

res\_dd=Label(gui,text=rsd.replace('-',''))

res\_mnth=Label(gui,text=rsm.replace('-',''))

res\_yyyy=Label(gui,text=rsy.replace('-',''))

ress.grid(row=11,column=0)

res\_day.grid(row=12,column=0)

res\_dd.grid(row=12,column=1)

res\_month.grid(row=13,column=0)

res\_mnth.grid(row=13,column=1)

res\_year.grid(row=14,column=0)

res\_yyyy.grid(row=14,column=1)

gui.configure(background = "light blue")

gui.geometry("640x800")

gui.title("Age Calculator")

dob=Label(gui,text="Enter Date of Birth",bg="light green")

dob\_day=Label(gui,text="Enter Day: ",bg="light green")

dob\_month=Label(gui,text="Enter Month: ",bg="light green")

dob\_year=Label(gui,text="Enter Year: ",bg="light green")

dob\_dd=Entry(gui)

dob\_mm=Entry(gui)

dob\_yyyy=Entry(gui)

dob.grid(row=0, column=0)

pres\_dt=Label(gui,text="Present Day",bg="orange")

pres\_day=Label(gui,text="Enter Day: ",bg="orange")

pres\_month=Label(gui,text="Enter Month: ",bg="orange")

pres\_year=Label(gui,text="Enter Year: ",bg="orange")

pres\_dd=Entry(gui)

pres\_mm=Entry(gui)

pres\_yyyy=Entry(gui)

dob\_day.grid(row=1,column=0)

dob\_dd.grid(row=1,column=1)

dob\_month.grid(row=2,column=0)

dob\_mm.grid(row=2,column=1)

dob\_year.grid(row=3,column=0)

dob\_yyyy.grid(row=3,column=1)

pres\_dt.grid(row=5,column=0)

pres\_day.grid(row=6,column=0)

pres\_dd.grid(row=6,column=1)

pres\_month.grid(row=7,column=0)

pres\_mm.grid(row=7,column=1)

pres\_year.grid(row=8,column=0)

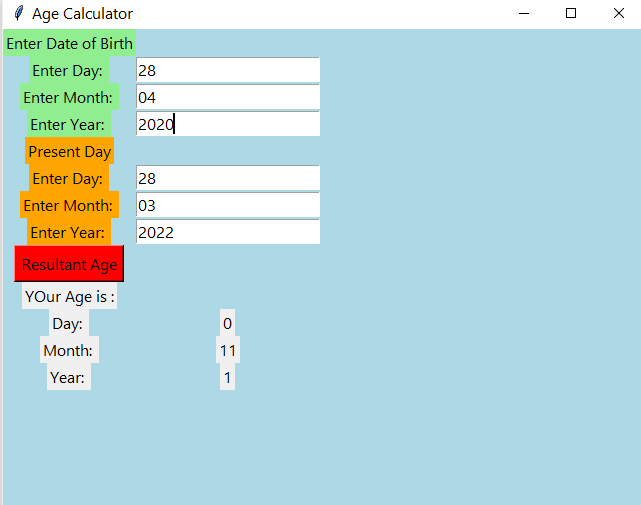
pres\_yyyy.grid(row=8,column=1)

resage = Button(gui, text = "Resultant Age", bg = "Red", command = agecalc)

resage.grid(row=9,column=0)

gui.mainloop()

output:-



**Practical 18:**

Write a python program which accepts two integers and arithmetic operator from the command line and perform the operation.

