

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

По дисциплине Объектно-ориентированное программирование

Тема работы Создание программы с помощью среды разработки Visual Studio.NET

Обучающийся Буров Глеб Максимович

Факультет факультет инфокоммуникационных технологий

Группа K3223

Направление подготовки 11.03.02 Инфокоммуникационные технологии и системы связи

Образовательная программа Программирование в инфокоммуникационных системах

Обучающийся

(дата)

(подпись)

Буров Г.М.
(Ф.И.О.)

Руководитель

(дата)

(подпись)

Иванов С.Е.
(Ф.И.О.)

СОДЕРЖАНИЕ

Стр.

ВВЕДЕНИЕ	3
1 Ход работы	4
1.1 Упражнение 1	4
1.2 Упражнение 2	5
1.3 Упражнение 3	5
1.4 Упражнение 4	7
1.5 Упражнение 5	9
ЗАКЛЮЧЕНИЕ	11

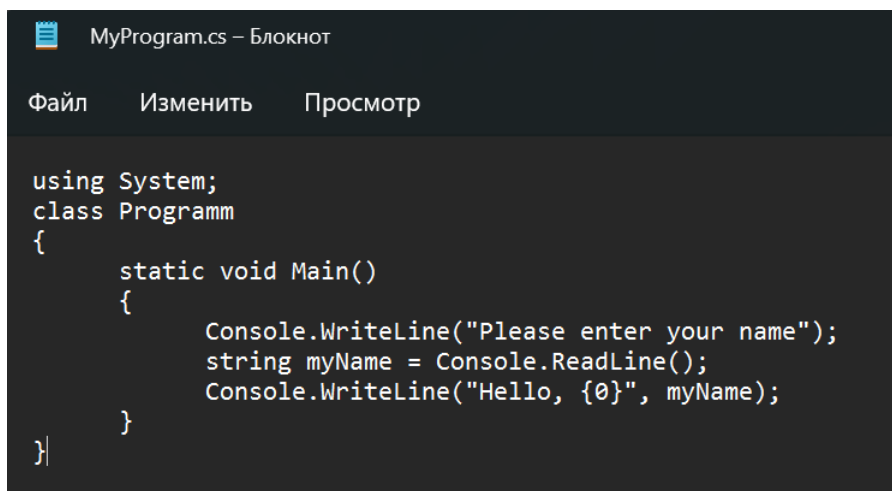
ВВЕДЕНИЕ

Целью данной лабораторной работы является ознакомление со средой разработки Visual Studio.NET, а также создание первых программ на языке программирования C#.

1 Ход работы

1.1 Упражнение 1

В первом упражнении необходимо реализовать программу, которая приветствует пользователя, на языке C#, используя обычный текстовый редактор (рис. 1.1).

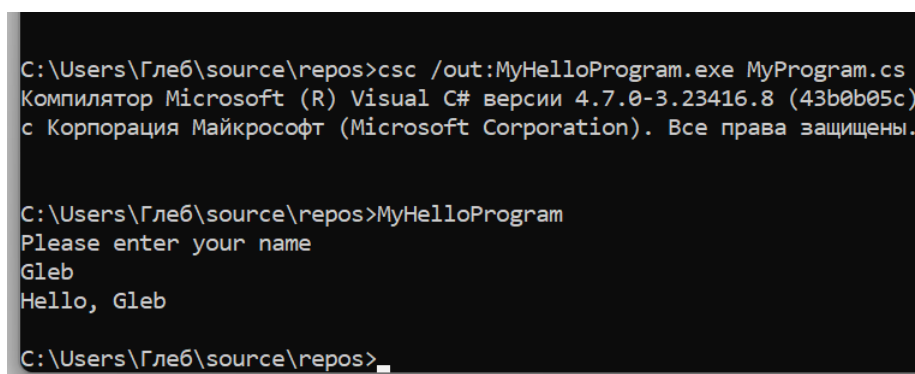


```
MyProgram.cs – Блокнот
Файл  Изменить  Просмотр

using System;
class Program
{
    static void Main()
    {
        Console.WriteLine("Please enter your name");
        string myName = Console.ReadLine();
        Console.WriteLine("Hello, {0}", myName);
    }
}
```

