**Program 1(A)**

**Aim: Write a python program to implement the following using recursion.**

**Factorial of given number.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Mar 1 07:36:12 2024 @author: it283

"""

def fact(n):

if n==0:

return 1

elif n==1:

return 1

else:

return n\*fact(n-1)

while True:

n=int(input("Enter N value" ))

if n<0:

print("please Enter positive number")

else:

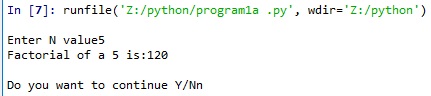
print("Factorial of a {} is:{}".format(n,fact(n)))

c=input("Do you want to continue Y/N")

if c=='N' or c=='n':

break

**Output:**



**Program 1(b)**

**b) Aim: Write a python program to implement Fibonacci series up to n using recursion**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Mar 1 07:46:19 2024@author: it283

"""

def fib(n):

if n==0:

return 0

elif n==1:

return 1

else:

return(fib(n-1)+fib(n-2))

while True:

n=int(input("enter N number"))

print("fibnocci series is:")

for i in range(n):

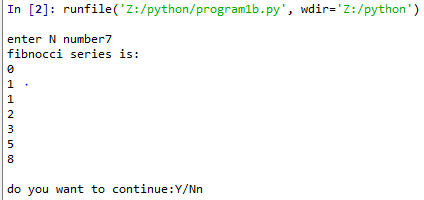
print(fib(i))

ch=input("do you want to continue:Y/N")

if ch=='y'or ch=='n':

break

**Output:**



**Program 1(c)**

**c) Aim: write a python program to print sum of digits in a given number.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Mar 1 07:54:33 2024@author: it283

"""

def sum(n):

if n==0:

return 0

else:

return(n%10+sum(int(n/10)))

while True:

n=int(input("enter N value"))

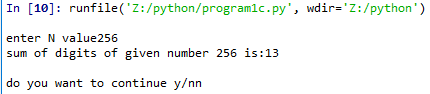
print("sum of digits of given number {} is:{}".format(n,sum(n)))

c=input("do you want to continue y/n")

if c=='N' or c=='n':

break

**Output:**



**Program 2(a)**

**Aim: write a python program to print dates 30 days before and after the given date.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Mar 15 03:06:06 2024@author: it283

"""

while True:

from datetime import datetime,timedelta

s=input("enter any date(dd:mm:yyyy):")

d=datetime.strptime(s,"%d:%m:%Y")

print("given date is:",d)

d1=d.date()

d2=d.date()

print("before given date is:")

for i in range(30):

d2-=timedelta(1)

print(d2)

print("after given date is:")

for i in range(30):

d1+=timedelta(1)

print(d1)

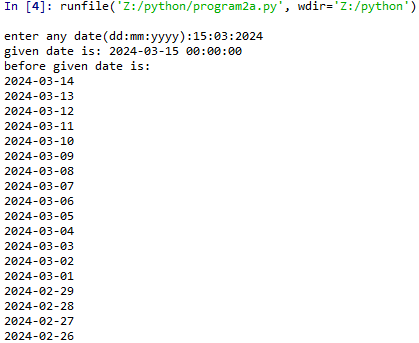
c=input("do you want to continue:Y/N")

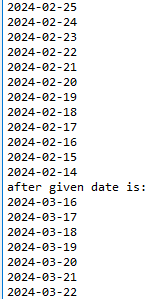
if c=='N' or c=='n':

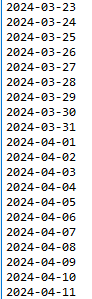
print("thank you,visit again")

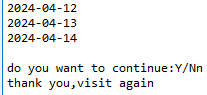
break

**Output:**









**Program 2(b)**

**Aim: Write a python program create 12 fixed dates from a specified date where the difference between to dates is 20.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Mar 15 04:18:21 2024@author: it283

"""

while True:

from datetime import datetime,timedelta

s=input("enter any date(dd-mm-yyyy):")

d=datetime.strptime(s,"%d-%m-%Y")

print("given date is:",d.date())

t1=20

for i in range(12):

t=d.date()+timedelta(days=t1)

t1=t1+20

print(t)

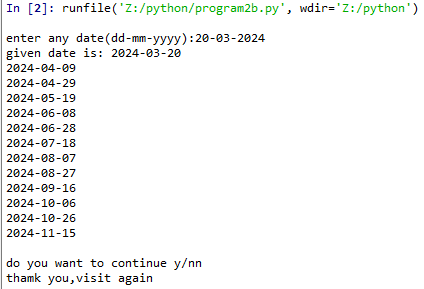
c=input("do you want to continue y/n")

if c=='N' or c=='n':

print("thamk you,visit again")

break

**Output:**



**Program 2(c)**

**Aim: Write a python program to calculate your age based on the given date or from the current date.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Tue Mar 19 04:25:31 2024@author: it283

"""

from datetime import datetime,date

a=input("enter date of birth:(dd:mm:yyyy)")

d=datetime.strptime(a,"%d:%m:%Y")

print("dob is:",d)

a1=input("enter current date:(dd:mm:yyyy)")

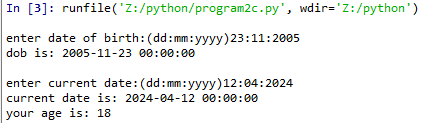
d1=datetime.strptime(a1,"%d:%m:%Y")

print("current date is:",d1)

a2=((d1-d)/365).days

print("your age is:",a2)

**Output:**



**Program 3(a)**

**Aim: Python program to read a text file named “bec.txt” and display the following:**

**The words start with t and a (or)T and A.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Wed Mar 20 15:57:01 2024@author: it283

"""

import re

def file():

try:

n=int(input("enter how many lines you want to enter into a file: "))

f=open("bec.txt","w")

for i in range(n):

print("enter {} line: ".format(i+1))

s=input()

f.write(s)

f.write(" \n ")

f.close()

f=open("bec.txt","r")

l=f.read()

f.close()

s=l.split(" ")

n=[]

for i in s:

if re.search("^[a,A,t,T]",i):

n.append(i)

else:

continue

except EOFError:

pass

print("list of words stating with a/A/t/T in given file is: ",n)

while True:

file()

ch=input("do you want to continue Y/N: ")