i) if no of arguments are less than 3 raise:”illegal number of arguments”

ii)if operator is not arithmetic operator raise:”Invalid operator Exception”

iii)if result is negative then raise: “Negative result exception”

program:-

ch='y'

def funop(\*args):

a=args[0]

b=args[1]

op=args[2]

d1=a+b

d2=a-b

d3=a\*b

d4=a/b

d5=a%b

#chkres(d1,d2,d3,d4,d5)

if(len(args)==3):

if(op=='+' and d1>0):

print("a+b:",d1)

elif(op=='-' and d2>0):

print("a-b:",d2)

elif(op=='\*' and d3>0):

print("a\*b:",d3)

elif(op=='/' and d4>0):

print("a/b:",d4)

elif(op=='%' and d5>0):

print("a%b:",d5)

else:

chkres(d1,d2,d3,d4,d5)

else:

raise Exception("Illegal No of Args")

def chkres(d1,d2,d3,d4,d5):

if(d1<0 or d2<0 or d3<0 or d4<0 or d5<0 ):

raise Exception("Negative result Exception")

else:

pass

while ch.lower()=='y':

x=int(input("Enter value for num1: "))

y=int(input("Enter value for num2: "))

op=input("Enter Operator: ")

lst\_op=['+','-','%','\*','/']

if(op in lst\_op):

funop(x,y,op)

else:

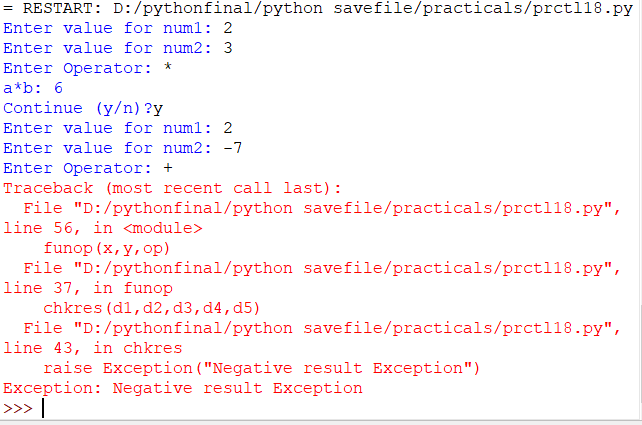
raise Exception("Invalid Operator")

ch=input("Continue (y/n)?")

#funop(x,y,op,"arg4")

#ch=input("Continue (y/n)?")

output:-



**Practical 19:**

Write a python program which contains method”displayColor” which take one character argument. Raise an error if the character is not an alphabet.If it is any other alphabet,report an error.

program:-

ch='y'

class colors:

def displaycolor(self,str1):

rainbow={'v':'voilet',

'i':'indigo',

'b':'blue',

'g':'green',

'y':'yellow',

'o':'orange',

'r':'red'}

if(str1.lower() in rainbow.keys()):

print(rainbow[str1.lower()])

else:

raise KeyError("Color not found")

while ch.lower()=='y':

str1=input("Enter Character: ")

color1=colors()

if((len(str1)==1 ) and(str1.isalpha()==True)) :

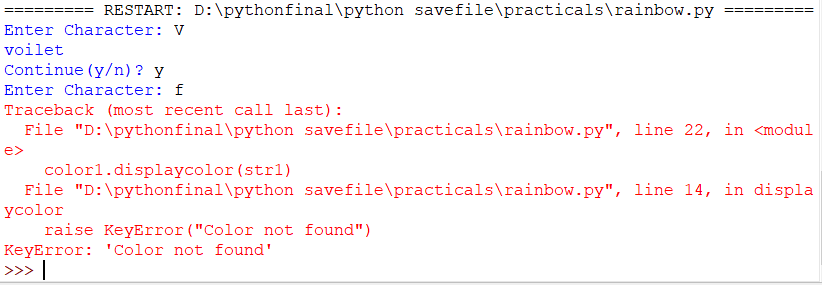
color1.displaycolor(str1)

ch=input("Continue(y/n)? ")

else:

raise Exception("only Alphabets are allowed")

output:-



**Practical 20:**

Write a python program to check the validity of password input by users.Validation atleast 1 letter between[a-z]and 1 leeter between [A-Z]atleast 1 letter 1 number[0-9] and alteast 1 character from[@,#,etc],minimum length of 6 character and maximum length of 16 characters

program:-

import re

ch='y'

while ch.lower()=='y':

str1=input("Enter Your String: ")

p1 = '[A-Z]'

p2='[a-z]'

p3 = '[0-9]'

p4='[$#@]'

n=len(str1)

if(n>=6 and n<=16 ):

if((re.search(p1,str1)) and (re.search(p2,str1)) and (re.search(p3,str1)) and (re.search(p4,str1))):

print("Valid Password")

else:

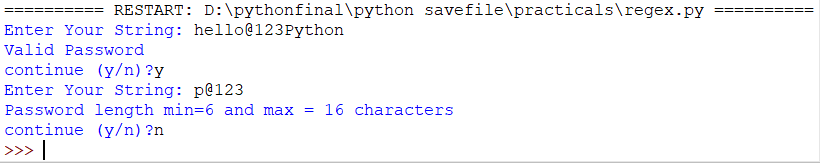
print("Passsword must contain 1 uppercase,1 lowecase alphabets,1 numeric and 1 special character")

else:

print("Password length min=6 and max = 16 characters")

ch=input("continue (y/n)?")

output:-



**Practical 21:**

Create abstract class shape with two abstract methods calculate the shape areas double abstract,double area,double perimeter .Write the following subclasses for shape class circle,rectangle,triangle, and rhombus.All the classes must implement abstract method inheritance of parent class.

program:-

from abc import ABC, abstractmethod

import math

ch='y'

class shape(ABC):

@abstractmethod

def area(self,a,b):

pass

@abstractmethod

def perimeter(self,a,b):

pass

class rectangle(shape):

def area(self,a,b):

r=a\*b

print("area of rectangle: ",r)

def perimeter(self,a,b):

s=a+b

r=2\*s

print("perimeter:",r)

class triangle(shape):

def area(self,a,b):

s=a\*b

r=0.5\*s

print("area of triangle: ",r)

def perimeter(self,a,b):

c=int(input("Enter Length for third side of traingle"))

s=a+b+c

print("perimeter:",s)

class circle(shape):

def area(self,a,b):

s=a\*b

r=3.14\*s\*s

print("area of circle: ",r)

def perimeter(self,a,b):

s=3.14\*a\*b

r=2\*s

print("perimeter:",r)

class rhombus(shape):

def area(self,a,b):

s=a\*b

r=s/2

print("area of rhombus: ",r)

def perimeter(self,a,b):

x=(a\*a)+(b\*b)

r=2\*math.sqrt(x)

print("perimeter:",r)

while ch.lower()=='y':

print("Select option to search from below \n")

print("1. rectangle ")

print("2. triangle")

print("3. circle")

print("4. rhombus")

print("5. show all")

d1=rectangle()

d2=triangle()

d3=circle()

d4=rhombus()

otp=int(input("Enter option"))

if(otp==1):

a=int(input("Enter length of rectangle: "))

b=int(input("Enter breadth of rectangle: "))

d1.area(a,b)

d1.perimeter(a,b)

elif(otp==2):

a=int(input("Enter base of triangle: "))

b=int(input("Enter height of traingle: "))

d2.area(a,b)

d2.perimeter(a,b)

elif(otp==3):

a=int(input("Enter radius of circle: "))

d3.area(a,1)

d3.perimeter(a,1)

elif(otp==4):

a=int(input("Enter d1 of rhombus: "))

b=int(input("Enter d2 of rhombus: "))

#c=int(input("Enter side of rhombus: "))

d4.area(a,b)

d4.perimeter(a,b)

elif(otp==5):

a1=int(input("Enter length of rectangle: "))

a2=int(input("Enter breadth of rectangle: "))

b1=int(input("Enter base of triangle: "))

b2=int(input("Enter height of traingle: "))

c1=int(input("Enter radius of circle: "))

e1=int(input("Enter d1 of rhombus: "))

e2=int(input("Enter d2 of rhombus: "))

#e3=int(input("Enter side of rhombus: "))

d1.area(a1,a2)

d1.perimeter(a1,a2)

d2.area(b1,b2)

d2.perimeter(b1,b2)

d3.area(c1,1)

d3.perimeter(c1,1)

d4.area(e1,e2)

