Міністерство освіти та науки України

Національний технічний університет України

«Київський Політехнічний Інститут ім. Ігоря Сікорського»

ФТІ

Кафедра ФТЗЗІ

**Лабораторна робота №6**

з дисципліни: «Програмування 4»

на тему:

**«Обмін інформацією з файлами та модульний принцип програмування»**

**Варіант 11**

Виконав:

Ст. гр. ФЕ-81

Корнеєнков Кирило

Перевірив:

доцент Прогонов Д.О.

Київ 2020

Мета роботи: Оволодіти навичками роботи з файлами даних та засвоїти модульний принцип програмування.

**1. Порядок виконання роботи**

1. Проаналізувати умову задачі.

2. Розробити алгоритм та створити програму розв’язання задачі згідно з номером варіанту. При цьому розмістити функції програми у двох (або більше) різних файлах, та об’єднати ці файли у спільну програму двома різними способами:   
1) реалізувати модульний принцип програмування, створивши проект, у якому всі функції, крім головної, містяться в окремому файлі;

3. Результати роботи оформити протоколом.

**2. Варіанти завдань**

Модифікувати створену під час виконання у лабораторній роботі № 5 програму згідно з номером варіанту, використовуючи потоки файлового вводу/виводу для зчитування вхідної інформації з одного файла та запису отриманого результату в інший.

Код реалізації:

**func.py**

class Rational():

    def \_\_init\_\_(self, numerator=0, demoniator=1):

        self.numer = numerator

        self.denom = demoniator

    def \_\_add\_\_(self,other):

        n1 = self.numer

        d1 = self.denom

        n2 = other.numer

        d2 = other.denom

        n3 = d2\*n1 + d1\*n2

        d3 = d1\*d2

        return Rational(n3,d3)

    def \_\_mul\_\_(self, other):

        n1 = self.numer

        d1 = self.denom

        n2 = other.numer

        d2 = other.denom

        n3 = n1\*n2

        d3 = d1\*d2

        return Rational(n3,d3)

    def gcd(self, a, b):

        if b == 0:

            return a

        return self.gcd(b, a % b)

    def reduce(self):

        d = 1

        if self.denom != 0 and self.numer != 0:

            d = self.gcd(self.numer, self.denom)

        if d > 1:

            self.numer /= d

            self.denom /= d

            a = Rational(int(self.numer),int(self.denom))

            return a

    def \_\_str\_\_(self):

        return "{0}/{1}".format(self.numer,self.denom)

**lab\_6.py**

from func import Rational

# Создаём два объекта рациональных чисел:

fil = open('data.txt', 'r')

x1 = int(fil.readline())

y1 = int(fil.readline())

r1 = Rational(x1, y1)

x2 = int(fil.readline())

y2 = int(fil.readline())

r2 = Rational(x2, y2)

# Проверяем перегруженную операцию сложения:

r3 = r1 + r2

# Проверяем перегруженный метод вывода:

filres = open('results.txt', 'w')

filres.write(str(r3))

filres.write('\n')

print(r3)

# Проверяем перегруженную операцию уножения:

r4 = r1 \* r2

filres.write(str(r4))

filres.write('\n')

print(r4)

# Проверяем метод сокращения числа:

r5 = r1.reduce()

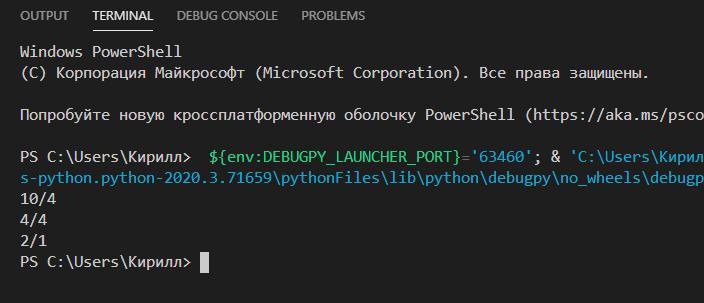
filres.write(str(r5))

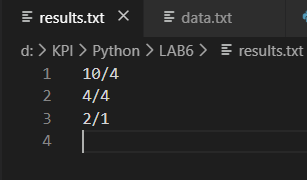
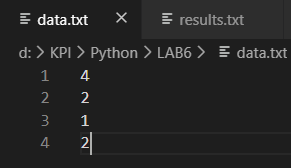
filres.write('\n')

print(r5)

fil.close()

filres.close()

Приклади виконання програми:

Файл *data.txt* з вхідними даними: Результат роботи програми у файлі *result.txt*

