Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №1
«Основные конструкции языка Python»

Выполнил:
Студент группы ИУ5-33Б
Лупарев Сергей
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2023 г.

Описание задания

Разработать программу для решения биквадратного уравнения.

1. Программа должна быть разработана в виде консольного приложения на языке Python.
2. Программа осуществляет ввод с клавиатуры коэффициентов A, B, C, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты A, B, C могут быть заданы в виде параметров командной строки (вариант задания параметров приведен в конце файла с примером кода). Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. Описание работы с параметрами командной строки.
4. Если коэффициент A, B, C введен или задан в командной строке некорректно, то необходимо проигнорировать некорректное значение и вводить коэффициент повторно пока коэффициент не будет введен корректно. Корректно заданный коэффициент - это коэффициент, значение которого может быть без ошибок преобразовано в действительное число.
5. Дополнительное задание 1 (*). Разработайте две программы на языке Python - одну с применением процедурной парадигмы, а другую с применением объектно-ориентированной парадигмы.
6. Дополнительное задание 2 (*). Разработайте две программы - одну на языке Python, а другую на любом другом языке программирования (кроме C++).

Текст программы

```python
import sys
import math

#ф-я вычисления коэффициентов
def get_coef(index, prompt):
    if (index < len(sys.argv)):
        try:
            return float(sys.argv[index])
        except:
            return get_coef(404, prompt)

    print(prompt, end = ' ')
    try:
        return float(input())
    except:
        return get_coef(index, prompt)


# Вычисление корней
def get_roots(a, b, c):
    D = b*b - 4*a*c
    if D < 0:
        return list()

    sqD = math.sqrt(D)
    root1 = (-b + sqD) / (2.0*a)
    root2 = (-b - sqD) / (2.0*a)
    res = list()
    if (root1 >= 0):
        res.append(math.sqrt(root1))
        res.append(-math.sqrt(root1))
    if (root2 >= 0):
        res.append(math.sqrt(root2))
        res.append(-math.sqrt(root2))
    return list(set(res))
```

```
def main():
    a = 0
    while(a==0):
        a = get_coef(1, 'Введите коэффициент A:')
    b = get_coef(2, 'Введите коэффициент B:')
    c = get_coef(3, 'Введите коэффициент C:')
    roots = get_roots(a,b,c)
    len_roots = len(roots)
    if len_roots == 0:
        print('Нет корней')
    else:
        print('Количество корней уравнения {}. Корни: '.format(len_roots), end = '')
        for i in roots:
            print(i, end = '  ')


if __name__ == "__main__":
    main()
```

ООП-подход

```
import sys
import math

class Solver:
    def __init__(self, mass):
        while (1):
            for i in range(len(mass)):
                if mass[i] == None:
                    print("Введите коэффициент номер {}: ".format(i+1), end = '')
                    try:
                        mass[i] = float(input())
                    except:
                        print("Ошибка")
            if mass[0] == 0:
                mass[0] = None
            if(mass[0]!=None and mass[1]!=None and mass[2]!=None):
                break
        self.a = mass[0]
        self.b = mass[1]
        self.c = mass[2]
        self.roots = list()
        self.D = self.b*self.b - 4*self.a*self.c

    def solve(self):
        if self.D < 0:
            return

        sqD = math.sqrt(self.D)
        root1 = (-self.b + sqD) / (2.0*self.a)
        root2 = (-self.b - sqD) / (2.0*self.a)
        res = list()
        if (root1 >= 0):
            self.roots.append(math.sqrt(root1))
            self.roots.append(-math.sqrt(root1))
        if (root2 >= 0):
            self.roots.append(math.sqrt(root2))
            self.roots.append(-math.sqrt(root2))
        self.roots = list(set(self.roots))
```

```
    def show(self):
        len_roots = len(self.roots)
        if len_roots == 0:
            print('Нет корней')
        else:
            print('Количество корней уравнения {}. Корни: '.format(len_roots), end = '')
            for i in self.roots:
                print(i, end = '  ')



def main():
    sys2 = [0 for i in range(3)]
    for i in [1, 2, 3]:
        try:
            sys2[i-1] = sys.argv[i]
        except:
            sys2[i-1] = None

    equation = Solver(sys2)
    equation.solve()
    equation.show()


if __name__ == "__main__":
    main()
```

Примеры выполнения программы

```
Python 3.7.2 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: D:\CBeer\Labs_PICKUP\lab1\lab1.py =================
Введите коэффициент A: 1
Введите коэффициент B: абв
Введите коэффициент B: ыфэххСхэзм
Введите коэффициент B: -17
Введите коэффициент C: 16
Количество корней уравнения 4. Корни: 1.0  4.0  -4.0  -1.0
>>>
```

<u>ООП-подход</u>

```
================ RESTART: D:\CBeer\Labs_PICKUP\lab1\lab1_2.py ================
Введите коэффициент номер 1: 1
Введите коэффициент номер 2: арпаррп
Ошибка
Введите коэффициент номер 3: 4
Введите коэффициент номер 2: ааа
Ошибка
Введите коэффициент номер 2: 4
Нет корней
>>>
```