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 ИУ5-31Б
Паронько Д.И.
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++).

Текст программы

```
import math
import os
import sys

from colorama import Style, Fore, init

init()

def get_coeff_from_user(index, prompt) -> str:
    try:
        return float(sys.argv[index])
    except:
        while True:
            print(prompt, end = ' ')
            try:
                return float(input())
            except:
                print(Fore.YELLOW + "Incorrent coefficient. Try again")
                print(Style.RESET_ALL, end = '')
```

```
def print_answer(roots):
    print("Roots of your equation:")

    for i in range(len(roots)):
        print(Fore.GREEN + "x_" + str(i + 1), ' = ', roots[i])

def solve_incomplete_quadrate_equataion(quadr_coeff: float, free_coeff: float):
    if quadr_coeff * free_coeff > 0:
        return "No real roots"
    root = math.sqrt(-free_coeff / quadr_coeff)
    return [-root, root]

def solve_biquadrate_equation(a: float, b: float, c: float):
    if a == 0:
        if b == 0:
            return "Equation is not biquadrate"

        if c == 0:
            return [0.]

        return  solve_incomplete_quadrate_equataion(b, c)


    discrimannt = b**2 - 4 * a * c

    roots = list()

    if discrimannt < 0:
        return "No real roots"
    if discrimannt == 0:
        t = -b / (2 * a)
        if t < 0:
            return "No real roots"
        if t == 0:
            [0.]

        root = math.sqrt(t)
        return [-root, root]

    t1 = (-b + math.sqrt(discrimannt)) / (2 * a)
    t2 = (-b - math.sqrt(discrimannt)) / (2 * a)

    if t1 >= 0:
        root1 = math.sqrt(t1)
        roots.extend([-root1, root1])

    if t2 >= 0:
        root2 = math.sqrt(t2)
        roots.extend([-root2, root2])

    return sorted(roots)
```

```
def run():
    command_args = sys.argv[1:]


    coeff_a = get_coeff_from_user(1, "Enter coefficient A:")
    coeff_b = get_coeff_from_user(2, "Enter coefficient B:")
    coeff_c = get_coeff_from_user(3, "Enter coefficient C:")

    answer = solve_biquadrate_equation(coeff_a, coeff_b, coeff_c)

    if isinstance(answer, str):
        print(Fore.RED + answer)
        return

    print_answer(answer)

def main():
    try:
        run()
    except Exception as e:
        print(f'Error: {e}')

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

Результат выполнения

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py 2 -4 2
Roots of your equation:
x_1  =  -1.0
x_2  =  1.0
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py 2 3 4
No real roots
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py -10 9 3
Roots of your equation:
x_1  =  -1.0765093329543833
x_2  =  1.0765093329543833
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py 1 -5 4
Roots of your equation:
x_1  =  -2.0
x_2  =  -1.0
x_3  =  1.0
x_4  =  2.0
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py 1 0 -4
Roots of your equation:
x_1  =  -1.4142135623730951
x_2  =  1.4142135623730951
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py 1 0 0
Roots of your equation:
x_1  =  0.0
x_2  =  -0.0
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py 1 kenbbrb -6
Enter coefficient B: nvunrbji
Incorrent coefficient. Try again
Enter coefficient B: rbr
Incorrent coefficient. Try again
Enter coefficient B: 12
Roots of your equation:
x_1  =  -0.6933546699978738
x_2  =  0.6933546699978738
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py
Enter coefficient A: -32
Enter coefficient B: 45
Enter coefficient C: 100
Roots of your equation:
x_1  =  -1.6141849559978174
x_2  =  1.6141849559978174
```

```
juve@DESKTOP-HDCAHDS:~/git/BMSTU-PCPL-Labs-2025/lab_1$ python3 main.py
Enter coefficient A: 1
Enter coefficient B: j
Incorrent coefficient. Try again
Enter coefficient B: k
Incorrent coefficient. Try again
Enter coefficient B: -1
Enter coefficient C: 0
Roots of your equation:
x_1  =  -1.0
x_2  =  -0.0
x_3  =  0.0
x_4  =  1.0
```