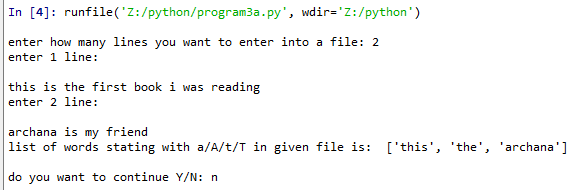
if ch=='Y' or ch=="y":

continue

else:

break

**Output:**



**Program 3(b)**

**Aim: Write a python program to find even length of words in a given file**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Wed Mar 20 16:01:40 2024@author: it283

"""

def file():

n=input("enter file name: ")

try:

f=open(n,"r")

a=[]

l=f.read()

f.close()

s=l.split(" ")

for j in s:

if len(j)%2==0:

a.append(j)

except EOFError:

pass

print("list of even length words in given file is: ",a)

while True:

file()

ch=input("do you want to continue Y/N: ")

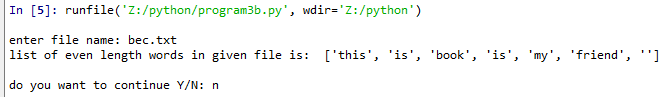
if ch=='Y' or ch=="y":

continue

else:

break

**Output:**



**Program 3(c)**

**Aim: Write a Python program to read a binary file named “book.dat” has structure[ book name , book no., author ,price]**

**i).write a user defined function create file() to input data for a record and add to “book.dat”**

**ii).write a user defined function countRec(author) which accepts author are stored in binary file “book.dat” and return result.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Wed Mar 20 15:43:40 2024@author: it283

"""

import pickle

def createfile():

f=open("book.dat","ab")

name=input("enter book name: ")

bno=int(input("enter book number: "))

author=input("enter author name: ")

price=int(input("enter price of the book: "))

record=[name,bno,author,price]

pickle.dump(record,f)

f.close()

def count():

a=input("enter author name you want to search: ")

f=open("book.dat","rb")

cnt=0

try:

while True:

record=pickle.load(f)

if record[2]==a:

cnt+=1

except EOFError:

pass

print("count= ",cnt)

f.close()

while True:

createfile()

ch=input("do you want to continue Y/N: ")

if ch=='Y' or ch=="y":

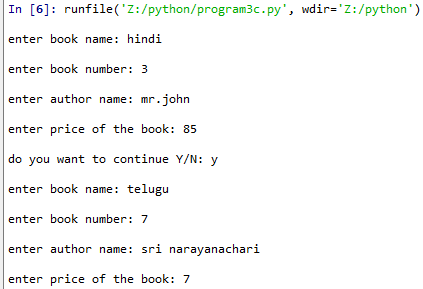
continue

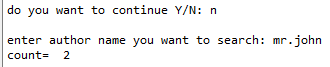
else:

break

count( )

**Output:**





**Program 4(a)**

**Aim: write a python program to implement data stricter using stack using list.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Wed Mar 20 14:06:49 2024@author: it283

"""

class stack:

def \_\_init\_\_(self):

self.ele=0

def push(self):

self.ele=int(input("enter which element you want to insert:"))

list.append(self.ele)

print("Inserted successfully")

def pop(self):

if len(list)==0:

print("stack is empty,you can't delete")

else:

print("deleted element is {}".format(list.pop()))

print("Deleted successfully")

def display(self):

print("stack elements are")

n=len(list)

for i in range(n-1,-1,-1):

print(list[i])

list=[]

ob=stack()

while True:

print("Stack operations")

print("1.Push\n2.Pop\n3.Display\n4.exit\n")

ch=int(input("enter your choice:"))

if ch==1:

ob.push()

elif ch==2:

ob.pop()

elif ch==3:

ob.display()

elif ch==4:

break

else:

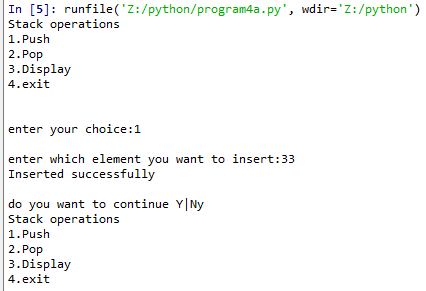
print("Invalid choice")

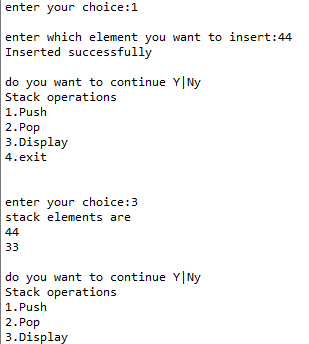
c=input("do you want to continue Y|N")

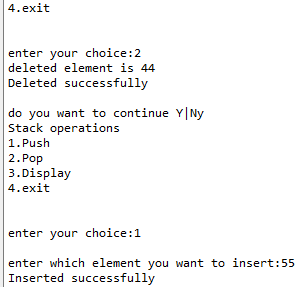
if c=='n' or c=='N':

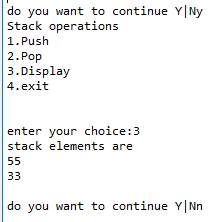
break

**Output:**









**Program 4(b)**

**Aim: write a python program to implement data stricter using queue using list.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Wed Mar 20 14:17:16 2024@author: it283"""

class queue:

def \_\_init\_\_(self):

self.ele=0

def enqueue(self):

self.ele=int(input("enter which element you want to insert:"))

list.append(self.ele)

print("Inserted successfully")

def dequeue(self):

if len(list)==0:

print("queue is empty,you can't delete")

else:

print("deleted element is {}".format(list.pop(0)))

print("Deleted successfully")

def display(self):

print("queue elements are")

n=len(list)

for i in range(n):

print(list[i])

list=[]

ob=queue()

while True:

print("Queue operations")

print("1.Enqueue\n2.Dequeue\n3.Display\n4.exit\n")

ch=int(input("enter your choice:"))

if ch==1:

ob.enqueue()

elif ch==2:

ob.dequeue()

elif ch==3:

ob.display()

elif ch==4:

break

else:

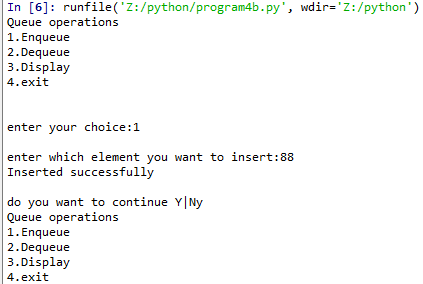
print("invalid choice")

