1.Write a Python program to find the factorial of a given number using recursion

**SOURCE CODE**:

def fact(n):

if(n==1) or(n==0):

return 1

else:

return n\*fact(n-1)

while True:

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

print("factorial of",n,"is :",fact(n))

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

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

continue

else:

break

**Output:**

**Text

Description automatically generated**

2.Create a mapping from three character month name to month number.Ask the user for month either in lower or upper .print the month number corresponding to the month user entered.

**SOURCE CODE**:

months="jan feb mar apr may jun jul aug sep oct nov dec"

months=months.split(" ")

months1={}

for i in range(len(months)):

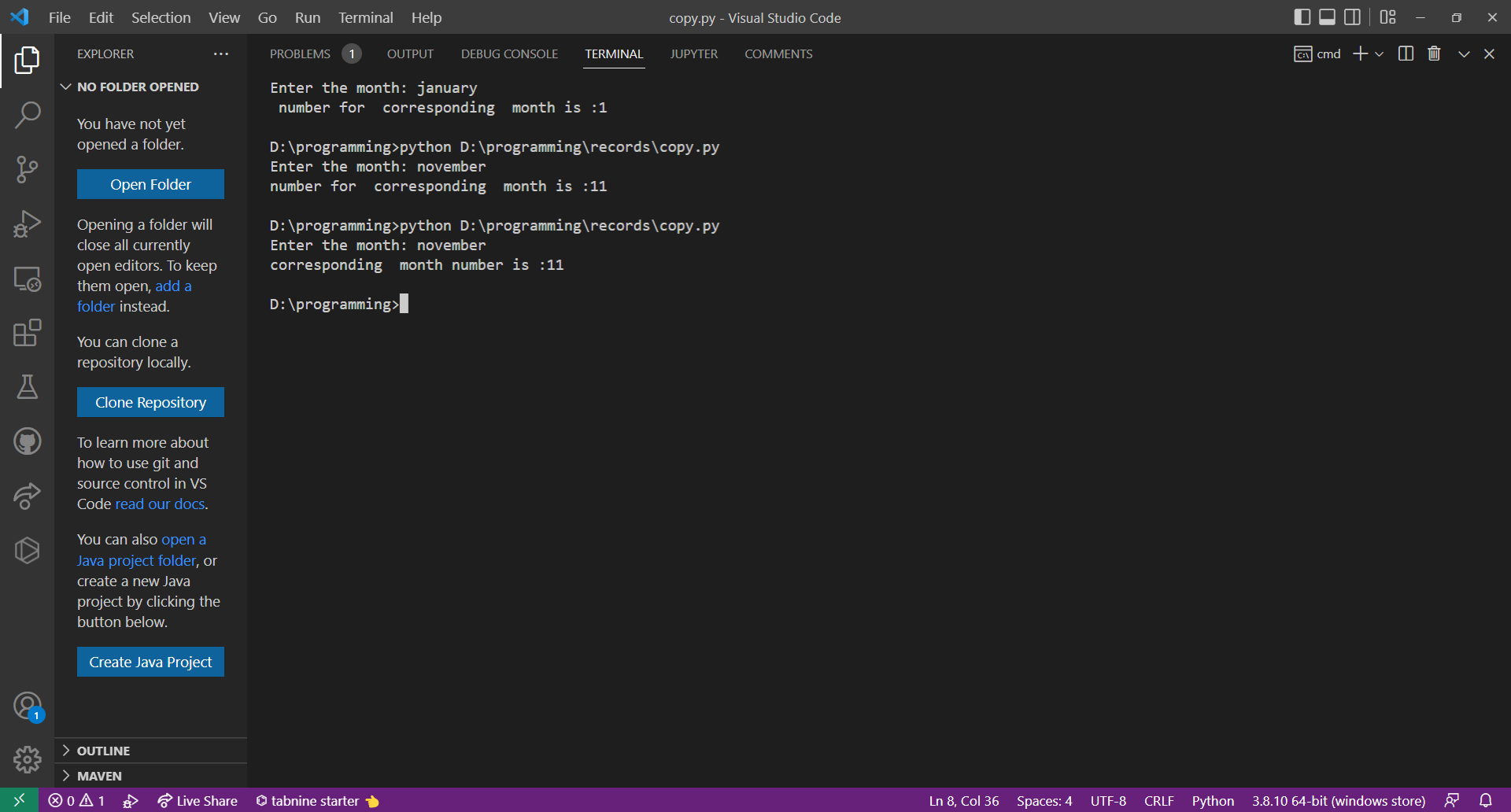
months1[months[i]]=i+1

text=input(“Enter the month: ”)

mon=text[:3].lower()

print(“corresponding month number is :{}”.format(months1[mon]))

**Output:**

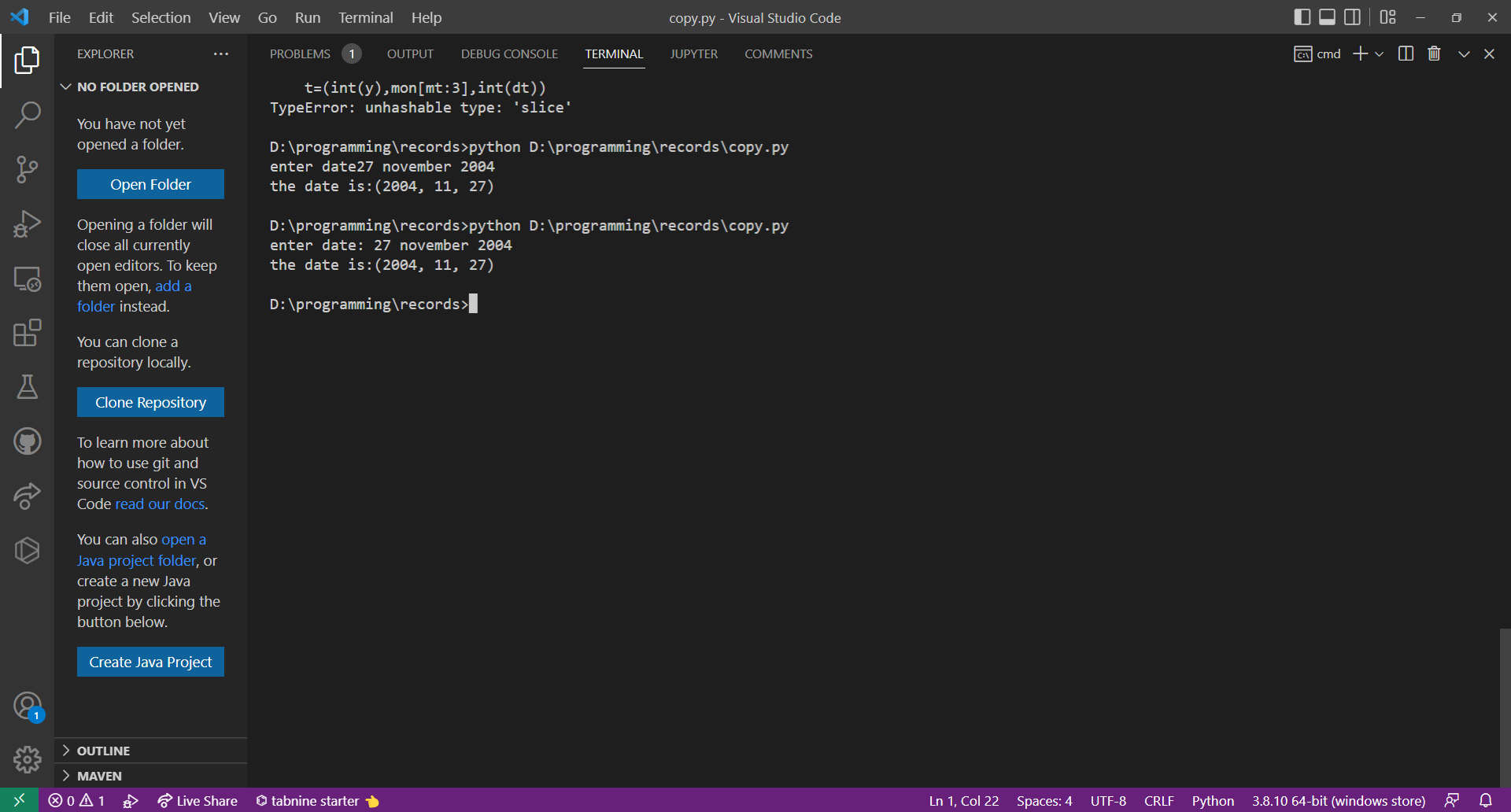


4.You are given data strings of the form “29 Jul,2009” or “4 January 2008”, in other words a number, a string and another number, with a comma sometimes separating the items. Write a program that takes such a string as input and print a tuple as output with all integers as date string.

