

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

## **ЛАБОРАТОРНАЯ РАБОТА №3**

### **Использование выражений**

Студент:  
*Швалов Даниил Андреевич*  
Факультет ИКТ  
Группа: K32211  
Преподаватель:  
*Иванов Сергей Евгеньевич*

## Упражнение 1. Реализация операторов выбора

### Задание 1.1. Применение конструкции if-else-if

Выполнив все шаги из задания, я получил следующий исходный код:

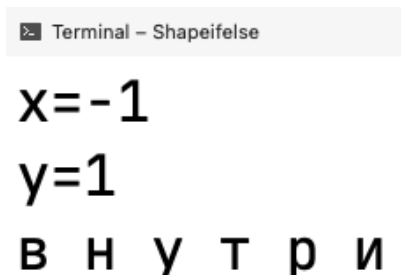
---

```
namespace Shapeifelse;
class Program
{
    static void Main(string[] args)
    {
        Console.Write("x=");
        float x = float.Parse(Console.ReadLine());
        Console.Write("y=");
        float y = float.Parse(Console.ReadLine());

        if (x * x + y * y < 9 && y > 0)
        {
            Console.WriteLine("внутри");
        }
        else if (x * x + y * y > 9 || y < 0)
        {
            Console.WriteLine("вне");
        }
        else
        {
            Console.WriteLine("на границе");
        }
    }
}
```

---

Пример работы программы представлен на рис. 1.1, 1.2 и 1.3.



```
Terminal - Shapeifelse
x=-1
y=1
внутри
```

Рис. 1.1: Ввод координат внутри области

```
Terminal - Shapeifelse

x=-3
y=3
в н е
```

Рис. 1.2: Ввод координат вне области

```
Terminal - Shapeifelse

x=-3
y=0
н а г р а н и ц е
```

Рис. 1.3: Ввод координат на границе области

## Задание 1.2. Применение оператора switch

Проделав все шаги из задания, я получил следующий исходный код:

---

```
namespace Calc_switch;
class Program
{
    static void Main(string[] args)
    {
        Console.Write("A = ");
        double a = double.Parse(Console.ReadLine());
        Console.Write("OP = ");
        char op = char.Parse(Console.ReadLine());
        Console.Write("B = ");
        double b = double.Parse(Console.ReadLine());

        bool ok = true;
        double res = 0;

        switch (op)
        {
            case '+':
                res = a + b;
                break;
            case '-':
                res = a - b;
                break;
            case '*':
                res = a * b;
```

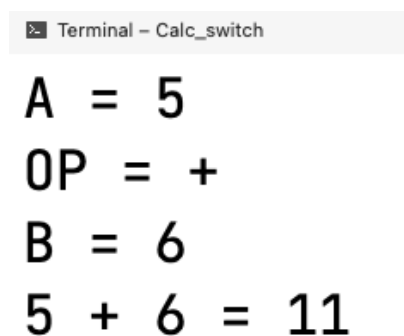
```

        break;
    case '/':
    case ':':
        res = a / b;
        break;
    default:
        ok = false;
        break;
}

if (ok)
{
    Console.WriteLine("{0} {1} {2} = {3}", a, op, b, res);
}
else
{
    Console.WriteLine("Операция не определена");
}
}
}

```

На рис. 1.4 представлена работа приложения с корректными данными.



```

Terminal - Calc_switch
A = 5
OP = +
B = 6
5 + 6 = 11

```

Рис. 1.4: Работа приложения с корректными данными

На рис. 1.5 показан случай деления числа на ноль. В результате работы программы мы получили бесконечность. Это связано с тем, что при делении числа с плавающей запятой на ноль по стандарту IEEE 754 должно получиться число, представляющее бесконечность.

```
Terminal - Calc_switch  
A = 5  
OP = /  
B = 0  
5 / 0 = ∞
```

Рис. 1.5: Деление числа на ноль

На рис. 1.6 показан случай деления нуля на ноль. Также, как и предыдущем случае, стандарт IEEE 754 указывает, что при делении нуля на ноль (т. е. при любой математически неопределенной операции) результатом должен быть NaN (not a number).

```
Terminal - Calc_switch  
A = 0  
OP = :  
B = 0  
0 : 0 = NaN
```

Рис. 1.6: Деление нуля на ноль

На рис. 1.7 представлен случай использования неправильного символа операции. В данном случае программа отработала корректно и сообщила об этом пользователю.

```
Terminal - Calc_switch  
A = 5  
OP = ?  
B = 2  
О п е р а ц и я   н е   о п р е д е л е н а
```

Рис. 1.7: Использование неизвестного оператора

### Задание 1.3. Определение високосного года

В своей программе я считываю год со стандартного ввода, а затем проверяю, кратен ли год 4 и при этом не кратен 100, или же кратен ли он 400. Если одно из этих условий выполняется, я выведу YES, а иначе – NO. Исходный код программы представлен в следующем листинге:

---

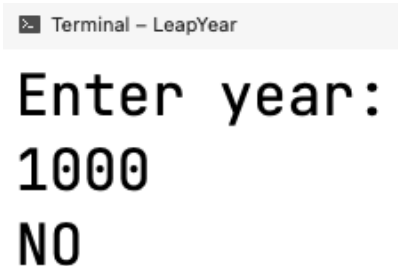
```
namespace LeapYear;  
class Program
```

```
{
    static void Main(string[] args)
    {
        Console.WriteLine("Enter year:");
        int year = int.Parse(Console.ReadLine());

        if (year % 4 == 0 && year % 100 != 0 || year % 400 == 0)
        {
            Console.WriteLine("YES");
        }
        else
        {
            Console.WriteLine("NO");
        }
    }
}
```

---

На рис. 1.8, 1.9 и 1.10 я проверяю работу программы для разных годов с разной кратностью.



```
> Terminal - LeapYear
Enter year:
1000
NO
```

Рис. 1.8: Год кратный 4 и кратный 100



```
> Terminal - LeapYear
Enter year:
2000
YES
```

Рис. 1.9: Год кратный 4, кратный 100 и кратный 400

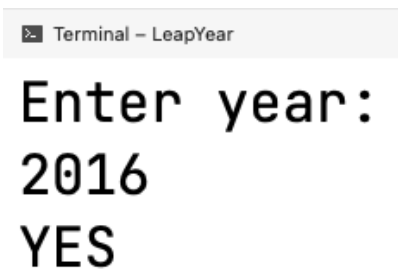


Рис. 1.10: Год кратный 4

## Упражнение 2. Реализация циклов при работе с данными размерных типов

### Задание 2.1. Использование операторов цикла while, do while и for

Я выполнил все шаги из задания и получил следующий исходный код:

---

```
namespace Loop;
class Program
{
    static void Main(string[] args)
    {
        Console.Write("n = ");
        int n = int.Parse(Console.ReadLine());

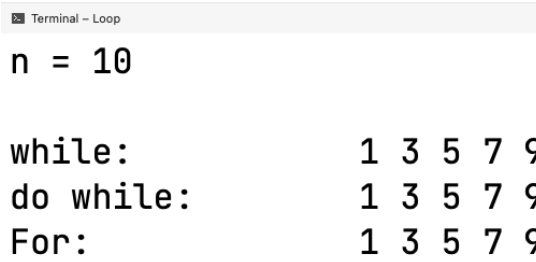
        Console.Write("\nwhile: \t\t");
        int i = 1;
        while (i ≤ n)
        {
            Console.Write(" " + i);
            i += 2;
        }

        Console.Write("\ndo while: \t");
        i = 1;
        do
        {
            Console.Write(" " + i);
            i += 2;
        } while (i ≤ n);

        Console.Write("\nFor: \t\t");
        for (i = 1; i ≤ n; i += 2)
        {
            Console.Write(" " + i);
        }
    }
}
```

---

Результат работы программы показан на рис. 2.1.



```
Terminal - Loop
n = 10

while:      1 3 5 7 9
do while:   1 3 5 7 9
For:        1 3 5 7 9
```

Рис. 2.1: Результат работы программы

После этого я реализовал следующую программу, использующую цикл с постусловием. Эта программа принимает на вход левую и правую границу интервала, после чего выводит значение синуса для каждого  $x$  с шагом 0.01. Исходный код программы представлен в следующем листинге:

---

```
namespace Loop;
class Program
{
    static void Main(string[] args)
    {
        double x, x1, x2, y;

        Console.WriteLine("Enter x1:");
        x1 = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter x2:");
        x2 = double.Parse(Console.ReadLine());

        Console.WriteLine("x\tsin(x)");
        x = x1;
        do
        {
            y = Math.Sin(x);
            Console.WriteLine("{0:f3}\t{1:f3}", x, y);
            x = x + 0.01;
        }
        while (x ≤ x2);
    }
}
```

---

На рис. 2.2 представлен результат работы программы.



```
Terminal - Loop

Enter x1:
5
Enter x2:
5.1

x          sin(x)
5.000      -0.959
5.010      -0.956
5.020      -0.953
5.030      -0.950
5.040      -0.947
5.050      -0.944
5.060      -0.940
5.070      -0.937
5.080      -0.933
5.090      -0.930
5.100      -0.926
```

Рис. 2.2: Результат работы программы

Затем я выполнил следующее задание и реализовал программу с циклом с пред-  
условием. В этой программе используется алгоритм Евклида для поиска НОД. Ее  
исходный код представлен в следующем листинге:

---

```
namespace Loop;
class Program
{
    static void Main(string[] args)
    {
        int a, b, temp;

        Console.WriteLine("Enter a:");
        a = int.Parse(Console.ReadLine());
        Console.WriteLine("Enter b:");
        b = int.Parse(Console.ReadLine());

        temp = a;
        while (temp != b)
        {
            a = temp;
            if (a < b)
            {
```

```

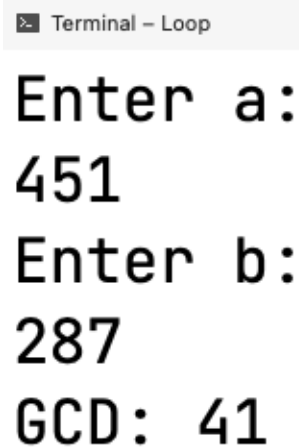
        temp = a;
        a = b;
        b = temp;
    }
    temp = a - b;
    a = b;
}

Console.WriteLine("GCD: {0}", a);
}
}

```

---

На рис. 2.3 представлен результат работы программы.



```

Terminal - Loop

Enter a:
451
Enter b:
287
GCD: 41

```

Рис. 2.3: Результат работы программы

После этого задачу вывода значений функции с помощью цикла с предусловием. Единственное изменение, которое необходимо внести, чтобы программы остались эквивалентными – проверка на равенство текущего значения  $x$  и значения  $x1$ . Если этого не сделать, то в случае, когда  $x > x2$ , цикл не будет выполнен ни разу. Исходный код решения представлен в следующем листинге:

---

```

namespace Loop;
class Program
{
    static void Main(string[] args)
    {
        double x, x1, x2, y;

        Console.WriteLine("Enter x1:");
        x1 = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter x2:");
        x2 = double.Parse(Console.ReadLine());

        Console.WriteLine("x\tsin(x)");
    }
}

```

```

x = x1;

while (x ≤ x2 || x == x1)
{
    y = Math.Sin(x);
    Console.WriteLine("{0:f3}\t{1:f3}", x, y);
    x = x + 0.01;
}
}

```

---

Затем я реализовал алгоритм Евклида с помощью цикла с постусловием. Кроме изменения самого цикла, пришлось добавить проверку на неравенство аргументов а и b. Это необходимо, поскольку иначе, если на вход будут переданы равные значения а и b, программа попадет в бесконечный цикл. Исходный код программы представлен в следующем листинге:

---

```

namespace Loop;
class Program
{
    static void Main(string[] args)
    {
        int a, b, temp;

        Console.WriteLine("Enter a:");
        a = int.Parse(Console.ReadLine());
        Console.WriteLine("Enter b:");
        b = int.Parse(Console.ReadLine());

        temp = a;
        do
        {
            a = temp;
            if (a < b)
            {
                temp = a;
                a = b;
                b = temp;
            }
            if (a ≠ b)
            {
                temp = a - b;
                a = b;
            }
        }
        while (temp ≠ b);

        Console.WriteLine("GCD: {0}", a);
    }
}

```

```
}  
}
```

---

## Задание 2.2. Расчет суммы, используя операторы перехода

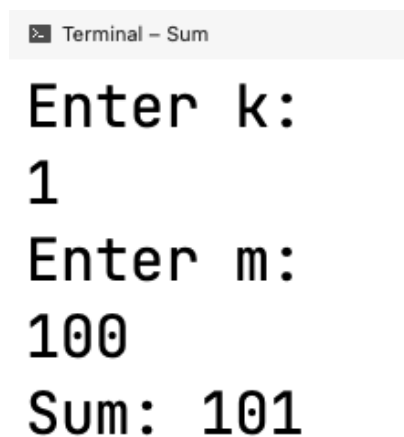
Выполнив все шаги из задания, я получил следующий исходный код:

---

```
namespace Sum;  
class Program  
{  
    static void Main(string[] args)  
    {  
        int k, m;  
  
        Console.WriteLine("Enter k:");  
        k = int.Parse(Console.ReadLine());  
        Console.WriteLine("Enter m:");  
        m = int.Parse(Console.ReadLine());  
  
        int s = 0;  
        for (int i = 1; i ≤ 100; i++)  
        {  
            if (i > k && i < m) continue;  
            s += i;  
        }  
        Console.WriteLine("Sum: {0}", s);  
    }  
}
```

---

На рис. 2.4 показан пример работы программы.



```
Terminal - Sum  
Enter k:  
1  
Enter m:  
100  
Sum: 101
```

Рис. 2.4: Пример работы программы Sum

## Задание 2.3. Стрельба по мишени

Моя программа разработана для второго варианта. В своей программе я сначала генерирую случайные значения координат мишени. После этого запускаю цикл из трех выстрелов. При каждом выстреле я считываю значения координат от пользователя. Если пользователю не повезло и ему помешали, его значения координат заменяются на случайные. После этого я рассчитываю расстояние до центра мишени. В зависимости от того, в какое место попал пользователь, начисляется разное количество очков. После каждого выстрела пользователю сообщается текущее количество очков. Исходный код программы представлен в следующем листинге:

---

```
namespace TargetShooting;
class Program
{
    static void Main(string[] args)
    {
        const int shotsCount = 3;
        Console.WriteLine("Get ready to make {0} shots", shotsCount);

        Random random = new Random();

        int targetX = random.Next(0, 4);
        int targetY = random.Next(0, 4);

        int score = 0;

        for (int i = 0; i < shotsCount; ++i)
        {
            Console.WriteLine("Enter x:");
            int x = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter y:");
            int y = int.Parse(Console.ReadLine());

            if (random.Next(0, 10) == 5)
            {
                Console.WriteLine("Oops! Someone interrupted you!");
                x = random.Next(0, 4);
                y = random.Next(0, 4);
            }

            double distance = Math.Sqrt(
                Math.Pow(targetX - x, 2) + Math.Pow(targetY - y, 2));

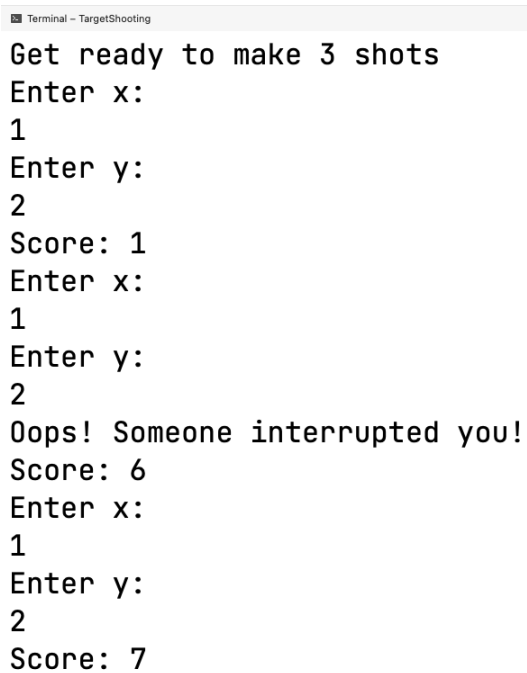
            if (distance <= 1)
            {
                score += 10;
            }
            else if (distance <= 2)
            {
                score += 5;
            }
        }
    }
}
```

```

    }
    else if (distance ≤ 3)
    {
        score += 1;
    }
    else
    {
        score += 0;
    }
    Console.WriteLine("Score: {0}", score);
}
}
}

```

На рис. 2.5 показан пример работы программы.



```

Terminal - TargetShooting
Get ready to make 3 shots
Enter x:
1
Enter y:
2
Score: 1
Enter x:
1
Enter y:
2
Oops! Someone interrupted you!
Score: 6
Enter x:
1
Enter y:
2
Score: 7

```

Рис. 2.5: Пример работы программы