**Московский государственный технический университет им. Н.Э. Баумана**

Факультет «Радиотехнический»

Кафедра РТ5 «Системы обработки информации и управления» Отчёт по лабораторной работе №1

|  |  |
| --- | --- |
| Выполнил: | Проверил: |
| студент группы РТ5-31Б: | преподаватель каф. ИУ5 |
| Кузнецов Марат Дмитриевич | Гапанюк Ю. Е. |
| Подпись и дата: | Подпись и дата: |

Москва, 2023 г.

# Задание

Разработать программу для решения биквадратного уравнения.

Программа должна быть разработана в виде консольного приложения на языке Python.

Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и ДЕЙСТВИТЕЛЬНЫЕ корни уравнения (в зависимости от дискриминанта).

Коэффициенты А, В, С 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.

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.

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

Дополнительное задание 2 (\*). Разработайте две программы - одну на языке Python, а другую на любом другом языке программирования (кроме С++).

# Код программы (основное задание)

import sys

def CalculateDiscriminant(a, b, c): return b\*\*2 - 4\*a\*c

def CalculateRoots(a, b, discriminant):

roots = [] sq\_roots = [] if discriminant > 0: sq\_roots.append((-b + discriminant\*\*0.5) / (2\*a)) sq\_roots.append((-b - discriminant\*\*0.5) / (2\*a)) elif discriminant == 0: sq\_roots.append(-b / (2\*a)) for a in sq\_roots: if (a > 0):

roots.append(a\*\*(0.5)) roots.append(-a\*\*(0.5)) if (a == 0): roots.append(a\*\*(0.5)) return roots

def InputCoefficient(prompt): while True: try:

coefficient = float(input(prompt))

return coefficient except ValueError:

print("Ошибка ввода. Введите действительное число.")

if \_\_name\_\_ == "\_\_main\_\_": if len(sys.argv) == 4: try:

1. = float(sys.argv[1]) b = float(sys.argv[2]) c = float(sys.argv[3]) except ValueError:

print("Некорректные коэффициенты. Пожалуйста, введите действительные числа.") sys.exit(1) else:

print("Введите коэффициенты уравнения:") a = InputCoefficient("Коэффициент A: ") while a == 0:

print("Коэффициент 'a' не может быть равен нулю для биквадратного уравнения.") a = InputCoefficient("Коэффициент A: ")

1. = InputCoefficient("Коэффициент B: ")
2. = InputCoefficient("Коэффициент C: ")

discriminant = CalculateDiscriminant(a, b, c)

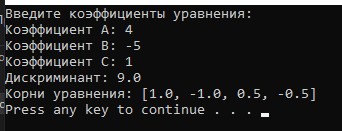
if discriminant >= 0:

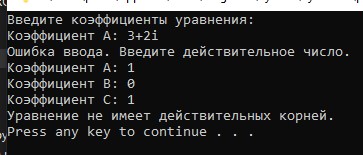
roots = CalculateRoots(a, b, discriminant) if len(roots) > 0:

print("Корни уравнения:", roots) else:

print("Уравнение не имеет действительных корней.")

# Результаты





**Код программы (дополнительное задание 1)**

import sys

class Equation: def \_\_init\_\_(self):

self.a = 0 self.b = 0 self.c = 0

def InputCoefficient(self, prompt): while True: try:

coefficient = float(input(prompt)) return coefficient except ValueError:

print("Ошибка ввода. Введите действительное число.")

def InsertCoefficient(self): if len(sys.argv) == 4: try:

self.a = float(sys.argv[1]) self.b = float(sys.argv[2]) self.c = float(sys.argv[3]) except ValueError:

print("Некорректные коэффициенты. Пожалуйста, введите действительные числа.") sys.exit(1) else:

print("Введите коэффициенты уравнения:") self.a = self.InputCoefficient("Коэффициент A: ") while self.a == 0:

print("Коэффициент 'a' не может быть равен нулю для квадратного уравнения.") self.a = self.InputCoefficient("Коэффициент A: ") self.b = self.InputCoefficient("Коэффициент B: ") self.c = self.InputCoefficient("Коэффициент C: ")

class Calculator: def \_\_init\_\_(self): self.discriminant = 0 self.roots = [] self.sq\_roots = []

def CalculateDiscriminant(self, Equation): self.discriminant = Equation.b\*\*2 - 4\*Equation.a\*Equation.c

def CalculateRoots(self, Equation):

self.roots = [] self.sq\_roots = [] if self.discriminant > 0:

self.sq\_roots.append((-Equation.b + self.discriminant\*\*0.5) / (2\*Equation.a)) self.sq\_roots.append((-Equation.b - self.discriminant\*\*0.5) / (2\*Equation.a)) elif self.discriminant == 0 and Equation.a!=0:

self.sq\_roots.append(-Equation.b / (2\*Equation.a)) for r in self.sq\_roots: if (r > 0):

self.roots.append(r\*\*(0.5)) self.roots.append(-r\*\*(0.5)) if (r == 0): self.roots.append(r\*\*(0.5)) return self.roots

def PrintRoots(self): if len(self.roots) > 0:

print("Корни уравнения:", self.roots)

else:

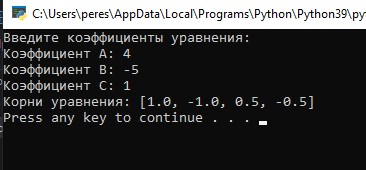
print("Уравнение не имеет действительных корней.")

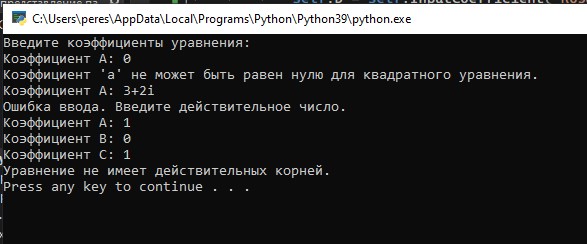
def execute(self, Equation): self.CalculateDiscriminant(Equation) self.CalculateRoots(Equation) self.PrintRoots()

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

Eq = Equation() Eq.InsertCoefficient() calc = Calculator() calc.execute(Eq)

# Результаты





# Код программы (дополнительное задание 2 (C#))

using System;

using System.Collections.Generic;

class Program

{

static void Main(string[] args)

{

double a, b, c;

if (args.Length == 3)

{

// Вариант с параметрами командной строки

if (double.TryParse(args[0], out a) && double.TryParse(args[1], out b) &&

double.TryParse(args[2], out c))

{

SolveAndPrintResults(a, b, c);

} else

{

Console.WriteLine("Ошибка ввода параметров командной строки. Введите коэффициенты с клавиатуры.");

} } else

{

// Ввод с клавиатуры a = ReadCoefficient("Введите коэффициент A: "); b = ReadCoefficient("Введите коэффициент B: "); c = ReadCoefficient("Введите коэффициент C: ");

SolveAndPrintResults(a, b, c);

}

}

static double ReadCoefficient(string prompt)

{

double coefficient; while (true)

{

Console.Write(prompt);

if (double.TryParse(Console.ReadLine(), out coefficient))

{

return coefficient;

} else {

Console.WriteLine("Ошибка ввода. Повторите ввод.");

}

}

}

static void SolveAndPrintResults(double a, double b, double c)

{

Stack<double> roots = new Stack<double>();

double discriminant = b \* b - 4 \* a \* c;

if (discriminant < 0)

{

Console.WriteLine("Уравнение не имеет действительных корней.");

}

else if ((discriminant == 0) & (-b / (2 \* a) > 0))

{

roots.Push(Math.Sqrt(-b / (2 \* a))); roots.Push(-Math.Sqrt(-b / (2 \* a)));

Console.WriteLine($"Корни уравнения: {roots}");

} else

{

if ((-b + Math.Sqrt(discriminant)>0) | (-b - Math.Sqrt(discriminant)>0))

{

if (-b + Math.Sqrt(discriminant)>0)

{

roots.Push(Math.Sqrt((-b + Math.Sqrt(discriminant)) / (2 \* a))); roots.Push(-Math.Sqrt((-b + Math.Sqrt(discriminant)) / (2 \* a)));

}

if (-b - Math.Sqrt(discriminant)>0)

{

roots.Push(Math.Sqrt((-b - Math.Sqrt(discriminant)) / (2 \* a))); roots.Push(-Math.Sqrt((-b - Math.Sqrt(discriminant)) / (2 \* a)));

}

Console.WriteLine($"Корни уравнения:");

foreach (double root in roots){

Console.WriteLine(root);

}

}

}

}

}

**Результаты**

