МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет “Информатика и системы управления”
Кафедра “Системы обработки информации и управления”

Дисциплина “Парадигмы и конструкции языков программирования”

Отчет по лабораторной работе №1
“Основные конструкции языка Python”

Выполнил:
Студент группы ИУ5-35Б
Рябов М.А.
Преподаватель:
Гапанюк Ю.Е.

Москва 2025

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 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++).

Листинг кода

main.py

```
import sys

def get_coefficient(coefficient_index, coefficient_name):
    coefficient = 0

    try:
        coefficient = float(sys.argv[coefficient_index])
    except:
        while True:
            try:
                coefficient = float(input(f"Введите коэффициент
{coefficient_name}: "))
```

```
                break
            except ValueError:
                print("Введите действительное число")

    return coefficient

def solve_biquadrate_equation(a, b, c):
    roots = set()

    discriminant = b * b - 4.0 * a * c

    if discriminant == 0.0:
        root = -b / (2.0 * a)
        if root >= 0:
            roots.add(root ** 0.5)
            roots.add(-1 * (root ** 0.5))
    elif discriminant > 0:
        squared_discriminant = discriminant ** 0.5

        first_root = (-b + squared_discriminant)/(2.0 * a)
        second_root = (-b - squared_discriminant)/(2.0 * a)

        if first_root >= 0:
            roots.add(first_root ** 0.5)
            roots.add(-1 * (first_root ** 0.5))

        if second_root >= 0:
            roots.add(second_root ** 0.5)
            roots.add(-1 * (second_root ** 0.5))

    return roots

def main():
    a = get_coefficient(1, "A")
    b = get_coefficient(2, "B")
    c = get_coefficient(3, "C")

    roots = solve_biquadrate_equation(a, b, c)

    roots_amount = len(roots)

    if roots_amount == 0:
        print("Корней нет")
    elif roots_amount == 1:
        print(f"Корень уравнения: {roots[0]}")
    else:
        print(f"Корни уравнения:", *roots, sep=" ")

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

extra.py

```
import sys

class Coefficient:
    def __init__(self, index, name):
        self.index = index
        self.name = name

        try:
            self.value = float(sys.argv[self.index])
        except:
            while True:
                try:
                    self.value = float(input(f"Введите коэффициент {self.name}:
"))
                    break
                except ValueError:
                    print("Введите действительное число")

class BiquadrateEquation:
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c

    def solve(self):
        roots = set()

        discriminant = self.b.value * self.b.value - 4.0 * self.a.value *
self.c.value

        if discriminant == 0.0:
            root = -self.b.value / (2.0 * self.a.value)
            if root >= 0:
                squered_root = root ** 0.5
                roots.add(squered_root)
                roots.add(-1 * squered_root)
        elif discriminant > 0:
            squared_discriminant = discriminant ** 0.5

            first_root = (-self.b.value + squared_discriminant)/(2.0 *
self.a.value)
            second_root = (-self.b.value - squared_discriminant)/(2.0 *
self.a.value)

            if first_root >= 0:
                squared_first_root = first_root ** 0.5
                roots.add(squared_first_root)
                roots.add(-1 * squared_first_root)
```

```
            if second_root >= 0:
                squared_second_root = second_root ** 0.5
                roots.add(squared_second_root)
                roots.add(-1 * squared_second_root)

        return roots

def main():
    a = Coefficient(1, "A")
    b = Coefficient(2, "B")
    c = Coefficient(3, "C")

    equation = BiquadrateEquation(a, b, c)

    roots = equation.solve()
    roots_amount = len(roots)

    if roots_amount == 0:
        print("Корней нет")
    elif roots_amount == 1:
        print(f"Корень уравнения: {roots[0]}")
    else:
        print(f"Корни уравнения:", *roots, sep=" ")

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

Выполнение программы

```
mmmm@LAPTOPIC:~/projects/pcpl_2025/lab_1$ python3 main.py
Введите коэффициент A: 1
Введите коэффициент B: -3
Введите коэффициент C: 2
Корни уравнения: 1.4142135623730951 -1.0 1.0 -1.4142135623730951
```