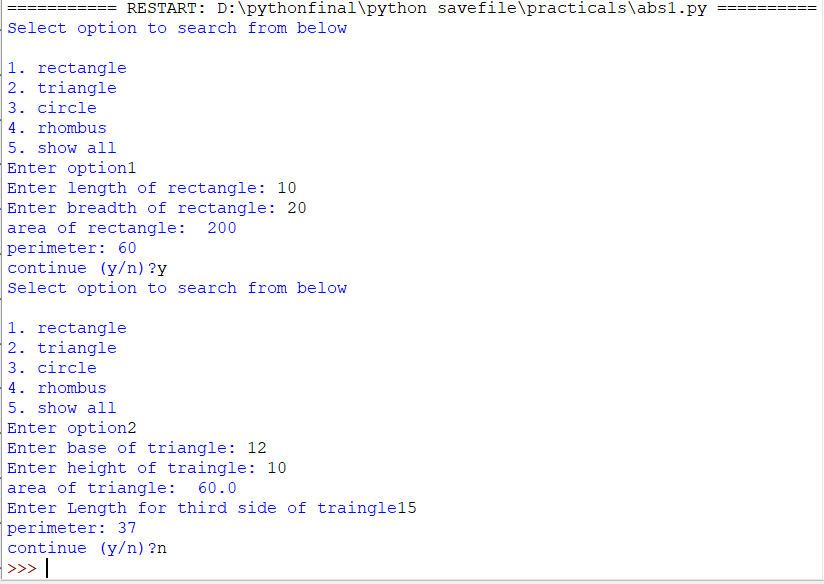
d4.perimeter(e1,e2)

else:

print("please select suitable option")

ch=input("continue (y/n)?")

output:-



**Practical 22:**

A bank given simple interest 9.25% per annum for deposit mode in bank it also give compound interest 8.5% for deposit.Acustomer would like to deposit 25000 for 5 years write a python program to implement interface

program:-

from abc import ABC, abstractmethod

ch='y'

class myclass(ABC):

@abstractmethod

def si(self,p,r,n):

pass

@abstractmethod

def ci(self,p,r,n):

pass

class simple(myclass):

def si(self,p,r,n):

rsi=(p\*r\*n)/100

print("Simple Interest ",rsi)

def ci(self,p,r,n):

pass

class compound(myclass):

def si(self,p,r,n):

pass

def ci(self,p,r,n):

c=p \* (pow((1 + r / 100), n))

print("Compound Interest ",c)

while ch.lower()=='y':

p=int(input("Enter principle ammount"))

r1=9.25

r2=8.5

#=int(input("Enter rate of interest"))

n=int(input("Enter Time in years"))

print("Select option to search from below \n")

print("1. simple Interest \n 2. compound Interest \n 3. show both")

d1=simple()

d2=compound()

otp=int(input("Enter option: "))

if(otp==1):

d1.si(p,r1,n)

#d1.ci(p,r2,n)

elif(otp==2):

#d2.si(p,r1,n)

d2.ci(p,r2,n)

elif(otp==3):

d1.si(p,r1,n)

#d1.ci(p,r2,n)

#d2.si(p,r,n)

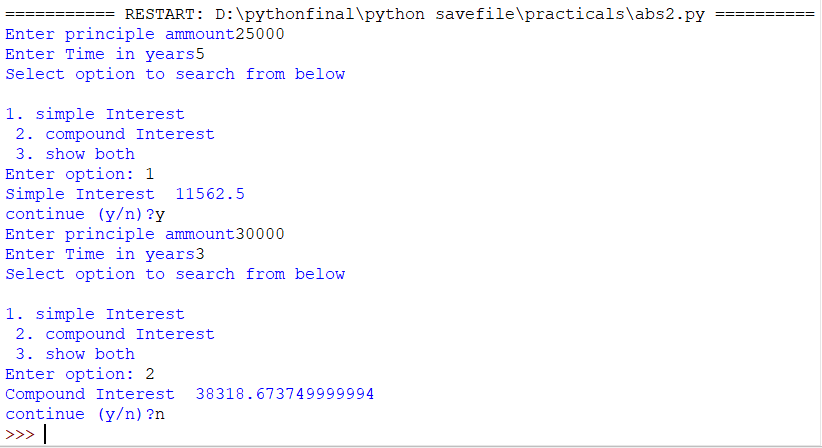
d2.ci(p,r2,n)

else:

print("please select suitable option")

ch=input("continue (y/n)?")