**SOURCE CODE**:

m=input("enter date: ")  
m=m.replace(',',' ')  
dt,mt,y=m.split()  
mon={'Jan':1,'Feb':2,'Mar':3,'Apr':4,'May':5,'Jun':6,'Jul':7,'Aug':8,'Sep':9,  
 'Oct':10,'Nov':11,'Dec':12}  
mt=mt.capitalize()  
t=(int(y),mon[mt],int(dt))  
print(“the date is : {}”.format(t))

**Output:**



4.Read the pendulum.txt file. Print only the second column of the file.

**SOURCE CODE:**

f=open('pendulum.txt','r')

w=open('copy.txt','w')

for i in f:

try:

fields=i.split()

w.write(fields[1]+'\n')

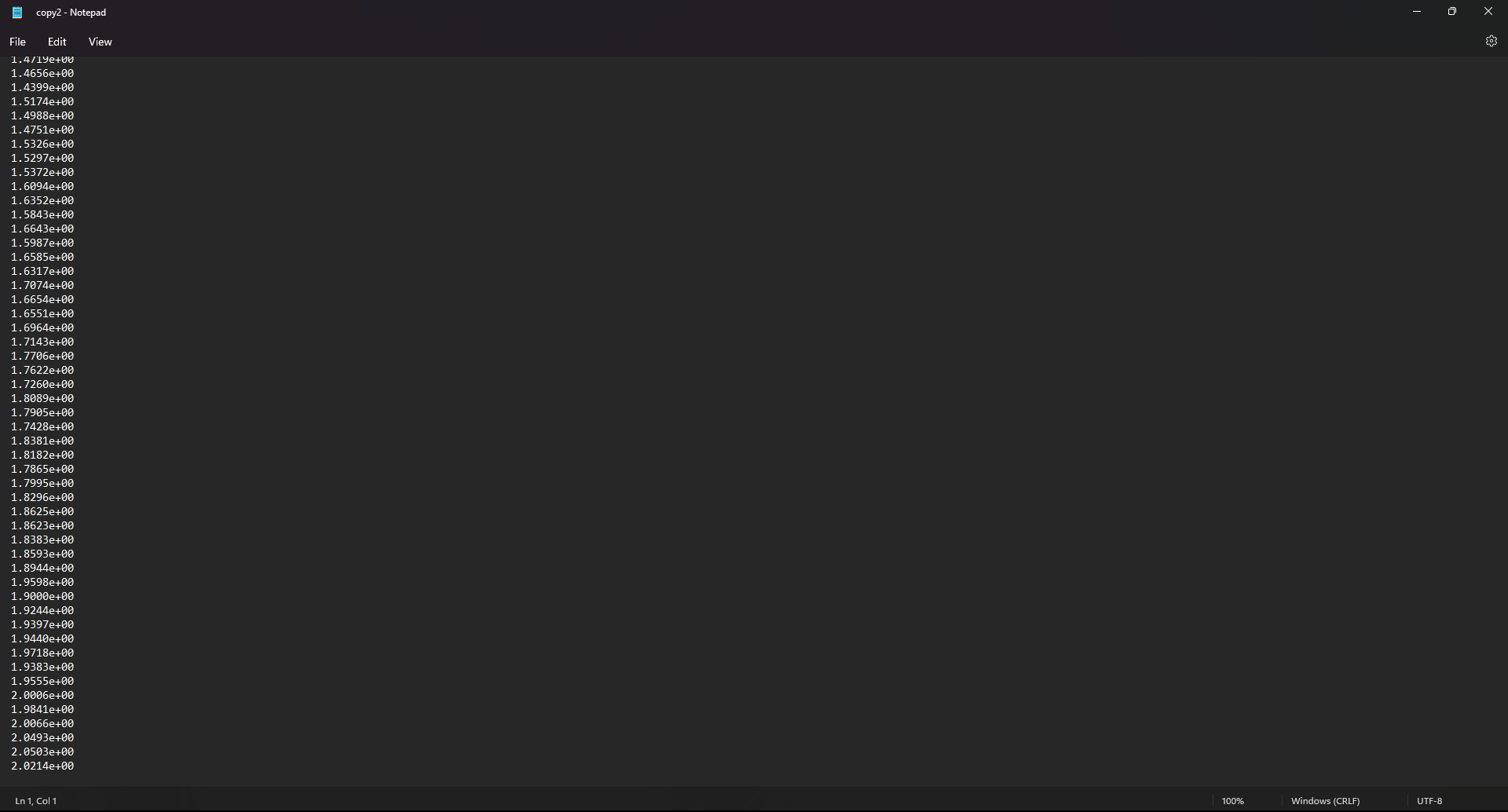
except:

print(" ")

f.close()

w.close()

**Output:**

****

**MBOX**

From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008

Return-Path: <postmaster@collab.sakaiproject.org>

Received: from murder (mail.umich.edu [141.211.14.90])

by frankenstein.mail.umich.edu (Cyrus v2.3.8) with LMTPA;

Sat, 05 Jan 2008 09:14:16 -0500

X-Sieve: CMU Sieve 2.3

Received: from murder ([unix socket])

by mail.umich.edu (Cyrus v2.2.12) with LMTPA;

Sat, 05 Jan 2008 09:14:16 -0500

Received: from holes.mr.itd.umich.edu (holes.mr.itd.umich.edu [141.211.14.79])

by flawless.mail.umich.edu () with ESMTP id m05EEFR1013674;

Sat, 5 Jan 2008 09:14:15 -0500

Received: FROM paploo.uhi.ac.uk (app1.prod.collab.uhi.ac.uk [194.35.219.184])

BY holes.mr.itd.umich.edu ID 477F90B0.2DB2F.12494 ;

5 Jan 2008 09:14:10 -0500

Received: from paploo.uhi.ac.uk (localhost [127.0.0.1])

by paploo.uhi.ac.uk (Postfix) with ESMTP id 5F919BC2F2;

Sat, 5 Jan 2008 14:10:05 +0000 (GMT)

Message-ID: <200801051412.m05ECIaH010327@nakamura.uits.iupui.edu>

Mime-Version: 1.0

Content-Transfer-Encoding: 7bit

Received: from prod.collab.uhi.ac.uk ([194.35.219.182])

by paploo.uhi.ac.uk (JAMES SMTP Server 2.1.3) with SMTP ID 899

for <source@collab.sakaiproject.org>;

Sat, 5 Jan 2008 14:09:50 +0000 (GMT)

Received: from nakamura.uits.iupui.edu (nakamura.uits.iupui.edu [134.68.220.122])

by shmi.uhi.ac.uk (Postfix) with ESMTP id A215243002

for <source@collab.sakaiproject.org>; ……………………………………………….

5.Write a program to read a file called mbox.txt and display the no .of lines containing a string @ucb.ac.za.

**Source code:**

import re

f=open('mbox.txt','r')

pattern=re.compile("@uct.ac.za")

