**Московский государственный технический**

**Университет им Н.Э.Баумана**

Факультет «Информатика и системы управление»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №1

Выполнил: Проверил:

Студент: Марянян А.А. Преподаватель каф. ИУ5

Группа: ИУ5-34Б Нардид А.Н.

Подпись и дата: Подпись и дата:

**Постановка задачи**

**Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание**

**Код**

namespace Lab1;  
  
internal abstract class BiquadraticEquationSolver  
{  
 private static void Main(string[] args)  
 {  
 double a, b, c;  
   
 if (args.Length == 3)  
 {  
 bool aParsed = double.TryParse(args[0], out a);  
 bool bParsed = double.TryParse(args[1], out b);  
 bool cParsed = double.TryParse(args[2], out c);  
  
 if (!aParsed || !bParsed || !cParsed)  
 {  
 Console.ForegroundColor = ConsoleColor.**Red**;  
 Console.WriteLine("Ошибка: некорректные коэффициенты в параметрах командной строки.");  
 Console.ResetColor();  
 return;  
 }  
 }  
 else  
 {  
 a = InputCoefficient("A");  
 b = InputCoefficient("B");  
 c = InputCoefficient("C");  
 }  
   
 SolveBiquadraticEquation(a, b, c);  
 }  
  
 private static double InputCoefficient(string coefficientName)  
 {  
 double coefficient;  
 Console.Write($"Введите коэффициент {coefficientName}: ");  
 while (!double.TryParse(Console.ReadLine(), out coefficient))  
 {  
 Console.Write($"Некорректный ввод для {coefficientName}. Введите {coefficientName} повторно: ");  
 }  
 return coefficient;  
 }  
  
 private static void SolveBiquadraticEquation(double a, double b, double c)  
 {  
 if (a == 0)  
 {  
 Console.ForegroundColor = ConsoleColor.**Red**;  
 Console.WriteLine("Ошибка: коэффициент A не может быть равен нулю.");  
 Console.ResetColor();  
 return;  
 }  
  
 double discriminant = b \* b - 4 \* a \* c;  
  
 if (discriminant < 0)  
 {  
 Console.ForegroundColor = ConsoleColor.**Red**;  
 Console.WriteLine("Корней нет.");  
 Console.ResetColor();  
 }  
 else  
 {  
 double sqrtDiscriminant = Math.Sqrt(discriminant);  
 double z1 = (-b + sqrtDiscriminant) / (2 \* a);  
 double z2 = (-b - sqrtDiscriminant) / (2 \* a);  
   
 bool hasSolutions = false;  
  
 if (z1 >= 0)  
 {  
 var x1 = Math.Sqrt(z1);  
 var x2 = -Math.Sqrt(z1);  
 Console.ForegroundColor = ConsoleColor.**Green**;  
 Console.WriteLine($"Корни уравнения: x1 = {x1}, x2 = {x2}");  
 Console.ResetColor();  
 hasSolutions = true;  
 }  
  
 if (z2 >= 0)  
 {  
 var x3 = Math.Sqrt(z2);  
 var x4 = -Math.Sqrt(z2);  
 Console.ForegroundColor = ConsoleColor.**Green**;  
 Console.WriteLine($"Корни уравнения: x3 = {x3}, x4 = {x4}");  
 Console.ResetColor();  
 hasSolutions = true;  
 }  
  
 if (hasSolutions) return;  
 Console.ForegroundColor = ConsoleColor.**Red**;  
 Console.WriteLine("Действительных корней нет.");  
 Console.ResetColor();  
 }  
 }  
}

**Построение проекта**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**Вывод**

**Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**