output:-



**Practical 23:**

write a python program to demonstrate three technique to implement multithreading

program:-

import time

from threading import Thread

def myfunc(i):

print("Sleeping 5 sec from thread %d\n" %i)

time.sleep(5)

print("Finished sleeping from thread %d\n" %i)

print("Thread without using a Class")

for i in range(10):

t=Thread(target=myfunc,args=(i,))

t.start()

print("1 method multithreading is completed")

class MyThread(Thread):

def run(self):

for i in range(1,6):

print("\n Thread %i going to sleep for 5 second." %i)

time.sleep(5)

print("\n Thread %i is awake now" %i)

t1=MyThread()

print("Creating thread using subclass")

t1.start()

t1.join()

print("2 method multithreading is completed")

class MyThread1:

def sleep(self):

for i in range(1,6):

print("Thread %i going to sleep for 5 seconds." %i)

time.sleep(5)

print("Thread %i is awake now" %i)

obj=MyThread1()

t2=MyThread1(target=obj.sleep())

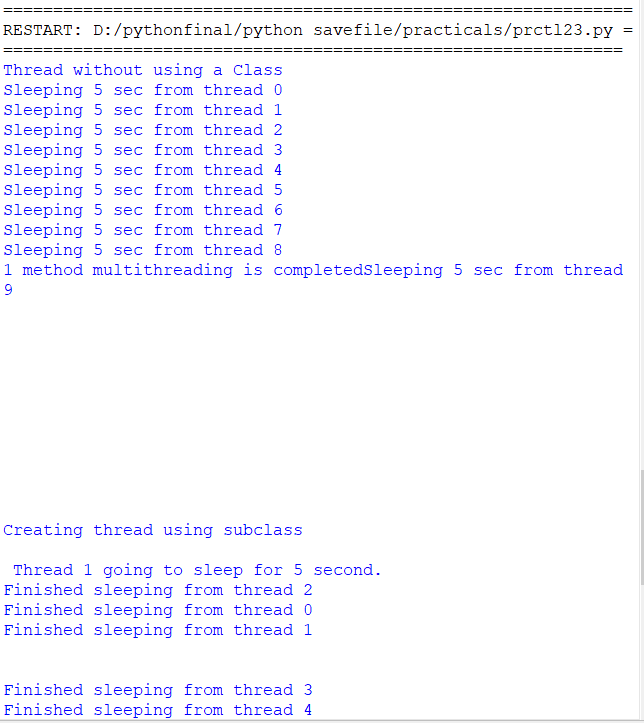
print("Creating thread without using subclass")

t2.start()

t2.join()

print("3 method multithreading is complete")

output:-



**Practical 24:**

write a python program to demonstrate multithreading through shared value

program:-

import threading

x=0

COUNT=1000000

def foo():

global x

for i in range(COUNT):

x+=1

def fbar():

global x

for i in range(COUNT):

x=-1

t1=threading.Thread(target=foo)

t2=threading.Thread(target=fbar)

t1.start()

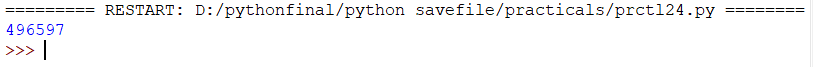
t2.start()

t1.join()

t2.join()

print(x)

output:-



**Practical 25:**

write a python program to implement semaphores in multithreading

program:-

import threading

import time

done=threading.Semaphore(0)

item=None

def producer():

global item

print("I m the producer and I produce data")

print("Producer is going to sleep")

time.sleep(5)

item="Hello"

print("Producer is alive.Signalling the consumer:")

done.release()

def consumer():

print(" I am a consumer and I wait for data")

print("Consumer is waiting")

done.acquire()

print("Consumer got",item)

t1=threading.Thread(target=producer)

t2=threading.Thread(target=consumer)

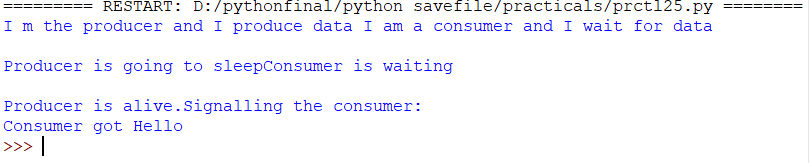
t1.start()

t2.start()

t1.join()

t2.join()

output:-



**Practical 26:**

write a python program to illustrate database & table creation using sqlite3 and mysql

program:-

sqlite3

import sqlite3

conn=sqlite3.connect('test.db')

conn.execute('''CREATE TABLE COMPANY1 (ID INT PRIMARY KEY NOT NULL,NAME TEXT NOT NULL,AGE INT NOT NULL,ADDRESS CHAR(50),SALARY REAL);''')

print("Table created")

conn.close()

mysql

import mysql.connector as mysql

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host=cred[0],user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

mcursor.execute('''CREATE TABLE COMPANY

(ID INT PRIMARY KEY NOT NULL,

NAME TEXT NOT NULL,

AGE INT NOT NULL,

ADDRESS CHAR(50),

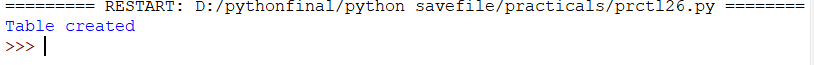
SALARY REAL);''')

print('table created successfully')

mydb.close()

output:-

sqlite3



mysql



**Practical 27:**

write a python program to insert n no of records using sqlite3 and mysql

program:-

sqlite3

import sqlite3

conn=sqlite3.connect('saumik.db')

print("open database successfully")

n=int(input("Enter how many records you want to insert"))

for i in range(0,n):

a=int(input("ID: "))

b=input("NAME: ")

c=int(input("Age: "))

d=input("Address: ")

e=int(input("Salary; "))

print("\n before insertion")

cursor=conn.execute("SELECT id,name,age,address,salary from company")

for row in cursor:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

str1="Insert into COMPANY(id,name,age,address,salary)values('"+ str(a) +"','"+ b+ "','"+ str(c) +"','"+ d +"','"+ str(e)+"')"

conn.execute(str1)

conn.commit()

print("Records inserted successfully")

print("\n after insertion")

cursor=conn.execute("SELECT id,name,age,address,salary from company")

for row in cursor:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

conn.close()

mysql

import mysql.connector as mysql

import mysql.connector as mysql

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host="localhost",user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

def beforeins():

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host=cred[0],user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

mcursor.execute("SELECT id,name,age,addr,salary from emp")

mcur=mcursor.fetchall()

for row in mcur:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

rc=[]

n=int(input("Enter how many records you want to insert"))

for i in range(0,n):

a=int(input("ID: "))

b=input("NAME: ")

c=int(input("Age: "))

d=input("Address: ")

e=int(input("Salary; "))

val1=(a,b,c,d,e)

rc.append(val1)

print("\n before insertion")

beforeins()

str1="Insert into emp(id,name,age,addr,salary) values(%s,%s,%s,%s,%s)"

#print(rc)

if len(rc)>1:

mcursor.executemany(str1,rc)

else:

mcursor.execute(str1,rc[0])

mydb.commit()

print(mcursor.rowcount,"Records inserted successfully")

print("\n after insertion")

beforeins()

mcursor.close()

