**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «ПиКЯП»

Отчет по лабораторной работе №1

## «Основные конструкции языка Python.»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34Б |  | преподаватель каф. ИУ5 |
| Сафронов Андрей |  | Нардид А. Н. |
|  |  |  |

Москва, 2024 г.

**Цель лабораторной работы:** изучение основных конструкций языка Python.

**Задание:** Разработать программу для решения [биквадратного уравнения.](https://ru.wikipedia.org/wiki/%D0%A3%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_%D1%87%D0%B5%D1%82%D0%B2%D1%91%D1%80%D1%82%D0%BE%D0%B9_%D1%81%D1%82%D0%B5%D0%BF%D0%B5%D0%BD%D0%B8#%D0%91%D0%B8%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D1%83%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5)

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.
5. Дополнительное задание 1 (\*). Разработайте две программы на языке Python - одну с применением процедурной парадигмы, а другую с применением объектно-ориентированной парадигмы.

Код программы:

lab1.py

import sys  
  
  
def enter(letter):  
 while True:  
 try:  
 return float(input(f"Please, enter {letter}: "))  
 except:  
 print("Wrong! Try again!")  
  
  
def enter\_coefficients():  
 if len(sys.argv) > 1:  
 a = float(sys.argv[1])  
 if len(sys.argv) > 2:  
 b = float(sys.argv[2])  
 else:  
 b = enter("B")  
 if len(sys.argv) > 3:  
 c = float(sys.argv[3])  
 else:  
 c = enter("C")  
 else:  
 a = enter("A")  
 b = enter("B")  
 c = enter("C")  
 return a, b, c  
  
  
def calculate(a, b, c):  
 d = b \*\* 2 - 4 \* a \* c  
 solution = []  
 if d == 0:  
 r = -b / (2 \* a)  
 solution.append(r)  
 elif d > 0:  
 r1 = (-b - d \*\* 0.5) / (2 \* a)  
 r2 = (-b + d \*\* 0.5) / (2 \* a)  
 solution.append(r1)  
 solution.append(r2)  
 return solution  
  
  
def output(a, b, c, solution):  
 print(f"A: {a}; B: {b}; C: {c}. ")  
 if len(solution) == 0:  
 print("No roots :( ")  
 elif len(solution) == 1:  
 print(f"One root: {solution[0]} :)")  
 else:  
 print(f"Two roots: {solution[0]} and {solution[1]} ^U^")  
  
  
def main():  
 a, b, c = enter\_coefficients()  
 solution = calculate(a, b, c)  
 output(a, b, c, solution)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

lab1\_oop.py

import sys  
  
def enter(letter):  
 while True:  
 try:  
 return float(input(f"Please, enter {letter}: "))  
 except:  
 print("Wrong! Try again!")  
  
class Equation:  
 def \_\_init\_\_(self):  
 self.a, self.b, self.c, self.solution = 0, 0, 0, []  
  
 def enter\_coefficients(self):  
 if len(sys.argv) > 1:  
 self.a = float(sys.argv[1])  
 if len(sys.argv) > 2:  
 self.b = float(sys.argv[2])  
 else:  
 self.b = enter("B")  
 if len(sys.argv) > 3:  
 self.c = float(sys.argv[3])  
 else:  
 self.c = enter("C")  
 else:  
 self.a = enter("A")  
 self.b = enter("B")  
 self.c = enter("C")  
  
 def calculate(self):  
 d = self.b \*\* 2 - 4 \* self.a \* self.c  
 if d == 0:  
 r = -self.b / (2 \* self.a)  
 self.solution.append(r)  
 elif d > 0:  
 r1 = (-self.b - d \*\* 0.5) / (2 \* self.a)  
 r2 = (-self.b + d \*\* 0.5) / (2 \* self.a)  
 self.solution.append(r1)  
 self.solution.append(r2)  
  
  
 def print(self):  
 print(f"A: {self.a}; B: {self.b}; C: {self.c}. ")  
 if len(self.solution) == 0:  
 print("No roots :( ")  
 elif len(self.solution) == 1:  
 print(f"One root: {self.solution[0]} :)")  
 else:  
 print(f"Two roots: {self.solution[0]} and {self.solution[1]} ^U^")  
  
  
 def solve(self):  
 self.enter\_coefficients()  
 self.calculate()  
 self.print()  
  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 eq = Equation()  
 eq.solve()

Результаты:

