March 15, 2024

Darshil Solanki

PYTHON

assignment

Certificate

This is to certify that **DARSHILKUMAR MANSUKHBHAI SOLANKI** of MCA SEM-2 with PRN NO: 8023057262 has successfully completed the **PYTHON PROGRAMMING** lab work during academic year 2023-24.

Date of Completion: 15/03/2024

Signature of HOD Signature of Professor

Table of Contents

[1. Write a python program to design calculator. 4](#_Toc161419728)

[2. Write a program to find the area of circle. 4](#_Toc161419729)

[3. Write a program to generate two random number and perform left and right shift of generated number. 5](#_Toc161419730)

[4. Write a program to swap two numbers. 5](#_Toc161419731)

[5. Write a Python program to get a string made of the first 2 and the last 2 chars from a given a string. If the string length is less than 2, return instead the empty string. 5](#_Toc161419732)

[6. Write a program to find particular character form given string and create new list for the index of that character. 6](#_Toc161419733)

[7. Write a program to compute the frequency of largest element in list. 6](#_Toc161419734)

[8. Substract two list elements and output new list if the element in the first list are greater. 6](#_Toc161419735)

[9. Print all the pelindrom numbers in a list. 6](#_Toc161419736)

[10. Find the max length sublist in a nested list. 7](#_Toc161419737)

[11. Write a Python program to check if a 3 digit number is Armstrong number or not. 7](#_Toc161419738)

[12. Write Python Program to reverse a number and also find the Sum of digits in the reversed number. Prompt the user for input. 7](#_Toc161419739)

[13. Write Python code to check if a given year is a leap year or not. 8](#_Toc161419740)

[14. Write Python program to find the GCD of two positive numbers. 8](#_Toc161419741)

[15. Write Python program to count the total number of vowels, consonants and blanks in a String. 9](#_Toc161419742)

[16. Write Python code to find Mean, Variance and Standard Deviation for a list of numbers. 9](#_Toc161419743)

[17. Write a program that accepts a sentence and calculate the number of digits, uppercase and lowercase letters. 10](#_Toc161419744)

[18. Write Python code to sort a sequence of names according to their alphabetical order without using sort() function. 10](#_Toc161419745)

[19. Write Python code to find the factorial of a number 11](#_Toc161419746)

[20. Write Python program to perform a linear search for a given Key number in the list and report Success or Failure. 11](#_Toc161419747)

[21. Write Python Program to count the number of times an item appears in the list. 12](#_Toc161419748)

[22. Write a program to calculate min and max values from a list of tuples. 12](#_Toc161419749)

[23. Write a program to merge two list into list of tuples. 12](#_Toc161419750)

[24. Given a list of tuples containing both int and string remove all the string from list of tuples. 13](#_Toc161419751)

[25. WAP to find max and min value in a set and also find the length of the set 13](#_Toc161419752)

[26. WAP to create union and intersection of sets 13](#_Toc161419753)

[27. WAP to count no of vowels using set in a given string. 14](#_Toc161419754)

[28. Write a python program to create a dictionary for library. 14](#_Toc161419755)

[29. Write a python program to create computer accessory dictionary and replace dictionary value if price is less than 500. 15](#_Toc161419756)

[30. Write a python program to match key values in two dictionary. 16](#_Toc161419757)

[31. Write a function to calculate the power of number raised to other. 16](#_Toc161419758)

[32. Write a password generator function in python which should generate random passwords every time the user ask for a new password. Strong password should be a mix of lower case, uppercase, number and symbol. 16](#_Toc161419759)

[33. Write a program to read two double values from command line, perform division among them and display answer. Handle all possible errors using Exception Handling Mechanism and display appropriate message.(**Use one try and multiple catches**). 17](#_Toc161419760)

[34. Write a method to compute addition of two matrices to get a resultant matrix. Call this method in main to have A= B+C+D(where A,B,C,D are matrices). 17](#_Toc161419761)

[35. Write a python program to read file contain and if file is not available then handle it using appropriate exception handling. 18](#_Toc161419762)

[36. Write a python program to read two files and concate it output and write it in third file. 18](#_Toc161419763)

[37. Write a python class named circle constructed by a radius and two methods which will compute area and perometer. 19](#_Toc161419764)

[38. Create a class Demo and also create method test () in it. Overload test () in four ways. First version takes no parameter, the second takes one integer parameter, and the third takes two integer parameters and fourth takes one double parameter. 19](#_Toc161419765)

[39. create an abstract class shape and derive rectangle and circle from shape class. Implement abstract method of shape class in rectangle and circle class. Shape class contains: origin (x,y) as data member Display() and area() as abstract methods.Circle cotains : radious, Rectangle contains : length & width ( user inheritance , overloading and overriding concept.) 20](#_Toc161419766)

[40. Write a python program to convert date of yyyy-mm-dd to dd-mm-yyyy. 21](#_Toc161419767)

[41. Write a python program to find occurrence and position of substrings within a string. 21](#_Toc161419768)

[42. Write a python program to find sequences of one uppercase letter followed by lowercase letter. 21](#_Toc161419769)

[43. Write a python program to create calculator using Tkinter. 22](#_Toc161419770)

[44. Write a python program to create student registration form and collect the data provied by student in database. 23](#_Toc161419771)

### Write a python program to design calculator.

flag='1'

while(flag!='0'):

a=int(input("Enter first value:"))

b=int(input("Enter second value:"))

flag=input("Enter Operation or 0 to exit:")

if(flag=='+'):

print("A: ",a," B: ",b, "a+b:", a+b)

elif(flag=='-'):

print("A: ",a," B: ", b, "a-b:", a-b)

elif(flag=='\*'):

print("A: ",a," B: ", b, "a\*b:", a\*b)

elif(flag=='/'):

print("A: ",a," B: ", b, "a/b:", a/b)

elif(flag=='%'):

print("A: ",a," B: ", b, "a%b:", a%b)

elif(flag=='//'):

print("A: ",a," B: ", b, "a//b:", a//b)

elif(flag=='\*\*'):

print("A: ",a," B: ", b, "a\*\*b:", a\*\*b)

elif(flag=='0'):

break

else:

print("Unexpected Choice!!")

**Output:**

Enter first value:3

Enter second value:5

Enter Operation or 0 to exit: \*

A: 3 B: 5 a\*b: 15

Enter first value:0

Enter second value:4

Enter Operation or 0 to exit:0

### Write a program to find the area of circle.

r=int(input("Enter a radius:"))

print("Area of circle with radius ",r,"is: ",3.14\*r\*r)

**Output:**

Enter a radius:5

Area of circle with radius 5 is: 78.5

### Write a program to generate two random number and perform left and right shift of generated number.

import random

a=random.randrange(1,10)

b=random.randrange(1,10)

print("Generated Numbers are a: ",a," b: ",b)

print("a>>b:",a>>b)

print("Numbers are a: ",a," b: ",b)

print("b<<a: ",b<<a)

print("Numbers are a: ",a," b: ",b)

**Output:**

Generated Numbers are a: 5 b: 3

a>>b: 0

Numbers are a: 5 b: 3

b<<a: 96

Numbers are a: 5 b: 3

### Write a program to swap two numbers.

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

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

print("a: ",a," b: ",b)

a,b=b,a

print("after swapping")

print("a: ",a," b: ",b)

**Output:**

Enter a:3

Enter b:2

a: 3 b: 2

after swapping

a: 2 b: 3

### Write a Python program to get a string made of the first 2 and the last 2 chars from a given a string. If the string length is less than 2, return instead the empty string.

str=input("Enter a string:")

newStr=str if len(str)<3 else str[0:2]+str[-2:]

print("Newstr: ",newStr)

**Output:**

Enter a string:Hello

Newstr: Helo

### Write a program to find particular character form given string and create new list for the index of that character.

str=input("Enter a string:")

c=input("Enter a character to find:")

l=[]

for i in range(len(str)):

if(str[i]==c):

l.append(i)

print(l)

**Output:**

Enter a string:This is testing String.

Enter a character to find:i

[2, 5, 12, 19]

### Write a program to compute the frequency of largest element in list.

li=[12,223,223,32,21,34,2,3,32,12]

Maxli=max(li)

print(li.count(Maxli))

**Output:**

2

### Substract two list elements and output new list if the element in the first list are greater.

l1=[12,10,11,23,21,23,34,45,38,89]

l2=[11,23,34,12,34,4,23,34,22,10]

l3=[]

for i in range(len(l1)):

if(l1[i]>l2[i]):

l3.append(l1[i]-l2[i])

print(l3)

**Output:**

[1, 11, 19, 11, 11, 16, 79]

### Print all the pelindrom numbers in a list.

li=[123,32,23232,45,11,121,56]

for i in li:

rev=0

temp=i

while(temp>0):

rev= rev\*10 + temp%10

temp=int(temp/10)

if(i==rev):

print(i)

**Output:**

23232

11

121

### Find the max length sublist in a nested list.

nestedLi=[[1,2,3],[1,2],[1,2,3,4,5],[0],[],[1,2,3,4]]

maxLength=0

maxIndex=0

for i in range(len(nestedLi)):

if len(nestedLi[i])>maxLength :

maxLength = len(nestedLi[i])

maxIndex = i

print("Maximum subList index is: ",maxIndex)

print("Maximum subList: ",nestedLi[maxIndex])

**Output:**

Maximum subList index is: 2

Maximum subList: [1, 2, 3, 4, 5]

### Write a Python program to check if a 3 digit number is Armstrong number or not.

n=int(input("Enter a 3-digit Number:"))

sum=0

temp=n

while(temp>0):

sum+=(temp%10)\*\*3

temp= int(temp/10)

if(sum==n):

print("Number is Armstrong")

else:

print("Number is not Armstrong")

**Output:**

Enter a 3-digit Number:153

Number is Armstrong

### Write Python Program to reverse a number and also find the Sum of digits in the reversed number. Prompt the user for input.

rev=0

num=int(input("Enter a num:"))

temp=num

sum=0

while(temp>0):

sum+=temp%10

rev= rev\*10 + temp%10

temp=int(temp/10)

print("Number in reverse order is:",rev)

print("Sum of all digit in {} is {}".format(num,sum))

**Output:**

Enter a num:123

Number in reverse order is: 321

Sum of all digit in 123 is 6

### Write Python code to check if a given year is a leap year or not.

year=int(input("Enter a year:"))

if(year%4==0):

print("Leap Year")

else:

print("Not a leap year")

**Output:**

Enter a year:2002

Not a leap year

### Write Python program to find the GCD of two positive numbers.

n1=int(input("Enter a num:"))

n2=int(input("Enter a num:"))

fact1=[]

fact2=[]

gcd=1

nt1=n1

nt2=n2

for i in range(2,n1+1):

while(nt1%i==0):

fact1.append(i)

nt1=int(nt1/i)

for i in range(2,n2+1):

while(nt2%i==0):

fact2.append(i)

nt2=int(nt2/i)

for i in fact1:

if(i in fact2):

gcd\*=i

fact2.remove(i)

fact1.remove(i)

if(len(fact2)!=0):

for i in fact2:

if(i in fact1):

gcd\*=i

fact1.remove(i)

print("GCD of {} and {} is {}".format(n1,n2,gcd))

**Output:**

Enter a num:4

Enter a num:8

GCD of 4 and 8 is 4

### Write Python program to count the total number of vowels, consonants and blanks in a String.

vowel=['a','A','e','E','i','I','o','O','u','U']

st=input("Enter a string:")

sc=vc=cc=0

for i in st:

if(i.isspace()):

sc+=1

if(i.isalpha()):

if(i in vowel):

vc+=1

else:

cc+=1

print("Space count:",sc)

print("Vowel count:",vc)

print("Consonant count:",cc)

**Output:**

Space count: 3

Vowel count: 4

Consonant count: 12

### Write Python code to find Mean, Variance and Standard Deviation for a list of numbers.

import math

li=[-3,0,3,6,9]

sum=0

for i in li:

sum+=i

mean=sum/len(li)

varSum=0

for i in li:

varSum+=(i-mean)\*\*2

variance=varSum/len(li)

std\_variance=math.sqrt(variance)

print("Data:[-3,0,3,6,9]")

print("Mean is :",mean)

print("Variance is :",variance)

print("Standard variance is:",std\_variance)

**Output:**

Data:[-3,0,3,6,9]

Mean is : 3.0

Variance is : 18.0

Standard variance is: 4.242640687119285

### Write a program that accepts a sentence and calculate the number of digits, uppercase and lowercase letters.

sen=input("Enter a Sentence:")

dc,lc,uc=0

for i in sen:

if (i.isupper()):

uc+=1

if(i.islower()):

lc+=1

if(i.isdigit()):

dc+=1

print("digit count:",dc)

print("lowercase count:",lc)

print("uppercase count:",uc)

**Output:**

Enter a Sentence:This is Test String 1

digit count: 1

lowercase count: 13

uppercase count: 3

### Write Python code to sort a sequence of names according to their alphabetical order without using sort() function.

li=["darshil","ajay","nusarat","pratham","aaryan"]

newLi=[]

for i in li:

newLi.append(i.lower())

for i in range(len(newLi)):

for j in range(len(newLi)-i-1):

if(newLi[j]>newLi[j+1]):

newLi[j+1],newLi[j]=newLi[j],newLi[j+1]

print("Sorted list of names:\n",newLi)

**Output:**

Sorted list of names:

['aaryan', 'ajay', 'darshil', 'nusarat', 'pratham']

### Write Python code to find the factorial of a number

f=int(input("Enter a num:"))

fact=1

if(f==0):

print(1)

elif(f<0):

print("Negative num!!!")

else:

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

fact\*=i

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

**Output:**

Enter a num:5

factorial of 5 is 120

### Write Python program to perform a linear search for a given Key number in the list and report Success or Failure.

li=[5,43,4,32,41,22,0]

k=int(input("Enter key to find:"))

for i in li:

if(i==k):

print("Success key ",k," founded")

break

else:

print("failure!")

**Output:**

Enter key to find:41

Success key 41 founded

### Write Python Program to count the number of times an item appears in the list.

li=[1,21,131,13,45,1,45,56,1]

sLi=set(li)

for i in sLi:

print(i,"occured",li.count(i),"times")

**Output:**

1 occured 3 times

131 occured 1 times

45 occured 2 times

13 occured 1 times

21 occured 1 times

56 occured 1 times

### Write a program to calculate min and max values from a list of tuples.

li=[(1,2,3),(1,),(2,5,1),(3,1,9)]

for i in li:

print("Min value of",i,"is",min(i))

print("Max value of",i,"is",max(i))

print("")

**Output:**

Min value of (1, 2, 3) is 1

Max value of (1, 2, 3) is 3

Min value of (1,) is 1

Max value of (1,) is 1

Min value of (2, 5, 1) is 1

Max value of (2, 5, 1) is 5

Min value of (3, 1, 9) is 1

### Write a program to merge two list into list of tuples.

l1=[1,2,3,4,6,9,4,7,8]

l2=[6,3,5,3,2,3,8,5,0]

mergeLi=[]

for i in range(len(l1)):

mergeLi.append((l1[i],l2[i]))

print("L1 is:",l1)

print("L2 is:",l2)

print("Merged List is:",mergeLi)

**Output:**

L1 is: [1, 2, 3, 4, 6, 9, 4, 7, 8]

L2 is: [6, 3, 5, 3, 2, 3, 8, 5, 0]

Merged List is: [(1, 6), (2, 3), (3, 5), (4, 3), (6, 2), (9, 3), (4, 8), (7, 5), (8, 0)]

### Given a list of tuples containing both int and string remove all the string from list of tuples.

li=[(1,'1',3,'darshil',0),

(2,'hello',9,3,''),

(3,'hi',9.3,' ',4),

(7,'list',2,1),

(9,3,4,5,'tuple','array')]

newLi=[]

for i in li:

temp=[]

for j in i:

if type(j) !=type("str"):

temp.append(j)

newLi.append(tuple(temp))

print("List with string:",li)

print("List without string:",newLi)

**Output:**

List with string: [(1, '1', 3, 'darshil', 0), (2, 'hello', 9, 3, ''), (3, 'hi', 9.3, ' ', 4), (7, 'list', 2, 1), (9, 3, 4, 5, 'tuple', 'array')]

List without string: [(1, 3, 0), (2, 9, 3), (3, 9.3, 4), (7, 2, 1), (9, 3, 4, 5)]

### WAP to find max and min value in a set and also find the length of the set

s={1,2,3,2,4,9,5,8,6}

print("Set:",s)

print("Max value of set:",max(s))

print("Min value of set:",min(s))

print("Length of set:",len(s))

**Output:**

Set: {1, 2, 3, 4, 5, 6, 8, 9}

Max value of set: 9

Min value of set: 1

Length of set: 8

### WAP to create union and intersection of sets

s1={1,8,3,6,9,4,2,5,7,0}

s2={1,18,12,19,13,2,17,20,6}

print("Set1:",s1)

print("Set2:",s2)

print("Union of sets is:",s1 | s2)

print("Intersection of sets is:",s1 & s2)

**Output:**

Set1: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

Set2: {1, 2, 6, 12, 13, 17, 18, 19, 20}

Union of sets is: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 12, 13, 17, 18, 19, 20}

Intersection of sets is: {1, 2, 6}

### WAP to count no of vowels using set in a given string.

vowel=('a','A','e','E','i','I','o','O','u','U')

st=input("Enter a string:")

vc=0

for i in st:

if i in vowel:

vc+=1

print("Vowel count is:",vc)

**Output:**

Enter a string:This is testing string

Vowel count is: 5

### Write a python program to create a dictionary for library.

library={

"Books":{

"b1":{"title":"first book","author":"darshil"},

"b2":{"title":"second book","author":"darshil"},

"b3":{"title":"third book","author":"darshil"},

},

"bCategory":{

"cat1":"science",

"cat2":"data science",

"cat3":"data structure"

}

}

print("Library dictionary:\n",library)

**Output:**

Library dictionary:

{'Books': {'b1': {'title': 'first book', 'author': 'darshil'}, 'b2': {'title': 'second book', 'author': 'darshil'}, 'b3': {'title': 'third book', 'author': 'darshil'}}, 'bCategory': {'cat1': 'science', 'cat2': 'data science', 'cat3': 'data structure'}}

### Write a python program to create computer accessory dictionary and replace dictionary value if price is less than 500.

compAccessory={

"a1":{

"name":"keyboard",

"price":800

},

"a2":{

"name":"mouse",

"price":500

},

"a3":{

"name":"display",

"price":5000

},

"a4":{

"name":"motherboard",

"price":2000

},

"a5":{

"name":"pendrive",

"price":250

},

}

print("Accessory before changes:\n",compAccessory)

for key,value in compAccessory.items():

if(value["price"]<500):

compAccessory[key]["price"]=500

print("Accessory after changes:\n",compAccessory)

**Output:**

Accessory before changes:

{'a1': {'name': 'keyboard', 'price': 800}, 'a2': {'name': 'mouse', 'price': 500}, 'a3': {'name': 'display', 'price': 5000}, 'a4': {'name': 'motherboard', 'price': 2000}, 'a5': {'name': 'pendrive', 'price': 250}}

Accessory after changes:

{'a1': {'name': 'keyboard', 'price': 800}, 'a2': {'name': 'mouse', 'price': 500}, 'a3': {'name': 'display', 'price': 5000}, 'a4': {'name': 'motherboard', 'price': 2000}, 'a5': {'name': 'pendrive', 'price': 500}}

### Write a python program to match key values in two dictionary.

dic1={

1:"one",

2:"two",

3:"three",

4:"four",

5:"five"

}

dic2={

1:"one",

2:"two",

3:"3",

4:"fourth",

5:"five"

}

for i in dic1.keys():

if(dic1[i]!=dic2[i]):

print("Value of dic1 and dic2 for key {} not matched".format(i))

**Output:**

Value of dic1 and dic2 for key 3 not matched

Value of dic1 and dic2 for key 4 not matched

### Write a function to calculate the power of number raised to other.

a=int(input("Enter a num1:"))

b=int(input("Enter a num2:"))

pow=1

for i in range(b):

pow\*=a

print("Power of {} raised to {} is: {}".format(a,b,pow))

**Output:**

Enter a num1:2

Enter a num2:4

Power of 2 raised to 4 is: 16

### Write a password generator function in python which should generate random passwords every time the user ask for a new password. Strong password should be a mix of lower case, uppercase, number and symbol.

import random

import string

def generate\_password():

length = 15

characters = string.ascii\_letters + string.digits + string.punctuation

password = ''.join(random.choice(characters) for i in range(length))

return password

password = generate\_password()

print("Generated Password:", password)

**Output:**

Generated Password: OhkUYs)1>r1D["b

### Write a program to read two double values from command line, perform division among them and display answer. Handle all possible errors using Exception Handling Mechanism and display appropriate message.(**Use one try and multiple catches**).

try:

n1=int(input("Enter a num1:"))

n2=int(input("Enter a num2:"))

print("{}/{} is {}".format(n1,n2,(n1/n2)))

except ZeroDivisionError:

print("Cann't divide by zero.")

except Exception as e:

print("Error:",e)

**Output:**

Enter a num1:5

Enter a num2:0

Cann't divide by zero.

### Write a method to compute addition of two matrices to get a resultant matrix. Call this method in main to have A= B+C+D(where A,B,C,D are matrices).

1. Use command line arguments
2. Raise and handle define exception if size of matrix is improper for doing matrix addition.

import sys

def add(b,c,d):

temp=[[b[0][i]+c[0][i]for i in range(3)],

[b[1][i]+c[1][i]for i in range(3)]]

return [[temp[0][i]+d[0][i] for i in range(3)],[temp[1][i]+d[1][i] for i in range(3)]]

try:

a= [ [ int(sys.argv[i]) for i in range(1,4)],[int(sys.argv[i]) for i in range(4,7)] ]

print("From cmdLine:\n",a)

print("Addition:")

print(add(a,a,a))

except IndexError:

print("Provide enough numbers")

except Exception as e:

print("Error occured:",e)

**Output:**

From cmdLine:

[[1, 2, 3], [4, 5, 6]]

Addition:

[[3, 6, 9], [12, 15, 18]]

### Write a python program to read file contain and if file is not available then handle it using appropriate exception handling.

try:

with open('demo.txt','r') as file:

content = file.read()

print("File Data:\n",content)

except FileNotFoundError:

print("Error:File not found. Make sure path is correct and file exists")

except Exception as e:

print("Error occured:",e)

**Output:**

File Data:

This is text file reading demo using python.

### Write a python program to read two files and concate it output and write it in third file.

try:

with open('concate1.txt','r') as file1, open('concate2.txt','r') as file2:

data1=file1.read()

data2=file2.read()

concated\_data=data1+"\n"+data2

with open('concated.txt','w') as file:

file.write(concated\_data)

print("Files concatenated in concated.txt")

except FileNotFoundError:

print("One or both input files is missing")

except Exception as e:

print("Error occured:",e)

**Output:**

Files concatenated in concated.txt

concated.txt

This is demo txt file 1 for concatenation

This is demo txt file 2 for concatenation

### Write a python class named circle constructed by a radius and two methods which will compute area and perometer.

import math

class circle:

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

self.radius=radius

def area(self):

return math.pi\*self.radius\*self.radius

def perimeter(self):

return 2\*math.pi\*self.radius

c=circle(5)

print("Area of circle:",c.area())

print("Area of perimeter:",c.perimeter())

**Output:**

Area of circle: 78.53981633974483

Area of perimeter: 31.41592653589793

### Create a class Demo and also create method test () in it. Overload test () in four ways. First version takes no parameter, the second takes one integer parameter, and the third takes two integer parameters and fourth takes one double parameter.

class Demo:

def test(self,\*vargs):

if(len(vargs)==0):

print("Test with zero parameter.")

elif(len(vargs)==1):

print("Test with one parameter.")

elif(len(vargs)==2):

print("Test with two parameter.")

else:

pass

d = Demo()

d.test()

d.test(1)

d.test(1,2)

d.test(1.2)

**Output:**

Test with zero parameter.

Test with one parameter.

Test with two parameter.

Test with one parameter.

### create an abstract class shape and derive rectangle and circle from shape class. Implement abstract method of shape class in rectangle and circle class. Shape class contains: origin (x,y) as data member Display() and area() as abstract methods.Circle contains : radious, Rectangle contains : length & width ( user inheritance , overloading and overriding concept.)

from abc import ABC,abstractmethod

import math

class Shape(ABC):

@abstractmethod

def display():

pass

@abstractmethod

def area():

pass

class Rectangle(Shape):

def \_\_init\_\_(self,length,width):

self.length=length

self.width=width

def display(self):

print("length:",self.length)

print("width:",self.width)

def area(self):

return self.length \* self.width

class Circle(Shape):

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

self.radius=radius

def area(self):

return math.pi\*self.radius\*self.radius

def display(self):

print(f"Circle radius:{self.radius}")

r=Rectangle(3,4)

r.display()

print("Area of rectangle:",r.area())

c=Circle(5)

c.display()

print("Area of circle:",c.area())

**Output:**

length: 3

width: 4

Area of rectangle: 12

Circle radius:5

Area of circle: 78.53981633974483

### Write a python program to convert date of yyyy-mm-dd to dd-mm-yyyy.

from datetime import date

def dateConvert(date):

return date.strftime('%d-%m-%Y')

d=date(2024,1,27)

print("Date before Converting:",d)

new\_date=dateConvert(d)

print("Date after Converting:",new\_date)

**Output:**

Date before Converting: 2024-01-27

Date after Converting: 27-01-2024

### Write a python program to find occurrence and position of substrings within a string.

string=input("Enter main String:")

substring=input("Enter sub string:")

print(string.find(substring) if string.find(substring)>0 else "Not found")

print(string.count(substring))

**Output:**

Enter main String:This is a main String

Enter sub string:is

2

2

### Write a python program to find sequences of one uppercase letter followed by lowercase letter.

import re

st=input("Enter a string:")

pattern=re.compile(r'([A-Z][a-z]+)')

matched = re.findall(pattern,st)

print("Matched Sequence of one uppercase followed by lowercase letter:",matched)

**Output:**

Matched Sequence of one uppercase followed by lowercase letter: ['This', 'Is', 'Test', 'String']

### Write a python program to create calculator using Tkinter.

import tkinter as tk

op=['+','-','/','\*','%']

def button\_click(number):

current = entry.get()

entry.delete(0, tk.END)

if current=="0" and str(number).isdigit():

entry.insert(0, str(number))

elif(current[-1] in op or (current[-1]=="." and str(number)==".")):

entry.delete(0,tk.END)

entry.insert(0,current[:len(current)-1]+str(number))

else:

entry.insert(0, current+str(number))

def clear():

if(entry.get()!="0"):

if len(entry.get())==1:

entry.delete(0,0)

entry.insert(0,"0")

entry.delete(len(entry.get())-1,tk.END)

def allclear():

entry.delete(0,tk.END)

entry.insert(0,"0")

def equal():

try:

result = eval(entry.get())

entry.delete(0, tk.END)

entry.insert(0, result)

except:

entry.delete(0, tk.END)

entry.insert(0, "Error")

def on\_key\_press():

pass

root = tk.Tk()

root.title("Calculator by Darshil Solanki")

root.geometry("385x350")

entry = tk.Entry(root, width=35, borderwidth=5)

entry.grid(row=0, column=0, columnspan=4)

entry.insert(0,"0")

button\_1 = tk.Button(root, text="1", padx=40, pady=20, command=lambda: button\_click(1))

button\_2 = tk.Button(root, text="2", padx=40, pady=20, command=lambda: button\_click(2))

button\_3 = tk.Button(root, text="3", padx=40, pady=20, command=lambda: button\_click(3))

button\_4 = tk.Button(root, text="4", padx=40, pady=20, command=lambda: button\_click(4))

button\_5 = tk.Button(root, text="5", padx=40, pady=20, command=lambda: button\_click(5))

button\_6 = tk.Button(root, text="6", padx=40, pady=20, command=lambda: button\_click(6))

button\_7 = tk.Button(root, text="7", padx=40, pady=20, command=lambda: button\_click(7))

button\_8 = tk.Button(root, text="8", padx=40, pady=20, command=lambda: button\_click(8))

button\_9 = tk.Button(root, text="9", padx=40, pady=20, command=lambda: button\_click(9))

button\_0 = tk.Button(root, text="0", padx=40, pady=20, command=lambda: button\_click(0))

button\_dot = tk.Button(root, text=".", padx=40, pady=20, command=lambda: button\_click("."))

button\_add = tk.Button(root, text="+", padx=40, pady=20, command=lambda: button\_click("+"))

button\_subtract = tk.Button(root, text="-", padx=40, pady=20, command = button\_click("-"))

button\_multiply = tk.Button(root, text="\*", padx=40, pady=20, command= button\_click("\*"))

button\_divide = tk.Button(root, text="/", padx=40, pady=20, command= button\_click("/"))

button\_modulo = tk.Button(root, text="%", padx=40, pady=20, command= button\_click("%"))

button\_allclear = tk.Button(root, text="AC", width=1,padx=40, pady=20, command=allclear)

button\_clear = tk.Button(root, text="C", padx=40, pady=20, command=clear)

button\_equal = tk.Button(root, text="=", padx=40, pady=20, command=equal)

button\_1.grid(row=2, column=0)

button\_2.grid(row=2, column=1)

button\_3.grid(row=2, column=2)

button\_4.grid(row=3, column=0)

button\_5.grid(row=3, column=1)

button\_6.grid(row=3, column=2)

button\_7.grid(row=4, column=0)

button\_8.grid(row=4, column=1)

button\_9.grid(row=4, column=2)

button\_0.grid(row=5, column=1)

button\_dot.grid(row=5,column=0)

button\_add.grid(row=1, column=3)

button\_subtract.grid(row=2, column=3)

button\_multiply.grid(row=3, column=3)

button\_divide.grid(row=4, column=3)

button\_modulo.grid(row=5,column=3)

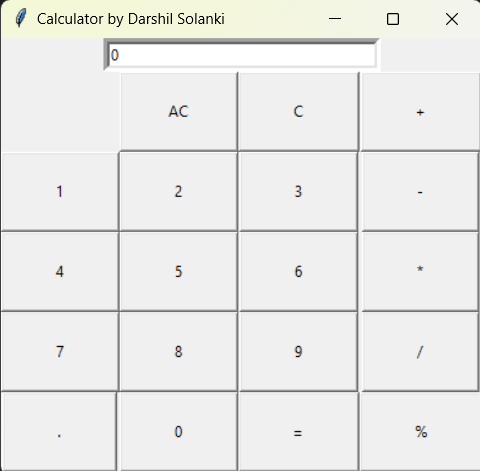
button\_allclear.grid(row=1, column=1)

button\_clear.grid(row=1, column=2)

button\_equal.grid(row=5, column=2)

root.mainloop()

**Output:**



### Write a python program to create student registration form and collect the data provied by student in database.

import sqlite3 as db

conn = db.connect('database.db')

cursor = conn.cursor()

def create\_table():

try:

cursor.execute('''

CREATE TABLE STUDENT(

PRN INTEGER PRIMARY KEY,

SNAME VARCHAR2(50),

DOB DATE,

ADDRESS VARCHAR2(60),

GENDER CHAR DEFAULT 'M',

COURSE VARCHAR2(10)

)

''')

conn.commit()

print("Table created Successfully.")

except Exception as e:

print("Error occured:",e)

def insert\_record(prn,sname,dob,address,gender,course):

try:

cursor.execute('INSERT INTO STUDENT VALUES(?,?,?,?,?,?)',[prn,sname,dob,address,gender,course])

conn.commit()

print("Record inserted Successfully.")

except Exception as e:

print("Error Occured:",e)

create\_table()

insert\_record(8023057262,'Darshilkumar','12-02-2002','Porbandar','M','MCA')

**Output:**

Table created Successfully.

Record inserted Successfully.