Output:-

sqlite3



mysql



**Practical 28:**

write a python program to illustrate retrieve n no of records using sqlite3 and mysql

program:-

sqlite3

import sqlite3

conn=sqlite3.connect('saumik.db')

print("open database successfully")

cursor=conn.execute("SELECT id,name,age,address,salary from company")

for row in cursor:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

print("Operation done successfully")

conn.close()

mysql

import mysql.connector as mysql

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host=cred[0],user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

mcursor.execute("SELECT id,name,age,addr,salary from emp")

mcur=mcursor.fetchall()

print("records\n")

for row in mcur:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

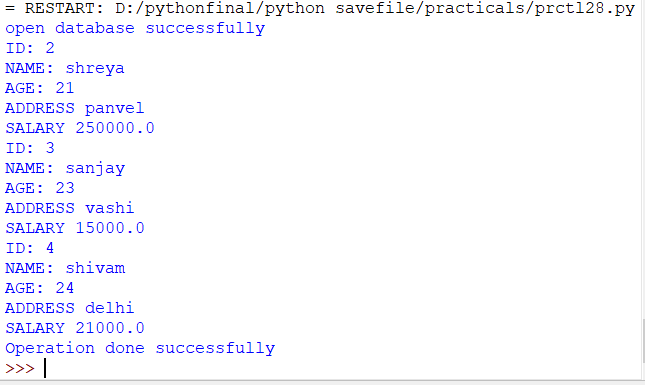
print("ADDRESS",row[3])

print("SALARY",row[4])

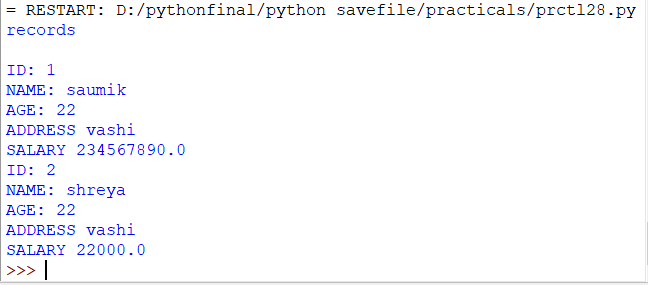
mcursor.close()

output:-

sqlite3



mysql



**Practical 29:**

write a python program to illustrate updation of records using sqlite3 and mysql

program:-

sqlite3

import sqlite3

import sys

def beforeupd():

conn=sqlite3.connect('saumik.db')

print("open database successfully")

cursor=conn.execute("SELECT id,name,age,address,salary from company")

for row in cursor:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

print("Operation done successfully")

try:

con=sqlite3.connect('saumik.db')

cursor=con.cursor()

id=int(input("Enter id:"))

name=input("Enter the name:")

print("\n before update")

beforeupd()

cursor.execute("update company set name='"+name+"' where id = '"+str(id)+"' ")

con.commit()

print("\n after update")

beforeupd()

finally:

con.close()

mysql

import mysql.connector as mysql

import mysql.connector as mysql

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host="localhost",user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

def beforeins():

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host=cred[0],user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

mcursor.execute("SELECT id,name,age,addr,salary from emp")

mcur=mcursor.fetchall()

for row in mcur:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

s\_id=int(input("Enter id:"))

name=input("Enter the name:")

print("\n before updation")

beforeins()

strupd="update emp set name = %s where id =%s"

val=(name,s\_id)

mcursor.execute(strupd,val)

mydb.commit()

print("\n after updation")

beforeins()

