МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Объектно-ориентированное программирование»

Тема: Использование методов

Студенты гр. 3363 _____ Овсейчик Н.И.
Минко Д.А.

Преподаватель _____ Новакова Н.Е.

Санкт-Петербург
2024

Цель работы

Цель лабораторной работы состоит в освоении применения методов в программировании, включая использование параметров, ссылочных и выходных параметров, а также освоение рекурсивных функций для решения практических задач. В ходе работы студенты создадут программы, сравнивающие числа, обменивающие их значениями, вычисляющие факториалы с использованием циклов и рекурсии, что позволит закрепить навыки разработки и отладки кода в среде Microsoft Visual Studio.

Ход работы

1. Использование параметров в методах, возвращающих значение.

В ходе выполнения задания в среде Visual Studio 2008 был создан новый проект типа Console Application. В классе Utils реализован метод Greater, который принимает два целочисленных параметра и возвращает большее из них. В классе Test в методе Main были определены переменные для ввода значений с консоли, сравнения чисел и вывода результата. После вызова метода Greater результат был сохранен в переменную и выведен на консоль. Программа скомпилирована, проведена корректировка ошибок, и результат сохранен (рис. 1).

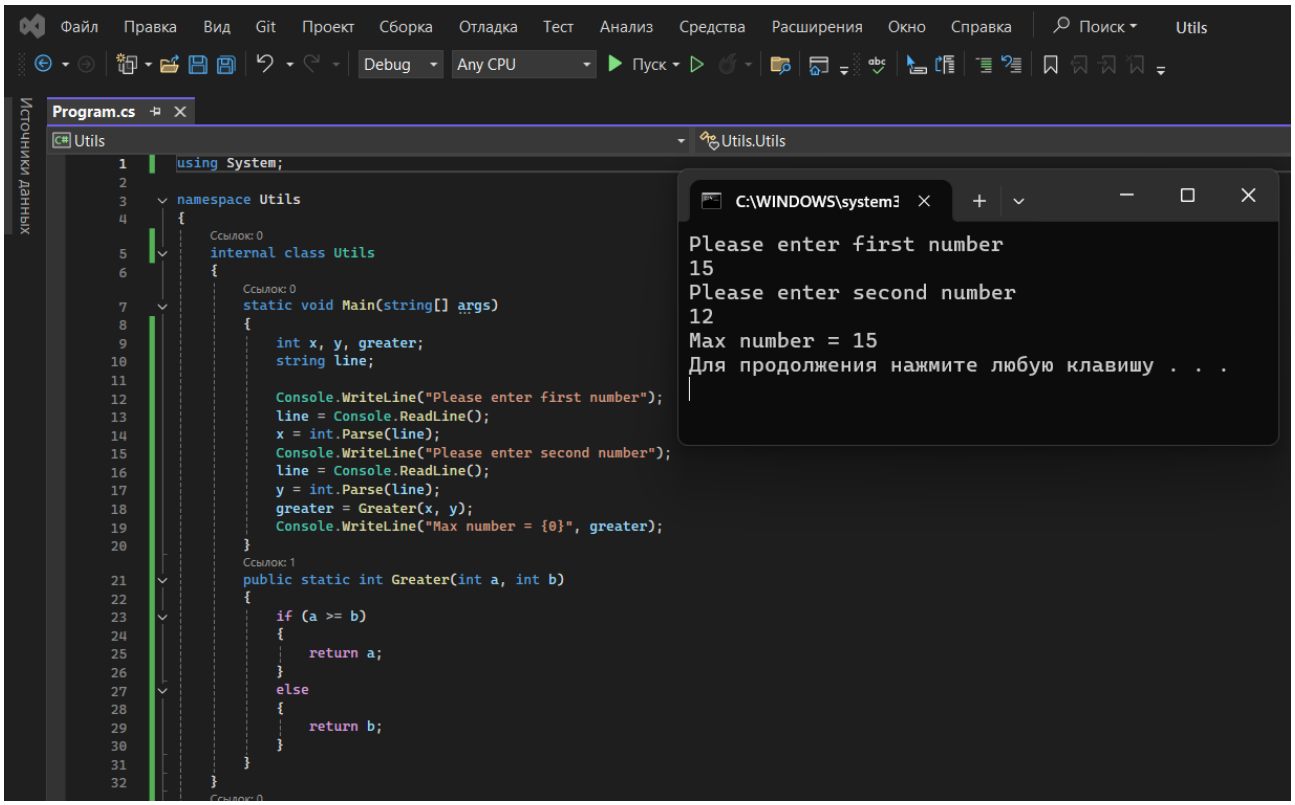

Рисунок 1 – Реализация метода Greater для сравнения двух целочисленных параметров

2. Использование методов со ссылочными параметрами.

В программе, созданной в предыдущем упражнении, был добавлен метод Swap в класс Utils, который принимает два целочисленных параметра по ссылке

и меняет их значения с помощью вспомогательной переменной. В методе Main были выведены исходные значения переменных, вызван метод Swap, после чего на консоль выведены измененные значения переменных. Программа скомпилирована, проведена корректировка ошибок и сохранен результат (рис. 2).

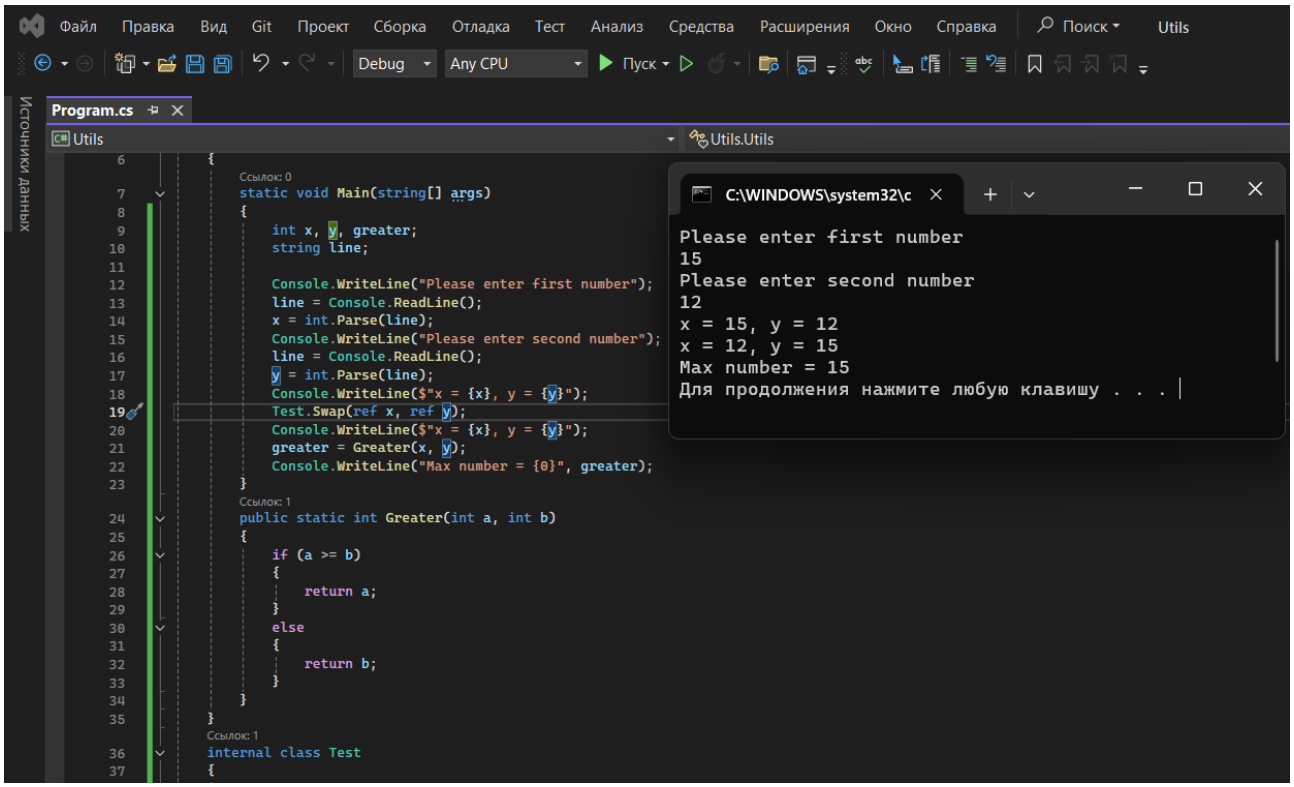

Рисунок 2 – Реализация метода Swap для обмена значениями двух целочисленных параметров, передаваемых по ссылке

3. Использование методов с выходными параметрами.