count=0

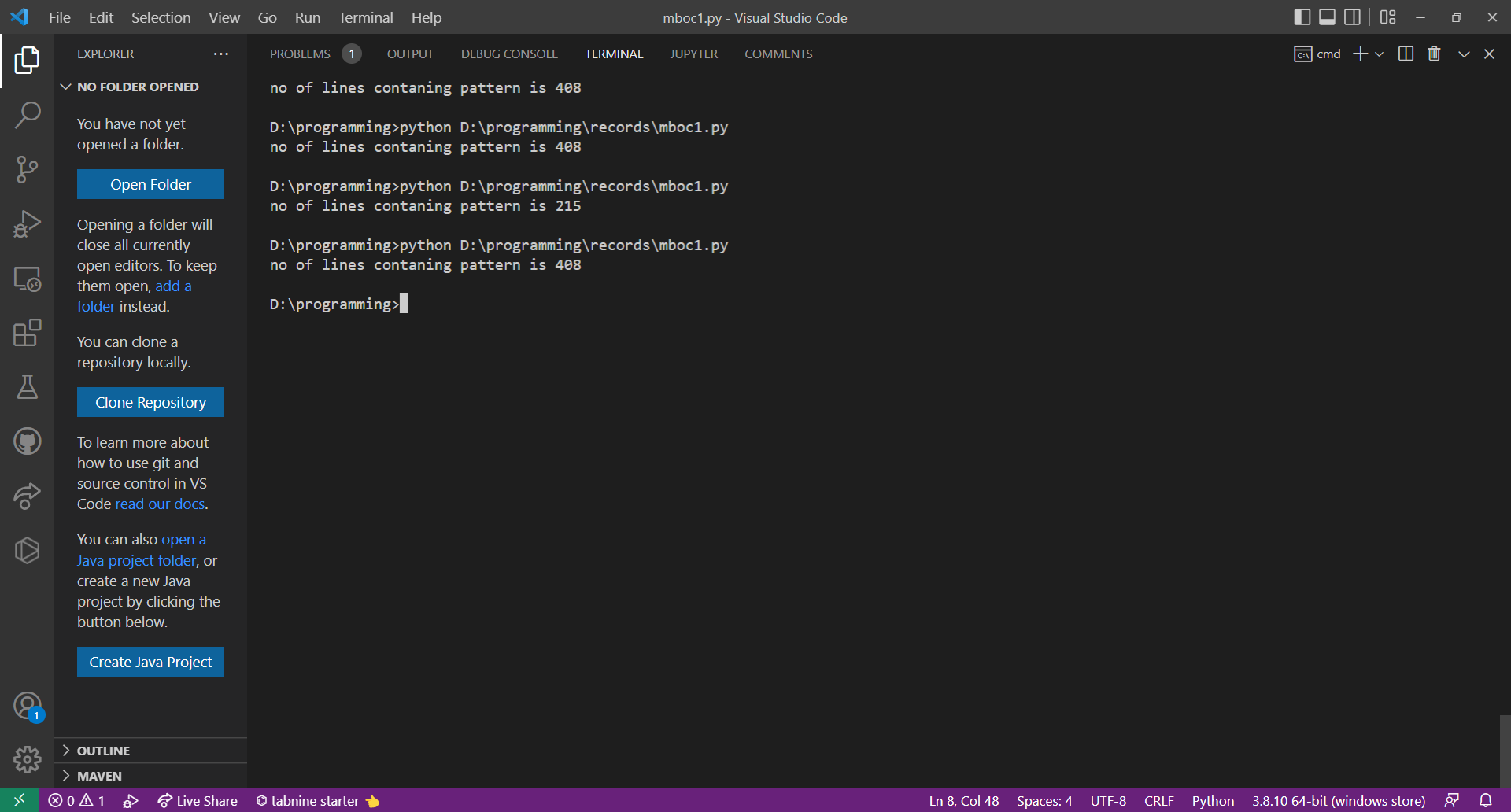
for i in f:

for match in re.finditer(pattern,i):

count+=1

print(“No lines that has @uct.ac.za is: {}”.format(count))

**Output:**



6.Write a python program to read file called mbox.txt display all the lines that starts with ‘X’ ‘-‘(ex:X-name:).

**SOURCE CODE:**

import re

f=open('mbox.txt','r')

text="X-"

lines=f.readlines()

for line in lines:

if text in line:

line=line.split(" ")

if text in line[0]:

print (line)

**Output:**

Text

Description automatically generated

7.Write a python program to demonstrate the shape class

**SOURCE CODE:**

class Shape:

def area(self):

print("Area of the shape")

class rectangle(Shape):

def area(self):

A=int(input("Enter length of the rectangle:"))

B=int(input("Enter breadth of the rectangle:"))

print("Area of the rectangle is: {}".format(A\*B))

class circle(Shape):

def area(self):

A=int(input("Enter radius of the circle:"))

print("Area of the circle is: {}".format(3.14\*pow(A,2)))

obj1=Shape()

obj2=rectangle()

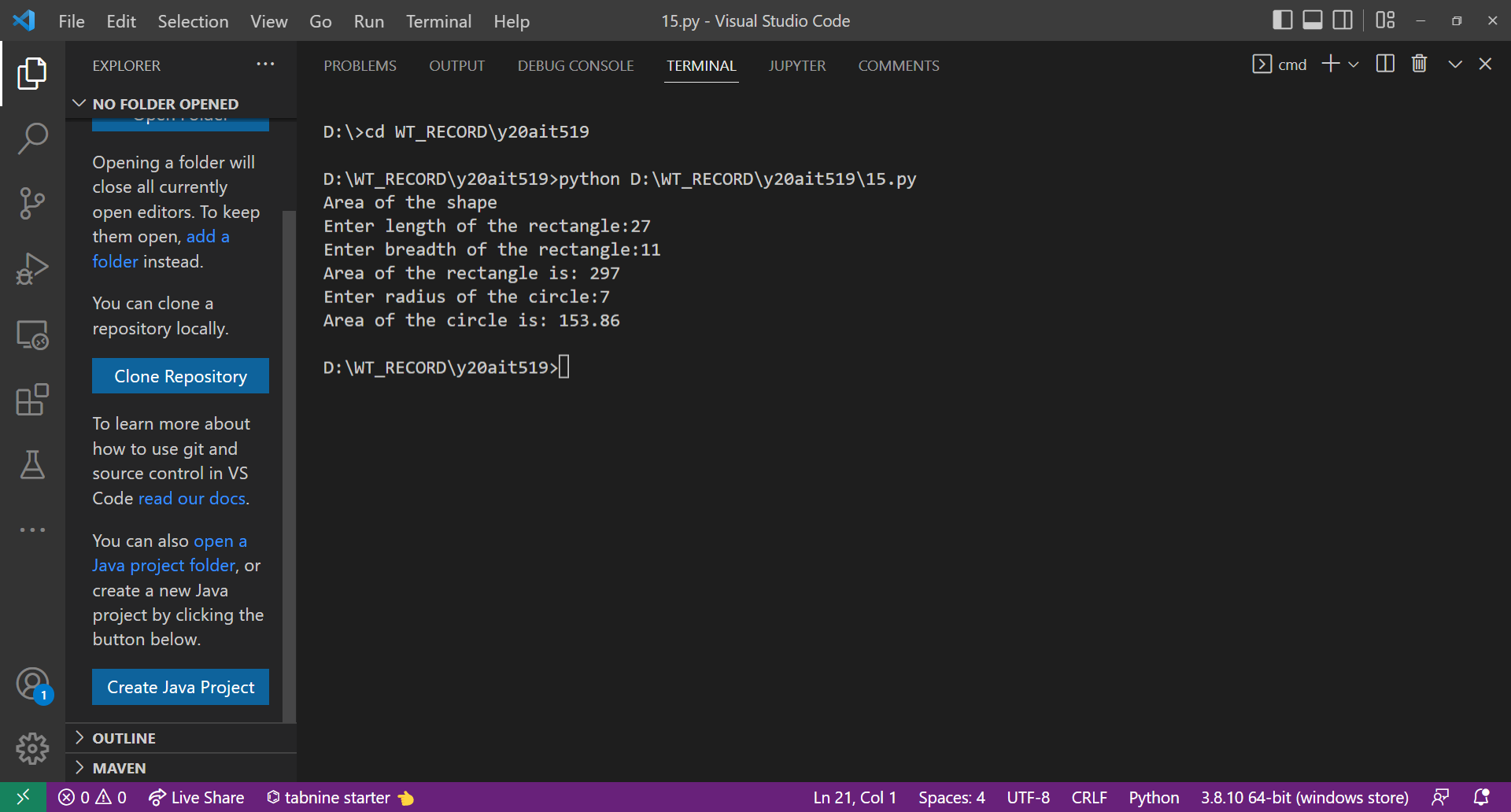
obj3=circle()

obj1.area()

obj2.area()

obj3.area()

**OUTPUT:**



7.Write a python program to calculate the student marks by using methods create, search, delete, display etc.

**SOURCE CODE:**

class student:

def init(self):

self.rollno = 0

self.regno=0

self.sub1=0

self.sub2=0

self.sub3=0

self.sub4=0

self.sub5=0

self.name=" "

self.perc=0

def create(self):

self.name=input("enter your name")

self.rollno=int(input("enter your rollno"))

self.regno=input("enter your regno")

self.sub1=int(input("enter maths marks"))

self.sub2=int(input("enter DBMS marks"))

self.sub3=int(input("enter PYTHON marks"))

self.sub4=int(input("enter DAA marks"))

self.sub5=int(input("enter WT marks"))

self.perc=((self.sub1+self.sub2+self.sub3+self.sub4+self.sub5)/500)\*100

print("student record is created")

def display(self):

regg=input("Enter register number")

for i in list1:

if i.regno==regg:

print("student name: ",i.name)

print("roll number: ",i.rollno)

print("register number: ",i.regno)

print(" maths marks: ",i.sub1)

print(" DBMS marks: ",i.sub2)

print(" PYTHON marks: ",i.sub3)

print(" DAA marks: ",i.sub4)

print(" WT marks: ",i.sub5)

def search(self):

c=input("enter regno to search record: ")

for i in list1:

if i.regno==c:

print("student name: ",i.name)

print("regdno: ",i.regno)

print("rollno: ",i.rollno)

print("percentage: " ,i.perc)

def update(self):

n=input("enter regno to update record:")

for i in list1:

if i.regno==n:

while True:

print("which content you want to update")

print("1.name\n2.rollno\n3.maths marks\n4.DBMS marks\n5.PYTHON marks\n6.DAA marks\n7.WT marks")

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

if ch==1:

n1=input("enter updated name:")

i.name=n1

elif ch==2:

n1=int(input("enter updated roll number :"))

i.rollno=n1

elif ch==3:

n1=int(input("enter updated s1 marks :"))

i.sub1=n1

elif ch==4:

n1=int(input("enter updated s2 marks :"))

i.sub2=n1

elif ch==5:

n1=int(input("enter updated s3 marks :"))

i.sub3=n1

elif ch==6:

n1=int(input("enter updated s4 marks :"))

i.sub4=n1

elif ch==7:

n1=int(input("enter updated s5 marks :"))

i.sub5=n1

else:

print("please enter a valid input 1-7")

choice=input("do you to update any other y/n:")

if choice=="n" or ch=="N":

break

def delete(self):

rgg=input("enter register number")

for i in list1:

if i.regno==rgg:

list1.remove(i)

print("student record is deleted")

list1=[]

ch=1

print("student records")

while ch!=0:

ob=student()

print("\n1.create\n2.display\n3.search\n4.delete\n5.update\n6.exit")

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

if (ch==1):

ob.create()

list1.append(ob)

elif (ch==2):

for i in list1:

print("student name:",i.name)

print("regdno:",i.regno)

print("rollno:",i.rollno)

print("percentage:",i.perc)

elif (ch==3):

ob.search()

elif (ch==4):

ob.delete()

elif (ch==5):

ob.update()

elif (ch==6):

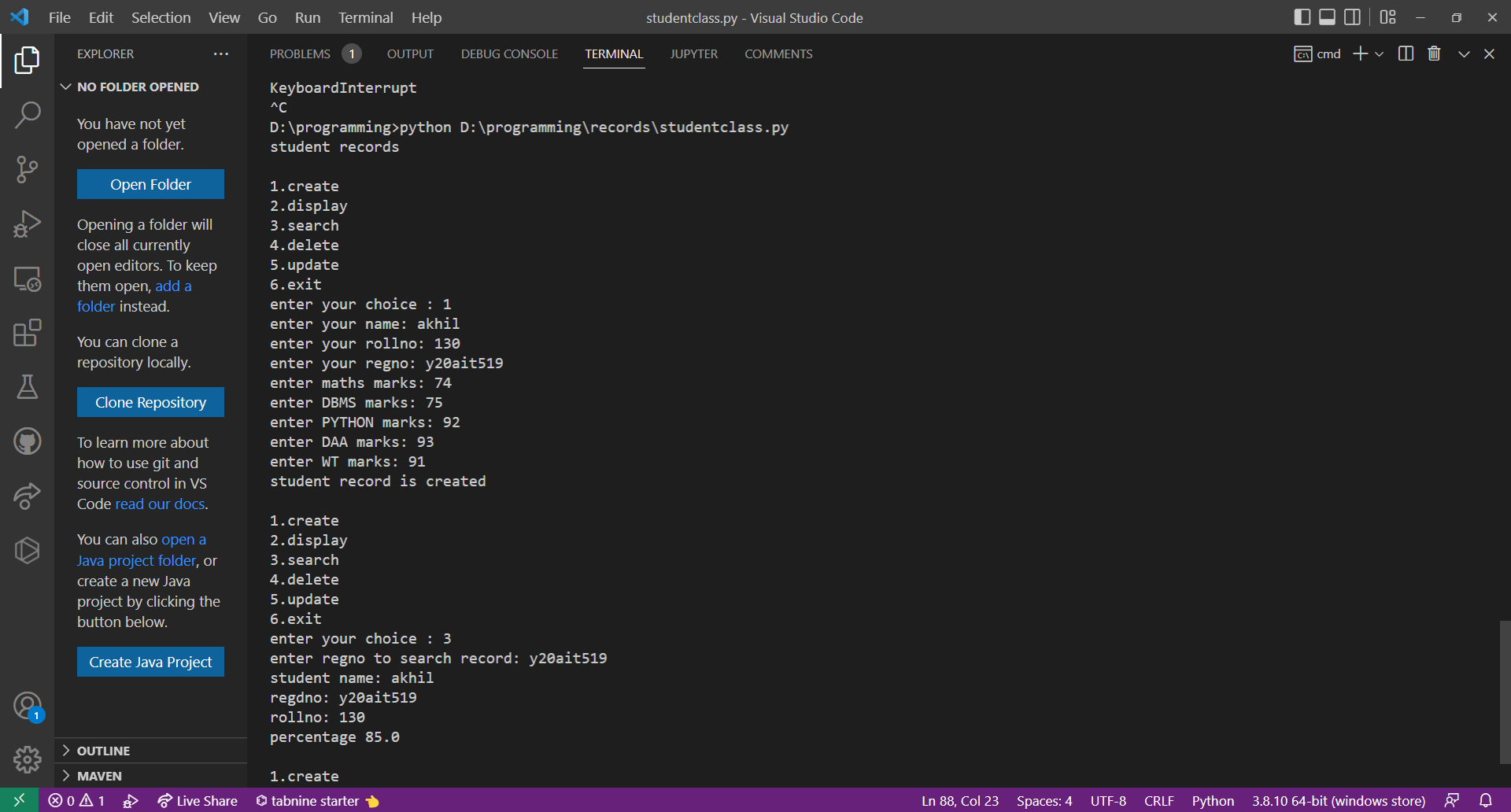
print("exiting")

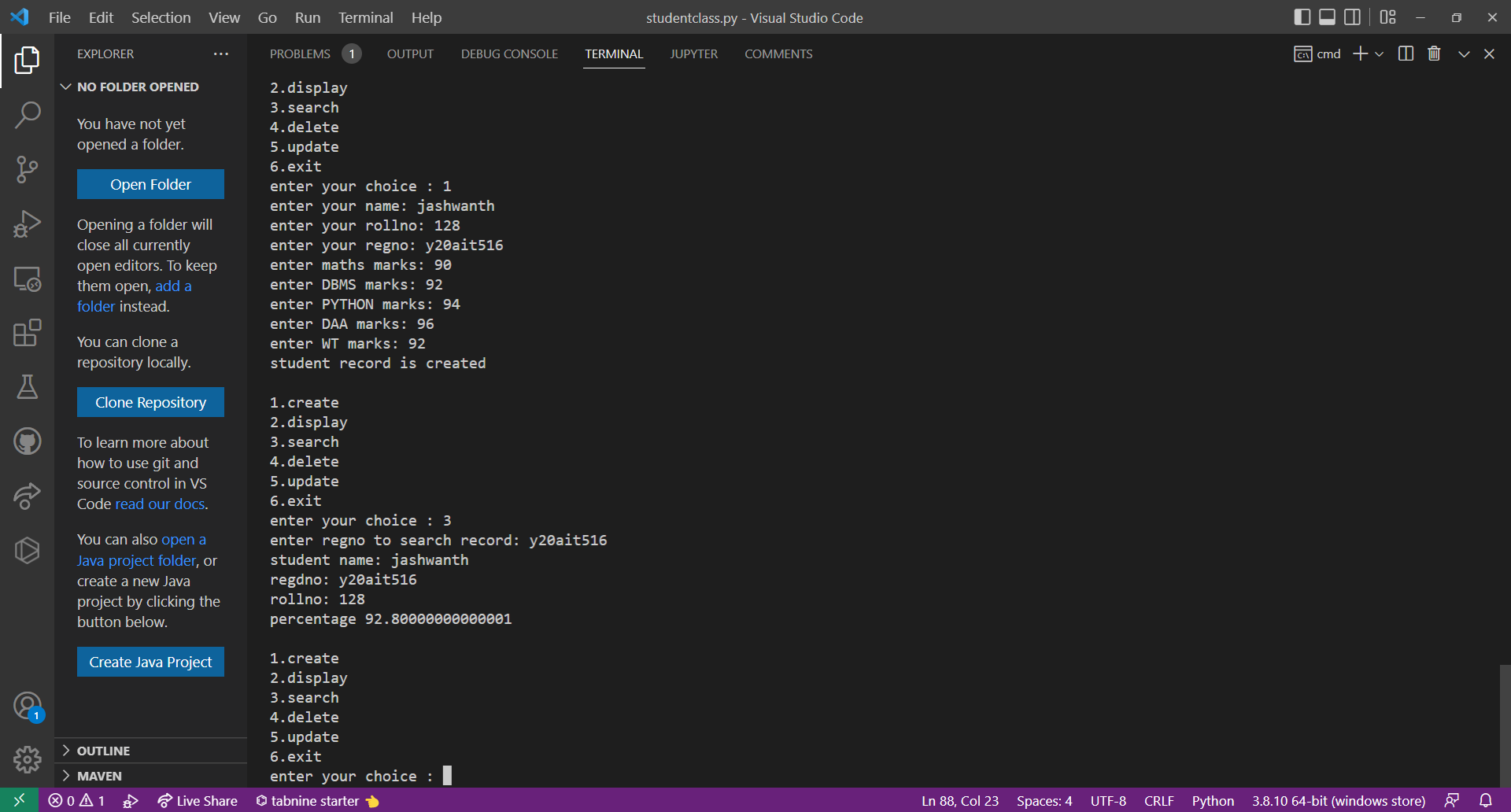
break

else:

print("invalid choice")

**output:**

****

****

A screenshot of a computer

Description automatically generated with medium confidence

**A screenshot of a computer

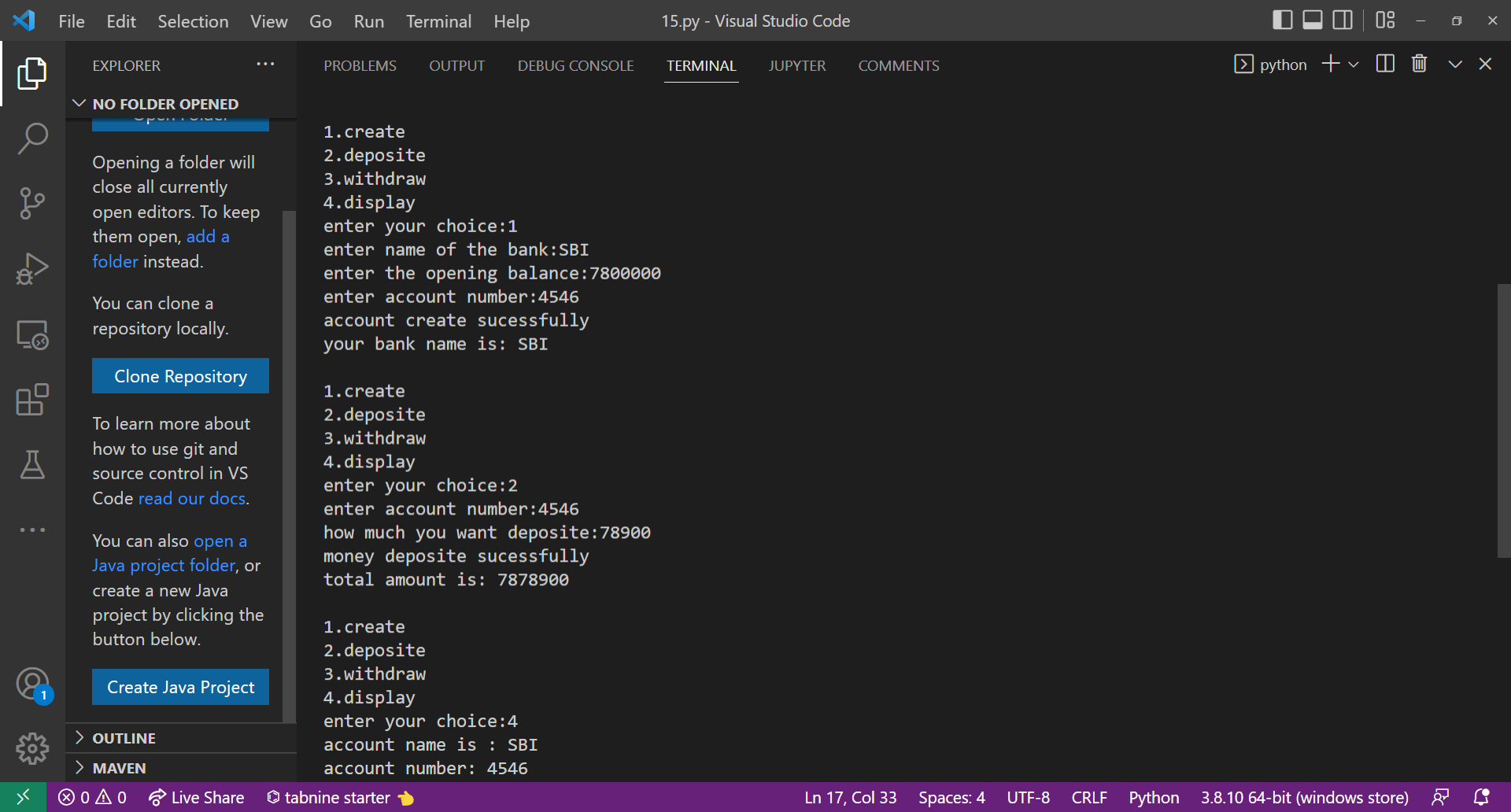
Description automatically generated with medium confidence**

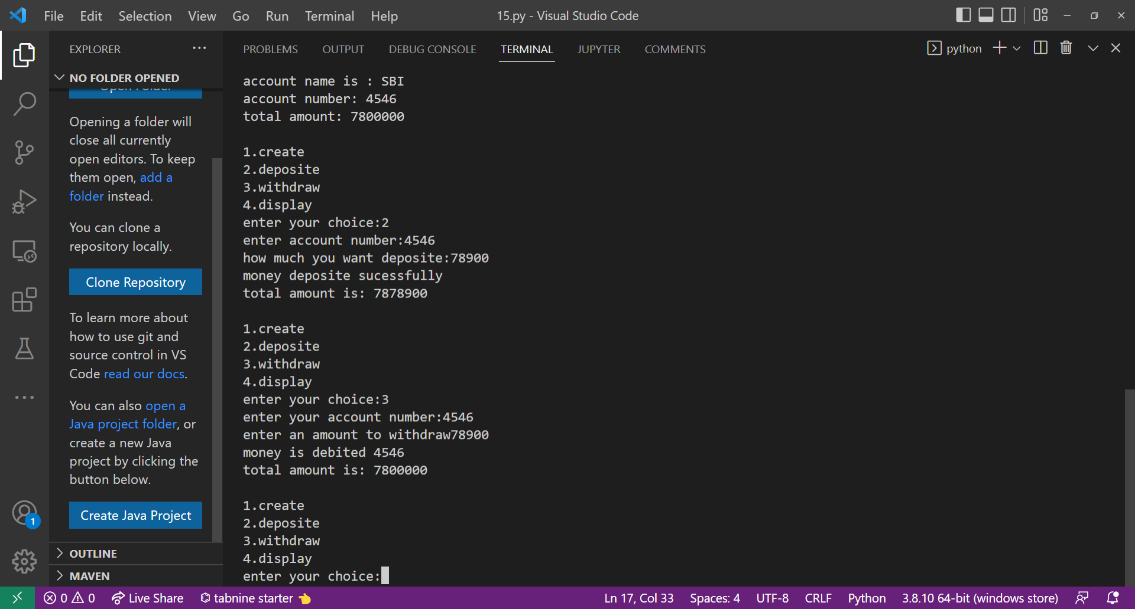
9.Write a python program to demonstrate the Bank account using methods create, deposite, withdraw, display etc.