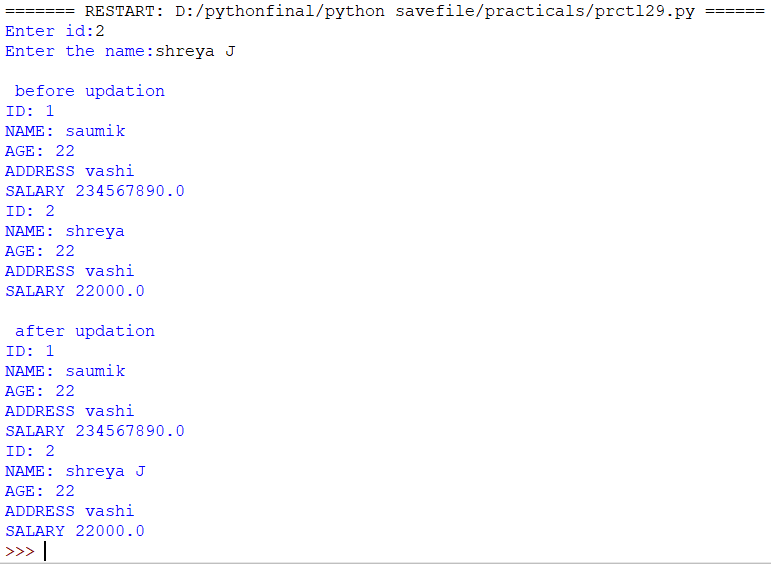
mydb.close()

output:-

sqlite3



mysql



**Practical 30:**

write a python program to deletion of records using sqlite3 and mysql

program:-

sqlite3

import sqlite3

import sys

def beforeupd():

conn=sqlite3.connect('saumik.db')

print("open database successfully")

cursor=conn.execute("SELECT id,name,age,address,salary from company")

for row in cursor:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

print("Operation done successfully")

try:

con=sqlite3.connect('saumik.db')

cursor=con.cursor()

id=int(input("Enter id:"))

print("\n before delete")

beforeupd()

#name=input("Enter the name:")

cursor.execute("delete from company where id = '"+str(id)+"' ")

print("Record Delete successfully")

con.commit()

print("\n after delete")

beforeupd()

finally:

con.close()

mysql

import mysql.connector as mysql

import mysql.connector as mysql

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host="localhost",user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

def beforeins():

cred=["localhost","root","","saumik"]

mydb=mysql.connect(host=cred[0],user=cred[1],password="",database=cred[3])

mcursor=mydb.cursor()

mcursor.execute("SELECT id,name,age,addr,salary from emp")

mcur=mcursor.fetchall()

for row in mcur:

print("ID:",row[0])

print("NAME:",row[1])

print("AGE:",row[2])

print("ADDRESS",row[3])

print("SALARY",row[4])

n=int(input("enter id: "))

print("\n before deletion")

beforeins()

str1="delete from emp where id= %s"

val=(n,)

mcursor.execute(str1,val)

mydb.commit()

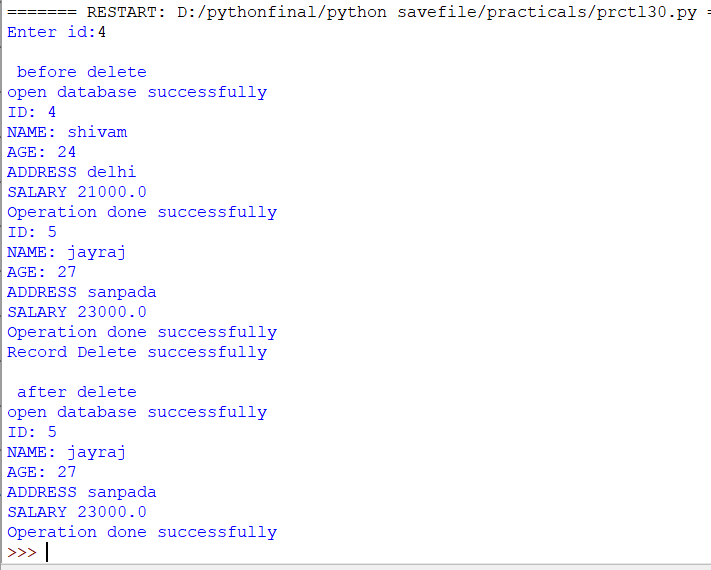
print("\n after deletion")

beforeins()

mydb.close()

output:-

sqlite3



mysql