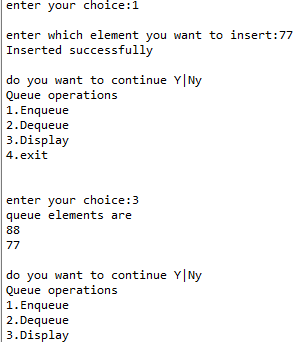
c=input("do you want to continue Y|N")

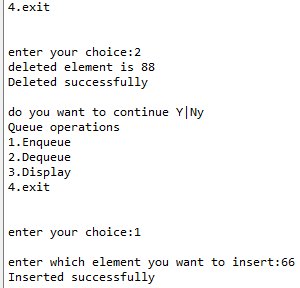
if c=='n' or c=='N':

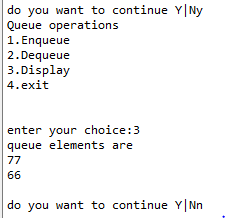
break

**Output:**









**Program 4(c)**

**Aim: write a python program to implemented data structer using linked list.**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Apr 12 03:14:28 2024@author: it283

"""

class Node:

def \_\_init\_\_(self,data):

self.data=data

self.next=None

class sll:

def \_\_init\_\_(self):

self.head=None

def create(self):

d=int(input("enter node data: "))

n1=Node(d)

self.head=n1

c=input("do you want to create another node(Y/N): ")

if c=='y' or c=='Y':

while True:

d1=int(input("enter next node data: "))

n2=Node(d1)

n1.next=n2

n1=n2

c=input("do you want to create another node(Y/N): ")

if c=='y' or c=='Y':

continue

else:

break

def traverse(self):

if self.head==None:

print("List is empty")

else:

temp=self.head

while temp is not None:

print(temp.data,end="->")

temp=temp.next

def insert\_at\_begin(self):

d=int(input("enter node data: "))

new=Node(d)

new.next=self.head

self.head=new

def delete\_at\_begin(self):

if self.head==None:

print("List is empty")

else:

s=self.head

print("deleted element is {}".format(s.data))

self.head=s.next

def insert\_at\_specific\_pos(self):

i=1

d=int(input("enter node data: "))

new=Node(d)

p=int(input("enter position to insert: "))

temp=self.head

if p==1:

new.next=self.head

self.head=new

else:

while(i<(p-1)):

temp=temp.next

i=i+1

new.next=temp.next

temp.next=new

def delete\_at\_specific\_pos(self):

i=1

p=int(input("enter position to delete: "))

temp=self.head

temp1=self.head

if self.head==None:

print("List is empty")

else:

if p==1:

s=self.head

print("deleted element is {}".format(s))

self.head=self.head.next

else:

while(i<(p)):

temp1=temp

temp=temp.next

i=i+1

temp1.next=temp.next

ob=sll()

ob.head=Node(None)

while True:

print("\n1.create\n2.traverse\n3.insert at begin\n4.delete at begin\n5.insert at specific position\n6.delete at specific position\n7.exit")

ch=int(input("Enter your choice:"))

if ch==1:

ob.create()

elif ch==2:

ob.traverse()

elif ch==3:

ob.insert\_at\_begin()

elif ch==4:

ob.delete\_at\_begin()

elif ch==5:

ob.insert\_at\_specific\_pos()

elif ch==6:

ob.delete\_at\_specific\_pos()

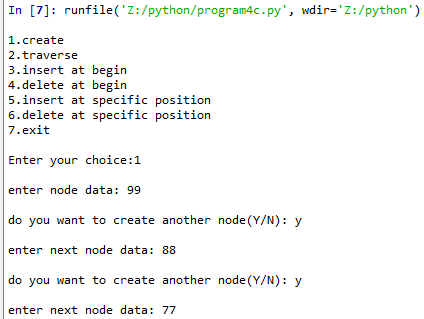
elif ch==7:

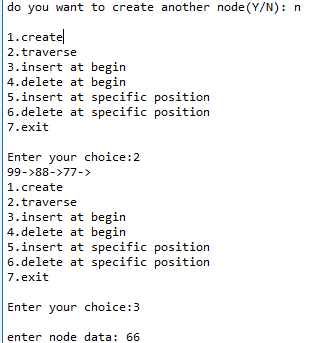
break

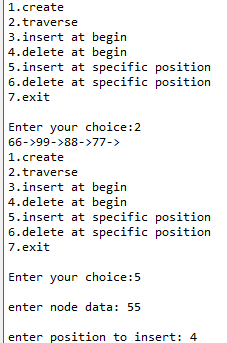
else:

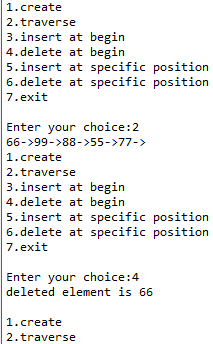
print("Invalid option")

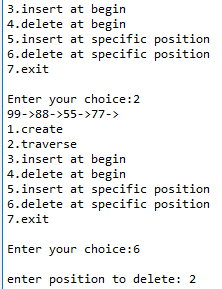
**Output:**

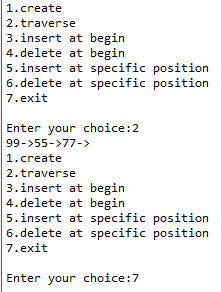












**Program 5**

**Aim: Write a python program to implement Bank Management System with following**

**a) Create an account b) Deposit c) Withdrawal**

**d) Check balance e) Mini-statement**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Apr 12 03:51:19 2024@author: it283

"""

import re

import datetime

import random

class Bank:

def \_init\_(self):

self.acc=0

self.name=" "

self.bal=0

self.pwd=" "

def create(self):

self.acc=random.randint(100000,100099)

self.name=input("Enter account holder name:")

self.bal=int(input("Enter opening balance:"))

if self.bal<500:

print("Opening balance should be grater then 500")

return

self.pwd=input("create a strong password containing atleast one capital,onesmall and a special character of length minimum 5 :")

if len(self.pwd)<5 or not re.search('[a-z]',self.pwd) or not re.search('[A-Z]',self.pwd) or not re.search('\W',self.pwd):

print("enter proper password")

return

self.t=datetime.datetime.now()

print("Account is created succesfully at {}".format(self.t))

print("Your account number is{}".format(self.acc))

print("Your password is {}".format(self.pwd))

f=open("trans.txt","a")

