

Python Practical’s

# TASK 6

Smit Joshi | 21-09-2023

View On [**github.com/smit-joshi814**](https://github.com/smit-joshi814/Learning-python/tree/main/collage/Task6)

# Practical 1

Write a python program for creating class Fraction with members numerator and

denominator to represent rational numbers like 1/2 and -3/8. Create a constructor to

initialize the objects and write the method for addition and multiplication of two fractions

and display the result in proper fraction format.

class Fraction(object):

    def \_\_init\_\_(self, numerator=1,denominator=1):

        self.numerator=numerator

        self.denominator=denominator

    def add(self,fraction):

        if self.denominator==fraction.denominator:

            return (self.numerator+fraction.numerator),self.denominator

        else:

            n1=(self.numerator\*fraction.denominator)+(self.denominator\*fraction.numerator)

            n2=self.denominator\*fraction.denominator

            return n1,n2

    def multiply(self,fraction):

        if self.denominator==fraction.denominator:

            return (self.numerator\*fraction.numerator),self.denominator

        else:

            n1=(self.numerator\*fraction.denominator)\*(self.denominator\*fraction.numerator)

            n2=self.denominator\*fraction.denominator

            return n1,n2

def main():

    obj1=Fraction(1,2)

    obj2=Fraction(3,8)

    num,deno=obj2.add(obj1)

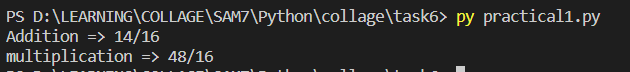
    print(f"Addition => {num}/{deno}")

    num,deno=obj2.multiply(obj1)

    print(f"multiplication => {num}/{deno}")

main()

### Output:



# Practical 2

Write a python program to create a class Room with members length and breadth. Initialize

the attributes and find the area of the room and display the result.

class Area(object):

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

        self.length=length

        self.bredth=bredth

    def area(self):

        return self.bredth\*self.length

def main():

    obj1=Area(4,3)

    print(f"Area of Room Is : {obj1.area()}");

main()

### Output:



# Practical 3

Create class point with members x and y. Create method for finding the euclidian distance

between two points.

import math

class PointOperations(object):

    def \_\_init\_\_(self, x,y):

        self.x=x

        self.y=y

    def getX(self):

        return self.x

    def getY(self):

        return self.y

    def distance(self,point):

        return math.sqrt(((self.x-point.x)\*\*2)+((self.y-point.y)\*\*2))

def main():

    obj1=PointOperations(4,3)

    obj2=PointOperations(0,0)

    print(f"Distance is: {obj2.distance(obj1)}")

main()

### Output:



# Practical 4

Write a python program to demonstrate the concept of default arguments. Create the

employee class: name, department and basic\_salary. Write a function to to calculate total

salary of the employee where total salary is sum of basic salary, DA and HRA. DA is 10%

of basic salary and HRA is 15% of basic salary. If basic salary is missing, take default

argument of basic salary as Rs. 9000.

class Employee():

    def \_\_init\_\_(self,name,department,basic\_salary=9000):

        self.name=name

        self.department=department

        self.basic\_salary=basic\_salary

    def getTotalSalary(self):

        da=self.basic\_salary\*0.10

        hra=self.basic\_salary\*0.15

        total\_salary=da+hra+self.basic\_salary

        return total\_salary

def main():

    emp1=Employee(name='Smit Joshi',department='MSCIT',basic\_salary=10000)

    print('total salary is: ',emp1.getTotalSalary())

main()

### Output:



# Practical 5

Write a python program to create class student with data name, roll no, sem, marks of 5

subject. Create a method to calculate percentage of the student

class Student():

    def \_\_init\_\_(self,name,rollNo,sem,marks):

        self.name=name

        self.rollNo=rollNo

        self.sem=sem

        self.marks=marks

    def calculatePercentage(self):

        sum=0

        for i in self.marks:

            sum+=i

        return (sum/500)\*100

def main():

    smit=Student(name='Smit Joshi',sem='1',rollNo='18',marks=[80,70,50,90,89])

    print('your percentage is : ',smit.calculatePercentage())

main()

### Output:



# Practical 6

create class complex with members real and imaginary and create a method for addition

and multiplication of two complex numbers

class Complex():

    def \_\_init\_\_(self,real=0.0,imaginary=0.0):

        self.real=real

        self.imaginary=imaginary

    def add(self,n1,n2):

        self.real=n1.real+n2.real

        self.imaginary=n1.imaginary+n2.imaginary

        return Complex(self.real,self.imaginary)

def main():

    n1=Complex(20,1)

    n2=Complex(30,1)

    temp=Complex()

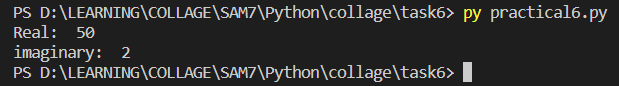
    temp=temp.add(n1,n2)

    print('Real: ',temp.real)

    print('imaginary: ',temp.imaginary)

main()

### Output:



# Practical 7

Create class date with members day, month and year. Create constructor for initialising the

values. Write a function for validation of the date. If date is not valid display proper

message.

class CustomDate():

    def \_\_init\_\_(self,day,month,year) :

        self.day=day

        self.month=month

        self.year=year

    def validate(self):

        if self.isYear() and self.isMonth() and self.isDay():

            print("Date is Correct")

        else:

            print("Invalid Date")

    def isLeapYear(self):

        if self.year%4==0:

            if self.year % 100 != 0 or self.year %400 ==0:

                return True

        else: return False

    def isYear(self):

        if self.year>=1 and self.year<=9999:

            return True

        else: return False

    def isMonth(self):

        if self.month >=1 and self.month<=12:

            return True

        else: return False

    def isDay(self):

        feb=28

        if self.isLeapYear(): feb=29

        match(self.month):

            case 1, 3, 5, 7, 8, 10, 12:

                return self.day>=1 and self.day<=31

            case 2:

                return self.day>=1 and self.day <= feb

            case default:

                return self.day >= 1 and self.day <= 30

def main():

    mdate=CustomDate(day=10,month=8,year=2023)

    mdate.validate()

main()

### Output:



# Practical 8

Write a python program for creating class Height with members feet and inch. Create a

constructor to initialize the objects and write the method for addition of two Height

objects , validate and display the result. (For validation: Inch<12).

class Height():

    def \_\_init\_\_(self,feet1,feet2,inch1,inch2):

        self.feet1=feet1

        self.feet2=feet2

        self.inch1=inch1

        self.inch2=inch2

    def sumFitInch(self):

        Sumfeet=self.feet1+self.feet2

        SumInchs=self.inch1+self.inch2

        if SumInchs>12:

            Sumfeet+=SumInchs//12

            SumInchs=SumInchs%12

        return Sumfeet,SumInchs

def main():

    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: "))

    if inch1>11 or inch2 >11:

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

    else:

        obj=Height(feet1=feet1,feet2=feet2,inch1=inch1,inch2=inch2)

        totalFeets,totalInchs=obj.sumFitInch()

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

main()

### Output:

