|  |  |  |
| --- | --- | --- |
| Защищено:  Гапанюк Ю.Е.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |  | Демонстрация:  Гапанюк Ю.Е.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |

**Отчет по лабораторной работе № 1 по курсу**

**Базовые компоненты интернет-технологий**

**ГУИМЦ**

#### Тема работы: " Биквадратное уравнение "

9

(количество листов)

Вариант № **4**

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-52Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Чиварзин А.Е. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |

Москва, МГТУ - 2020

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ**

[1. Описание задания 3](#_Toc50574201)

[2. Диаграмма классов 3](#_Toc50574202)

[3. Текст программы 3](#_Toc50574203)

[4. Результаты выполнения программы 8](#_Toc50574204)

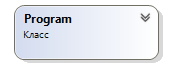
# Описание задания

Разработать программу для решения биквадратного уравнения.

Программа должна соответствовать следующим требованиям:

1. Программа должна быть разработана в виде консольного приложения на языке C#.
2. Программа 2. Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и корни уравнения (в зависимости от дискриминанта).
3. Если коэффициент А, В, С введен некорректно, то необходимо проигнорировать некорректное значение и ввести коэффициент повторно.
4. Первой строкой программа выводит ФИО разработчика и номер группы.
5. Корни уравнения выводятся зеленым цветом. Если корней нет, то сообщение выводится красным цветом.
6. ДОПОЛНИТЕЛЬНОЕ ТРЕБОВАНИЕ. Коэффициенты А, В, С задаются в виде параметров командной строки. Если они не указаны, то вводятся с клавиатуры в соответствии с пунктом 2. Проверка из пункта 3 в этом случае производится для параметров командной строки без повторного ввода с клавиатуры

# Диаграмма классов



# Текст программы

using System;

namespace LAB\_1

{

class Program

{

static double? A = null, B = null, C = null;

static double[] roots = new double[4] { 0, 0, 0, 0 }; //Действительная часть

static double[] cPart = new double[4] { 0, 0, 0, 0 }; //Комплексная часть

//static double[] c2part = new double[4] { 0, 0, 0, 0 }; //i^2

static bool dIsComplex = false;

static void Main(string[] args)

{

Console.WriteLine("Чиварзин А. Е. ИУ5Ц-52Б");

if (args.Length == 3)

readFromArgs(args);

else

input();

//А, В и C != NULL (см. input())

double d = (double)(B \* B - 4 \* A \* C);

if (d < 0) //Комплексный внутренний корень

{

noROOTs();

/\*

\* d \*= -1;

\* d = Math.Sqrt(d);

\* dIsComplex = true;

\* double ba = (double)(-B / 2 \* A); //Действительная часть внутреннего корня

\* double cROOT\_1 = (double)(d / (2 \* A)); //Комплексная часть внутреннего корна 1

\* double cROOT\_2 = (double)((-d) / (2 \* A)); //Комплексная часть внутреннего корня 2

\*/

return;

}

d = Math.Sqrt(d);

if (d == 0)

{

double ba = (double)(-B / 2 \* A);

if (ba == 0)

printROOTs(1, false); //1 корень = 0

if (ba > 0)

{

roots[0] = Math.Sqrt(ba);

roots[1] = (-1) \* Math.Sqrt(ba);

printROOTs(2, false);

}

}

else if (d > 0)

{

double ba1 = (double)((-B + d) / (2 \* A));

double ba2 = (double)((-B - d) / (2 \* A));

if (ba1 > 0 && ba2 > 0) //Не комплексные

{

roots[0] = Math.Sqrt(ba1);

roots[1] = Math.Sqrt(ba2);

roots[2] = -roots[0];

roots[3] = -roots[1];

printROOTs(4, false);

}

else if (ba1 < 0 && ba2 < 0)

{

cPart[0] = Math.Sqrt(-ba1);

cPart[1] = Math.Sqrt(-ba2);

cPart[2] = -cPart[0];

cPart[3] = -cPart[1];

printROOTs(4, true);

}

else if (ba1 < 0 && ba2 > 0)

{

cPart[0] = Math.Sqrt(-ba1);

roots[1] = Math.Sqrt(ba2);

cPart[2] = -cPart[0];

roots[3] = -roots[1];

printROOTs(4, true);

}

else if (ba1 > 0 && ba2 < 0)

{

roots[0] = Math.Sqrt(ba1);

cPart[1] = Math.Sqrt(-ba2);

roots[2] = -roots[0];

cPart[3] = -cPart[1];

}

}

}

static void input()

{

while (true)

{

String temp;

Console.WriteLine("Введите A, B и C");

try //Ввод A

{

Console.Write("A >>> ");

temp = Console.ReadLine();

A = Convert.ToDouble(temp);

if (A == null || A == 0)

{

Console.WriteLine("Некорректный ввод коеффициента \"A\"");

continue;

}

} catch (FormatException)

{

Console.WriteLine("Некорректный ввод коеффициента \"A\"");

continue;

}

try //Ввод B

{

Console.Write("B >>> ");

temp = Console.ReadLine();

B = Convert.ToDouble(temp);

if (B == null)

{

Console.WriteLine("Некорректный ввод коеффициента \"B\"");

continue;

}

}

catch (FormatException)

{

Console.WriteLine("Некорректный ввод коеффициента \"B\"");

continue;

}

try //Ввод C

{

Console.Write("C >>> ");

temp = Console.ReadLine();

C = Convert.ToDouble(temp);

if (C == null)

{

Console.WriteLine("Некорректный ввод коеффициента \"C\"");

continue;

}

}

catch (FormatException)

{

Console.WriteLine("Некорректный ввод коеффициента \"C\"");

continue;

}

break; //Конец цикла. Ввод выполнен без ошибок.

}

Console.WriteLine("Коеффициенты введены успешно!");

}

static void printROOTs(int rCount, bool complex)

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine("Корни уравнения:");

if (complex) //Комплексные корни

{

for (int i = 0; i < rCount; i++)

{

if (cPart[i] >= 0)

Console.WriteLine("{0:d}: {1:f4} +{2:f4}i", i + 1, roots[i], cPart[i]);

else

Console.WriteLine("{0:d}: {1:f4} {2:f4}i", i + 1, roots[i], cPart[i]);

}

}

else //Действительные корни

{

for (int i = 0; i < rCount; i++)

{

Console.WriteLine("{0:d}: {1:f4}", i + 1, roots[i]);

}

}

Console.ResetColor();

}

static void noROOTs()

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Корней нет!");

Console.ResetColor();

}

static void readFromArgs(string[] args) //Ввод A, B и C из аргумментов коммандной строки

{

try

{

A = Convert.ToDouble(args[0]);

if (A == 0)

throw new FormatException();

B = Convert.ToDouble(args[1]);

C = Convert.ToDouble(args[2]);

} catch (FormatException)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Аргумменты коммандной строки введены некорректно.");

Console.ResetColor();

input();