# f.write( "Accno Date deposit withdraw balance password")

#print("\n")

# f.write(' \n')

f.write("Acc:")

f.write(str(self.acc))

f.write(" ")

f.write("Time:")

f.write(str(self.t))

f.write(" ")

f.write("Bal:")

f.write(str(self.bal))

f.write(" ")

f.write("pwd:")

f.write(str(self.pwd))

#print("\n")

f.write("\n")

f.close()

def deposit(self):

ch=int(input("Enter the account number you want to deposit:"))

p=input("Enter the password:")

for account in accounts:

if account.acc==ch and account.pwd==p:

self.t=datetime.datetime.now()

amount=int(input("Enter the amount you want to deposit:"))

account.bal+=amount

print("deposited succesfully")

f=open("trans.txt","a")

f.write("Acc:")

f.write(str(ch))

f.write(" ")

f.write("Time:")

f.write(str(self.t))

f.write(" ")

f.write("Dep:")

f.write(str(amount))

f.write(" ")

f.write("Bal:")

f.write(str(account.bal))

f.write(" ")

f.write("Pwd:")

f.write(p)

f.write("\n")

#print("\n")

f.close()

else:

print("check password and account number")

return

def withdraw(self):

c=int(input("Enter the account number you want to wthdraw:"))

p=input("Enter the password:")

for account in accounts:

if account.acc==c and account.pwd==p:

self.t=datetime.datetime.now()

n=int(input("Enter the amount you want to withdraw:"))

if account.bal>=n:

account.bal-=n

print("withdraw succesful")

f=open("trans.txt","a")

f.write("Acc:")

f.write(str(c))

f.write(" ")

f.write("Time:")

f.write(str(self.t))

f.write(" ")

f.write("Wd:")

f.write(str(n))

f.write(" ")

f.write("Bal:")

f.write(str(account.bal))

f.write(" ")

f.write("Pwd:")

f.write(p)

f.write("\n")

f.close()

# print(self.bal)

else:

print("Insufficient balance")

else:

print("Check account number and password")

def check(self):

ch=int(input("Enter the account number you want to check the balance:"))

p=input("Enter the password:")

for account in accounts:

if account.acc==ch and account.pwd==p:

print("Your account has {} balance left".format(account.bal))

return

else:

print("Check the account number and password")

def mini(self):

f=open("trans.txt","r")

l=f.readlines()

f.close()

for i in l:

print(i.strip())

#print(l)

accounts=[]

while True:

ob=Bank()

print("\n1.create\n2.deposit\n3.withdraw\n4.check balance\n5.mini statement")

ch=int(input("Enter your choice:"))

if ch==1:

ob.create()

accounts.append(ob)

elif ch==2:

ob.deposit()

elif ch==3:

ob.withdraw()

elif ch==4:

ob.check()

elif ch==5:

ob.mini()

elif ch==6:

break

else:

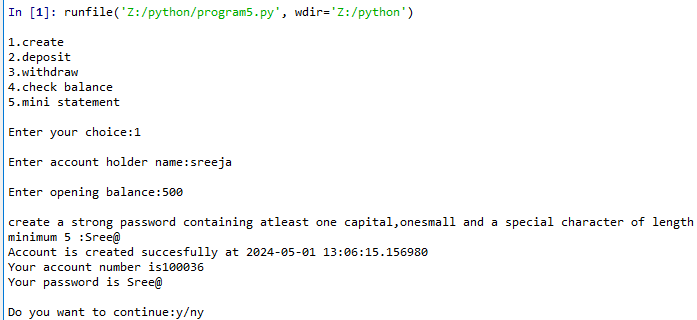
print("Invalid statement")

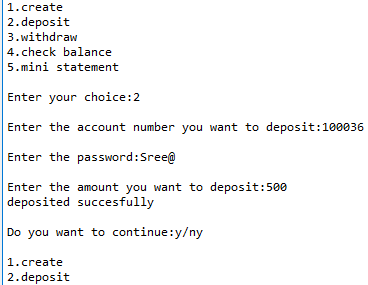
c=input("Do you want to continue:y/n")

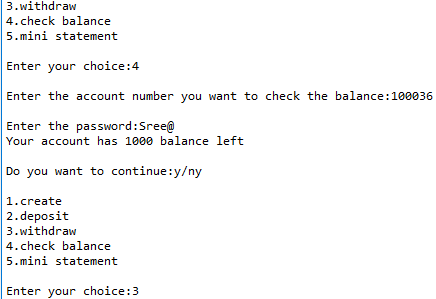
if c=='n' or c=='N':

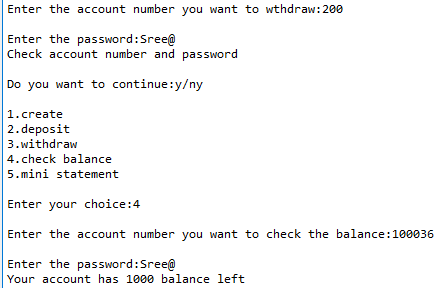
break

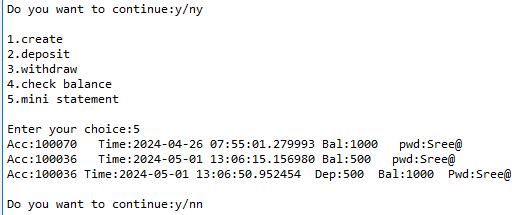
**Output:**











**Program 6**

**Aim: Write a python program to implement Library Management System with following modules using oops concepts.**

**a) Add book details b) Display book details c) Update book details**

**d) Delete book details e) Search book details**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Apr 19 03:48:22 2024@author: it283