**SOURCE CODE**:

class bank\_account():  
    def \_\_init\_\_(self):  
        self.balance=0  
        self.account=0  
        [self.name](http://self.name/)=""  
    def create(self):  
        [self.name](http://self.name/)=input("enter name of the bank")  
        self.balance=int(input("enter the opening balance"))  
        self.account=int(input("enter account number"))  
        print("account create sucessfully")  
        print("your bank name is:",[self.name](http://self.name/))  
    def deposite(self):  
       d=int(input("enter account number"))  
       if d==self.account:  
           b=int(input("how much you want deposite"))  
           self.balance+=b  
           print("money deposite sucessfully")  
           print("total amount is:",self.balance)  
  
    def withdraw(self):  
        w=int(input("enter your account number"))  
        if w==self.account:  
            h=int(input("enter an amount to withdraw"))  
            self.balance-=h  
            print("money is debited",w)  
            print("total amount is:",self.balance)  
    def display(self):  
        print("account name is :",[self.name](http://self.name/))  
        print("account number:",self.account)  
        print("total amount",self.balance)  
           
s=bank\_account()  
while True:  
    print("\n1.create\n2.deposite\n3.withdraw\n4.display")  
    ch=int(input("enter your choice"))  
    if ch==1:  
        s.create()  
    elif ch==2:  
        s.deposite()  
    elif ch==3:  
        s.withdraw()  
    elif ch==4:  
        s.display()  
    else:  
        print("invalid choice")  
        c=input("do you want to continue Y/N")  
        if c=='N' or c=='n':  
             break;

**Output:**





A screenshot of a computer

Description automatically generated with medium confidence

**10.**Write a python program to demonstrate database operations to persist student details using methods create, update, search, delete etc.,

**SOURCE CODE**:

import sqlite3

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

cur = conn.cursor()

cur.execute(''' SELECT count(name) FROM sqlite\_master WHERE type='table' AND name='STUDENTS2' ''')

if cur.fetchone()[0]==0:

cur.execute('CREATE TABLE STUDENTS2 ( REGD varchar(10) PRIMARY KEY ,NAME VARCHAR(18),FATHERNAME VARCHAR(18),ADDRESS VARCHAR,MATH\_MARKS NUMBER,OOPS\_MARKS NUMBER,DS\_MARKS NUMBER,PERCENTAGE NUMBER )')

print("table created")

conn.commit()

students=[]

class student:

percentage=0

def create(self):

self.regd=input("enter the regd number:")

self.name=input("enter the your name:")

self.fname=input("enter the father name:")

self.address=input("enter your address:")

self.maths\_marks=int(input("enter your math’s marks:"))

self.oops\_marks=int(input("enter your oops marks:"))

self.ds\_marks=int(input("enter your ds marks:"))

self.percentage=int((self.maths\_marks+self.oops\_marks+self.ds\_marks)/3)

cur.execute('insert into STUDENTS2 values(?,?,?,?,?,?,?,?)',(self.regd,self.name,self.fname,self.address,self.maths\_marks,self.oops\_marks,self.ds\_marks,self.percentage))

print(“record created successfully”)

conn.commit

def update(self):

print("1.regdno\n2name\n3.father name\n4.address\n5.math\_marks\n6.oops\_marks\n7.ds\_marks")

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

i=input("enter the regd number:")

if ch==1:

c=input("enter the regd number you want to change:")

cur.execute('update students2 set regd = ? where regd =?',(i,c))

print(“updated successfully”)

elif ch==2:

c=input("enter the your name you want to change:")

cur.execute('update students2 set name = ? where regd =?',(c,i))

print(“updated successfully”)

elif ch==3:

c=input("enter the your father name you want to change:")

cur.execute('update students2 set father name = ? where regd =?',(c,i))

print(“updated successfully”)

elif ch==4:

c.address=input("enter the your address:")

cur.execute('update students2 set address = ? where regd=?',(c,i))

print(“updated successfully”)

elif ch==5:

c=int(input("enter math’s marks:"))

cur.execute('update students2 set math\_marks=? where regd=?',(c,i))

print(“updated successfully”)

elif ch==6:

c=int(input("enter oops marks you want to modify:"))

cur.execute('update students2 set oops\_marks=? where regd=?',(c,i))

print(“updated successfully”)

elif ch==7:

c=int(input("enter ds marks you want to modify:"))

cur.execute('update students2 set ds\_marks=? where regd=?',(c,i))

print(“updated successfully”)

else:

print("wrong choice")

if ch>4 and ch<8:

cur.execute('select \* from students2 where regd=?',(i,))

row = cur.fetchone()

sum=row[4]+row[5]+row[6]

percentage=sum/3

cur.execute('update students2 set percentage=? where regd=?',(percentage,i)

conn.commit()

def search(self):

c=input("enter the regd number you want to search:")

cur.execute('select \* from students2 where regd=?',(c,))

for records in cur:

print(records)

def delete(self):

c=input("enter the regd number you want to delete:")

cur.execute('delete from students2 where regd=?',(c,))

conn.commit()

def display(self):

cur.execute('select \* from students2')

f=cur.fetchall()

for e in f:

print(e)

while True:

o1=student()

ch=int(input("1.create\n2.update\n3.search\n4.delete\n 5.display"))

if ch==1:

o1.create()

students.append(o1)

elif ch==2:

o1.update()

elif ch==3:

o1.search()

elif ch==4:

o1.delete()

elif ch==5:

ad=input("enter admin login:")

if ad=="1234":

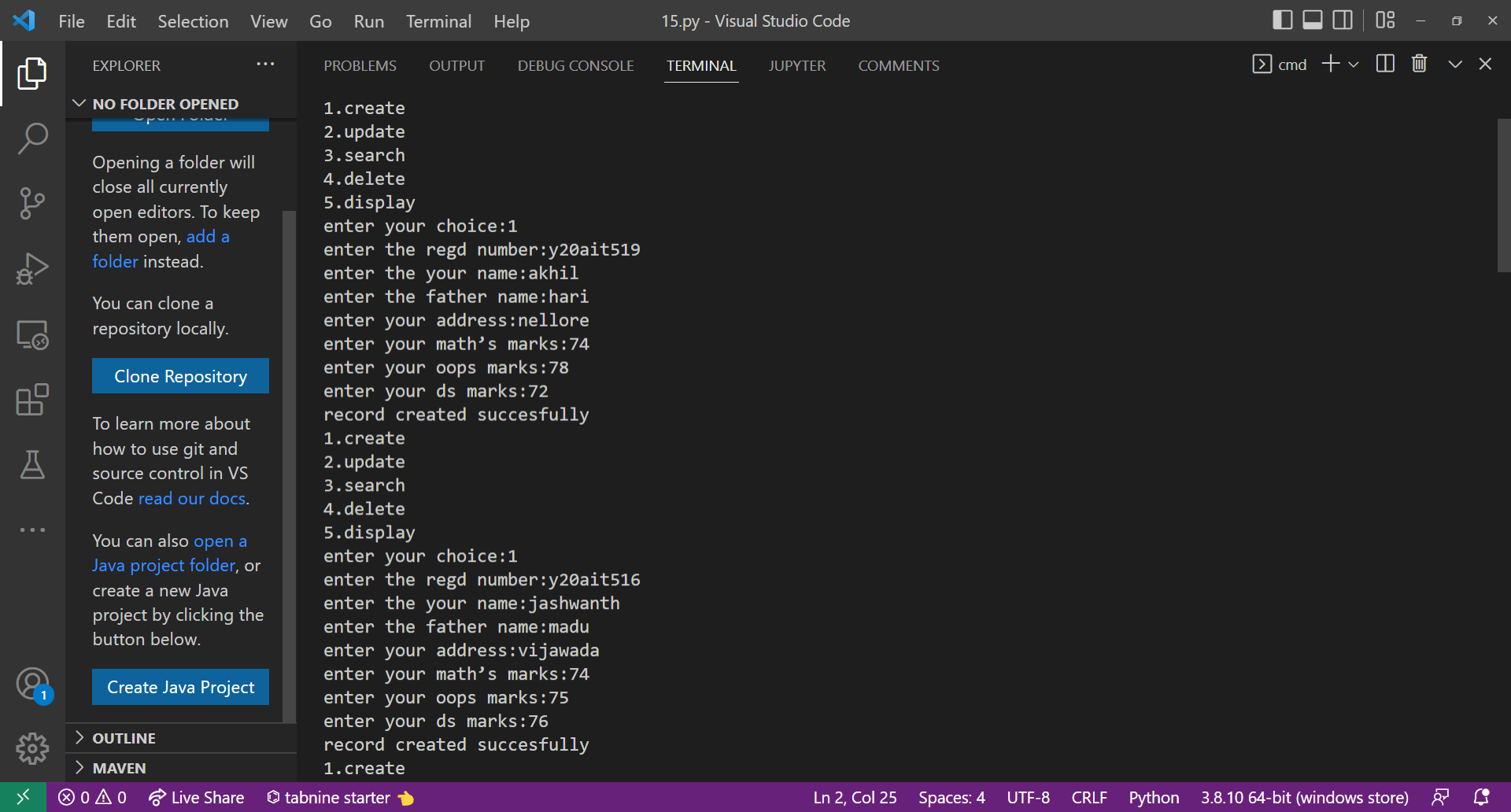
o1.display()

