

**Санкт-Петербургский национальный исследовательский  
университет информационных технологий, механики и оптики**

## **ЛАБОРАТОРНАЯ РАБОТА №4**

**Создание и использование методов**

Студент:

*Швалов Даниил Андреевич*

*Факультет ИКТ*

*Группа: K32211*

Преподаватель:

*Иванов Сергей Евгеньевич*

## Упражнение 1. Использование параметров в методах, возвращающих значения

Выполнив все шаги из задания, я получил следующий код:

---

```
using System;
namespace Utils
{
    public class Utils
    {
        public static int Greater(int a, int b)
        {
            if (a > b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }
    }
}
```

---

---

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

        Console.WriteLine("Введите первое число:");
        x = int.Parse(Console.ReadLine());
        Console.WriteLine("Введите второе число:");
        y = int.Parse(Console.ReadLine());

        int greater = Utils.Greater(x, y);
        Console.WriteLine("Большим из чисел {0} и {1} является {2} ", x, y, greater);
    }
}
```

---

На рис. 1.1 приведен пример работы программы.

```
Terminal - Utils
В в е д и т е   п е р в о е   ч и с л о   :
5
В в е д и т е   в т о р о е   ч и с л о   :
6
Б о л ь ш и м   и з   ч и с е л   5   и   6   я в л я е т с я   6
```

Рис. 1.1: Пример работы программы

## Упражнение 2. Использование в методах параметров, передаваемых по ссылке

Выполнив все шаги из задания, я получил следующий код:

---

```
using System;
namespace Utils
{
    public class Utils
    {
        public static int Greater(int a, int b)
        {
            if (a > b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }

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

---

---

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

        Console.WriteLine("Введите первое число:");
```

```

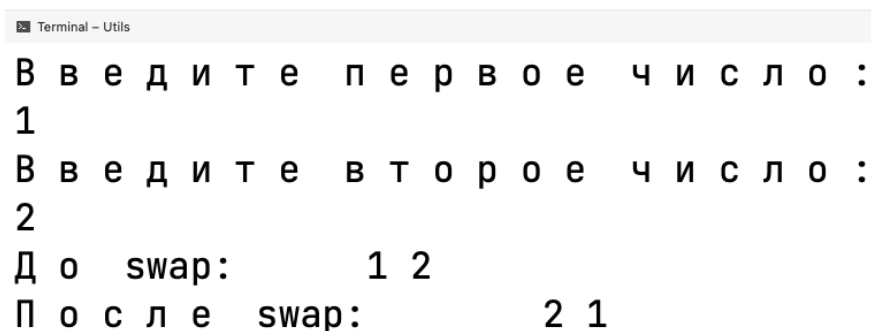
    x = int.Parse(Console.ReadLine());
    Console.WriteLine("Введите второе число:");
    y = int.Parse(Console.ReadLine());

    Console.WriteLine("До swap: \t" + x + " " + y);
    Utils.Swap(ref x, ref y);
    Console.WriteLine("После swap: \t" + x + " " + y);
}
}

```

---

На рис. 2.1 приведен пример работы программы.



```

Terminal - Utils
В в е д и т е   п е р в о е   ч и с л о :
1
В в е д и т е   в т о р о е   ч и с л о :
2
Д о   s w a p :           1 2
П о с л е   s w a p :           2 1

```

Рис. 2.1: Пример работы программы

### Упражнение 3. Использование возвращаемых параметров в методах

Выполнив все шаги из задания, я получил следующий код:

---

```

using System;
namespace Utils
{
    public class Utils
    {
        public static int Greater(int a, int b)
        {
            if (a > b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }

        public static void Swap(ref int a, ref int b)
        {
            int temp = a;

```

```

        a = b;
        b = temp;
    }

    public static bool Factorial(int n, out int answer)
    {
        int k;
        int f = 1;
        bool ok = true;

        try
        {
            checked
            {
                for (k = 2; k ≤ n; ++k)
                {
                    f = f * k;
                }
            }
        }
        catch (Exception)
        {
            f = 0;
            ok = false;
        }

        answer = f;
        return ok;
    }
}

```

---

```

namespace Utils;
class Program
{
    static void Main(string[] args)
    {
        int f, x;
        bool ok;

        Console.WriteLine("Number for factorial:");
        x = int.Parse(Console.ReadLine());
        ok = Utils.Factorial(x, out f);

        if (ok)
        {
            Console.WriteLine("Factorial(" + x + ") = " + f);
        }
        else

```

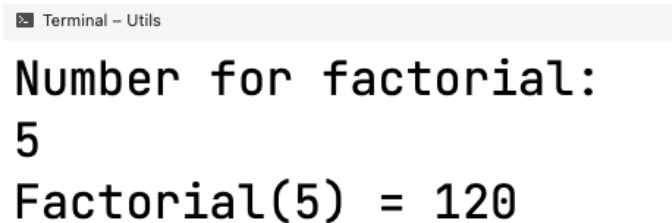
```

    {
        Console.WriteLine("Cannot compute this factorial");
    }
}

```

---

На рис. 3.1 и 3.2 приведен пример работы программы.

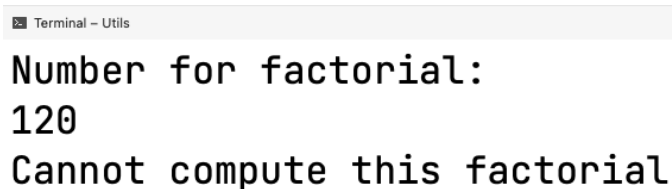


```

Terminal - Utils
Number for factorial:
5
Factorial(5) = 120

```

Рис. 3.1: Пример работы программы с небольшими значениями



```

Terminal - Utils
Number for factorial:
120
Cannot compute this factorial

```

Рис. 3.2: Пример работы программы с значениями, вызывающими переполнение

#### Упражнение 4. Расчет площади треугольника с помощью метода

В методе `IsTriangle` я проверяю, что каждая сторона строго больше нуля, а также каждая сторона меньше суммы двух других сторон треугольника.

В методе `Square` для трех аргументов я сначала вызываю метод `IsTriangle`. Если пользователь передал некорректные данные, то функция выбрасывает исключение. После проверки я с помощью формулы Герона нахожу площадь треугольника.

В методе `Square` для одного аргумента я вызываю метод `Square` для трех аргументов, используя одну и ту же сторону в качестве аргументов. Исходный код программы приведен в следующем листинге:

---

```

using System;
namespace Square
{
    public class Operation
    {
        public static double Square(double a, double b, double c)
        {
            if (!IsTriangle(a, b, c))
            {

```

```

        throw new ArgumentException(
            "The sides must satisfy the condition of the existence of a triangle"
        );
    }

    double p = (a + b + c) / 2;
    return Math.Sqrt(p * (p - a) * (p - b) * (p - c));
}

public static double Square(double side)
{
    return Square(side, side, side);
}

private static bool IsTriangle(double a, double b, double c)
{
    return a > 0 && b > 0 && c > 0 &&
        a + b > c && a + c > b && b + c > a;
}
}
}

```

---

В методе main я сначала спрашиваю у пользователя, какой тип треугольника использовать. После этого в зависимости от типа я считываю либо одну сторону, либо три стороны. Затем я вызываю метод Square и вывожу значение площади в стандартный вывод. Исходный код программы приведен в следующем листинге:

---

```

namespace Square;
class Program
{
    static void Main(string[] args)
    {
        try
        {
            Console.WriteLine("Is it an equilateral triangle? (y/N)");
            if (Console.ReadLine().Trim().ToLower() == "y")
            {
                Console.WriteLine("Enter the side of the triangle:");
                double side = double.Parse(Console.ReadLine());
                Console.WriteLine("Square: {0}", Operation.Square(side));
            }
            else
            {
                Console.WriteLine("Enter the first side of the triangle:");
                double a = double.Parse(Console.ReadLine());
                Console.WriteLine("Enter the second side of the triangle:");
                double b = double.Parse(Console.ReadLine());
                Console.WriteLine("Enter the third side of the triangle:");
            }
        }
        catch { }
    }
}

```

```

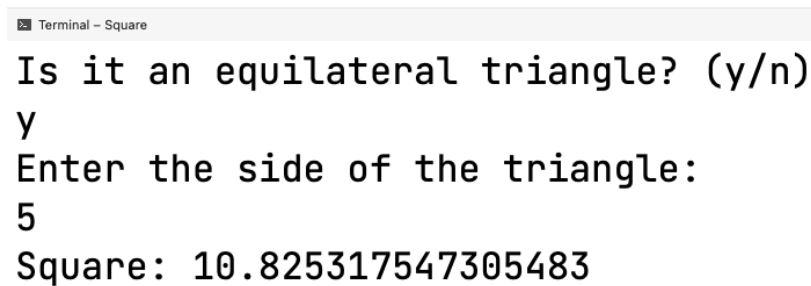
        double c = double.Parse(Console.ReadLine());

        Console.WriteLine("Square: {0}", Operation.Square(a, b, c));
    }
}
catch (ArgumentException e)
{
    Console.WriteLine(e.Message);
}
}
}

```

---

На рис. 4.1 приведен пример расчета площади равностороннего треугольника.



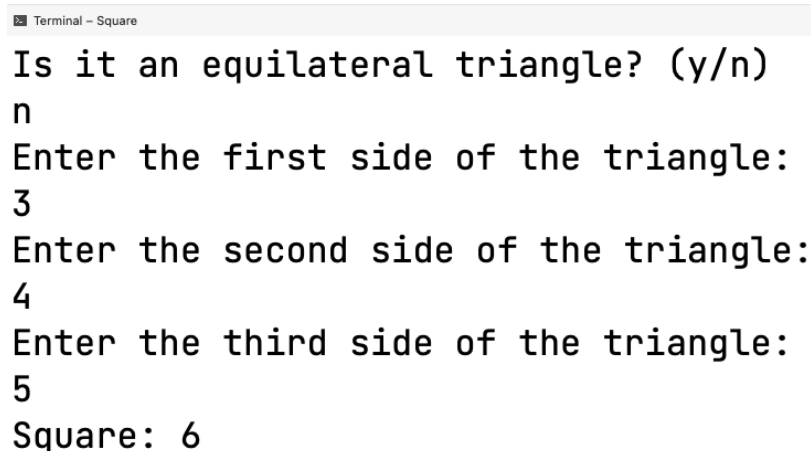
```

Terminal - Square
Is it an equilateral triangle? (y/n)
y
Enter the side of the triangle:
5
Square: 10.825317547305483

```

Рис. 4.1: Расчет площади равностороннего треугольника

На рис. 4.2 приведен пример расчета площади обычного треугольника.



```

Terminal - Square
Is it an equilateral triangle? (y/n)
n
Enter the first side of the triangle:
3
Enter the second side of the triangle:
4
Enter the third side of the triangle:
5
Square: 6

```

Рис. 4.2: Расчет площади обычного треугольника

На рис. 4.3 приведен пример расчета площади некорректного треугольника.



```
Terminal - Square
Is it an equilateral triangle? (y/n)
n
Enter the first side of the triangle:
3
Enter the second side of the triangle:
4
Enter the third side of the triangle:
10
The sides must satisfy the condition of the existence of a triangle
```

Рис. 4.3: Расчет площади некорректного треугольника

## Упражнение 5. Вычисление корней квадратного уравнения

В методе FindRoots я нахожу дискриминант уравнения, после чего, в зависимости от значения дискриминанта я нахожу корни. Исходный код программы представлен в следующем листинге:

---

```
using System;
namespace EquationRoots
{
    public class Equation
    {
        public static int FindRoots(double a, double b, double c,
                                    ref double x1, ref double x2)
        {
            double discriminant = b * b - 4 * a * c;
            if (discriminant > 0)
            {
                x1 = (-b + Math.Sqrt(discriminant)) / (2 * a);
                x2 = (-b - Math.Sqrt(discriminant)) / (2 * a);
                return 1;
            }
            else if (discriminant == 0)
            {
                x1 = x2 = -b / (2 * a);
                return 0;
            }
            else
            {
                return -1;
            }
        }
    }
}
```

---

В методе main я сначала считываю значения параметров уравнения a, b и c, а после этого вызываю метод FindRoots. В зависимости от возвращаемого значения

я вывожу ответ на экран. Исходный код программы представлен в следующем листинге:

---

```
namespace EquationRoots;
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Enter a:");
        double a = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter b:");
        double b = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter c:");
        double c = double.Parse(Console.ReadLine());

        double x1 = 0;
        double x2 = 0;

        int result = Equation.FindRoots(a, b, c, ref x1, ref x2);
        if (result == 1)
        {
            Console.WriteLine(
                "a = {0}, b = {1}, c = {2}, x1 = {3}, x2 = {4}",
                a, b, c, x1, x2
            );
        }
        else if (result == 0)
        {
            Console.WriteLine(
                "a = {0}, b = {1}, c = {2}, x1 = x2 = {3}",
                a, b, c, x1
            );
        }
        else
        {
            Console.WriteLine(
                "a = {0}, b = {1}, c = {2}, no roots",
                a, b, c
            );
        }
    }
}
```

---

На рис. 5.1 показан пример выражения, не имеющего корней.

```
Terminal - EquationRoots
Enter a:
1
Enter b:
-5
Enter c:
9
a = 1, b = -5, c = 9, no roots
```

Рис. 5.1: Пример, не имеющий корней

На рис. 5.2 показан пример выражения, имеющего только один корень.

```
Terminal - EquationRoots
Enter a:
1
Enter b:
-4
Enter c:
4
a = 1, b = -4, c = 4, x1 = x2 = 2
```

Рис. 5.2: Пример, имеющий только один корень

На рис. 5.3 показан пример выражения, имеющего два корня.

```
Terminal - EquationRoots
Enter a:
1
Enter b:
3
Enter c:
-4
a = 1, b = 3, c = -4, x1 = 1, x2 = -4
```

Рис. 5.3: Пример, имеющий два корня