В проект, созданный ранее, был добавлен новый метод Factorial в класс Utils. Этот метод принимает два параметра: входной целочисленный параметр n и выходной параметр answer, вычисляющий факториал числа с помощью цикла for. Метод возвращает значение типа bool для проверки успешности выполнения. В методе Main был вызван метод Factorial, после чего программа скомпилирована, протестирована, исправлены ошибки и сохранен результат (рис. 3).

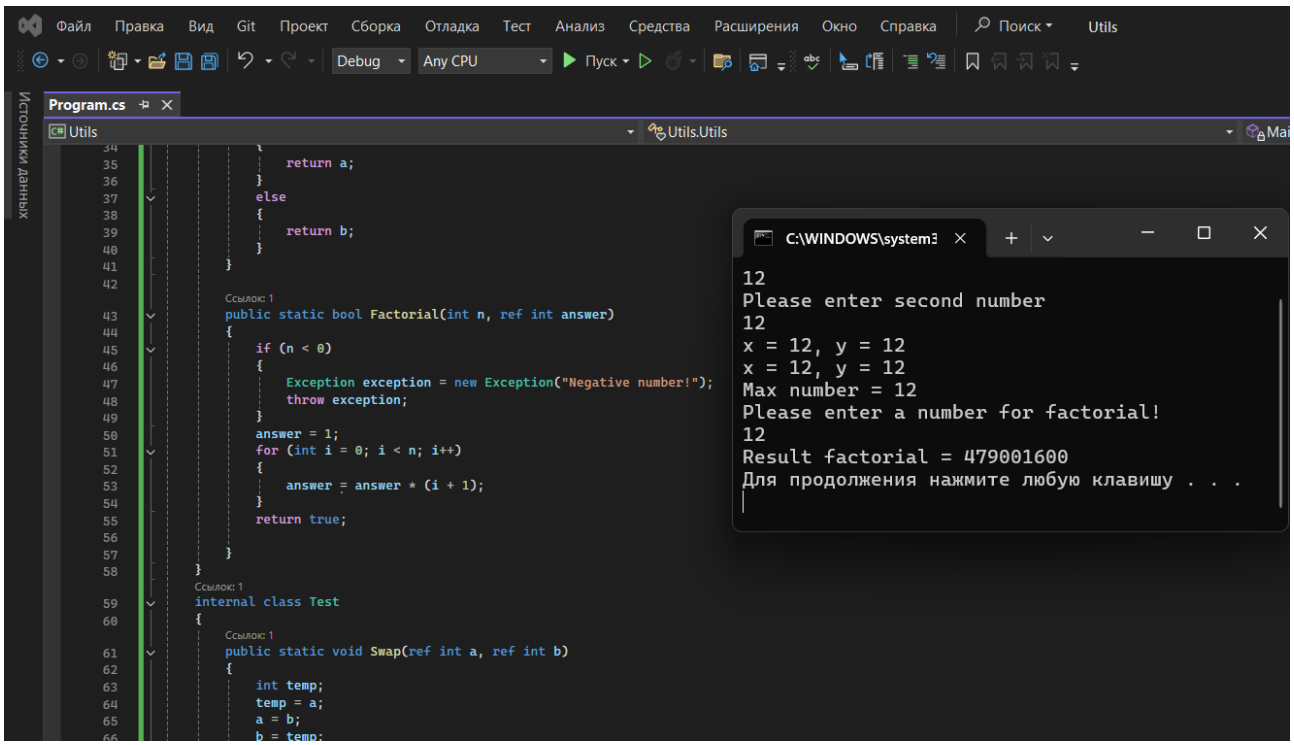

Рисунок 3 – Реализация метода Factorial для вычисления факториала числа

4. Использование альтернативного варианта вычисления факториала с применением рекурсии (рис. 4).

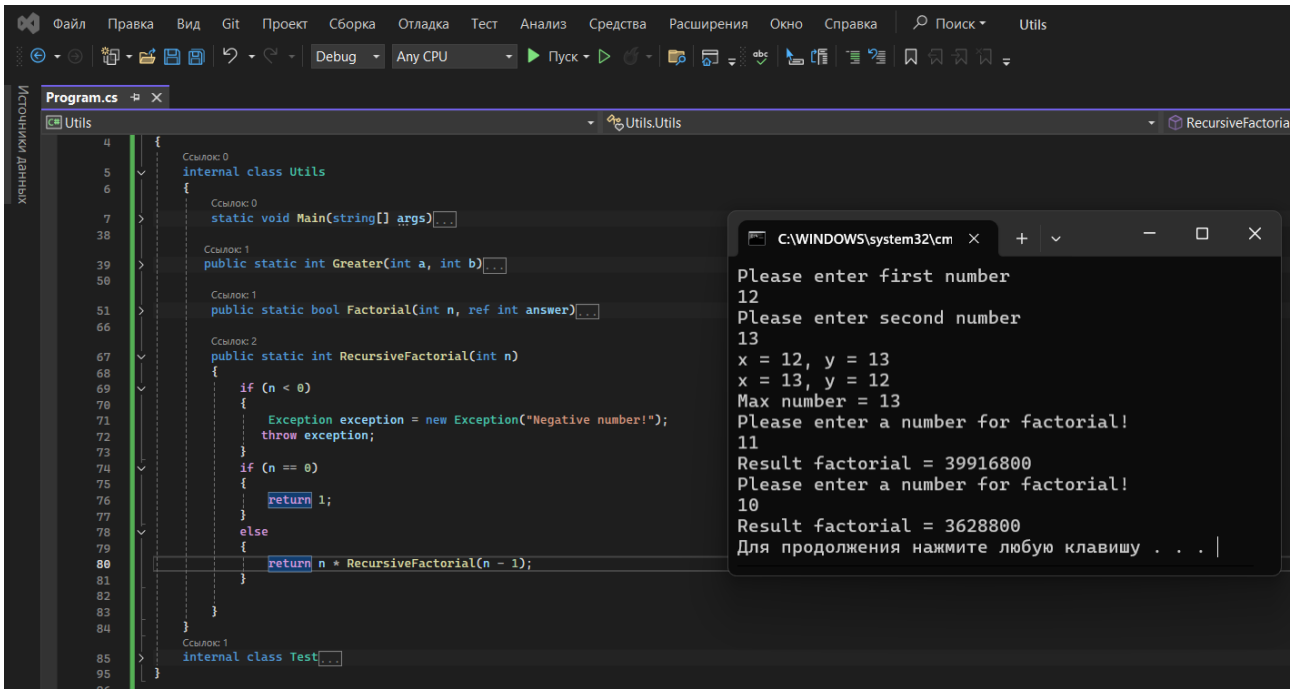

Рисунок 4 – Реализация метода Factorial для вычисления факториала числа с использованием рекурсии

Вывод

В ходе лабораторной работы были изучены и применены методы с различными типами параметров: обычными, ссылочными и выходными. Также был реализован расчет факториала с использованием циклов и рекурсивного подхода. Работа способствовала закреплению навыков программирования, отладки и компиляции в среде Microsoft Visual Studio.

ИСХОДНЫЙ КОД

Упражнение 1:

```
using System;

namespace Utils
{
    internal class Utils
    {
        static void Main(string[] args)
        {
            int x, y, greater;
            string line;

            Console.WriteLine("Please enter first number");
            line = Console.ReadLine();
            x = int.Parse(line);
            Console.WriteLine("Please enter second number");
            line = Console.ReadLine();
            y = int.Parse(line);
            greater = Greater(x, y);
            Console.WriteLine("Max number = {0}", greater);
        }
        public static int Greater(int a, int b)
        {
            if (a >= b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }
    }
    internal class Test
    {

    }
}
```