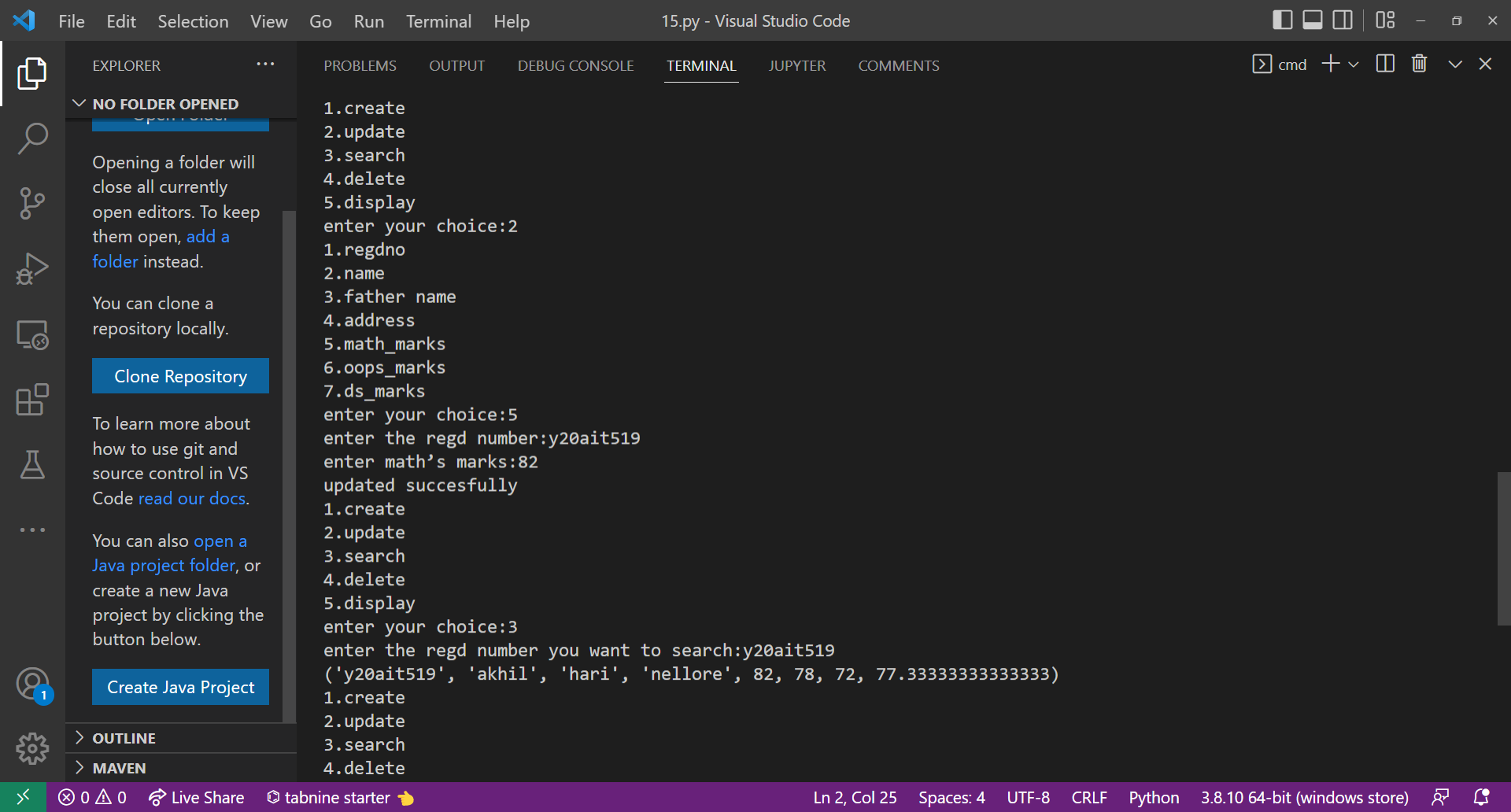
else:

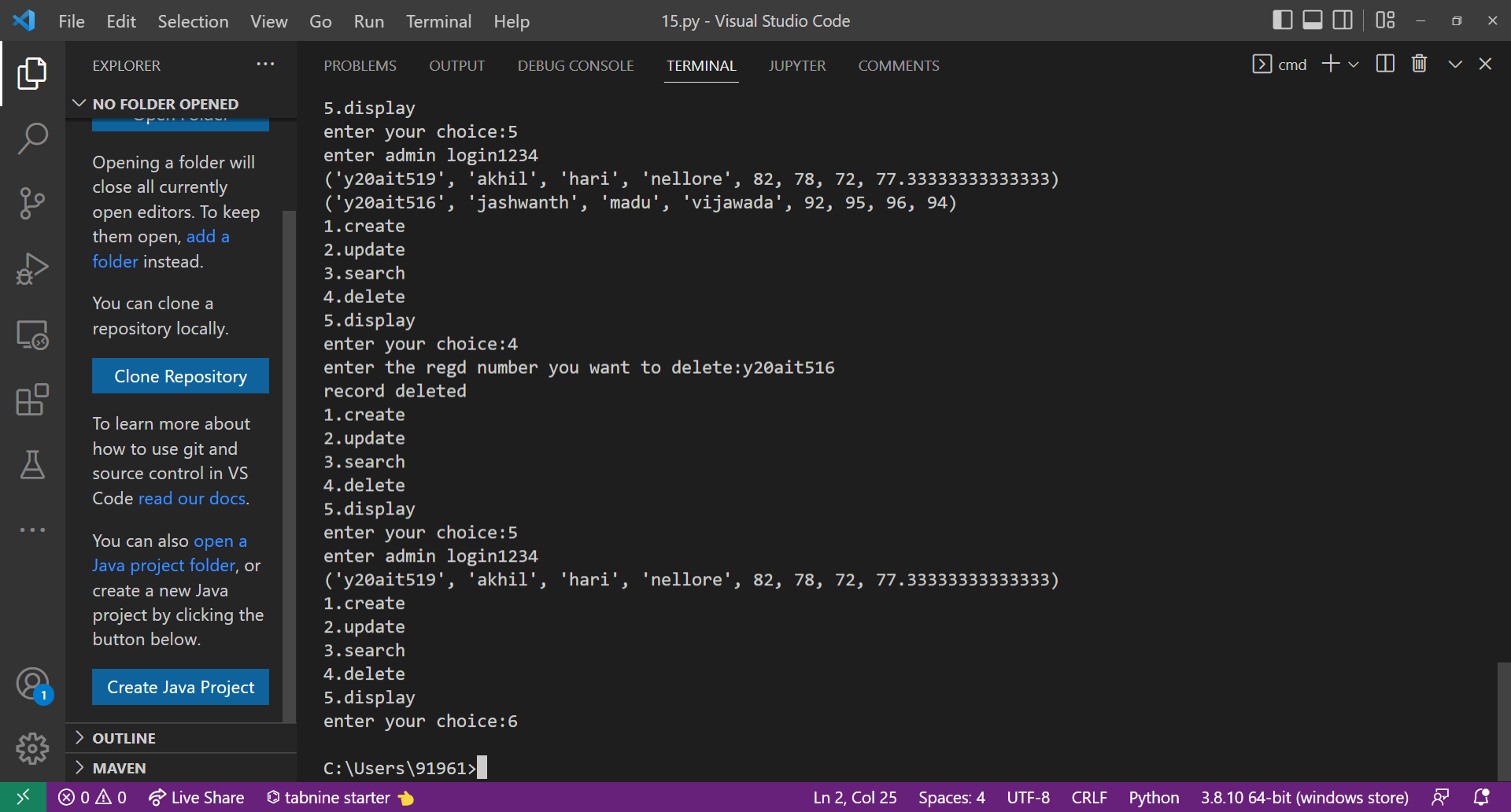
cur.close()

break

**Output:**

****

****

****

11. Write a python program to demonstrate queue using python

**Source code:**

class queue:

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

self.ele=0

def enqueue(self):

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

l.append(self.ele)

print("inserted sucessfully")

def dequeue(self):

if(len(l)==0):

print("the queue is empty")

else:

print("the deleted element is:",l.pop(0))

def display(self):

if(len(l)==0):

print("the queue is empty")

else:

for i in range(0,len(l),+1):

print(l[i],end=' ')

obj=queue()

**l=[]**

print("\*\* queue implementation in python\*\* \n")

while True:

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

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

if ch==1:

obj.enqueue()

elif ch==2:

obj.dequeue()

elif ch==3:

obj.display()

else:

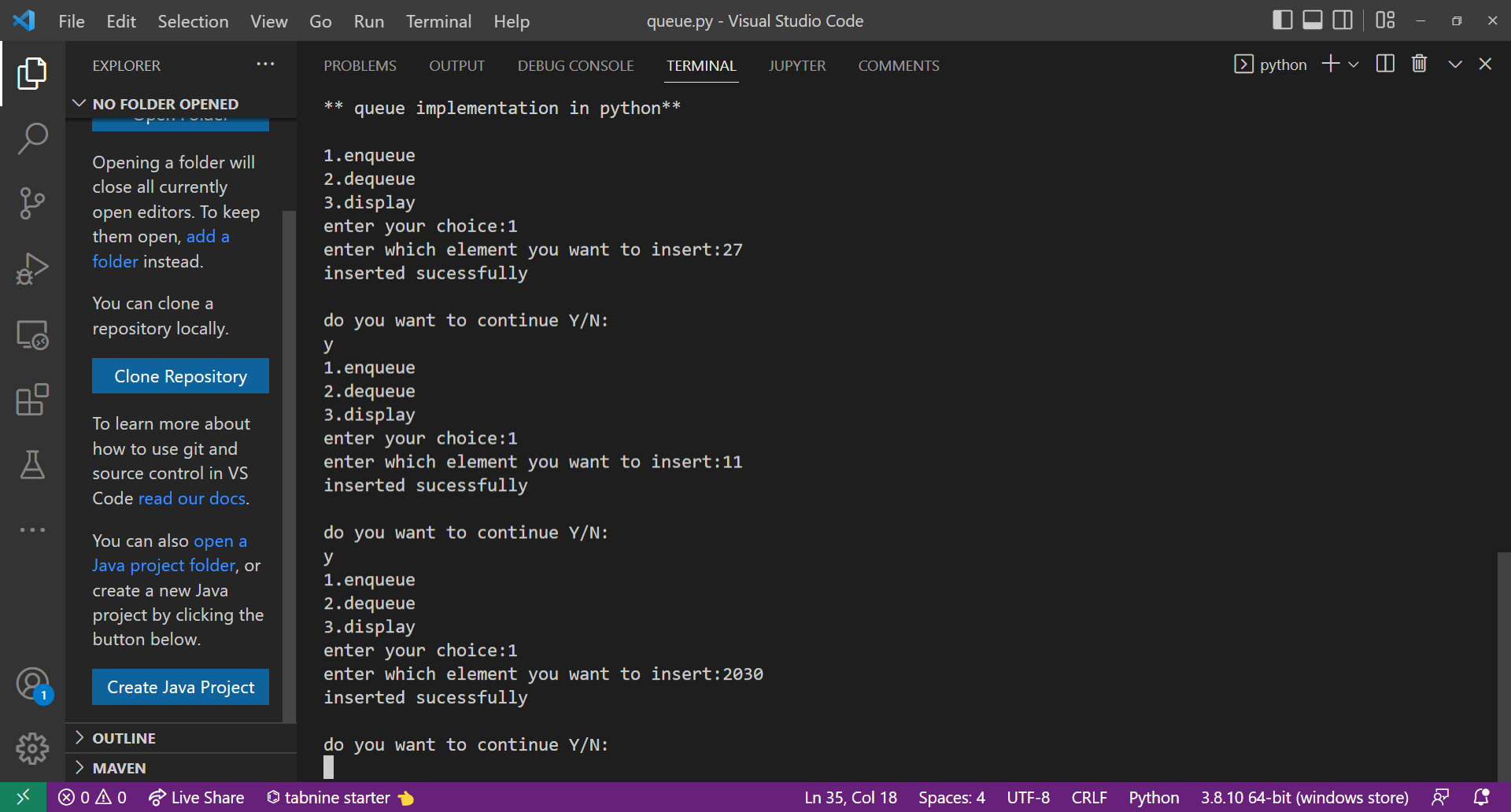
print("invalid choice\n")

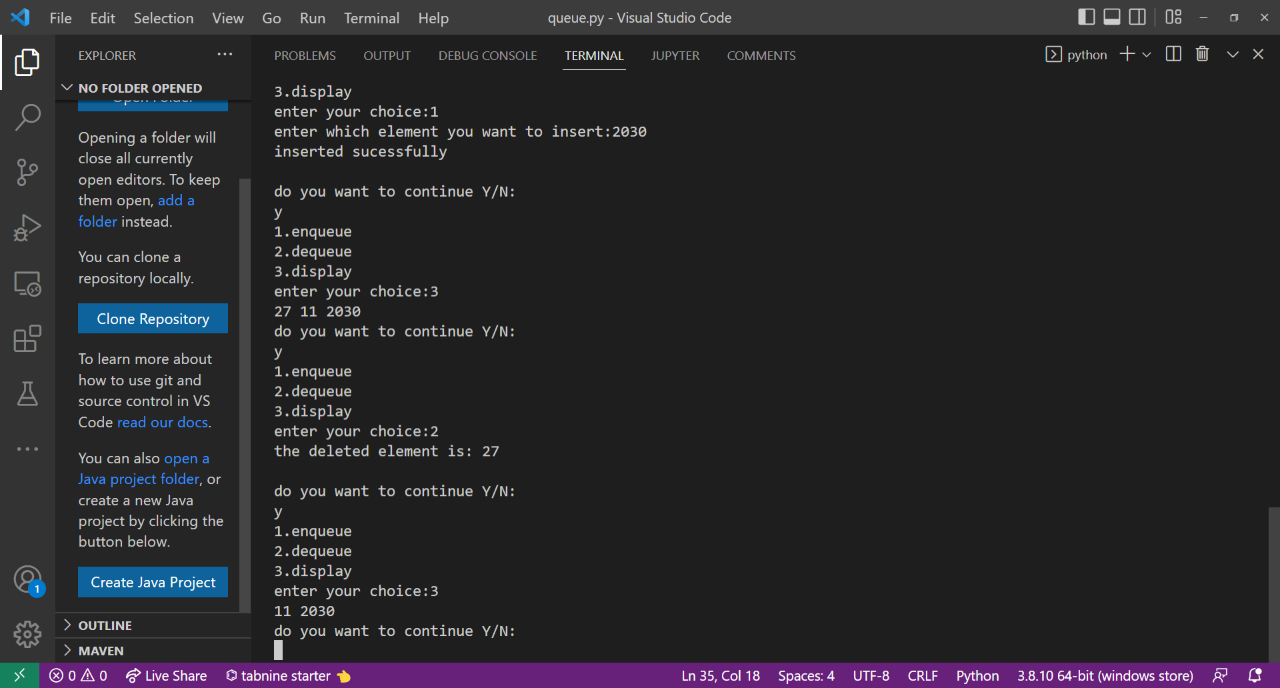
ask=input("do you want to continue Y/N: \n")

if ask=='n' or ask=='N':

break

OUTPUTS:





12. Write a python program to demonstrate the stack using python

Source code:

class stack:

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

self.ele=0

def push(self):

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

l.append(self.ele)

print("the element inserted sucessfully")

def pop1(self):

if(len(l)==0):

print("the stack is empty")

else:

print("the deleted element is",l.pop())

def display(self):

if(len(l)==0):

print("the stack is empty")

else:

print("the stack elements are")

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

print(l[i-1])

obj=stack()

l=[]

while True:

print("stack implementing using python")

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

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

if ch==1:

obj.push()

elif ch==2:

obj.pop1()

elif ch==3:

obj.display()

else:

print("invalid choice\n")

ask=input("do you want to continue Y/N:\n")

if ask=='n' or ask=='N':

break

Outputs:

