**COMPUTER SCIENCE**

**PRACTICAL FILE**

**DAV CENTENARY PUBLIC SCHOOL**

**NAME:- JAYESH NALWA**

**CLASS:- 12 A**

Q1) WAP to determine how many times a given letter occurs in a string

SCRIPT:-

count = 0

s = input("Enter a string : ")

l = input("Enter the letter to be found : ")

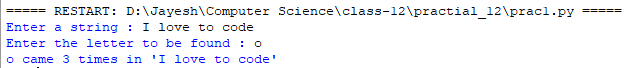
for i in s:

if i == l:

count+=1

print(f"{l} came {count} times in '{s}'")

OUTPUT :-



Q2) WAP to take a word from user and check if it is a palindrome

SCRIPT:-

word = input("Enter a word :")

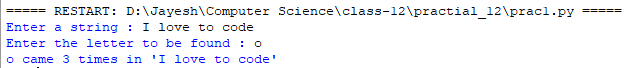
if word.lower() == word[::-1].lower():

print("It is a Palindrome")

else:

print("It is not a Palindrome")

OUTPUT :-



Q3) WAP to take the first number (a) ,common difference (d) and number of terms (n) from the user and display the arithmetic progression

SCRIPT :-

a = int(input("Enter the first number : "))

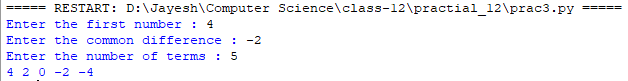
d = int(input("Enter the common difference : "))

n = int(input("Enter the number of terms : "))

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

print(f"{a}",end=' ')

a+=d

OUTPUT:- 

Q4) WAP to recursively find the factorial of a natural number

SCRIPT :-

def factorial(n):

if n<2:

return 1

else:

return n\*factorial(n-1)

n=int(input("Enter the number :"))

print("Factorial of",n,"is",factorial(n))

num = int(input("Enter the number :"))

fact = factorial(num)

print(fact)

OUTPUT :-



Q5) WAP to recursively find the fiboniccai series upto n tems and take n from the user

def fib(n):

if n==1:

return 0

elif n==2:

return 1

else:

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

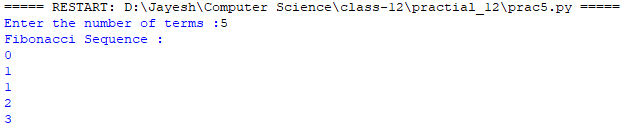
nterms = int(input("Enter the number of terms :"))

print("Fibonacci Sequence :")

for i in range(nterms):

print(fib(i))

OUTPUT :-



Q6) Write a recursive code to find the sum of all elements in a list

SCRIPT :-

def sum\_arr(arr,size):

if size == 0:

return 0

else:

return arr[size]+ sum\_arr(arr,size-1)

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

a= []

for i in range(0,n):

ele = int(input("Enter the element : "))

a.append(ele)

print("The list is :")

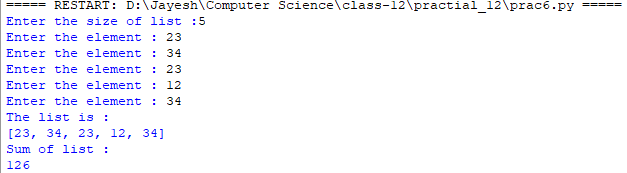
print(a)

print("Sum of list :")

b = sum\_arr(a,n)

print(b)

OUTPUT :-



Q7) Write a python menu driven program to show all stack operations performed on a list taken from the user

SCRIPT :-

def isEmpty(stk):

if stk == []:

return True

else:

return False

def Push(stk,item):

stk.append(item)

top = len(stk)-1

def Pop(stk):

if isEmpty(stk):

return "Underflow"

else:

item = stk.pop()

if len(stk) == 0:

top = None

else:

top = len(stk)-1

return item

def Peek(stk):

if isEmpty(stk):

return "Underflow"

else:

top = len(stk)-1

return stk[top]

def Display(stk):

if isEmpty(stk):

print("Stack is empty")

else:

top = len(stk)-1

print(stk[top],"<- top")

for a in range(top-1,-1,-1):

print(stk[a])

stack = []

top = None

while True:

print("STACK OPERATIONS")

print("1. Push")

print("2. Pop")

print("3. Peek")

print("4. Display")

print("5. Exit")

ch = int(input("Enter the choice (1-5) :"))

if ch == 1:

item = int(input("Enter the item :"))

Push(stack,item)

elif ch == 2:

item = Pop(stack)

if item == "Underflow":

print("Underflow ! Stack is empty !")

else:

print("Popped item is",item)

elif ch == 3:

item = Peek(stack)

if item == "Underflow":

print("Underflow ! Stack is empty !")

else:

print("Topmost item is",item)

elif ch == 4:

Display(stack)

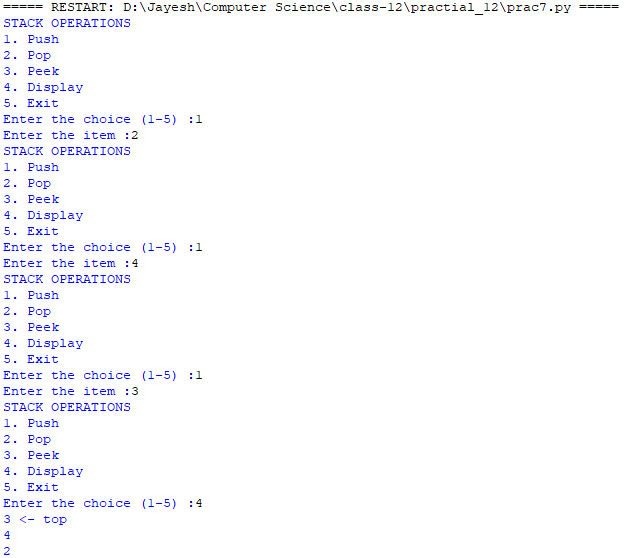
elif ch == 5:

break

else:

print("Invalid Choice !!!!")

OUTPUT :-



Q8) Write a python menu driven program to show all queue operations performed on a list taken from the user

SCRIPT:-

def isEmpty(qu):

if qu == []:

return True

else:

return False

def Enqueue(qu,item):

qu.append(item)

if len(qu) == 1:

front = rear = 0

else:

rear = len(qu)-1

def Dequeue(qu):

if isEmpty(qu):

return "Underflow"

else:

item = qu.pop(0)

if len(qu) == 0:

front = rear = None

return item

def Peek(qu):

if isEmpty(qu):

return "Underflow"

else:

front = 0

return qu[front]

def Display(qu):

if isEmpty(qu):

print("Queue Empty !")

elif len(qu) == 1:

print(qu[0],"<== front,rear")

else:

front = 0

rear = len(qu)-1

print(qu[front],"<- front")

for a in range(1,rear):

print(qu[a])

print(qu[rear],"<- rear")

queue = []

front = None

while True:

print("QUEUE OPERATIONS")

print("1. Enqueue")

print("2. Dequeue")

print("3. Peek")

print("4. Display")

print("5. Exit")

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

if ch == 1:

item = int(input("enter item :"))

Enqueue(queue,item)

elif ch == 2:

item = Dequeue(queue)

if item == "Underflow":

print("Underflow ! Queue is empty !")

else:

print("Dequeue-ed item is",item)

elif ch == 3:

item = Peek(queue)

if item == "Underflow":

print("Queue is empty !")

else:

print("Frontmost item is",item)

elif ch == 4:

Display(queue)

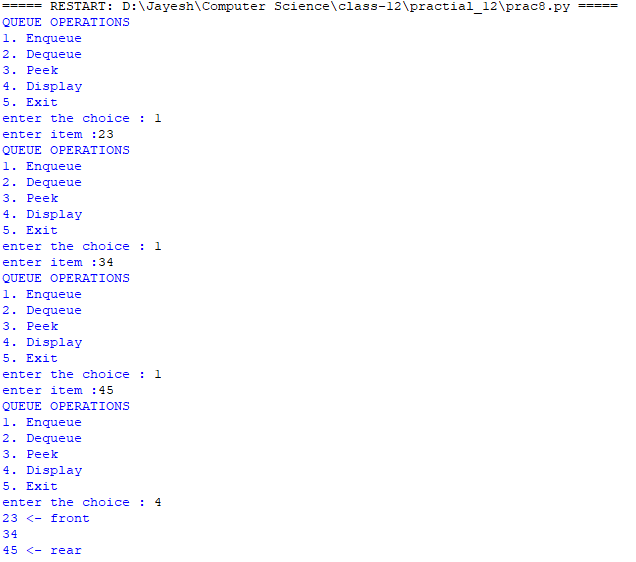
elif ch == 5:

break

else:

print("Invalid choice !")

OUTPUT :-



Q9) WAP to open a file 'practical.txt' and count the number of lines starting with T or t

SCRIPT :-

file = open("practical.txt","r")

lines = file.read().split('\n')

count = 0

for i in lines :

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

count += 1

print("the number of t's :",count)

OUTPUT :-



Q10) WAP to open a file word.txt and count the number of "the's" in the file

SCRIPT:-

file = open("word.txt","r")

lines = file.read().split()

count = 0

for i in lines :

if i == 'the' or i == 'The':

count += 1

print("the number of the's :",count)

OUTPUT:-



Q11) WAP to create five dictionaries(sample : Amit={'Name':'Amit','class':'12A','stream':'non-med','percentage':'97'}) where keys signifies the roll number. Write two functions storedata and loaddata using pickle module using binary file.

SCRIPT :-

import pickle

def storedata():

Amit = {"Name":"Amit",'class':'12 A','stream':'non-med','percentage':97}

Rahul = {"Name":"Rahul",'class':'12 B','stream':'med','percentage':96}

Garima = {"Name":"Garima",'class':'12 C','stream':'commerce','percentage':95}

Lia = {"Name":"Lia",'class':'12 D','stream':'Arts','percentage':94}

Camey = {"Name":"Camey",'class':'12 E','stream':'med','percentage':93}

db = {}

db['Amit'] = Amit

db['Rahul'] = Rahul

db['Garima'] = Garima

db['Lia'] = Lia

db['Camey'] = Camey

dbfile = open("pickle\_data","wb")

pickle.dump(db,dbfile)

dbfile.close()

def loaddata():

dfile = open("pickle\_data","rb")

dbl = pickle.load(dfile)

for keys in dbl:

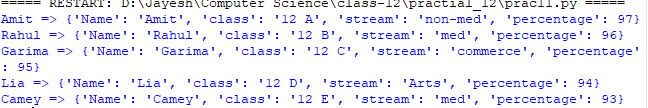
print(keys,"=>",dbl[keys])

dfile.close()

storedata()

loaddata()

OUTPUT :-



Q12) Rahul wants to play tambola and want a fair way to generate a number between 1 to 99. WAP to help Rahul with his game.

SCRIPT :-

import random

l = []

while True:

num = random.randint(1,99)

l.append(num)

ch = input("Do you want to continue ? :")

if ch == 'y':

for i in l:

if i == num:

break

print(i)

else:

break

OUTPUT :-