Упражнение 2:

```
using System;

namespace Utils
{
    internal class Utils
    {
        static void Main(string[] args)
        {
            int x, y, greater;
```

```
            string line;

            Console.WriteLine("Please enter first number");
            line = Console.ReadLine();
            x = int.Parse(line);
            Console.WriteLine("Please enter second number");
            line = Console.ReadLine();
            y = int.Parse(line);
            Console.WriteLine($"x = {x}, y = {y}");
            Test.Swap(ref x, ref y);
            Console.WriteLine($"x = {x}, y = {y}");
            greater = Greater(x, y);
            Console.WriteLine("Max number = {0}", greater);
        }
        public static int Greater(int a, int b)
        {
            if (a >= b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }
    }
    internal class Test
    {
        public static void Swap(ref int a, ref int b)
        {
            int temp;
            temp = a;
            a = b;
            b = temp;
        }
    }
}
```

Упражнение 3:

```
using System;

namespace Utils
{
    internal class Utils
    {
        static void Main(string[] args)
        {
            int x, y, greater, n;
            int answer = 0;
            string line;
```

```
            Console.WriteLine("Please enter first number");
            line = Console.ReadLine();
            x = int.Parse(line);
            Console.WriteLine("Please enter second number");
            line = Console.ReadLine();
            y = int.Parse(line);
            Console.WriteLine($"x = {x}, y = {y}");
            Test.Swap(ref x, ref y);
            Console.WriteLine($"x = {x}, y = {y}");
            greater = Greater(x, y);
            Console.WriteLine("Max number = {0}", greater);

            Console.WriteLine("Please enter a number for
factorial!");
            line = Console.ReadLine();
            n = int.Parse(line);
            Factorial(n, ref answer);
            Console.WriteLine($"Result factorial = {answer}");
        }
        public static int Greater(int a, int b)
        {
            if (a >= b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }

        public static bool Factorial(int n, ref int answer)
        {
            if (n < 0)
            {
                Exception exception = new Exception("Negative
number!");
                throw exception;
            }
            answer = 1;
            for (int i = 0; i < n; i++)
            {
                answer = answer * (i + 1);
            }
            return true;

        }
    }
    internal class Test
    {
        public static void Swap(ref int a, ref int b)
        {
            int temp;
```

```
                temp = a;
                a = b;
                b = temp;
            }
        }
    }
```

Упражнение 4:

```
using System;

namespace Utils
{
    internal class Utils
    {
        static void Main(string[] args)
        {
            int x, y, greater, n;
            int answer = 0;
            string line;

            Console.WriteLine("Please enter first number");
            line = Console.ReadLine();
            x = int.Parse(line);
            Console.WriteLine("Please enter second number");
            line = Console.ReadLine();
            y = int.Parse(line);

            Console.WriteLine($"x = {x}, y = {y}");
            Test.Swap(ref x, ref y);
            Console.WriteLine($"x = {x}, y = {y}");
            greater = Greater(x, y);
            Console.WriteLine("Max number = {0}", greater);

            Console.WriteLine("Please enter a number for
factorial!");
            line = Console.ReadLine();
            n = int.Parse(line);
            Factorial(n, ref answer);
            Console.WriteLine($"Result factorial = {answer}");

            Console.WriteLine("Please enter a number for
factorial!");
            line = Console.ReadLine();
            n = int.Parse(line);
            answer = RecursiveFactorial(n);
            Console.WriteLine($"Result factorial = {answer}");
        }

        public static int Greater(int a, int b)
        {
            if (a >= b)
            {
```

```
                return a;
            }
            else
            {
                return b;
            }
        }

        public static bool Factorial(int n, ref int answer)
        {
            if (n < 0)
            {
                Exception exception = new Exception("Negative
number!");
                throw exception;
            }
            answer = 1;
            for (int i = 0; i < n; i++)
            {
                answer = answer * (i + 1);
            }
            return true;

        }

        public static int RecursiveFactorial(int n)
        {
            if (n < 0)
            {
                Exception exception = new Exception("Negative
number!");
                throw exception;
            }
            if (n == 0)
            {
                return 1;
            }
            else
            {
                return n * RecursiveFactorial(n - 1);
            }

        }
    }
    internal class Test
    {
        public static void Swap(ref int a, ref int b)
        {
            int temp;
            temp = a;
            a = b;
            b = temp;
        }
```

```
        }
    }
```