Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана»

(национальный исследовательский университет)

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ____ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ_____

КАФЕДРА _____КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)_____

Отчет

по лабораторной работе № 3

Название лабораторной работы: Создание консольного приложения.

Дисциплина: Алгоритмизация и программирование

Студент гр. ИУ6-24Б **20.04.2024** **А.С. Воеводин**

(Подпись, дата) (И.О. Фамилия)

Преподаватель **20.04.2024** **О.А. Веселовская**

(Подпись, дата) (И.О. Фамилия)

Москва, 2024

Цель работы

Ознакомление со средой разработки Rider от компании JetBrains для написания программ на C#, написание простейшей консольной программы в целях ознакомления с синтаксисом объектно-ориентированного языка C#, работа с механизмами исключения, а также изучение ошибок и диагностических сообщений от компилятора, механизма работы с отладкой.

Задание

Часть 1. Изучить среду разработки консольных приложений на языке программирования C#.

В качестве примера использовать приложение, определяющее корни квадратного уравнения ax2+bx+c = 0. Предусмотреть возможность отсутствия корней при введенных пользователем коэффициентах, предусмотреть исключения при вводе нечисловых данных. Отладить и протестировать программу.

В отчете привести диаграмму классов, схему алгоритма метода Main, текст программы и результаты тестирования.

Часть 2. Изучить диагностические сообщения компилятора и средства отладки. Привести в отчете примеры диагностических сообщений, а также скриншоты результатов применения средств отладки для установки контрольных точек и просмотра промежуточных результатов.

Ход работы:

- Написание кода для первой части
- Тестирование
- Написание диаграмм классов, схемы алгоритма
- Отладка программы для второй части задания
- Вывод

Для начала напишем код для первой части, которая решает квадратное уравнение. Эта программа предусматривает все случаи, будь то вырожденные линейные уравнения, или обычное квадратное, на каждый случай предусмотрено сообщение в консоль для пользователя. Также предусмотрены исключения при вводе нечисловых данных. Ниже представлен код программы:

Листинг 1 – Код программы

```
namespace laba_3;

class Program
{
    static void Main(string[] args)
    {
        bool isBreak = false;
        while (!isBreak)
        {
            double a, b, c;
            bool flagA, flagB, flagC;
            Console.Write("Введите через пробел 3 числа: ");
            string st = Console.ReadLine();

            try
            {
                if (st.Split(' ').Length != 3)
                {
                    var err = new Exception("Вы ввели не 3 числа.
Попробуйте заново.");
                    throw err;
                }

                flagA = double.TryParse(st.Split(' ')[0], out a);
                flagB = double.TryParse(st.Split(' ')[1], out b);
                flagC = double.TryParse(st.Split(' ')[2], out c);
                if (!flagA || !flagB || !flagC)
                {
                    var err = new Exception("Вы ввели не числа.
Попробуйте заново");
                    throw err;
                }

                if (Math.Abs(a) < double.Epsilon)
                {
                    if (Math.Abs(b) < double.Epsilon)
                    {
                        if (Math.Abs(c) < double.Epsilon)
                        {
                            Console.WriteLine("Уравнение имеет
бесконечное количество решений.");
                        }
                        else
                        {
                            Console.WriteLine("Уравнение не имеет
корней.");
                        }
                    }
                    else
                    {
                        Console.WriteLine($"корень уравнения: {-c /
b}");
                    }
                }
                else
```

```
                {
                    double discriminant = b * b - 4 * a * c;

                    if (discriminant < 0)
                    {
                        Console.WriteLine("Уравнение не имеет
корней.");
                    }
                    else if (Math.Abs(discriminant) <
double.Epsilon)
                    {
                        Console.WriteLine($"Уравнение имеет один
корень: {-b / (2 * a)}");
                    }
                    else
                    {
                        double x1 = (-b - Math.Sqrt(discriminant)) /
(2 * a);
                        double x2 = (-b + Math.Sqrt(discriminant)) /
(2 * a);
                        Console.WriteLine($"Первый корень уравнения:
{x1}, второй корень уравнения: {x2}");
                    }
                }

                Console.WriteLine("Желаете ли вы закончить?
да/нет");
                string response = Console.ReadLine();
                if (response.ToLower() == "да")
                {
                    isBreak = true;
                }

                Console.Clear();
            }
            catch (Exception err)
            {
                Console.WriteLine(err.Message);
            }
        }
    }
}
```

Теперь протестируем получившуюся программу:

```
Введите через пробел 3 числа: 1 2 3
Уравнение не имеет корней.
Желаете ли вы закончить? да/нет
```