"""

class library:

def init(self):

self.Author=" "

self.bno=0

self.bname=" "

self.price=0

self.publisher=" "

def add(self):

self.bno=int(input("Enter book number:"))

self.bname=input("Enter book name:")

self.Author=input("Enter author name:")

self.price=int(input("Enter price of the book:"))

self.publisher=input("Enter publisher:")

def display(self):

c=0

if len(list)==0:

print("list is empty")

else:

for i in list:

c+=1

print("Book bumber is",i.bno)

print("Book name is",i.bname)

print("Price of the book",i.price)

print("Name of the publisher is",i.publisher)

if c==0:

print("Record is not found")

def update(self):

if len(list)==0:

print("list is empty")

else:

b=int(input("Enter the book number u want to update:"))

for i in list:

if i.bno==b:

print("Enter new details:")

i.bname=input("Enter book name:")

i.Author=input("Enter author name:")

i.price=int(input("Enter book price:"))

i.publisher=input("Enter publisher:")

print("Record updated successfully")

def delete(self):

c=0

if len(list)==0:

print("list is empty")

else:

a=input("Enter which author book u want to delete:")

for i in list:

if i.Author==a:

c+=1

list.remove(i)

print("Record deleted successfully")

if c==0:

print("Records not found")

def search(self):

if len(list)==0:

print("list is empty")

else:

c=0

a=input("Enter author name for searching:")

for i in list:

if i.Author==a:

c+=1

print("Record is found")

if c==0:

print("No records found")

list=[]

while True:

ob=library()

print("\n1.Add book details\n2.Display book details\n3.Update book details\n4.Delete book details\n5.Search book details\n")

ch=int(input("Enter your choice:"))

if ch==1:

ob.add()

list.append(ob)

elif ch==2:

ob.display()

elif ch==3:

ob.update()

elif ch==4:

ob.delete()

elif ch==5:

ob.search()

else:

print("Invalid choice")

c=input("Do u want to continue Y/N:")

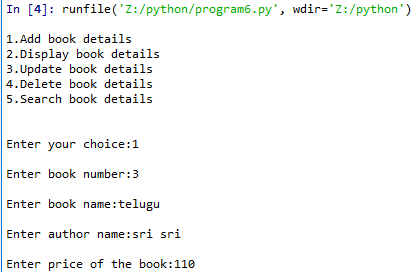
if c=='y' or c=='Y':

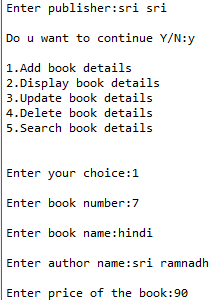
continue

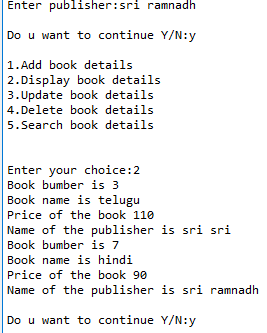
else:

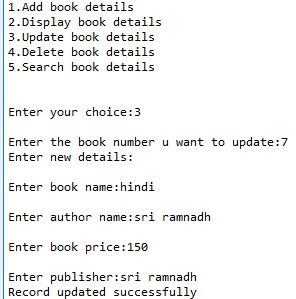
break

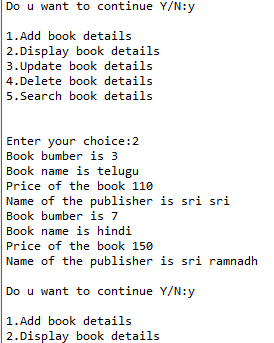
**Output:**

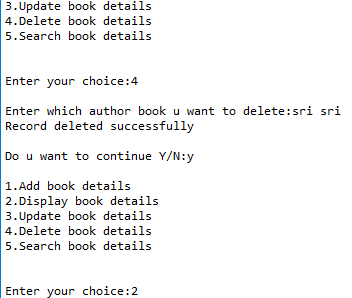


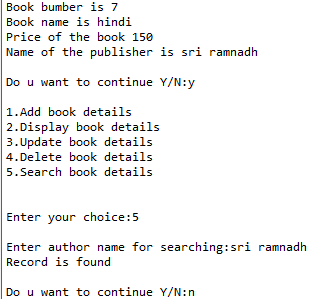












**Program 7**

**Aim: write a python program to select a type of user (General User, Prime User, Premium User) and find a discount of selected item based on following:**

**Type of User Discount**

**General User NO**

**Prime User 10%**

**Premium User 15%**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Wed May 1 13:27:18 2024@author: it283

"""

class General:

def \_\_init\_\_(self):

self.laptop=20000

self.mobile=15000

self.WashingMashine=40000

def cal\_lap(self,n):

return n\*self.laptop

def cal\_mob(self,n):

return n\*self.mobile

def cal\_wash(self,n):

return n\*self.WashingMashine

class Prime(General):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.cooler=35000

self.Fridge=30000

self.Ac=50000

def cal\_lap(self,n):

return n\*(self.laptop-self.laptop\*10/100)

def cal\_mob(self,n):

return n\*(self.mobile-self.mobile\*10/100)

def cal\_wash(self,n):

return n\*(self.WashingMashine-self.WashingMashine\*10/100)

def cal\_cool(self,n):

return n\*(self.cooler-self.cooler\*10/100)

def cal\_fri(self,n):

return n\*(self.Fridge-self.Fridge\*10/100)

def cal\_ac(self,n):

return n\*(self.Ac-self.Ac\*10/100)

class Advance(Prime):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.oven=35000

self.tv=30000

#self.Ac=50000

def cal\_lap(self,n):

return n\*(self.laptop-self.laptop\*15/100)

def cal\_mob(self,n):

return n\*(self.mobile-self.mobile\*15/100)

def cal\_wash(self,n):

return n\*(self.WashingMashine-self.WashingMashine\*15/100)

def cal\_cool(self,n):

return n\*(self.cooler-self.cooler\*15/100)

def cal\_fri(self,n):

return n\*(self.Fridge-self.Fridge\*15/100)

def cal\_ac(self,n):

return n\*(self.Ac\*15/100)

def cal\_ov(self,n):

return n\*(self.oven-self.oven\*15/100)

def cal\_tv(self,n):

return n\*(self.tv-self.tv\*15/100)

while True:

print("1.General\n2.Prime\n3.Advance")

u=int(input("Enter the type of user:"))

if u==1:

ob=General()

t=0

print("Sorry You have no discount")

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

while True:

print("The list of items you can purchase:")

print("1.laptop\n2.Mobile\n3.WashingMachine")

se=int(input("Select any items from the above list:"))

if se==1:

n=int(input("How many laptops you want to buy:"))

t+=ob.cal\_lap(n)

print(t)

elif se==2:

n=int(input("How many mobiles you want to buy:"))

t+=ob.cal\_mob(n)

print(t)

elif se==3:

n=int(input("How many washing Machines you want to buy:"))

t+=ob.cal\_wash(n)

else:

print("Invalid option")

c=input("Do you want to continue as general user Y/N:")

if c=="n" or c=="N":

# break

print("User name is:",name)

print("The bill of general user is:",t)

break

elif u==2:

ob1=Prime()

t1=0

print("You have 10% discount")

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

while True:

print("The list of items you can purchase:")

print("1.laptop\n2.Mobile\n3.WashingMachine\n4.cooler\n5.Fridge\n6.Ac")

se=int(input("Select any items from the above list:"))

if se==1:

n=int(input("How many laptops you want to buy:"))

t1+=ob1.cal\_lap(n)

print(t1)

elif se==2:

n=int(input("How many mobiles you want to buy:"))

t1+=ob1.cal\_mob(n)

print(t1)

elif se==3:

n=int(input("How many washing Machines you want to buy:"))

t1+=ob1.cal\_wash(n)

elif se==4:

n=int(input("How many coolers you want to buy:"))

t1+=ob1.cal\_cool(n)

print(t1)

elif se==5:

n=int(input("How many fridges you want to buy:"))

t1+=ob1.cal\_fri(n)

print(t1)

elif se==6:

n=int(input("How many acs you want to buy:"))

t1+=ob1.cal\_ac(n)

else:

print("Invalid option")

c=input("Do you want to continue as primel user Y/N:")

if c=="n" or c=="N":

# break

print("User name is:",name)

print("The bill of prime user is:",t1)

break

elif u==3:

ob2=Advance()

t2=0

print("You have 10% discount")

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

while True:

print("The list of items you can purchase:")

print("1.laptop\n2.Mobile\n3.WashingMashine\n4.cooler\n5.Fridge\n6.Ac\n7.Oven\n8.Tv")

se=int(input("Select any items from the above list:"))

if se==1:

n=int(input("How many laptops you want to buy:"))

t2+=ob2.cal\_lap(n)

print(t2)

elif se==2:

n=int(input("How many mobiles you want to buy:"))

t2+=ob2.cal\_mob(n)

print(t2)

elif se==3:

n=int(input("How many washing Mashines you want to buy:"))

t2+=ob2.cal\_wash(n)

elif se==4:

n=int(input("How many coolers you want to buy:"))

t2+=ob2.cal\_cool(n)

print(t2)

elif se==5:

n=int(input("How many fridges you want to buy:"))

t2+=ob2.cal\_fri(n)

print(t2)

elif se==6:

n=int(input("How many acs you want to buy:"))

t2+=ob2.cal\_ac(n)

elif se==7:

n=int(input("How many ovens do you want to buy:"))

t2+=ob2.cal\_ov(n)

elif se==8:

n=int(input("How many tvs do you want to buy:"))

t2+=ob2.cal\_tv(n)

else:

print("Invalid option")

c=input("Do you want to continue as advance user Y/N:")

if c=="n" or c=="N":

# break

print("User name is:",name)

print("The bill of advance user is:",t2)

break

else:

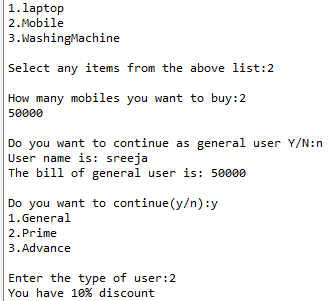
print("Invalid option")

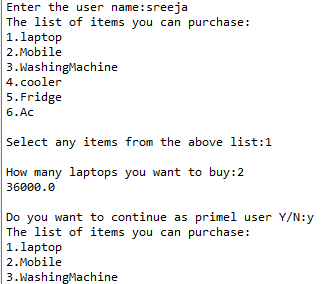
cc=input("Do you want to continue(y/n):")

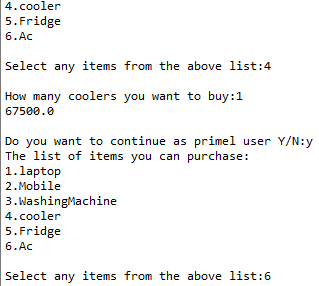
if cc=='n' or cc=='N':

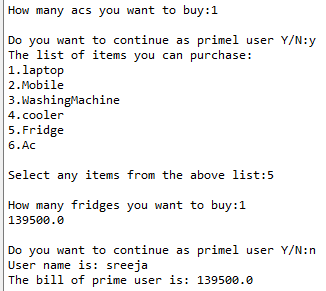
break

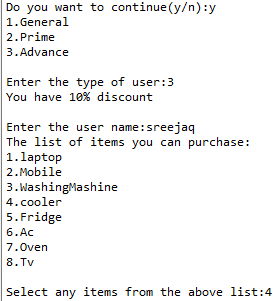
**Output:**

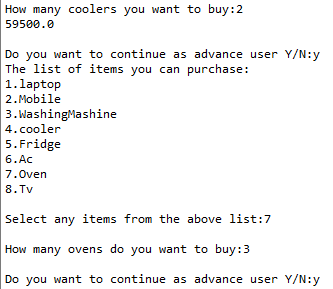


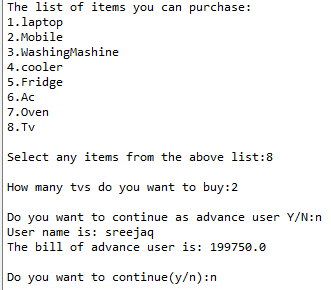












**Program 8**

**Aim: Write a python program to implement the following using SQLite3**

**a) Create a table b) Insert some data into the table**

**c) To read data from table d) Update a record in table**

**e) To delete a record from table**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Apr 19 03:31:13 2024@author: it283

"""

import sqlite3,re

from datetime import datetime

def create():

try:

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

query='''CREATE TABLE EMP(

ID INTEGER PRIMARYKEY,

NAME TEXT NOT NULL,

EMAIL TEXT NOT NULL UNIQUE,

JOINING\_DATE DATETIME,

SALARY REAL NOT NULL);'''

cursor=con.cursor()

print("successfully connected to SQLITE")

cursor.execute(query)

con.commit()

print("SQLite table created")

cursor.close()

except sqlite3.Error as error:

print("Error while creating sqlite table ",error)

finally:

if con:

con.close()

print("SQLite con closed")

def insert():

try:

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

cursor=con.cursor()

print("successfully connected to SQLITE")

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

name=input("Enter Name: ")

while True:

email=input("Enter email id: ")

if re.search("^[a-z1-9]\*@[a-z.]\*$",email):

break

else:

print("Enter valid email:")

j=input("Joining Date dd:mm:yyyy: ")

Join=datetime.strptime(j,"%d:%m:%Y").date()

while True:

Sal=int(input("Enter Salary: "))

if Sal<0:

print("Enter salary in positive")

else:

break

query="INSERT INTO EMP Values(?,?,?,?,?)"

cursor.execute(query,(id,name,email,Join,Sal))

con.commit()

print("Record is inserted successfully into emp table")

cursor.close()

except sqlite3.Error as error:

print("Error while creating sqlite table ",error)

finally:

if con:

con.close()

print("SQLite con closed")

def update():

try:

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

cursor=con.cursor()

print("successfully connected to SQLITE")

id=input("Enter id to update: ")

query="select \* from emp where id=?"

res=cursor.execute(query,(id,))

records=res.fetchall()

if len(records)==0:

print("Record not found")

else:

print("\n1.Name\n2.Email\n3.Joining date\n4.Salary\n")

ch=int(input("choose what you want to update: "))

if ch==1:

name=input("Enter value to update: ")

query="update emp set name=? where id=?"

cursor.execute(query,(name,id))

elif ch==2:

while True:

email=input("Enter email id: ")

if re.search("^[a-z1-9]\*@[a-z.]\*$",email):

break

else:

print("Enter valid email")

query="update emp set email=? where id=?"

cursor.execute(query,(email,id))

elif ch==3:

j=input("Enter date dd:mm:yyyy::: ")

joining\_date=datetime.strptime(j,"%d:%m:%Y").date()

query="update emp set joining\_date=? where id=?"

cursor.execute(query,(joining\_date,id))

elif ch==4:

salary=int(input("Enter value to update: "))

if salary<0:

print("Enter salary in positive number")

else:

query="update emp set salary=? where id=?"

cursor.execute(query,(salary,id))

else:

print("Invalid option")

con.commit()

print("Record updated successfully")

cursor.close()

except sqlite3.Error as error:

print("Error while creating sqlite table ",error)

finally:

if con:

con.close()

print("SQLite con closed")

def search():

try:

l=[]

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

cursor=con.cursor()

print("successfully connected to SQLITE")

i=int(input("enter id to search: "))

query='''select \* from emp;'''

res=cursor.execute(query)

l=res.fetchall()

l1=[]

count=0

for j in range(len(l)):

l1=(l[j])

if i in l1:

count+=1

if count>=1:

print("Record found")

query="select \* from emp where id=?"

res=cursor.execute(query,(i,))

rec=res.fetchall()

print(rec)

else:

print("Record not found")

cursor.close()

except sqlite3.Error as error:

print("Error while creating sqlite table ",error)

finally:

if con:

con.close()

print("SQLite con closed")

return count

def delete():

try:

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

cursor=con.cursor()

print("successfully connected to SQLITE")

id=input("Enter id you want to delete: ")

query="select \* from emp where id=?"

res=cursor.execute(query,(id,))

records=res.fetchall()

if len(records)==0:

print("Record not found")

else:

query="delete from emp where id=?"

cursor.execute(query,(id,))

con.commit()

print("record is deleted successfully")

cursor.close()

finally:

if con:

con.close()

print("SQLite con closed")

def display():

try:

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

cursor=con.cursor()

print("successfully connected to SQLITE")

query='''select \* from emp'''

res=cursor.execute(query)

records=res.fetchall()

print("total rows are: ",len(records))

print("id\tname\t\t\temail\t\t\tjoining date\t\tsalary")

for row in records:

print(row[0],"\t",row[1],"\t\t",row[2],"\t",row[3],"\t\t",row[4])

cursor.close()

except sqlite3.Error as error:

print("error while creating sqlite table ",error)

finally:

if con:

con.close()

print("SQLite con closed")

while True:

print("1.Create\n2.Insert\n3.Update\n4.Display\n5.Delete\n6.Search\n7.Exit")

ch=int(input("Enter your choice: "))

if ch==1:

create()

elif ch==2:

insert()

elif ch==3:

update()

elif ch==4:

display()

elif ch==5:

delete()

elif ch==6:

search()

elif ch==7:

break

else:

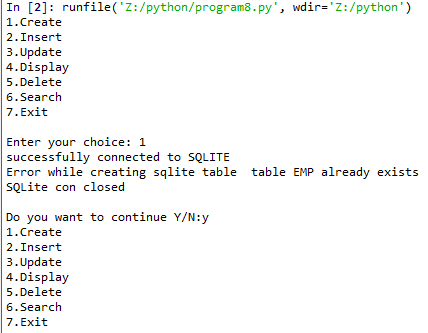
print("Invalid option")

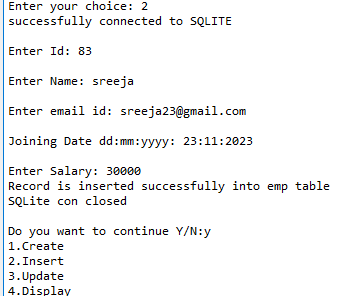
c=input("Do you want to continue Y/N:")

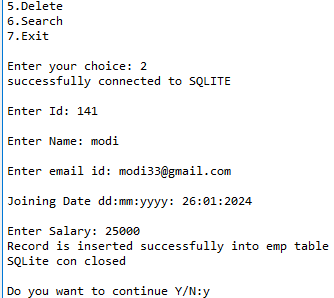
if c=='n' or c=='N':

break

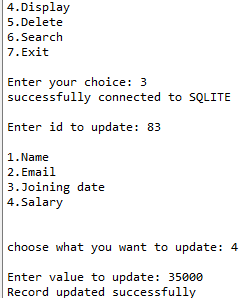
**Output:**

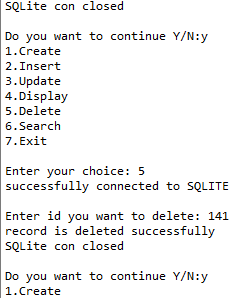




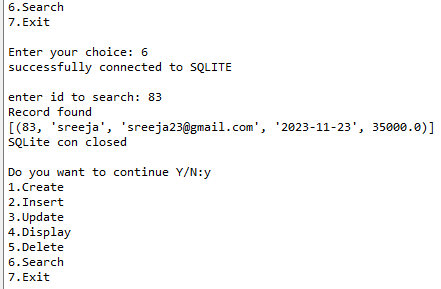














**Program 9**

**Aim: Write a python program to perform the following on two numbers using GUI module.**

**a) Addition b) Subtraction c) Multiplication d) Division**

**Source code:**

# -\*- coding: utf-8 -\*-

"""Created on Fri Apr 19 02:32:46 2024@author: it283

"""

from tkinter import \*

class Mywindow:

def \_\_init\_\_(self,win):

self.lbl1=Label(win,text='first number')

self.lbl2=Label(win,text='second number')

self.lbl3=Label(win,text='result')

self.t1=Entry()

self.t2=Entry()

self.t3=Entry()

self.lbl1.place(x=100,y=50)

self.t1.place(x=200,y=50)

self.lbl2.place(x=100,y=100)

self.t2.place(x=200,y=100)

self.b1=Button(win,text='add',command=self.add)

self.b2=Button(win,text='subtract',command=self.sub)

self.b3=Button(win,text='multiply',command=self.mul)

self.b4=Button(win,text='divide',command=self.div)

self.b1.place(x=100,y=150)

self.b2.place(x=200,y=150)

self.b3.place(x=100,y=200)

self.b4.place(x=200,y=200)

self.lbl3.place(x=100,y=250)

self.t3.place(x=200,y=250)

def add(self):

num1=int(self.t1.get())

num2=int(self.t2.get())

result=num1+num2

self.t3.insert(END,str(result))

def sub(self):

num1=int(self.t1.get())

num2=int(self.t2.get())

result=num1-num2

self.t3.insert(END,str(result))

def mul(self):

num1=int(self.t1.get())

num2=int(self.t2.get())

result=num1\*num2

self.t3.insert(END,str(result))

def div(self):

num1=int(self.t1.get())

num2=int(self.t2.get())

result=num1/num2

self.t3.insert(END,str(result))

window=Tk()

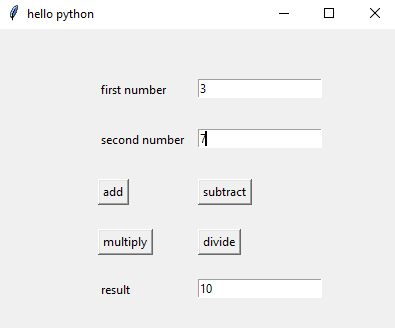
mywin=Mywindow(window)

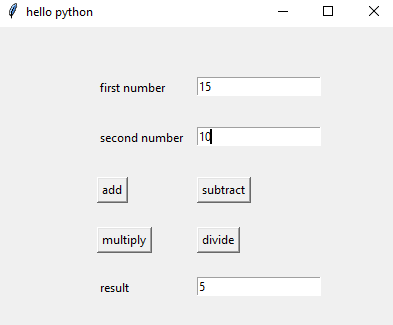
window.title('hello python')

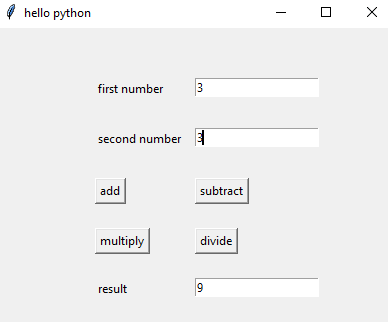
window.geometry("400x300+10+10")

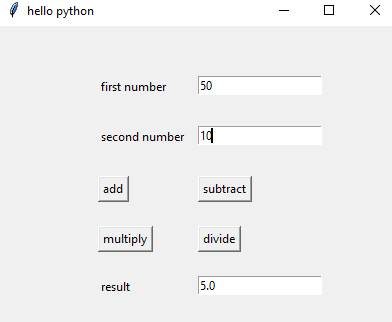
window.mainloop()

**Output:**









**Program 10**

**Aim: Write a python program to read excel file (Student.xlsx) using pandas library and print the following**

**a) Print all the students details of multiple sheets in a given excel file**

**b) Sort student details based on Marks(either sheet1 or sheet2)**

**c) Print student details whose name starts with ‘s’**

**d) Print student details based on specific range of marks**

**e) Print student details based on registration number**

**Source code:**

import pandas as pd

def read():

d=pd.read\_excel("student.xlsx",sheet\_name=0,index\_col=0)

d1=pd.read\_excel("student.xlsx",sheet\_name=1,index\_col=0)

read=pd.concat([d,d1])

print(read)

def sort():

d=pd.read\_excel("student.xlsx",sheet\_name=0,index\_col=0)

d1=pd.read\_excel("student.xlsx",sheet\_name=1,index\_col=0)

read=pd.concat([d,d1])

s=read.sort\_values(['Marks'],ascending=True)

print("Student details after sorting their marks: ")

print(s)

def startswith():

d=pd.read\_excel("student.xlsx",sheet\_name=0,index\_col=1)

d1=pd.read\_excel("student.xlsx",sheet\_name=1,index\_col=1)

read=pd.concat([d,d1])

filtered = read[read['Name'].str.startswith('s')]

print("List of student details whose name starts with s: ")

print(filtered)

def srange():

import pandas as pd

d=pd.read\_excel("student.xlsx",sheet\_name=0,index\_col=0)

d1=pd.read\_excel("student.xlsx",sheet\_name=1,index\_col=0)

read=pd.concat([d,d1])

def find(sr,er):

filtered=read[(read['Marks']>=sr)&(read['Marks']<=er)]

return filtered

sr=int(input("Enter start range of marks"))

er=int(input("Enter end range of marks"))

students=find(sr,er)

print("Students with marks between",sr,"and",er,":")

print(students)

def regno():

d=pd.read\_excel("student.xlsx",sheet\_name=0,index\_col=0)

d1=pd.read\_excel("student.xlsx",sheet\_name=1,index\_col=0)

read=pd.concat([d,d1])

def details(reg\_num):

sdetails=read[read['Regd.No']==reg\_num]

return sdetails

rg=input("Enter registration number you want to the details of by spaces :")

reg\_num=rg.split(" ")

for reg in reg\_num:

print("student details for registration number",reg,":")

print(details(reg))

print()

while True:

print("\n1.print all student details\n2.sorting of marks\n3.name starts with an alphabet s\n4.to print specific range of marks\n5.to print student details based on registration number\n6.exit\n")

ch=int(input("Enter your choice: "))

if ch==1:

read()

elif ch==2:

sort()

elif ch==3:

startswith()

elif ch==4:

srange()

elif ch==5:

regno()

elif ch==6:

break

else:

print("enter a valid choice(1/2/3/4/5/6)")

c=input("do you want to continue y/n")

if c=='n' or c=='N':

break

**Output:**

