**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Радиотехнический»

Кафедра «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №0

«Основные конструкции языка Python**.**»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы РТ5-31Б: |  | преподаватель кафедры ИУ5 |
| Молодцов М.В. |  | Гапанюк Ю. Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2024 г.

Постановка задачи

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 ( [вариант задания параметров приведен в конце файла с примером кода](https://github.com/ugapanyuk/BKIT_2022/blob/main/code/lab1_code) ). Если они не заданы, то вводятся с клавиатуры в соответствии с пунктом 2. [Описание работы с параметрами командной строки.](https://realpython.com/python-command-line-arguments/#the-command-line-interface)
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

Файл «op.py»

import math  
import argparse  
  
def get\_coefficient(name):  
 while True:  
 try:  
 coefficient = float(input(f"Введите коэффициент {name}: "))  
 return coefficient  
 except ValueError:  
 print(f"Ошибка! Коэффициент {name} должен быть числом.")  
  
def calculate\_discriminant(A, B, C):  
 return B \*\* 2 - 4 \* A \* C  
  
def solve\_quadratic(A, B, C):  
 D = calculate\_discriminant(A, B, C)  
 if D < 0:  
 return []  
 elif D == 0:  
 y = -B / (2 \* A)  
 if y >= 0:  
 return [math.sqrt(y), -math.sqrt(y)]  
 else:  
 return []  
 else:  
 sqrt\_D = math.sqrt(D)  
 y1 = (-B + sqrt\_D) / (2 \* A)  
 y2 = (-B - sqrt\_D) / (2 \* A)  
  
 roots = []  
 if y1 >= 0:  
 roots.extend([math.sqrt(y1), -math.sqrt(y1)])  
 if y2 >= 0:  
 roots.extend([math.sqrt(y2), -math.sqrt(y2)])  
  
 return roots  
  
def parse\_arguments():  
 parser = argparse.ArgumentParser(description="Решение биквадратного уравнения")  
 parser.add\_argument("-A", type=float, help="Коэффициент A")  
 parser.add\_argument("-B", type=float, help="Коэффициент B")  
 parser.add\_argument("-C", type=float, help="Коэффициент C")  
 return parser.parse\_args()  
  
def main():  
 print("Решение биквадратного уравнения Ax^4 + Bx^2 + C = 0")  
  
 args = parse\_arguments()  
  
 A = args.A if args.A is not None else get\_coefficient("A")  
 B = args.B if args.B is not None else get\_coefficient("B")  
 C = args.C if args.C is not None else get\_coefficient("C")  
  
 roots = solve\_quadratic(A, B, C)  
  
 if roots:  
 print("Корни уравнения:", roots)  
 else:  
 print("Решений нет.")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Экранные формы с примерами выполнения программы