Рисунок 1.1 — Итоговый код для упражнения 1

После этого необходимо откомпилировать и запустить программу из командной строки (рис. 1.2).



```
C:\Users\Глеб\source\repos>csc /out:MyHelloProgram.exe MyProgram.cs
Компилятор Microsoft (R) Visual C# версии 4.7.0-3.23416.8 (43b0b05c)
© Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

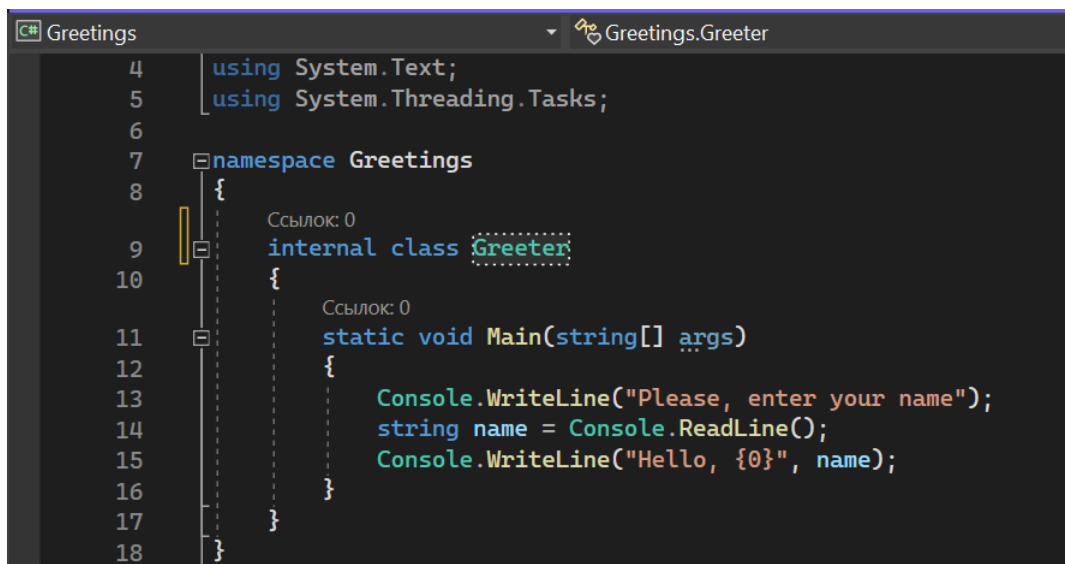
C:\Users\Глеб\source\repos>MyHelloProgram
Please enter your name
Gleb
Hello, Gleb

C:\Users\Глеб\source\repos>
```

Рисунок 1.2 — Результат работы программы из упражнения 1

1.2 Упражнение 2

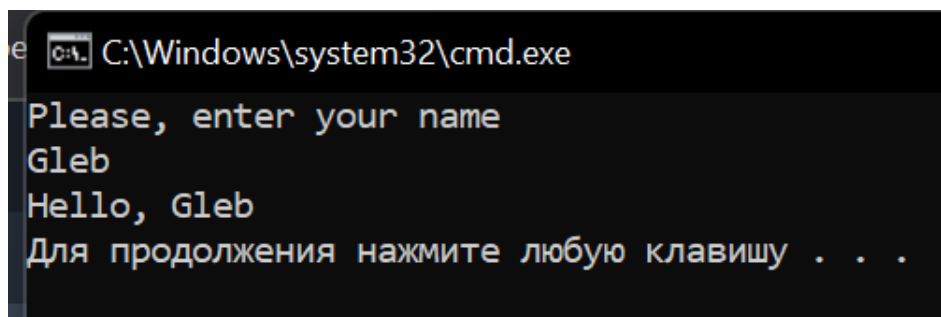
Во втором задании необходимо реализовать программу из прошлого упражнения с помощью среды разработки Visual Studio. Для этого был создан новый проект Greetings и класс Greeter (рис. 1.3).



```
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace Greetings
8  {
9      internal class Greeter
10     {
11         static void Main(string[] args)
12         {
13             Console.WriteLine("Please, enter your name");
14             string name = Console.ReadLine();
15             Console.WriteLine("Hello, {0}", name);
16         }
17     }
18 }
```

Рисунок 1.3 — Итоговый код для упражнения 2

Пример работы программы изображен на рисунке 1.4.



```
C:\Windows\system32\cmd.exe
Please, enter your name
Gleb
Hello, Gleb
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1.4 — Пример работы программы из упражнения 2

1.3 Упражнение 3

В третьем задании нужно изучить функционал работы интегрированного отладчика Visual Studio.NET: пройти программу по шагам и просмотреть изменения значений переменных. Например, установив точку остановки

программы на строчке, в которой выводится сообщение для пользователя «Please, enter your name», можно увидеть значение переменной myName до ввода данных (рис. 1.5).

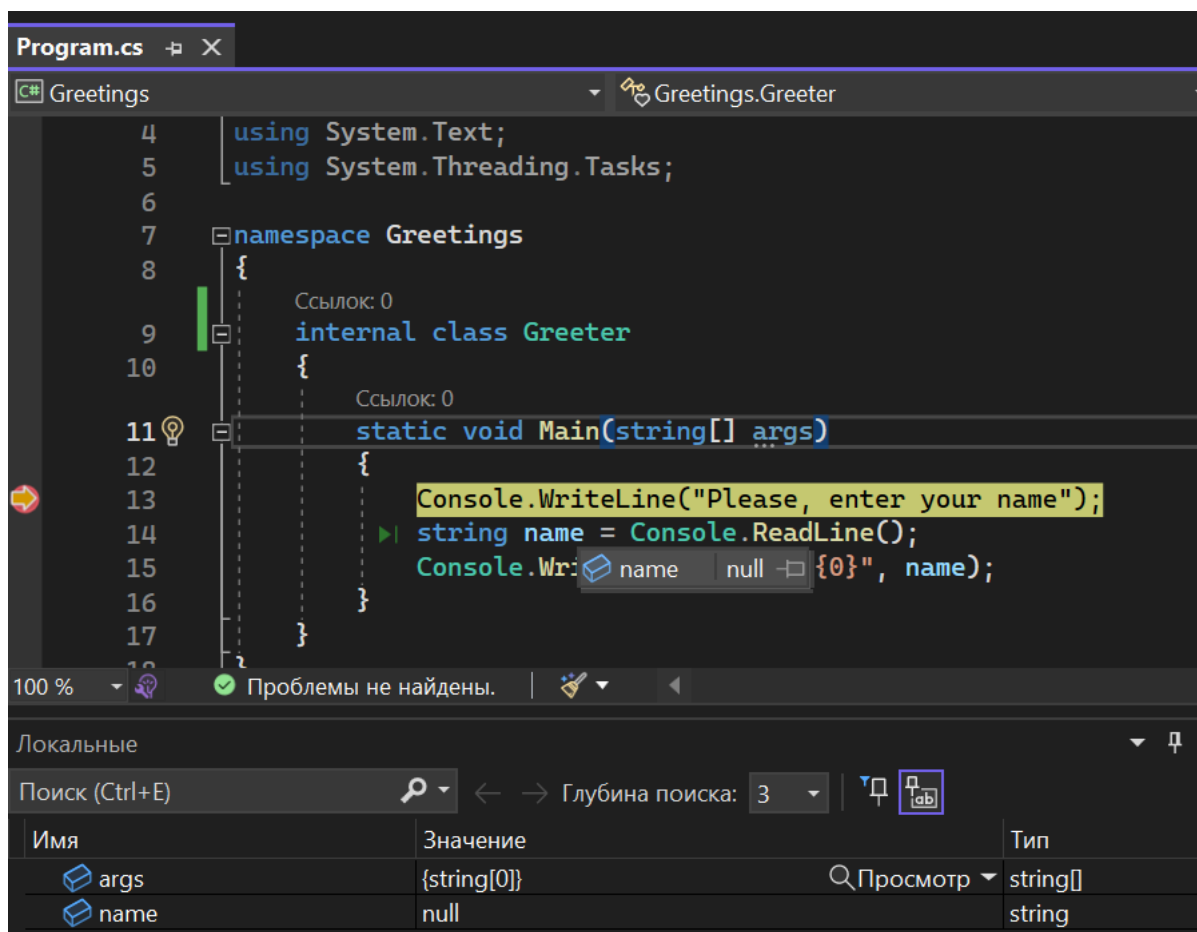


Рисунок 1.5 — Значение переменной myName до ввода данных

Аналогично, используя функционал Visual Studio — «Шаг вперёд» или «Шаг с заходом» — можно увидеть значение переменной myName после ввода имени (рис. 1.6).

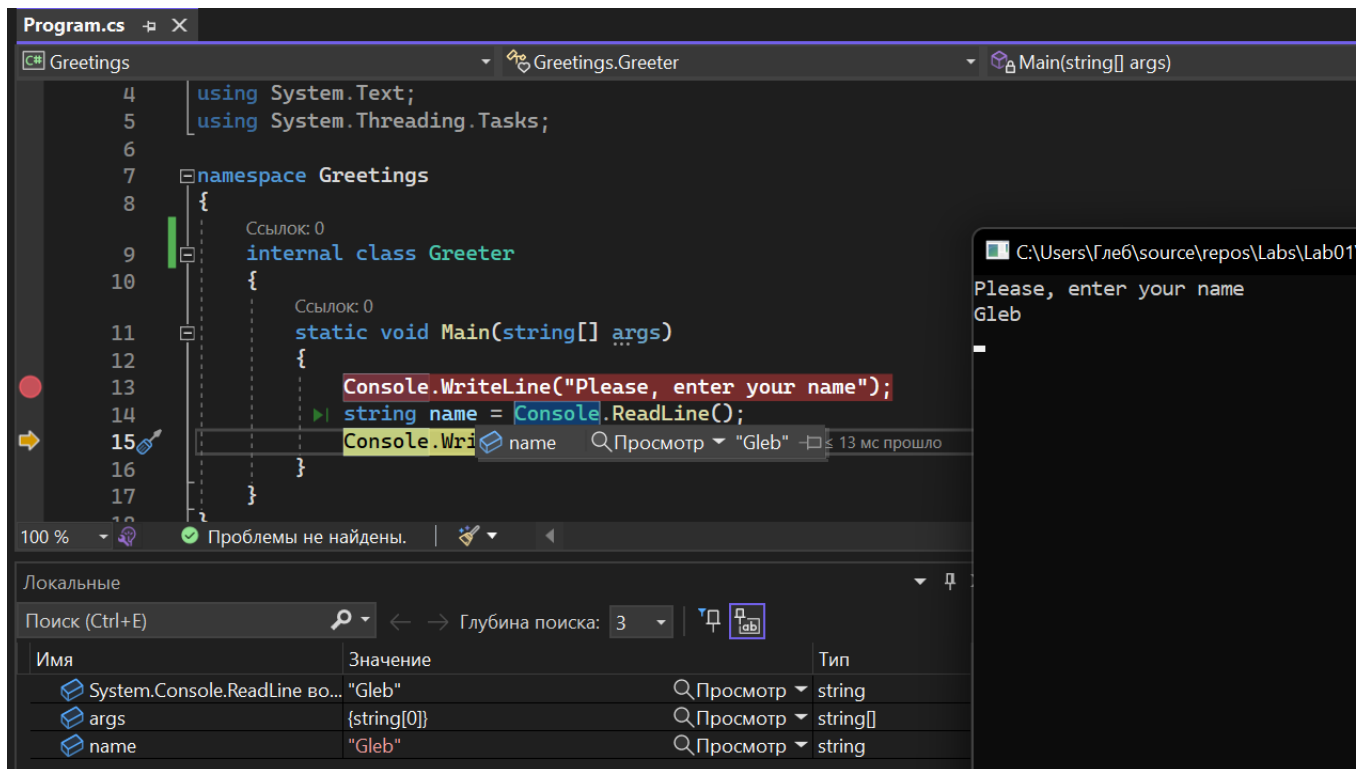


Рисунок 1.6 — Значение переменной `myName` после ввода данных

1.4 Упражнение 4

В четвертом упражнении необходимо реализовать программу, в которой будет использоваться обработчик исключительных ситуаций, который будет отлавливать ошибки времени выполнения. Программа должна запрашивать у пользователя два целых числа, делить первое на второе и выводить полученный результат.

Для этого был создан новый проект `Divisor` с классом `DivideIt`, который реализовывал необходимый функционал (рис. 1.7). Конструкция `try/catch` в программе необходима для обработки некорректных входных данных.

```

internal class DivideIt
{
    Ссылка: 0
    public static void Main(string[] args)
    {
        try
        {
            Console.WriteLine("Please enter the first integer");
            string temp = Console.ReadLine();
            int i = Int32.Parse(temp);
            Console.WriteLine("Please enter the second integer");
            temp = Console.ReadLine();
            int j = Int32.Parse(temp);
            int k = i / j;
            Console.WriteLine("The result of dividing {0} by {1} is {2}", i, j, k);
        }
        catch (FormatException e)
        {
            Console.WriteLine("Ошибка формата ввода: {0}", e);
        }
        catch (DivideByZeroException e)
        {
            Console.WriteLine("An exception was thrown: {0}", e.Message);
        }
    }
}

```

Рисунок 1.7 — Код программы из упражнения 4

Примеры работы программы, в том числе на некорректных входных данных, показаны на рисунке 1.8.

```

C:\Windows\system32\cmd.exe
Please enter the first integer
10
Please enter the second integer
0
An exception was thrown: Попытка деления на ноль.
Для продолжения нажмите любую клавишу . . .

C:\Windows\system32\cmd.exe
Please enter the first integer
4
Please enter the second integer
2
The result of dividing 4 by 2 is 2
Для продолжения нажмите любую клавишу . . .

C:\Windows\system32\cmd.exe
Please enter the first integer
13
Please enter the second integer
gfslhs
Ошибка формата ввода: System.FormatException: Входная строка имела неверный формат
в System.Number.StringToNumber(String str, NumberStyles options, NumberBuffer&

```

Рисунок 1.8 — Примеры работы программы из упражнения 4

1.5 Упражнение 5

В последнем упражнении нужно было самостоятельно реализовать программу, которая подсчитывает площадь равностороннего треугольника, периметр которого заранее известен. Также необходимо было реализовать диалог с пользователем: значение периметра должно вводиться с клавиатуры.

Для этого был создан новый проект Square с классом TriangleSquare, в методе Main которого объявляются 3 переменные типа double: perimeter (который впоследствии будет введен пользователем), side (используется для промежуточного расчета — длина стороны) и square (ответ — площадь треугольника) (рис. 1.9).

После того, как на экран выводится сообщение с просьбой пользователя ввести периметр, метод Parse преобразует введенные данные в тип double. Стоит отметить, что этот код находится внутри блока try, чтобы обработать случай ввода некорректных данных.

После этого рассчитывается длина стороны, площадь треугольника (по формуле Герона) и выводятся на экран.

The image shows a code editor with a dark background. It contains C# code for a class named TriangleSquare. The code includes a static Main method that prompts the user for a perimeter value, calculates the side length and area, and prints the results. It also includes a try-catch block to handle input errors. The code is as follows:

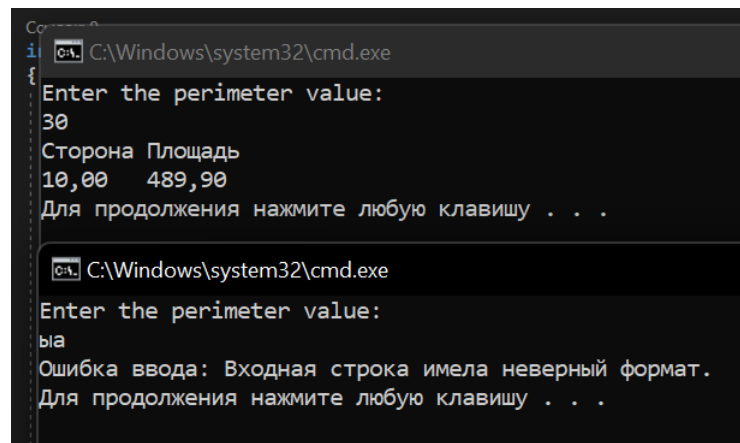
```
internal class TriangleSquare
{
    Ссылка: 0
    static void Main(string[] args)
    {
        try
        {
            double perimeter;
            double side;
            double square;

            Console.WriteLine("Enter the perimeter value: ");
            perimeter = double.Parse(Console.ReadLine());
            side = perimeter / 3;
            square = Math.Sqrt(perimeter * Math.Pow((perimeter - side), 3));

            Console.WriteLine("Сторона" + '\t' + "Площадь");
            Console.WriteLine("{0:F}" + '\t' + "{1:F2}", side, square);
        }
        catch (FormatException e) {
            Console.WriteLine("Ошибка ввода: {0}", e.Message);
        }
    }
}
```

Рисунок 1.9 — Код программы из упражнения 5

Примеры работы программы изображены на рисунке 1.10.



```
C:\Windows\system32\cmd.exe
Enter the perimeter value:
30
Сторона Площадь
10,00 489,90
Для продолжения нажмите любую клавишу . . .

C:\Windows\system32\cmd.exe
Enter the perimeter value:
ыя
Ошибка ввода: Входная строка имела неверный формат.
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1.10 — Примеры работы программы из упражнения 5

ЗАКЛЮЧЕНИЕ

В ходе этой лабораторной работы я ознакомился со средой разработки Visual Studio.NET, научился создавать новые проекты и классы, работать с переменными, обрабатывать исключения, а также изучил, как работает отладка в VS. Все упражнения были выполнены.