**Lab program-1:**

**Aim:1(a) Write a python program to find factorial of a given number using recursion.**

**Source Code:**

**”””**

Created on Tue Feb 27 08:50:55 2024

@author: it2117

"""

while True:

def fact(n):

if n==1:

return n

else:

return n\*fact(n-1)

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

print("factorial of {} is {}".format(n,fact(n)))

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

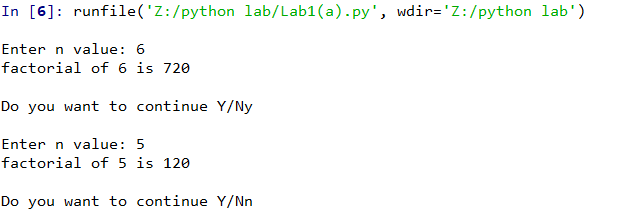
if ch=='y' or ch=='Y':

continue

else:

break

**Output:**

****

**Aim:1(b) Write a python program to print fibonacci series upto n terms using recursion.**

**Source Code:**

"""

Created on Tue Feb 27 09:21:56 2024

@author: it2117

"""

while True:

def fib(n):

if n<=1:

return n

else:

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

n=int(input("Enter the noumber of terms you want to print:"))

if n<0:

print("pls enter a positive integer")

else:

print("fibonacci sequence:")

for i in range(n):

print(fib(i))

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

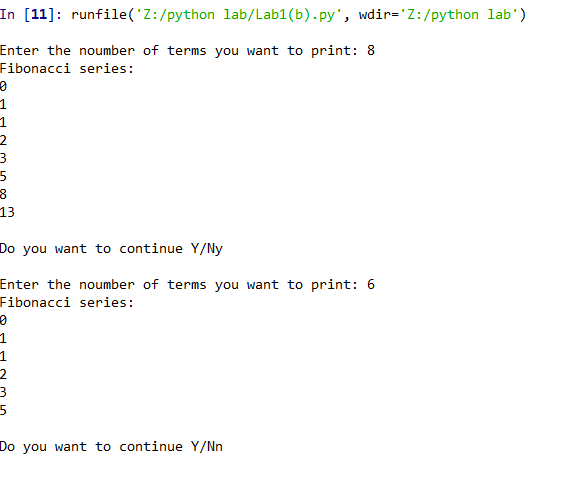
if ch=='y' or ch=='Y':

continue

else:

break

**Output:**

****

**Aim:1(c) Write a python program to print sum of digits of given number using recursion.**

**Source Code:**

"""

Created on Tue Feb 27 09:28:34 2024

@author: it2117

"""

while True:

def digit(n):

if n<=9:

return n

else:

rem=n%10

return rem+digit(n//10)

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

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

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

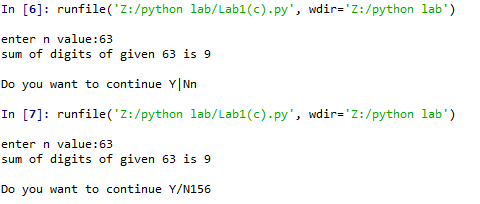
if ch=='y' or ch=='Y':

continue

else:

break

**Output:**



**Lab program-2:**

**Aim:(a) Write a python program to print 30 days before and after from the given date.**

**Source Code:**

from datetime import datetime,timedelta

s=input("enter any date( %d/%mm/%yyyy): ")

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

d1=d.date()

print("given date:",d1)

print(" 30 days after given date: ")

for i in range(30):

d1+=timedelta(1)

print(d1)

print(" 30 days before given date: ")

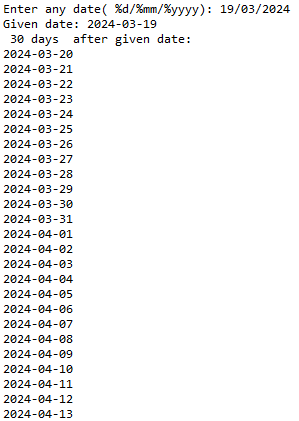
for i in range(30):

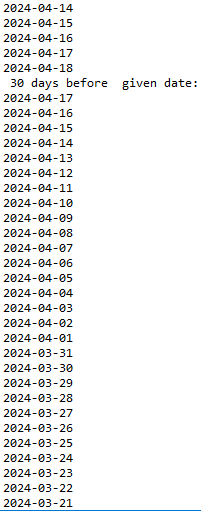
d1-=timedelta(1)

print(d1)

**Output:**









**Aim 2b)Write a python program to create 12 fixed dates from a specified date where the difference between two dates is ‘20’.**

**Source Code:**

"""

Created on Tue Mar 19 07:57:46 2024

@author: it2117

"""

from datetime import datetime,timedelta

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

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

d1=d.date()

print("Given date: ",d1)

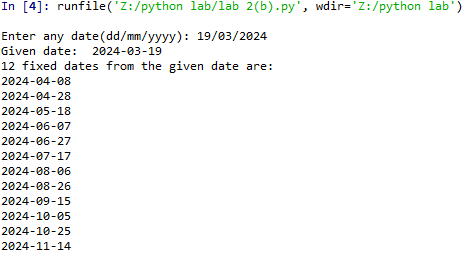
print("12 fixed dates from the given date are: ")

for i in range(12):

d1+=timedelta(days=20)

print(d1)

**Output:**

****

**Aim 2c):Write a python program to calaculate youyr age based on the given date from current date.**

**Source Code:**

"""

Created on Tue Mar 19 07:57:50 2024

@author: it2117

"""

from datetime import datetime

s=input("Enter you date of birth(dd/mm/yyyy):")

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

s=datetime.now()

ag=s-d

#ag=d-(s.date())

year=ag.days//365

mon=(ag.days%365)//30

days=(ag.days%365)%30

print("your age is: ",year,"years",mon,"months",days,"days")

**Output:**



**Lab program-3**

**Aim: Write a python program to read a textfile named “bec.txt” and display the following:**

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

**b)Even length words in a given file.**

**c)Write a user defined function Rec(author) which accept author name as argument and display the number of books of given author.**

**Source Code:**

"""

Created on Fri Mar 22 14:42:01 2024

@author: it2117

"""

while True:

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

n=input("Enter any data separated by space:")

f.write(n)

f.close()

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

s=a.read()

n=n.split(" ")

for i in n:

if i[0]=='t' or i[0]=='a' or i[0]=='A' or i[0]=='T':

print(i)

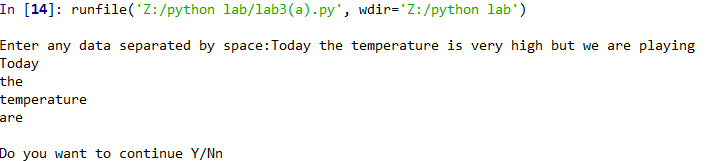
a.close()

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

if ch=='N' or ch=='n':

break

**Output:**

****

**b)Source Code:**

"""

Created on Fri Mar 22 14:43:07 2024

@author: it2117

"""

while True:

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

n=input("Enter any data separated by space:")

f.write(n)

f.close()

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

s=a.read()

n=s.split(" ")

for i in n:

if len(i)%2==0:

print(i)

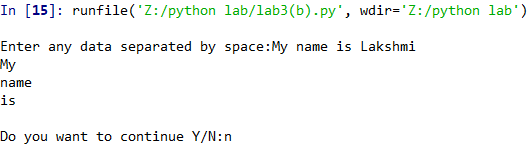
a.close()

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

if ch=='N' or ch=='n':

break

**Output:**

****

**c)Source Code:**

"""

Created on Fri Mar 22 14:42:46 2024

@author: it2117

"""

import pickle

while True:

def createFile():

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

Book\_Name=input("Enter book name:")

Book\_No=int(input("Enter book number:"))

Author=input("Enter author:")

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

record=[Book\_No,Book\_Name,Author,price]

pickle.dump(record,file)

file.close()

def count():

a=input("Enter author name")

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

cnt=0

try:

while True:

record=pickle.load(file)

if record[2]==a:

cnt+=1

except EOFError:

pass

print("Count=",cnt)

file.close()

while True:

createFile()

ch=input("add more records(Y/N)")

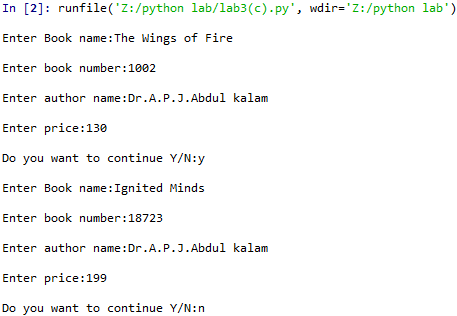
if ch=='Y' or ch=='y':

continue

else:

break

**Output:**

****

Enter author name: Dr . A. P .J. Abdul kalam

Count=2

**Lab program-4:**

**Aim: Write a python program to implement the following structures using oops concept:**

1. **Stack using list b) Queue using list c) Single linked list**

**Source Code:**

"""

Created on Fri Mar 22 14:44:57 2024

@author: it2117

"""

def push():

ele=int(input("Enter which element do you want to insert"))

list.append(ele)

print("Inserted successfully")

def pop():

if len(list)==0:

print("stack is empty")

else:

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

print("deleted successfully")

def display():

n=len(list)

if n==0:

print("stack is empty")

else:

print("stack element are")

for i in list[::-1]:

print(i)

list=[]

while True:

print("stack operations")

print("1.push\n2.pop\n3.display")

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

if ch==1:

push()

elif ch==2:

pop()

elif ch==3:

display()

else:

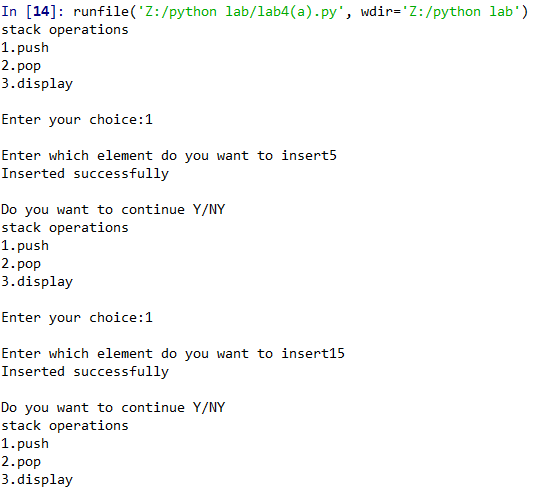
print("Invalid choice")

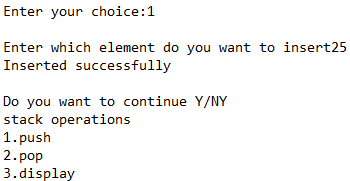
c=input("Do you want to continue Y/N")

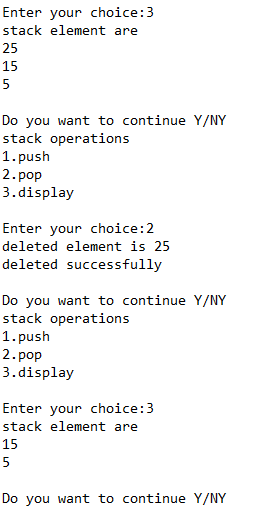
if c=='n' or c=='N':

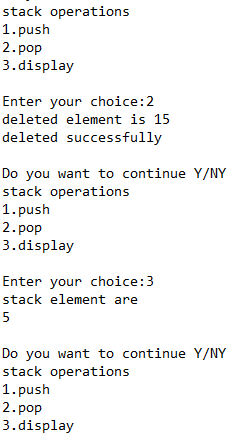
break

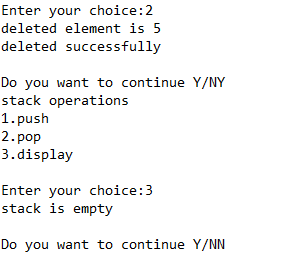
**Output:**

****

****

****

****

****

**Source Code (b):**

"""

Created on Fri Mar 22 14:44:57 2024

@author: it2117

"""

def enqueue():

ele=int(input("Which element do you want to insert"))

list.append(ele)

print("inserted successfully")

def dequeue():

if len(list)==0:

print("queue is empty")

else:

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

print("deleted successfully")

def display():

n=len(list)

if n==0:

print("queue is empty")

else:

print("queue elements are")

for i in range(n):

print(list[i])

list=[]

while True:

print("queue using list")

print("1.enqueue\n2.dequeue\n3.display")

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

if ch==1:

enqueue()

elif ch==2:

dequeue()

elif ch==3:

display()

else:

print("invalid choice")

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

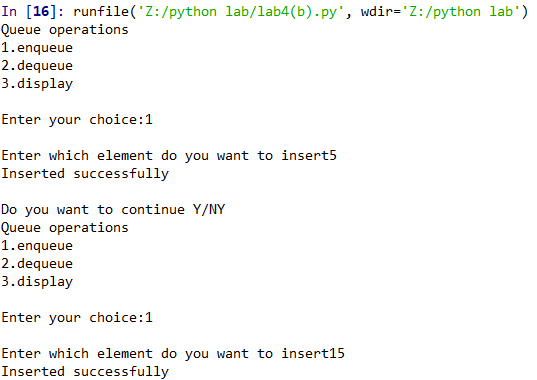
if c=='y' or c=='Y':

continue

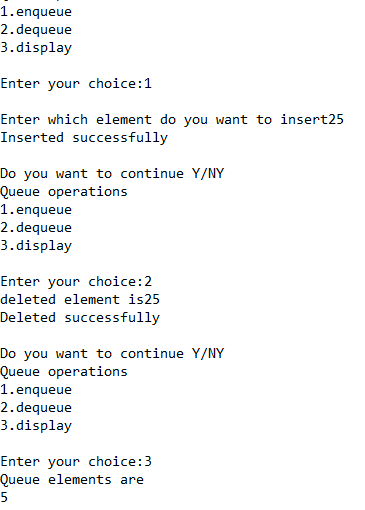
else:

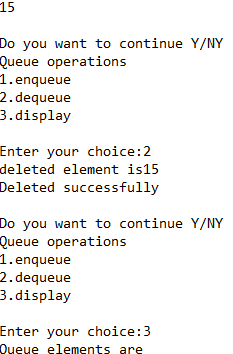
break

**Output:**

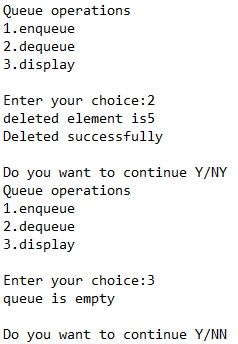
****

****

****

****

****

****

**Source Code (c):**

"""

Created on Sat Mar 23 13:45:31 2024

@author: it2117

"""

class node:

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

self.data=data

self.next=None

class linked\_list:

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

self.head=None

def insert(self,data):

if self.head:

temp=self.head

while(temp.next):

temp=temp.next

temp.next=data

else:

self.head=data

def traverse(self):

if self.head==None:

print("list is empty")

else:

temp=self.head

while(temp!=None):

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

temp=temp.next

def insert\_at\_begin(self,data):

if self.head==None:

self.head=data

else:

temp=self.head

self.head=data

self.head.next=temp

print("Inserted successfully")

def delete\_at\_begin(self):

if self.head==0:

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

else:

temp=self.head

print("deleted node is{}".format(temp.data))

self.head=temp.next

print("Deleted successfully")

def insert\_at\_location(self,d,p):

if p == 0:

self.insert\_at\_beginning(d)

return

new\_node = node(d)

current\_node = self.head

count=0

while current\_node:

if count == p - 1:

new\_node.next = current\_node.next

current\_node.next = new\_node

break

current\_node = current\_node.next

count += 1

else:

print("Invalid location.")

ll=linked\_list()

while True:

print("Linked list operations")

print("1.create\n2.traverse\n3.insert at begin\n4.delete at begin\n5.insertat specific location")

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

if ch==1:

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

n1=node(n)

ll.insert(n1)

elif ch==2:

ll.traverse()

elif ch==3:

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

n2=node(ne)

ll.insert\_at\_begin(n2)

elif ch==4:

ll.delete\_at\_begin()

elif ch == 5:

p = int(input("enter the position:"))

if p<0:

print("invalid position")

elif p==0:

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

ll.insert\_at\_begin(n)

else:

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

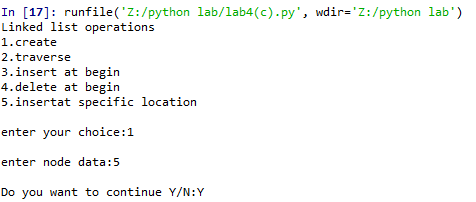
ll.insert\_at\_location(n,p)

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

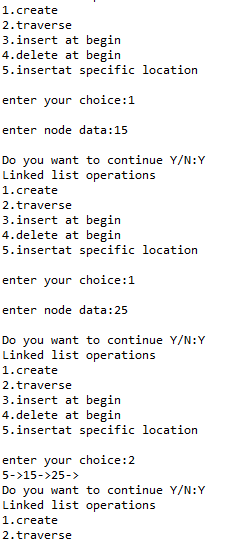
if c=='n' or c=='N':

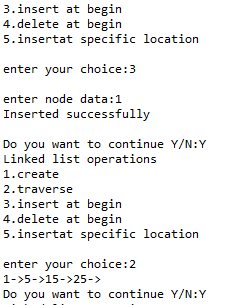
break

**Output:**

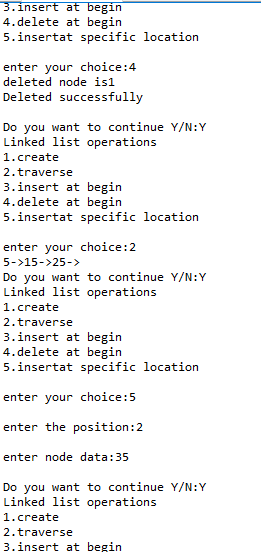


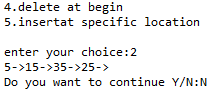
****

****

****

****

****

****

**Lab program-5:**

**Aim: Write a python program to implement bank management system with following attributes**

**a)create an account b)deposit c)withdraw d)check balance d)mini statement**

**Source Code:**

"""

Created on Tue Mar 26 07:15:59 2024

@author: it2117

"""

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 lenght 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("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))

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")

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()

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

break

else:

print("Check the account number and password")

break

def mini(self):

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

l=f.readlines()

f.close()

for i in l:

print(i.strip())

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:

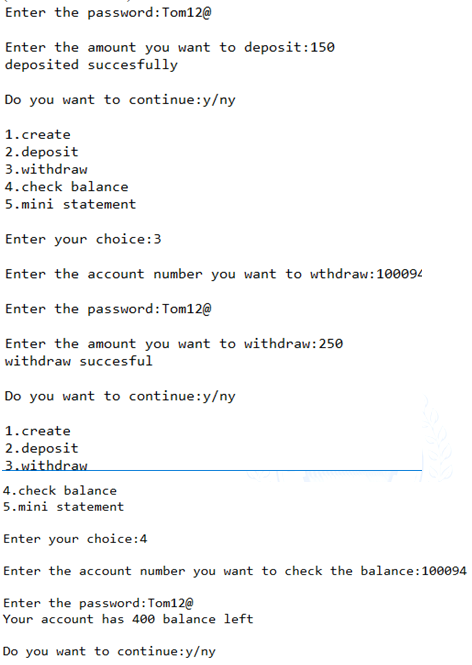
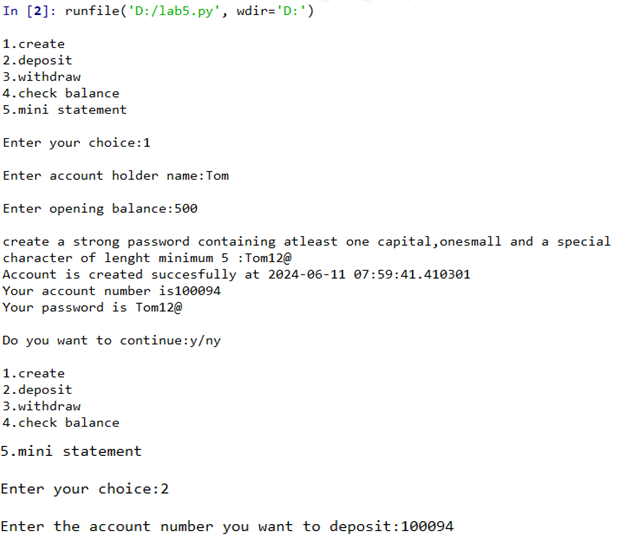
print("Invalid statement")

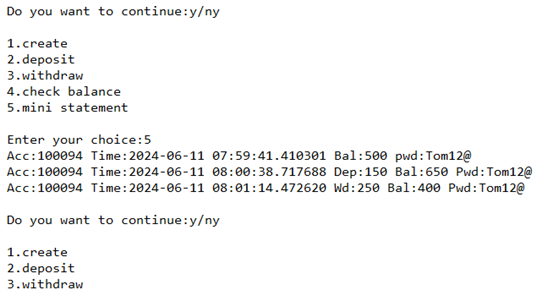
c=input("Do you want to continue:y/n")

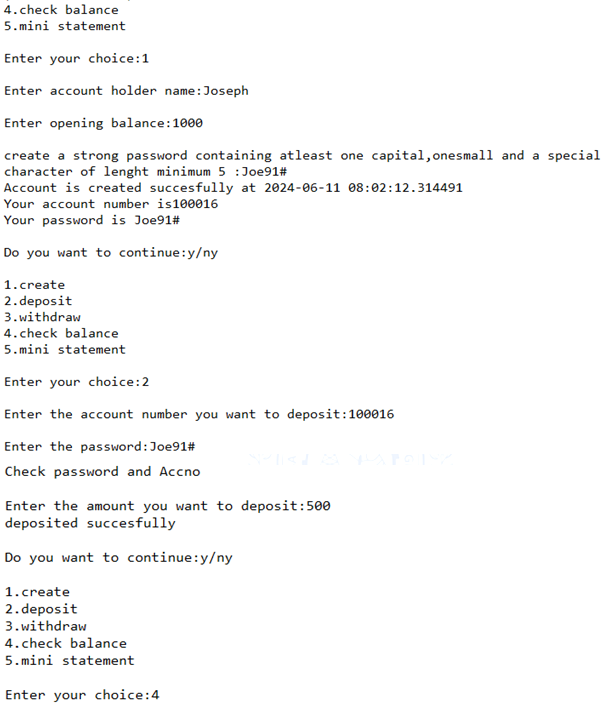
if c=='n' or c=='N':

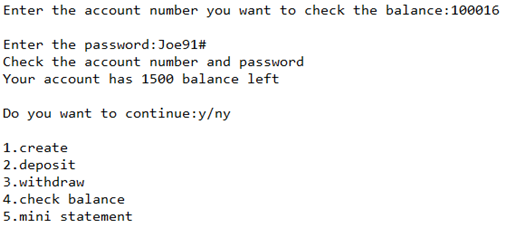
break

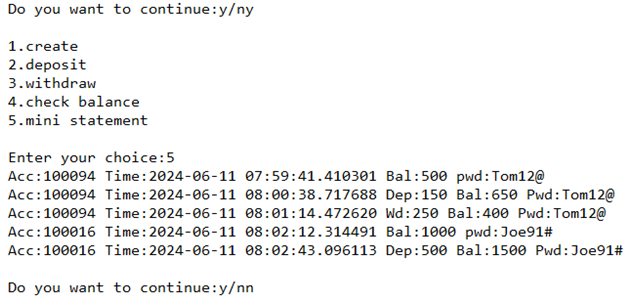
**Output**











**Lab program-6**

**Aim: Write a python program to implement library management system write the following modules using oops concept:**

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

**e) Delete book details**

**Source Code:**

"""

Created on Mon Apr 29 14:24:44 2024

@author: it2117

"""

class Library:

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

self.id=0

self.title=""

self.price=0

self.author=""

self.publisher=""

def insert(self):

self.title=input("enter book name: ")

self.id=int(input("enter book number: "))

self.price=int(input("enter book price: "))

self.author=input("enter author name: ")

self.publisher=input("enter publishers name: ")

def display(self,l):

if l==[]:

print("no records to display")

else:

j=1

for i in l:

print("Book {} Details".format(j))

print("Book Title:\t",i.title)

print("Book ID:\t",i.id)

print("Price:\t",i.price)

print("Author:\t",i.author)

print("Publisher\t",i.publisher)

j+=1

def update(self,l):

count=0

j=int(input("enter id you want to update: "))

for i in l:

if i.id==j:

print("1.title\n2.price\n3.author\n4.publisher")

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

if ch==1:

if i.id==j:

i.title=input("enter title: ")

elif ch==2:

if i.id==j:

i.price=int(input("enter price: "))

elif ch==3:

if i.id==j:

i.author=input("enter author name: ")

elif ch==4:

if i.id==j:

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

count+=1

if count==0:

print("record not found")

def delete(self,l):

j=int(input("enter id you want to delete: "))

for i in l:

if i.id==j:

l.remove(i)

print("record is deleted successfully")

else:

print("record not found: ")

def search(self,l):

j=int(input("enter id you want to search: "))

for i in l:

if i.id==j:

print("record found ")

print("Book {} Details".format(j))

print("Book Title:\t",i.title)

print("Book ID:\t",i.id)

print("Price:\t",i.price)

print("Author:\t",i.author)

print("Publisher\t",i.publisher)

else:

print("record not found: ")

list=[]

while True:

print("1.insert\n2.display\n3.update\n4.search\n5.delete\n6.exit")

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

if ch==1:

ob=Library()

ob.insert()

list.append(ob)

print("record is inserted successfully")

elif ch==2:

ob=Library()

ob.display(list)

elif ch==3:

ob=Library()

ob.update(list)

elif ch==4:

ob=Library()

ob.search(list)

elif ch==5:

ob=Library()

ob.delete(list)

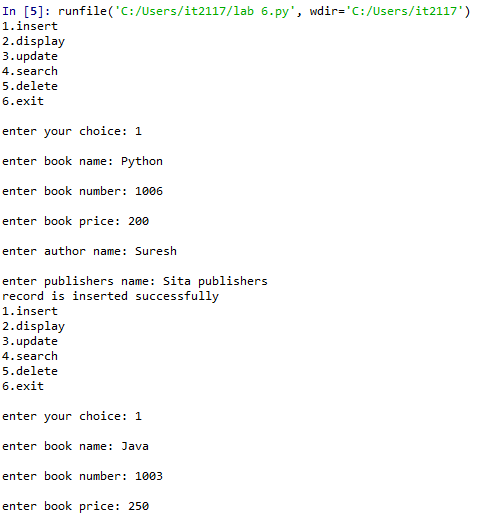
elif ch==6:

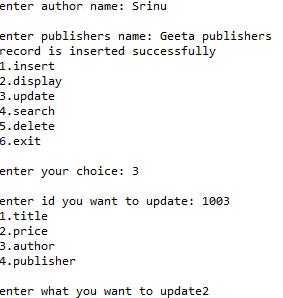
break

else:

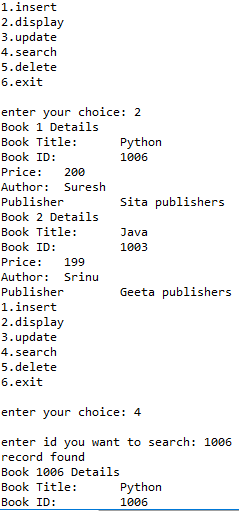
print("Invalid option")

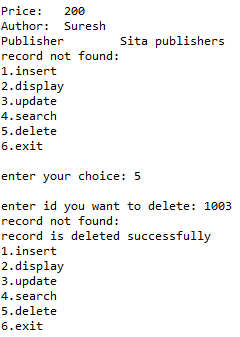
**Output:**

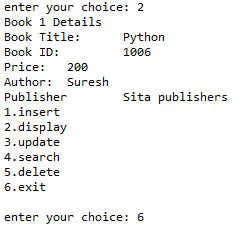












**Lab program-7:**

**Aim: Write a python program to implement price of selected item using inheritance for the following types of customers:**

|  |  |
| --- | --- |
| **TYPE OF USER** | **DISCOUNT** |
| **General user** | **No discount** |
| **Prime user** | **10% of base price** |
| **Advanced prime user** | **15% of base price** |

**Source Code:**

"""

Created on Mon Apr 29 16:02:37 2024

@author: it2117

"""

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.WashingMashine")

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 Mashines 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.WashingMashine\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 Mashines 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 15% 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:

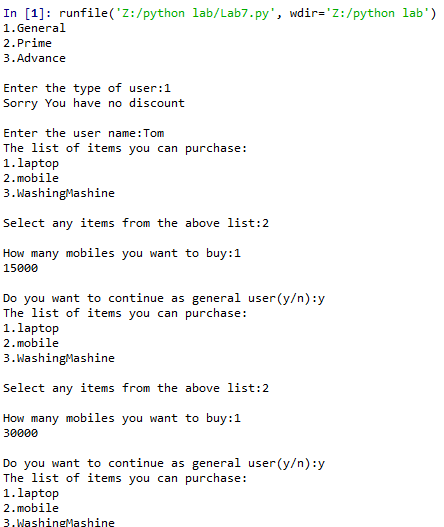
print("Invalid option")

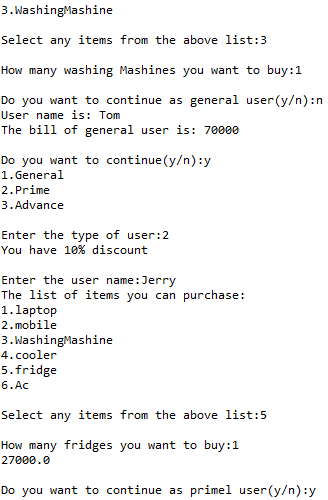
cc=input("Do you want to continue(y/n):")

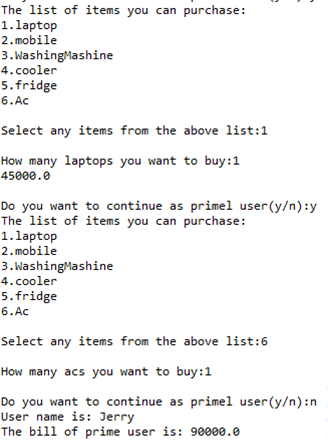
if cc=='n' or cc=='N':

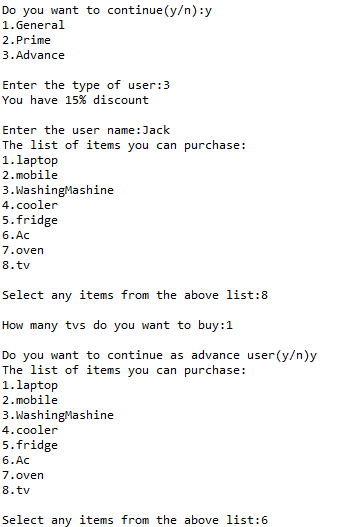
break

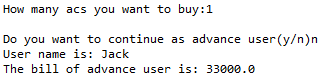
**Output:**













**Lab program-8:**

**Aim: Write a python program to create a table using mysqlite3.**

**Source Code:**

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 can 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 can 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 can 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()

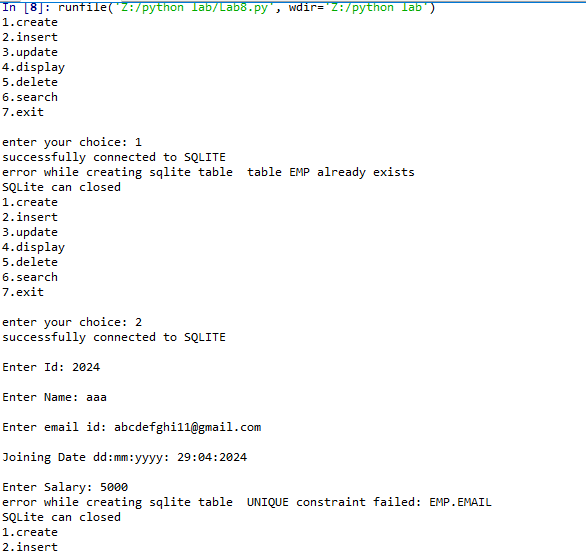
elif ch==7:

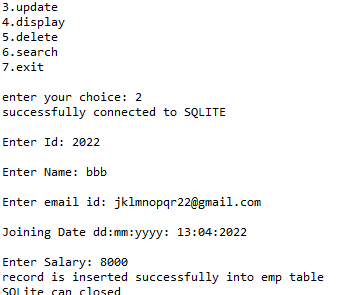
break

else:

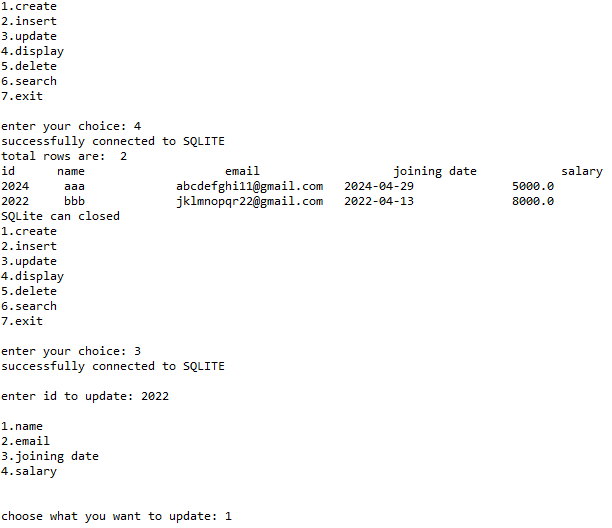
print("invalid option")

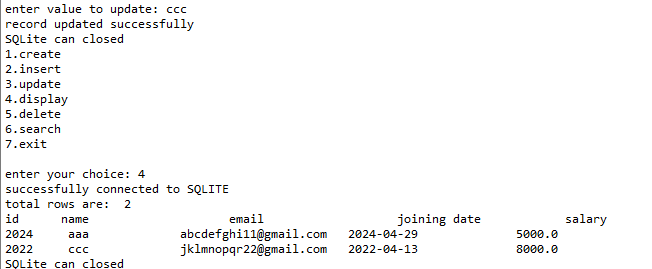
**Output:**

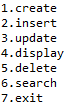




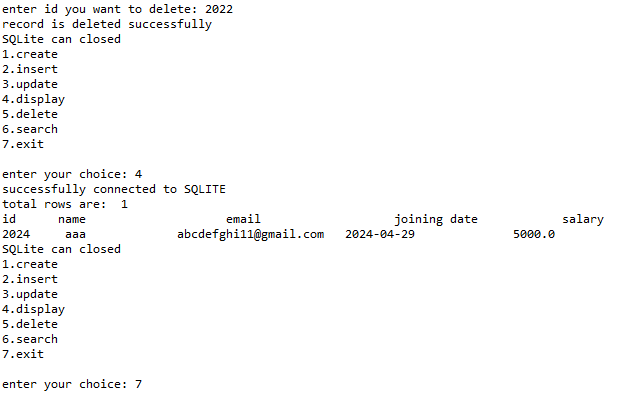












:

**Lab program-9:**

**Aim: Write a Python program to perform all arithmetic operations on two numbers using GUI module.**

**Source Code:**

"""

Created on Wed May 1 13:51:25 2024

@author: it2117

"""

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()

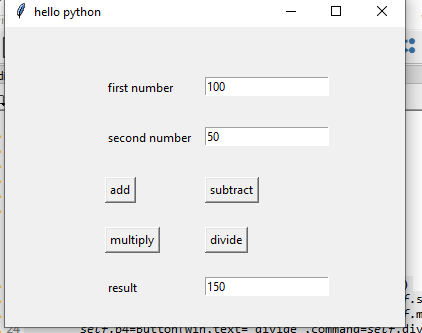
mywin=Mywindow(window)

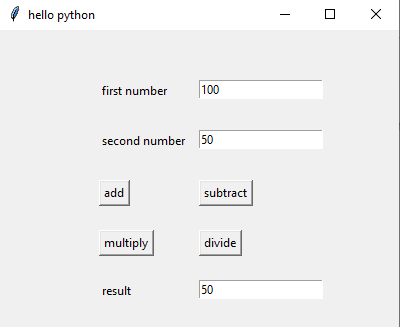
window.title('hello python')

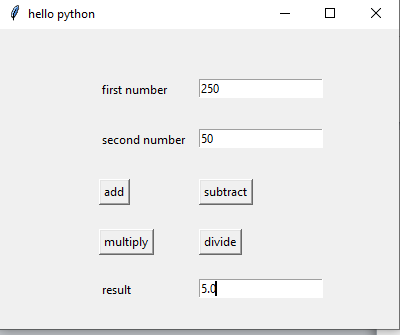
window.geometry("400x300+10+10")

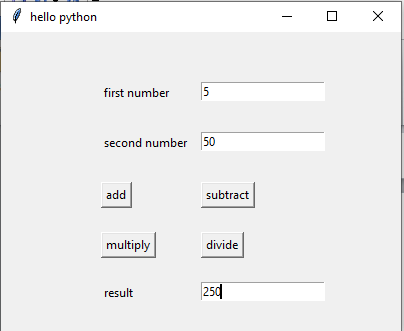
window.mainloop()

**Output:**

****

****

****

****

**Lab Program-10**

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

**a)to print all the student details**

**b)to print all the student details of multiple sheets**

**c)search student details based on marks either sheet1 or sheet2**

**d)print student details whose name starts with ‘s’**

**e)print student details based on specific range of marks**

**f)print student details based on Regd No.**

**Source Code:**

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

"""

Created on Tue Apr 30 09:17:40 2024

@author: it2117

"""

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['REDG 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()

def multi():

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)

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.student details of multiple sheets\n7.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:

multi()

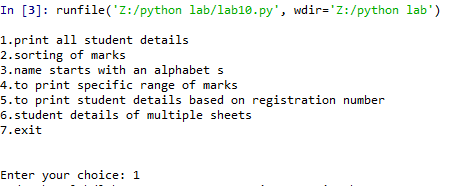
elif ch==7:

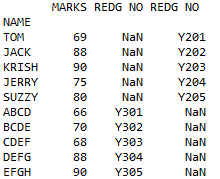
break

else:

print("enter a valid choice")

**Output:**



****

