|  |  |
| --- | --- |
| Name | Umar Hayyat |
| Roll no | 360 |

**Lab no 10**

**Introduction to Computing**

**Objectives:**

In this lab we will learn:

* How we can use OOP in python.
* How we can make a class using object.

**Task No. 1:**

Create a class called Rational Number for performing arithmetic with fractions. Use integer variables to represent the data of the class—the numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it is declared. The constructor should contain default values, in case no initializers are provided, and should store the fraction in reduced form (i.e., the fraction

2/4

would be stored in the object as 1 in the numerator and 2 in the denominator). Provide methods for each of the following:

a) Adding two Rational Numbers. The result should be stored in reduced form.

b) Subtracting two Rational Numbers. The result should be stored in reduced form.

c) Multiplying two Rational Numbers. The result should be stored in reduced form.

d) Dividing two Rational Numbers. The result should be stored in reduced form.

e) Printing Rational Numbers in the form **a/b**, where **a** is the numerator and **b** is the denominator.

f) Printing Rational Numbers in floating-point format.

**Code:**

**a)**

class rational(object):

def \_\_init\_\_(self,a,b):

self.a=a

self.b=b

def Add(self,other):

den=self.b\*other.b

num=self.a\*other.b+other.a\*self.b

print(str(num)+"/"+str(den))

return("num/den is equal ",num/den)

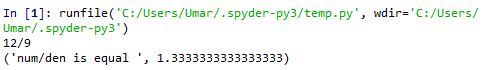
A=rational(2,3)

B=rational(2,3)

C=A.Add(B)

print(C)

**Output**:



Output of Task 1 part(a)

**b)**

class rational(object):

def \_\_init\_\_(self,a,b):

self.a=a

self.b=b

def Subtract(self,other):

den=self.b\*other.b

num=self.a\*other.b-other.a\*self.b

print(str(num)+"/"+str(den))

return("num/den is equal ",num/den)

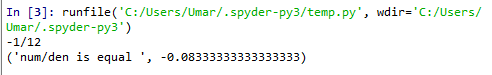
A=rational(2,3)

B=rational(3,4)

C=A.Subtract(B)

print(C)

**Output:**



Output of Task no 1 part(b)

**c)**

class rational(object):

def \_\_init\_\_(self,a,b):

self.a=a

self.b=b

def Multiple(self,other):

den=self.b\*other.b

num=self.a\*other.a

print(str(num)+"/"+str(den))

return("num/den is equal ",num/den)

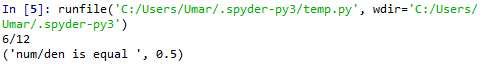
A=rational(2,3)

B=rational(3,4)

C=A.Multiple(B)

print(C)

**Output:**



Output of Task no 1 part (c)

**d)**

class rational(object):

def \_\_init\_\_(self,a,b):

self.a=a

self.b=b

def Divide(self,other):

den=self.b\*other.a

num=self.a\*other.b

print(str(num)+"/"+str(den))

return("num/den is equal ",num/den)

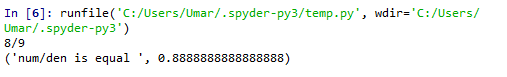
A=rational(2,3)

B=rational(3,4)

C=A.Divide(B)

print(C)

**Output:**



Output of Task no 1 part (d)

**e)**

**Code:**

class rational(object):

def \_\_init\_\_(self,n,d):

self.n=n

self.d=d

def \_\_str\_\_(self):

return (str(self.n)+"/"+str(self.d))

A=rational(2,5)

print(A)

**Output:**



Output of Task 1 part(e)

**f)**

**Code:**

class rational(object):

def \_\_init\_\_(self,n,d):

self.n=n

self.d=d

def float(self):

return(self.n/self.d)

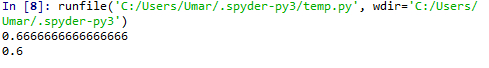
A=rational(6,9)

B=rational(3,5)

print(A.float())

print(B.float())

**Output:**



Output of Task 1 part(f)

**Task 2:**

Create a class Rectangle. The class has attributes length and width, each of which defaults to 1. It has methods that calculate the perimeter and the area of the rectangle. It has set and get methods for both length and width. The set methods should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0. Write a driver program to test the class.

**Code:**

class rectangle(object):

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

self.l=1

self.w=1

def set\_l(self):

number=float(input("enter a floating number: "))

if(number>0.0 and number<20.0):

self.l=number

else:

self.set\_l()

def set\_w(self):

number=float(input("enter a floating number: "))

if(number>0.0 and number<20.0):

self.w=number

else:

self.set\_w()

def get\_l(self):

return self.l

def get\_w(self):

return self.w

def perimeter(self):

result=2\*(self.l+self.w)

return result

def area(self):

result=self.l\*self.w

return result

y1=rectangle()

y1.set\_l()

y1.set\_w()

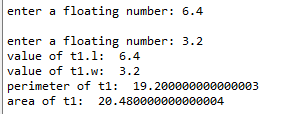
print("value of t1.l: ",y1.get\_l())

print("value of t1.w: ",y1.get\_w())

print("perimeter of t1: ",y1.perimeter())

print("area of t1: ",y1.area())

**Output:**



**Conclusion:**

In this lab,

I learnt use of different operation on rational number. I learnt how to make class and use it. I also learnt the use of getter and setter method.