

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

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

По дисциплине Объектно-ориентированное программирование

Тема работы Использование выражений

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
1.1.1 Задание 1	4
1.1.2 Задание 2	5
1.1.3 Задание 3	7
1.2 Упражнение 2	8
1.2.1 Задание 1	8
1.2.2 Задание 2	13
1.2.3 Задание 3	14
ЗАКЛЮЧЕНИЕ	17

ВВЕДЕНИЕ

Целью данной лабораторной работы является изучение и приобретение навыков использования управляющих конструкций для организации вычислений.

1 Ход работы

1.1 Упражнение 1

1.1.1 Задание 1

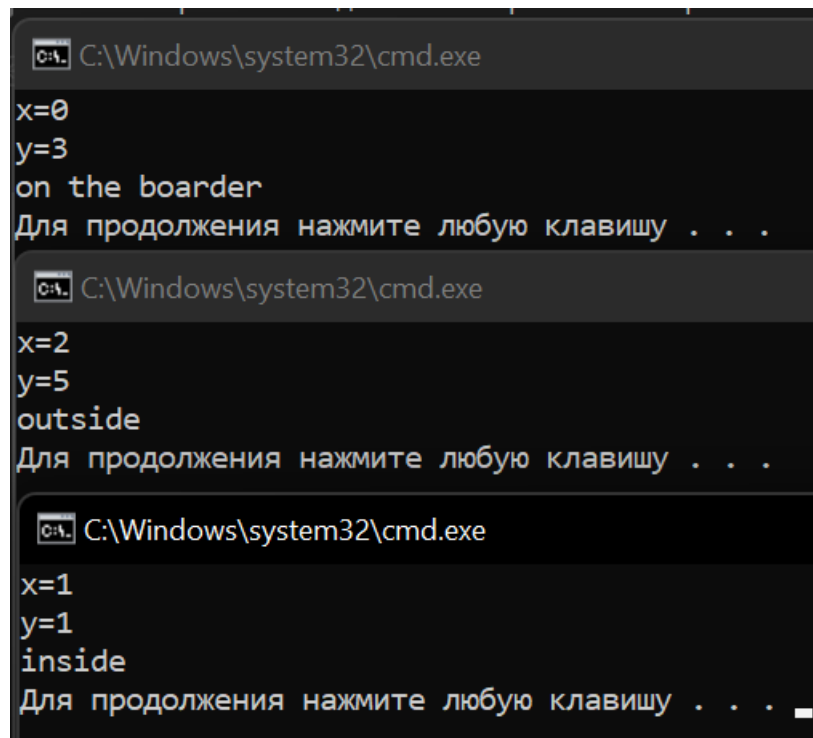
В первом задании нужно составить программу, которая выдает одно из сообщений «Да», «Нет», «На границе» в зависимости от того, лежит ли точка внутри заштрихованной области, вне заштрихованной области, или на границе. Реализовать это поможет условный оператор if/else if/ else (рис. 1.1).

```
internal 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("inside");
        }
        else if (x * x + y * y > 9 || y < 0)
        {
            Console.WriteLine("outside");
        }
        else
        {
            Console.WriteLine("on the boarder");
        }
    }
}
```

Рисунок 1.1 — Код задания 1

Примеры работы программы изображены на рис. 1.2.



The image displays three separate screenshots of a Windows command prompt window, each showing the output of a program. The title bar of each window reads 'C:\Windows\system32\cmd.exe'. The first screenshot shows the output: 'x=0', 'y=3', 'on the boarder', and 'Для продолжения нажмите любую клавишу . . .'. The second screenshot shows: 'x=2', 'y=5', 'outside', and 'Для продолжения нажмите любую клавишу . . .'. The third screenshot shows: 'x=1', 'y=1', 'inside', and 'Для продолжения нажмите любую клавишу . . .'. Each prompt ends with a small white cursor icon.

```
C:\Windows\system32\cmd.exe
x=0
y=3
on the boarder
Для продолжения нажмите любую клавишу . . .

C:\Windows\system32\cmd.exe
x=2
y=5
outside
Для продолжения нажмите любую клавишу . . .

C:\Windows\system32\cmd.exe
x=1
y=1
inside
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1.2 — Примеры работы программы из задания 1

1.1.2 Задание 2

Во втором задании нужно создать программу, моделирующую работу калькулятора. Пользователь должен ввести первый операнд, затем требуемую операцию и второй операнд. В зависимости от знака операции производится расчет операции.

Для этого был создан новый проект `Calc_switch`, в котором была реализована логика работы переключателя с помощью оператора `switch` (рис. 1.3)

```

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;

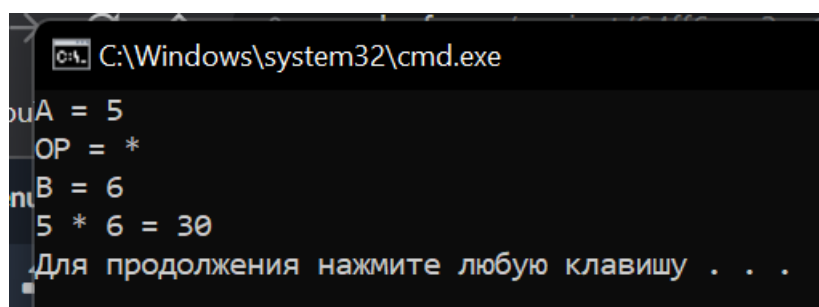
    switch (op)
    {
        case '+': res = a + b; break;
        case '-': res = a - b; break;
        case '*': res = a * b; break;
        case '/':
        case ':':
            res = a / b; break;
        default:
            ok = false; break;
    }

    if (ok)
    {
        Console.WriteLine("{0} {1} {2} = {3}", a, op, b, res);
    }
    else
    {
        Console.WriteLine("operation not defined");
    }
}

```

Рисунок 1.3 — Код задания 2

Пример работы программы показан на рис. 1.4.



```

C:\Windows\system32\cmd.exe
A = 5
OP = *
B = 6
5 * 6 = 30
Для продолжения нажмите любую клавишу . . .

```

Рисунок 1.4 — Пример работы программы из задания 2

Стоит отметить, что при вводе ненулевого первого значения и нуля в качестве второго значения, переменная `res` принимает значение `Infinity` (рис. 1.5).

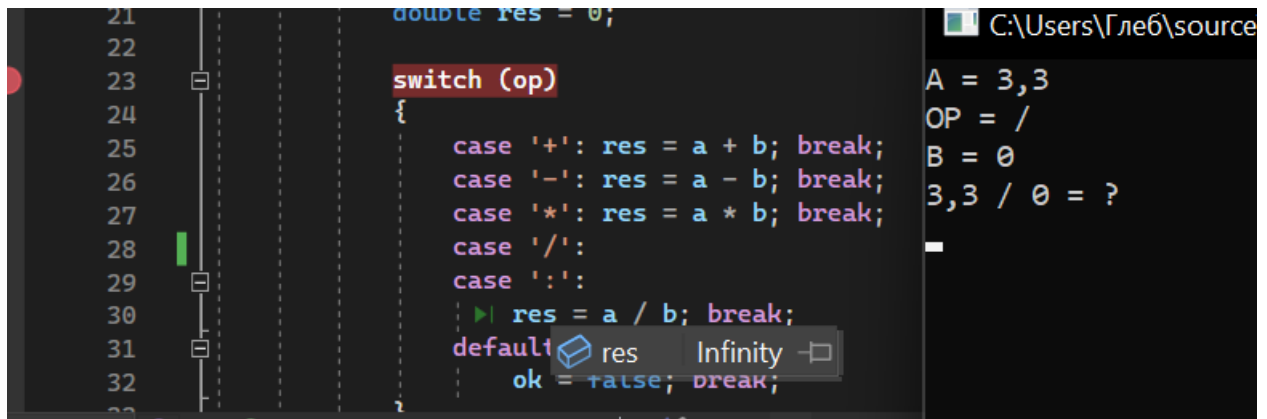


Рисунок 1.5 — Деление на ноль

Аналогично если в качестве обоих операндов будут введены нули, переменная `res` примет значение NaN (Not a Number). Это связано с тем, что мы преобразуем входные данные в тип `double`, в котором поддерживается реализация бесконечности и NaN.

1.1.3 Задание 3

В данном задании с помощью условных операторов требовалось определить, является ли год високосным. Номер года должен вводиться пользователем.

Программа была реализована в новом проекте с названием `Year_checker` (рис. 1.6).

```

namespace Year_checker
{
    Ссылка: 0
    internal class Program
    {
        Ссылка: 0
        static void Main(string[] args)
        {
            Console.Write("Введите номер года: ");
            int yearNumber = int.Parse(Console.ReadLine());

            if (yearNumber % 4 == 0 && yearNumber % 100 != 0 ||
                yearNumber % 400 == 0)
                Console.WriteLine("Год високосный");
            else
                Console.WriteLine("Год не високосный");
        }
    }
}

```

Рисунок 1.6 — Код задания 3

Пример работы программы показан на рис. 1.7.

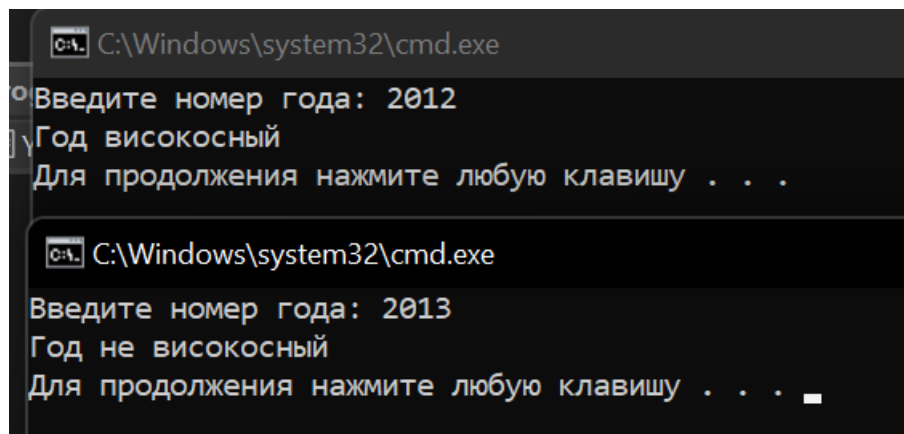


Рисунок 1.7 — Пример работы программы из задания 3

1.2 Упражнение 2

1.2.1 Задание 1

В первом задании нужно было написать программу, которая выводит на экран последовательность целых нечетных чисел в строчку через пробел с помощью трех операторов цикла: while, do while и for (рис. 1.8).


```

Ссылка: 0
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);
    }
}

```

Рисунок 1.8 — Код программы из задания 1

Пример работы программы на рис. 1.9.

```

C:\Windows\system32\cmd.exe
n = 10
while:      1 3 5 7 9
do while:   1 3 5 7 9
For:        1 3 5 7 9
Для продолжения нажмите любую клавишу . . .

```

Рисунок 1.9 — Пример работы программы из задания 1

После этого необходимо было реализовать программу, печатающую аргумент функции и ее значение (в данном случае, функция — \sin), с помощью цикла с постусловием. Пользователь вводит два числа — границы интервала — после этого выводится табличка со значениями функции на отрезке с шагом 0,01 (рис. 1.10).

```

internal class Program
{
    Ссылка: 0
    static void Main(string[] args)
    {
        double x;
        double y;

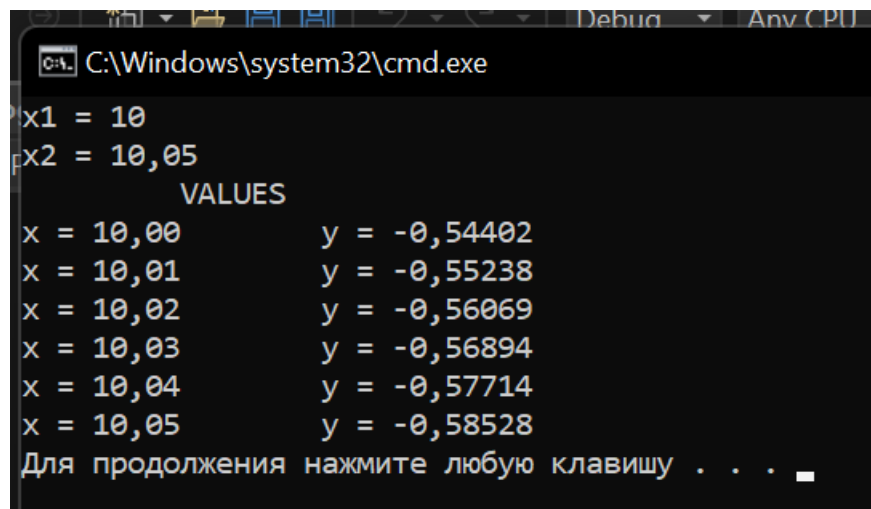
        Console.Write("x1 = ");
        double x1 = double.Parse(Console.ReadLine());
        Console.Write("x2 = ");
        double x2 = double.Parse(Console.ReadLine());

        Console.WriteLine("\t VALUES \t");
        x = x1;
        do
        {
            y = Math.Sin(x);
            Console.WriteLine("x = {0:F2} \t y = {1:F5} \t", x, y);
            x = x + 0.01;
        }
        while (x <= x2);
    }
}

```

Рисунок 1.10 — Цикл с постусловием из задания 1

Пример работы программы на рис. 1.11



```

C:\Windows\system32\cmd.exe
x1 = 10
x2 = 10,05
VALUES
x = 10,00      y = -0,54402
x = 10,01      y = -0,55238
x = 10,02      y = -0,56069
x = 10,03      y = -0,56894
x = 10,04      y = -0,57714
x = 10,05      y = -0,58528
Для продолжения нажмите любую клавишу . . .

```

Рисунок 1.11 — Результат программы с циклом do/while

Следующим этапом была реализация алгоритма Евклида с помощью цикла с предусловием (рис. 1.12)

```

Ссылка: 0
static void Main(string[] args)
{
    int a;
    int b;
    int temp;

    Console.Write("a = ");
    a = int.Parse(Console.ReadLine());
    Console.Write("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(a);
}

```

Рисунок 1.12 — Цикл с предусловием из задания 1

Пример работы программы на рис. 1.13

```

C:\Windows\system32\cmd.exe
a = 26
b = 13
13
Для продолжения нажмите любую клавишу . . .

```

Рисунок 1.13 — Результат программы с циклом while

После этого необходимо было реализовать задачу вывода функции с помощью цикла с предусловием (рис. 1.14) и алгоритм Евклида с помощью цикла с постусловием (рис. 1.15).

```

static void Main(string[] args)
{
    double x;
    double y;

    Console.Write("x1 = ");
    double x1 = double.Parse(Console.ReadLine());
    Console.Write("x2 = ");
    double x2 = double.Parse(Console.ReadLine());

    Console.WriteLine("\t VALUES \t");
    x = x1;
    while (x <= x2)
    {
        y = Math.Sin(x);
        Console.WriteLine("x = {0:F2} \t y = {1:F5} \t", x, y);
        x = x + 0.01;
    }
}

```

Рисунок 1.14 — Вывод функции, цикл с предусловием

```

static void Main(string[] args)
{
    int a;
    int b;
    int temp;

    Console.Write("a = ");
    a = int.Parse(Console.ReadLine());
    Console.Write("b = ");
    b = int.Parse(Console.ReadLine());
    temp = a;

    do
    {
        a = temp;
        if (a < b)
        {
            temp = a;
            a = b;
            b = temp;
        }
        temp = a - b;
        a = b;
    }
    while (temp != 0);
    Console.WriteLine(a);
}

```

Рисунок 1.15 — Алгоритм Евклида, цикл с постусловием

1.2.2 Задание 2

Во втором задании необходимо составить программу, реализующая сумму $s = \sum_{1 < i < 100} i$ для i , находящихся от 1 до k и от m до 100.

Для суммирования чисел в диапазоне были использованы цикл `for`, условный оператор `if` и оператор перехода `continue` (рис. 1.16).

```
internal class Program
{
    static void Main(string[] args)
    {
        int k;
        int m;
        Console.Write("k = ");
        k = int.Parse(Console.ReadLine());
        Console.Write("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(s);
    }
}
```

Рисунок 1.16 — Код программы из задания 2

Пример работы программы на рис. 1.17.

```
C:\Windows\system32\cmd.exe
k = 5
m = 98
312
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1.17 — Пример работы программы из задания 2

1.2.3 Задание 3

В последнем задании нужно было разработать программу, имитирующую стрельбу по мишени. Была реализована следующая функциональность: ввод данных о выстреле (в виде пары чисел — координат x и y) от пользователя, повтор ввода был организован в цикле.

Также дополнительно было реализовано: центр мишени задается случайным значением, чтобы стрелок не знал её местонахождения (стрельба вслепую), присутствует случайная помеха при выстреле. Релизовать этот функционал удалось с помощью экземпляра класса `Random`, с помощью методов `Next` (для генерации координат мишени) и `NextDouble` (для генерации помехи) (рис. 1.18).

```

static void Main(string[] args)
{
    int n;
    double x;
    double y;
    int centerX;
    int centerY;
    double noise;
    int s = 0;

    Random random = new Random();

    Console.Write("Введите количество выстрелов: ");
    n = int.Parse(Console.ReadLine());
    centerX = random.Next(10) - 5;
    centerY = random.Next(10) - 5;
    // Console.WriteLine("{0}, {1}", centerX, centerY);
    noise = random.NextDouble() - 0.5; // помеха [-0.5, 0.5]
    Console.WriteLine("Помеха: {0}", noise);

    for (int i = 0; i < n; i++)
    {
        Console.Write("x = ");
        x = int.Parse(Console.ReadLine());
        Console.Write("y = ");
        y = int.Parse(Console.ReadLine());

        double ratioX = x - centerX;
        double ratioY = y - centerY;

        if (ratioX * ratioX + ratioY * ratioY + noise <= 1)
        {
            s += 10;
            Console.WriteLine("10 баллов!");
        }

        if (ratioX * ratioX + ratioY * ratioY + noise > 1 &&
            ratioX * ratioX + ratioY * ratioY + noise <= 4)
        {
            s += 5;
            Console.WriteLine("5 баллов!");
        }

        if (ratioX * ratioX + ratioY * ratioY + noise > 4 &&
            ratioX * ratioX + ratioY * ratioY + noise <= 9)
        {
            s += 1;
            Console.WriteLine("1 балл!");
        }

        if (ratioX * ratioX + ratioY * ratioY + noise > 9)
        {
            Console.WriteLine("Вы не попали");
        }
    }

    Console.WriteLine("{0}, {1} - центр", centerX, centerY);
    Console.WriteLine("Вы набрали {0} баллов", s);
}

```

Рисунок 1.18 — Код программы из задания 3

Пример работы программы изображен на рис. 1.19.

```
C:\Windows\system32\cmd.exe
Введите количество выстрелов: 3
Помеха: 0,385558633546139
x = 2
y = 0
Вы не попали
x = -3
y = -2
1 балл!
x = -3
y = -4
1 балл!
(-1, -4) - центр
Вы набрали 2 баллов
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1.19 — Пример работы программы

ЗАКЛЮЧЕНИЕ

В ходе этой лабораторной работы я ознакомился с условными операторами if/else if/else, операторами цикла while/do while/for и оператором перехода continue. Все задания были выполнены.