**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка Интернет-Приложений»

Отчет по лабораторной работе №1

«Решение биквадратного уравнения»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-53Б |  | Юрий Евгеньевич Гапанюк |
| Ветошкин Артём |  |  |
| Подпись и дата:  30.09.21 |  | Подпись и дата: |

Москва, 2021 г.

**Задание:**

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 А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).
3. Коэффициенты А, В, С могут быть заданы в виде параметров командной строки. Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. Описание работы с параметрами командной строки.
4. Если коэффициент А, В, С 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.

**Текст программы:**

*import* sys

*import* math

def force\_conv\_number(string):

*try*:

*return* float(string)

*except*:

*return* None

def \_get\_coef(index, prompt):

    coef\_str = ''

*if* len(sys.argv) > index:

        coef\_str = sys.argv[index]

*else*:

        print(prompt)

        coef\_str = input()

*return* force\_conv\_number(coef\_str)

def get\_coef(index, prompt):

    tmp = None

*while* tmp == None:

        tmp = \_get\_coef(index, prompt)

*return* tmp

def complex\_sqrt(number):

*return* pow(number, 0.5)

def get\_roots(a, b, c):

    result = []

    D = b\*b - 4\*a\*c

    sqD = complex\_sqrt(complex(D, 0))

    root1 = (-b + sqD) / (2.0\*a)

    root2 = (-b - sqD) / (2.0\*a)

    result.append(complex\_sqrt(root1))

    result.append(-complex\_sqrt(root1))

    result.append(complex\_sqrt(root2))

    result.append(-complex\_sqrt(root2))

*return* result

def print\_roots(roots):

    len\_roots = len(roots)

*if* len\_roots == 0:

        print('Нет корней')

*elif* len\_roots == 1:

        print('Один корень: {}'.format(roots[0]))

*elif* len\_roots == 2:

        print('Два корня: {} и {}'.format(roots[0], roots[1]))

*elif* len\_roots == 3:

        print('Три корня: {}, {} и {}'.format(roots[0], roots[1], roots[2]))

*elif* len\_roots == 4:

        print('Четыри корня: {} и {}'.format(roots[0], roots[1], roots[3], roots[4]))

def input\_coef():

    a = get\_coef(1, 'Введите коэффициент А:')

    b = get\_coef(2, 'Введите коэффициент B:')

    c = get\_coef(3, 'Введите коэффициент C:')

*return* [a, b, c]

def main():

    [a, b, c] = input\_coef()

    roots = get\_roots(a,b,c)

    real\_roots = list(map(lambda x: x.real, filter(lambda number: number.imag == 0, roots)))

    print\_roots(real\_roots)

*if* \_\_name\_\_ == "\_\_main\_\_":

    main()

**Пример работы:**