Рисунок 1 – Тестовые данные 1

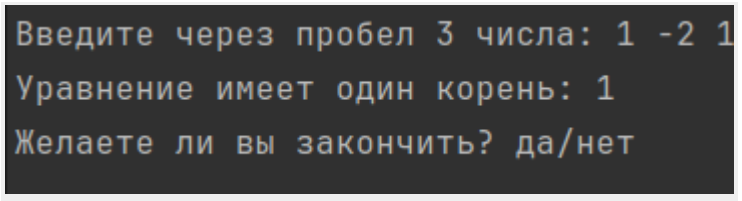


Рисунок 2 – Тестовые данные 2

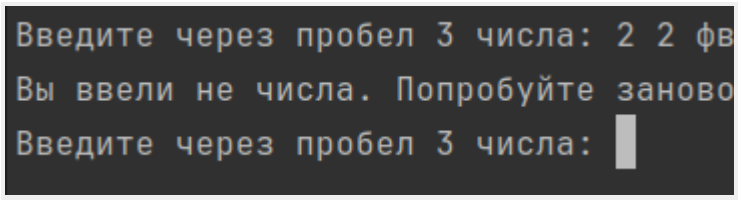


Рисунок 3 – Тестовые данные 3

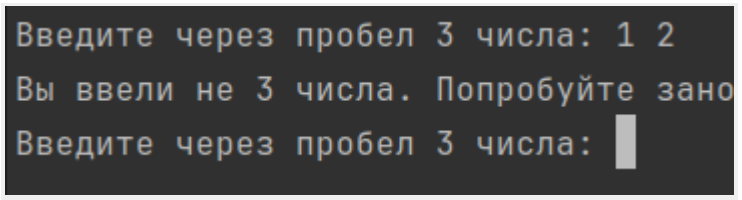


Рисунок 4 – Тестовые данные 4

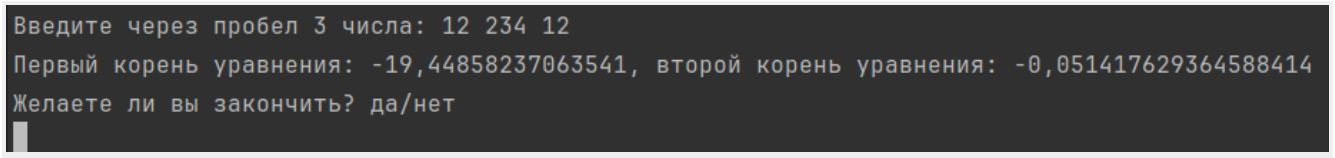
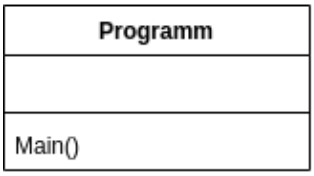


Рисунок 5 – Тестовые данные 5

Теперь сделаем схему алгоритма и диаграмму класса:



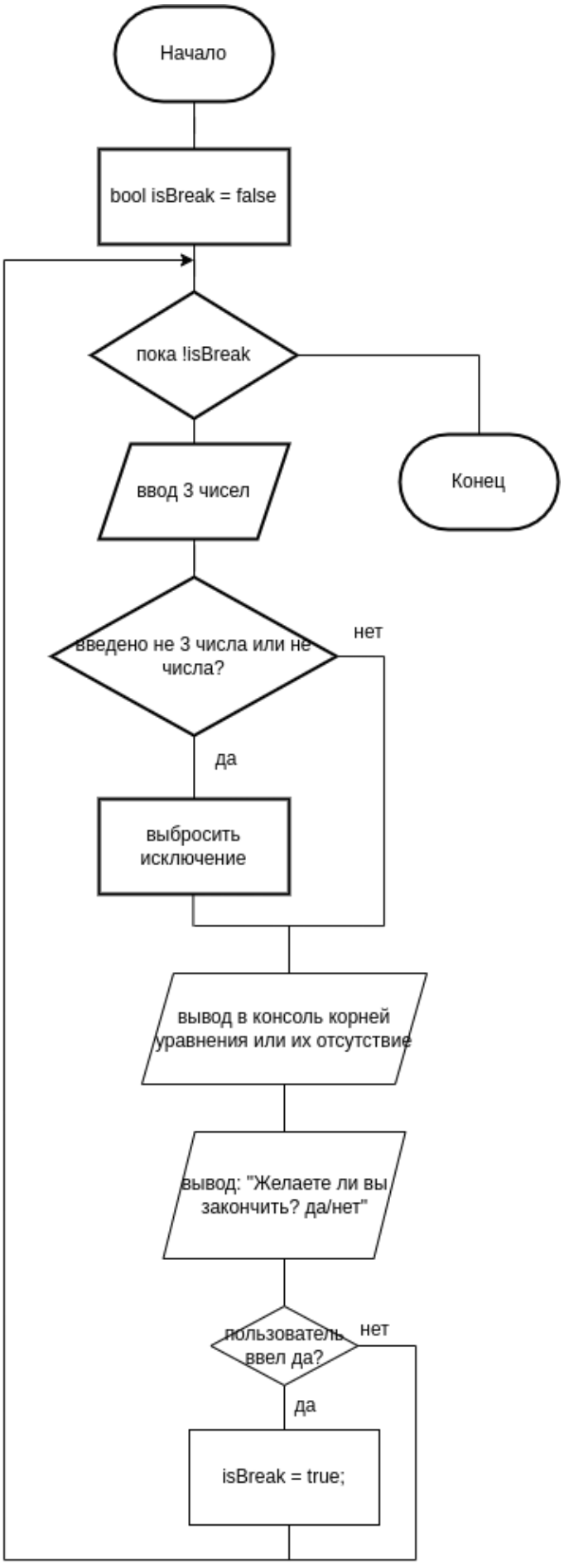


Рисунок 6 – Схема алгоритма и диаграмма класса

Первая часть задания выполнена, теперь приступим ко второй:

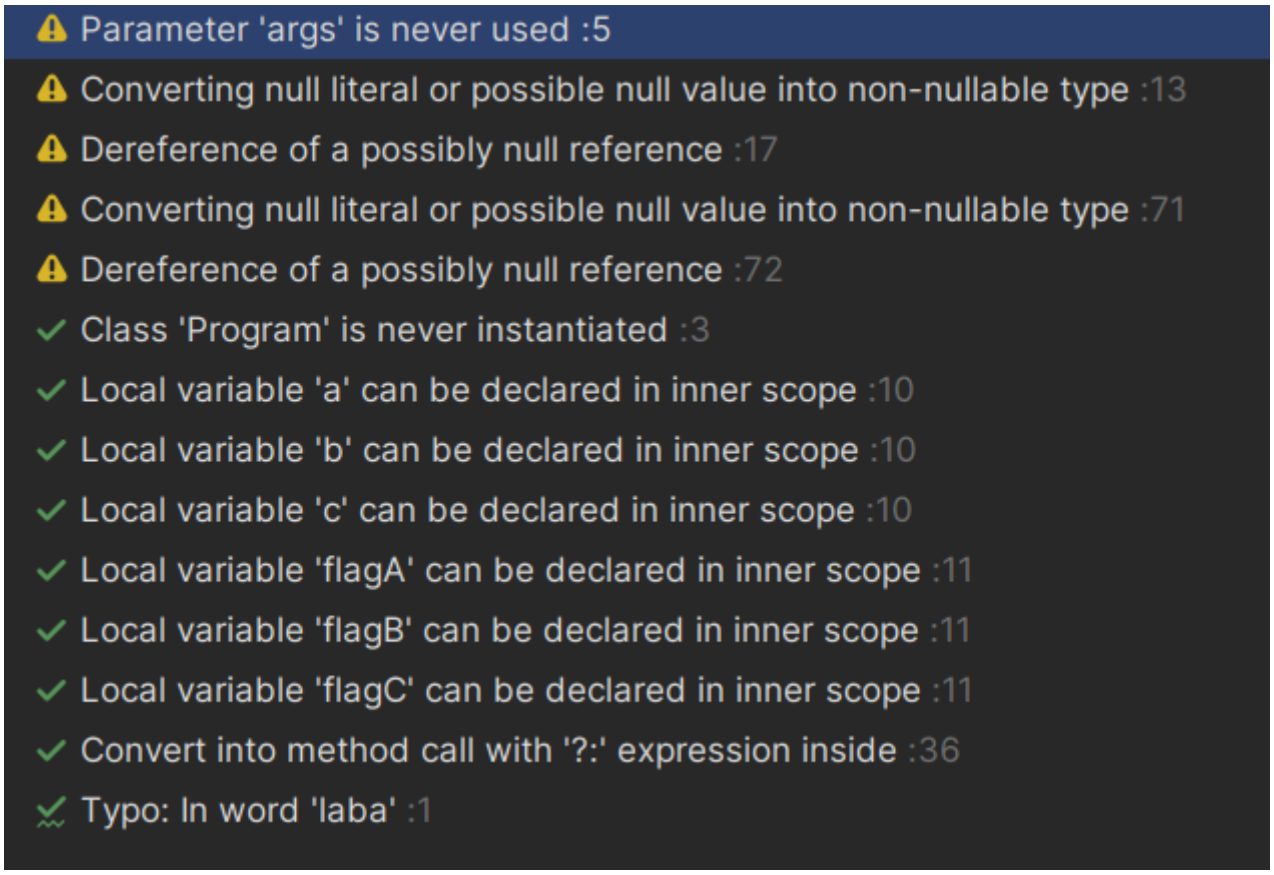


Рисунок 7 – Предупреждения от компилятора

Компилятор делает предупреждения, которые не являются обязательными, например о том, что параметры args в методе Main нигде не используются, а также даёт рекомендации о том, как и где стоит объявлять переменные. Теперь посмотрим на интерфейс в режиме отладки и изучим его:

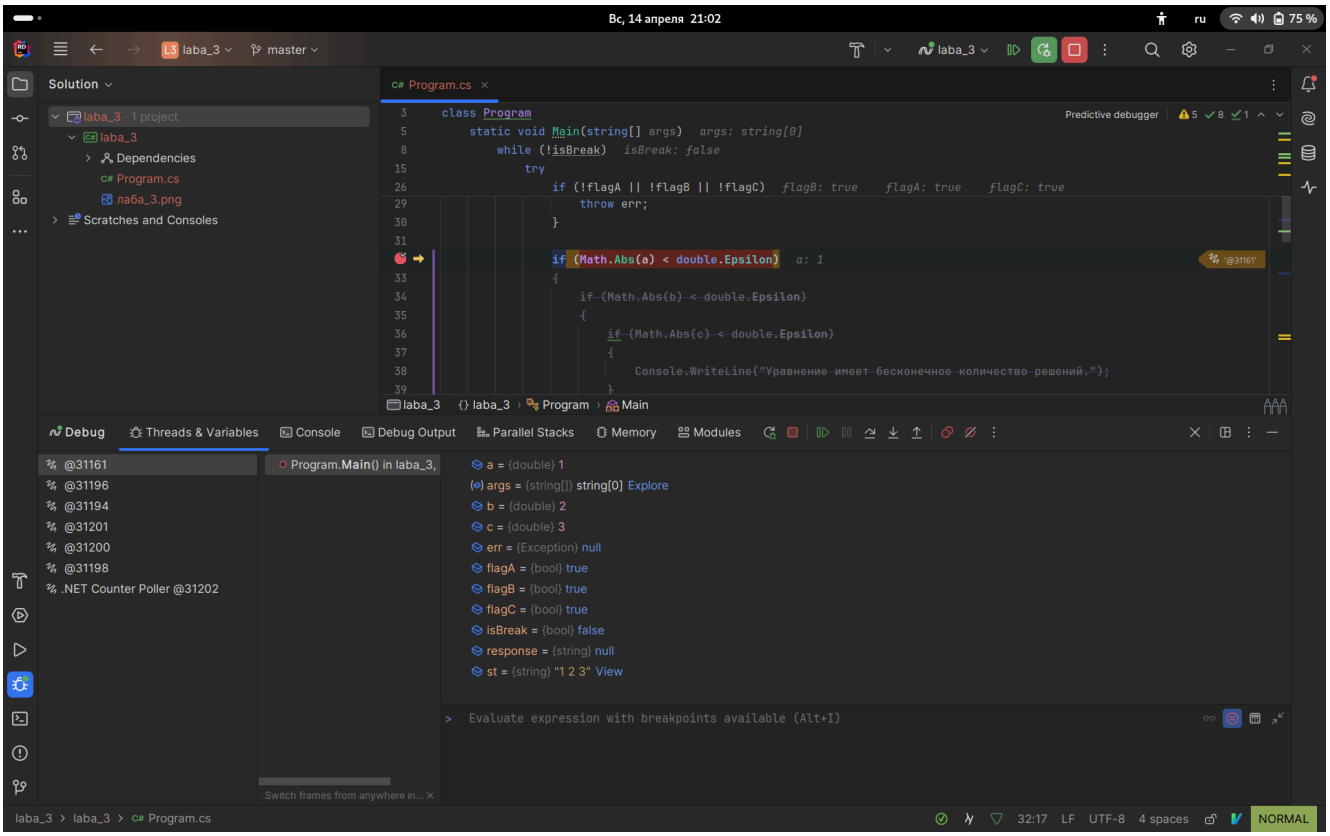


Рисунок 8 – Вид интерфейса при отладке

Как видно из рисунка, точка останова стоит на месте, когда уже введены 3 числа, также видно состояние других переменных.

Вывод

В ходе лабораторной работы были получены навыки работы с объектно-ориентированным языком C#, написание простейших консольных программ на нём и с отладчиком в профессиональной среде разработки Rider.