

Python Practical’s

# TASK 8

Smit Joshi | 14-10-2023

View On [**github.com/smit-joshi814**](https://github.com/smit-joshi814/Learning-python/tree/main/collage/Task8)

Practical 1

Create module for Simple Calculator functions such as addition, multiplication, division, subtraction. Create another file Scientific Calculator which includes functions like sin, cos, tan, power. Create package including these two modules. Create another file and demonstrate the use of defined packages.

# calculator/SimpleCalculator.py

# module simple calculator

def add(num1,num2):

    return num1 + num2

def sub(num1,num2):

    return num1 - num2

def mul(num1,num2):

    return num1 \* num2

def div(num1,num2):

    return num1 / num2

# calculator/ScientificCalculator.py

# Scintific calculator

import math

def sin1(num):

    return math.sin(num)

def cos1(num):

    return math.cos(num)

def tan1(num):

    return math.tan(num)

def power1(num,by):

    return math.pow(num,by)

# calculator/\_\_init\_\_.py

from . import ScientificCalculator,SimpleCalculator

# MasterCalculator.py

from calculator import SimpleCalculator as cal,ScientificCalculator as sci

def getNum():

    return int(input("Enter Number "))

while True:

    print("1.Add\n2.substract\n3.multiply\n4.divide\n5.sin\n6.cos\n7.tan\n8.power\n9.exit")

    choice=int(input("Enter Choice "))

    print()

    match choice:

        case 1:

            print("sum is",cal.add(getNum(),getNum()))

        case 2:

            print("substraction is",cal.sub(getNum(),getNum()))

        case 3:

            print("multiplication is",cal.mul(getNum(),getNum()))

        case 4:

            print("division is",cal.div(getNum(),getNum()))

        case 5:

            print("sin is",sci.sin1(getNum()))

        case 6:

            print("cos is",sci.cos1(getNum()))

        case 7:

            print("tan is",sci.tan1(getNum()))

        case 8:

            print("power is",sci.power1(getNum(),getNum()))

        case default:

            print("Exiting...")

            break

    print()

# Output:

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical1> py MasterCalculator.py

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 1

Enter Number 10

Enter Number 20

sum is 30

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 2

Enter Number 40

Enter Number 20

substraction is 20

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 3

Enter Number 2

Enter Number 2

multiplication is 4

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 4

Enter Number 4

Enter Number 2

division is 2.0

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 5

Enter Number 2

sin is 0.9092974268256817

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 6

Enter Number 2

cos is -0.4161468365471424

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 7

Enter Number 2

tan is -2.185039863261519

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 8

Enter Number 2

Enter Number 2

power is 4.0

1.Add

2.substract

3.multiply

4.divide

5.sin

6.cos

7.tan

8.power

9.exit

Enter Choice 9

Exiting...

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical1>

Practical 2

Create a module palindrome checker which takes input integer number and checks whether the number is palindrome or not. Create a file and use this module.

# PalindromeChecker.py

# is Pelendrom

def isPelendrome(num):

    rev=0

    temp=num

    while(temp>0):

        i=temp % 10

        rev= rev \* 10 + i

        temp=temp//10

    if rev == num: return True

    else: return False

# main.py

import PalindromeChecker as pel

if(pel.isPelendrome(int(input("Enter Number ")))):

    print("given Number is Palindrome")

else:

    print("Given Number is Not Palindrome")

# Output:

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical2> py main.py

Enter Number 121

given Number is Palindrome

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical2> py main.py

Enter Number 20

Given Number is Not Palindrome

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical2>

Practical 3

Create module consist of class point with members x and y and function euclidean\_distance to calculate the distance between two points.

# euclideanCalculator.py

import math

def euclidean\_distance(point1, point2):

  distance = math.sqrt((point1[0] - point2[0])\*\*2 + (point1[1] - point2[1])\*\*2)

  return distance

# main.py

import euclideanCalculator as cal

def getPoints(n):

   return int(input(f"Point {n} "))

point1=[]

point1.append(getPoints(1))

point1.append(getPoints(2))

print("---")

point2=[]

point2.append(getPoints(1))

point2.append(getPoints(2))

print("euclidean distance is",cal.euclidean\_distance(point1,point2))

# Output:

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical3> py main.py

Point 1 40

Point 2 20

---

Point 1 20

Point 2 60

euclidean distance is 44.721359549995796

Practical 4

Create a module consist of class distance with members feet & inch rewrite the function to add two object of distance and validate the resultant.

# distance.py

class Distance():

    def \_\_init\_\_(self,feet,inch):

        self.feet=feet

        self.inch=inch

    def sumFitInch(self,obj):

        Sumfeet=self.feet+obj.feet

        SumInchs=self.inch+obj.inch

        if SumInchs>12:

            Sumfeet+=SumInchs//12

            SumInchs=SumInchs%12

        return Sumfeet,SumInchs

# distanceCalculator.py

import distance as d

feet1=int(input("Enter value of Feet 1: "))

feet2=int(input("Enter value of Feet 2: "))

inch1=int(input("Enter value of Inch 1: "))

inch2=int(input("Enter value of Inch 2: "))

objdis1=d.Distance(feet1,inch1)

objdis2=d.Distance(feet2,inch2)

if inch1>11 or inch2 >11:

    print("Inch should not be more than 11")

else:

    totalFeets,totalInchs=objdis1.sumFitInch(objdis2)

    print(f"Feets: {totalFeets} \nInchs: {totalInchs}")

# Output:

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical4> py distanceCalculator.py

Enter value of Feet 1: 5

Enter value of Feet 2: 4

Enter value of Inch 1: 5

Enter value of Inch 2: 5

Feets: 9

Inchs: 10

Practical 5

Create package Geometry, with modules for calculation of area & perimeter of circle , rectangle, triangle. Write class and use.

# geometry/circle.py

import math

class Circle():

    def \_\_init\_\_(self, radius):

        self.radius = radius

    def area(self):

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

    def perimeter(self):

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

# geometry/rectangle.py

class Rectangle():

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

        self.length = length

        self.width = width

    def area(self):

        return self.length \* self.width

    def perimeter(self):

        return 2 \* (self.length + self.width)

# geometry/triangle.py

class Triangle:

    def \_\_init\_\_(self, base, height, side1, side2, side3):

        self.base = base

        self.height = height

        self.side1 = side1

        self.side2 = side2

        self.side3 = side3

    def area(self):

        return 0.5 \* self.base \* self.height

    def perimeter(self):

        return self.side1 + self.side2 + self.side3

# calculate.py

from geometry import circle, rectangle, triangle

circleObj=circle.Circle(radius=5)

rectangleObj = rectangle.Rectangle(length=4, width=6)

triangleObj =triangle.Triangle(base=4, height=3, side1=5, side2=5, side3=5)

print(f"Circle Area: {circleObj.area()}")

print(f"Circle Perimeter: {circleObj.perimeter()}")

print(f"Rectangle Area: {rectangleObj.area()}")

print(f"Rectangle Perimeter: {rectangleObj.perimeter()}")

print(f"Triangle Area: {triangleObj.area()}")

print(f"Triangle Perimeter: {triangleObj.perimeter()}")

# Output:

PS D:\LEARNING\COLLAGE\SAM7\Python\collage\task8\practical5> py calculate.py

Circle Area: 78.53981633974483

Circle Perimeter: 31.41592653589793

Rectangle Area: 24

Rectangle Perimeter: 20

Triangle Area: 6.0

Triangle Perimeter: 15