}

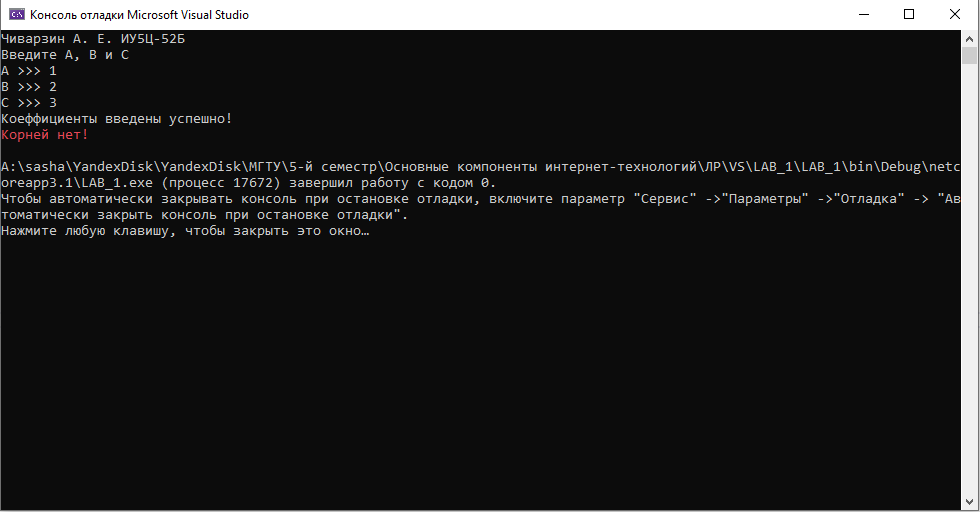
}

}

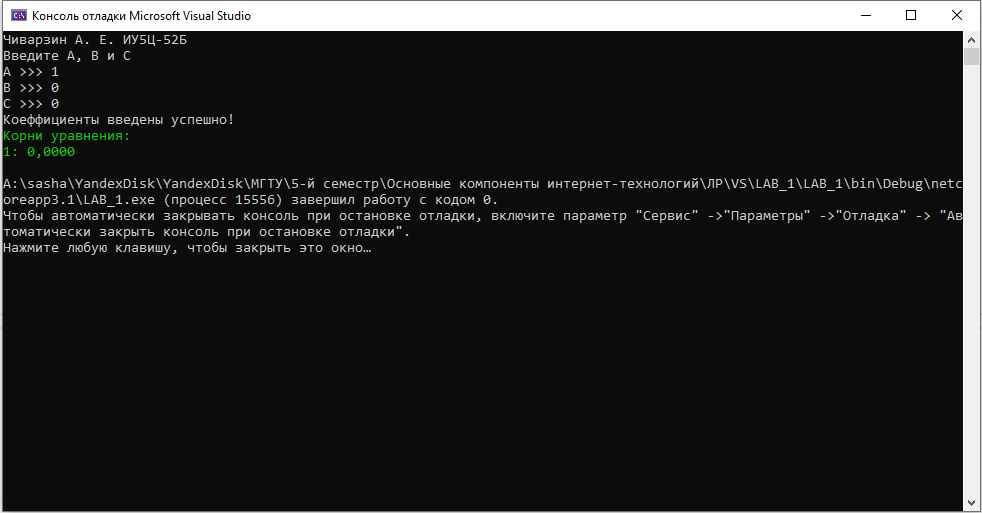
}

# Результаты выполнения программы

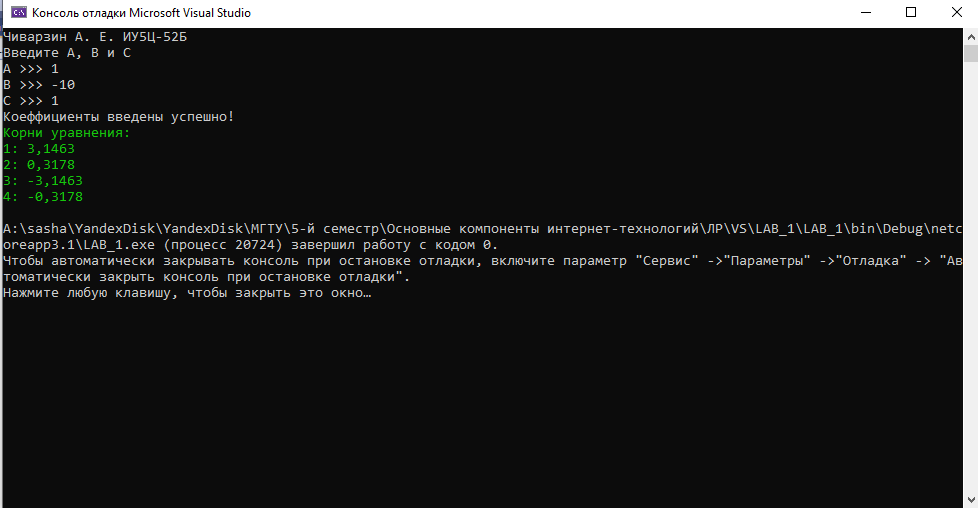
Корней нет



1 корень:



4 корня:



2 корня:

