

**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**ITMO University**

**ЛАБОРАТОРНАЯ РАБОТА №5**

**По дисциплине** Объектно-ориентированное программирование

**Тема работы** Создание и использование массивов

**Обучающийся** Крестьянова Елизавета Федоровна

**Факультет** факультет инфокоммуникационных технологий

**Группа** К3223

**Направление подготовки** 11.03.02 Инфокоммуникационные технологии и  
системы связи

**Образовательная программа** Программирование в  
инфокоммуникационных системах

<b>Обучающийся</b>	_____	_____	<u>Крестьянова Е.Ф.</u>
	(дата)	(подпись)	(Ф.И.О.)

<b>Руководитель</b>	_____	_____	<u>Иванов С.Е.</u>
	(дата)	(подпись)	(Ф.И.О.)

## СОДЕРЖАНИЕ

Стр.

ВВЕДЕНИЕ .....	3
1 Упражнение №1. Работа с массивом размерного типа данных	4
2 Упражнение №2. Перемножение матриц .....	7
3 Упражнение №3. Обработка данных массива.....	15
ЗАКЛЮЧЕНИЕ .....	20

## **ВВЕДЕНИЕ**

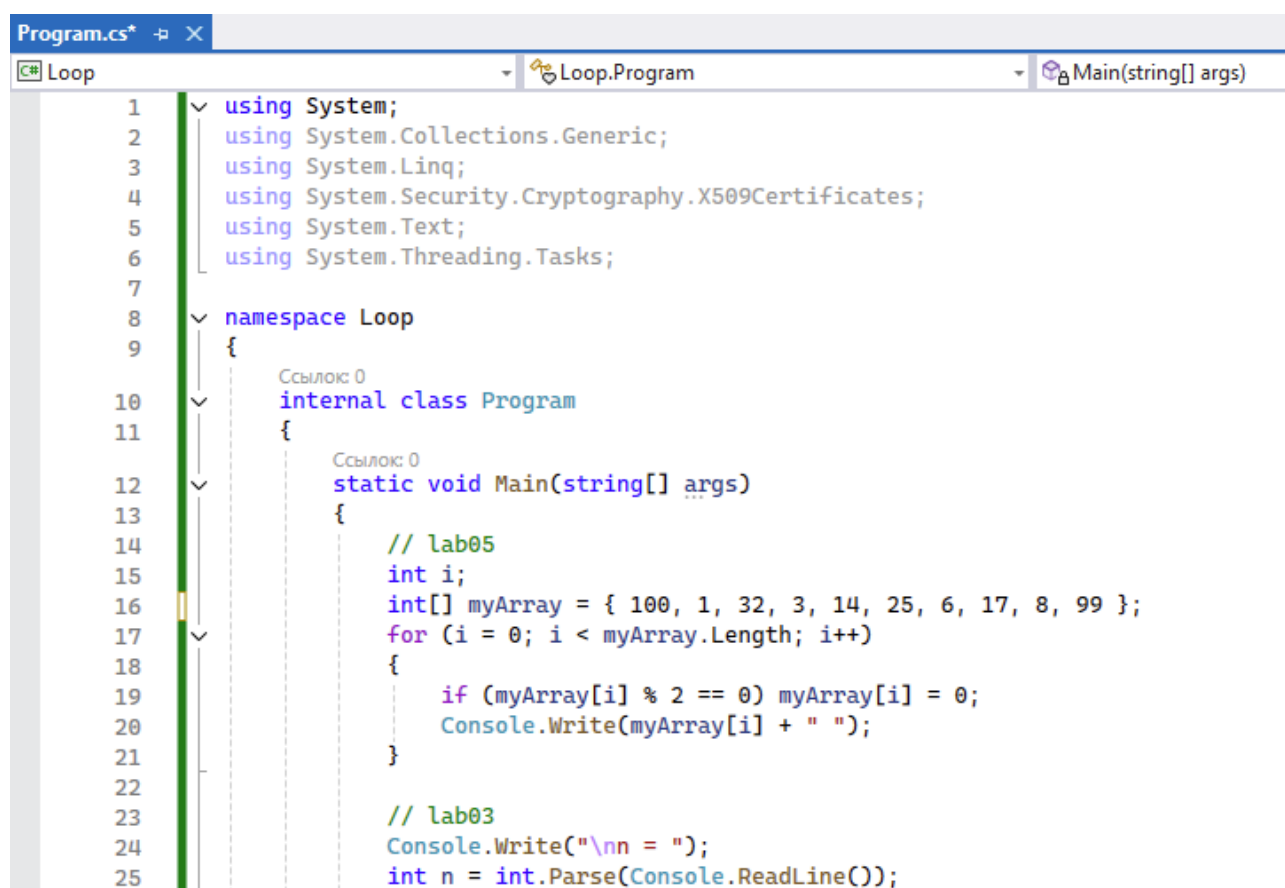
В данном отчёте представлено выполнение лабораторной работы по дисциплине «Объектно-ориентированное программирование».

Цель данной работы - изучение и приобретение навыков работы с массивами.

## 1 УПРАЖНЕНИЕ №1. РАБОТА С МАССИВОМ РАЗМЕРНОГО ТИПА ДАННЫХ

В данном упражнении было необходимо реализовать массив для хранения данных для проекта Loop из лабораторной работы №3.

Был создан массив myArray с заранее заданными элементами. Через цикл for выводятся все элементы массива, но сначала проверяется их чётность: если они таковы, они обнуляются. Код нового фрагмента проекта можно увидеть на рисунке 1.1



```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Security.Cryptography.X509Certificates;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace Loop
9  {
10     internal class Program
11     {
12         static void Main(string[] args)
13         {
14             // lab05
15             int i;
16             int[] myArray = { 100, 1, 32, 3, 14, 25, 6, 17, 8, 99 };
17             for (i = 0; i < myArray.Length; i++)
18             {
19                 if (myArray[i] % 2 == 0) myArray[i] = 0;
20                 Console.Write(myArray[i] + " ");
21             }
22
23             // lab03
24             Console.Write("\nn = ");
25             int n = int.Parse(Console.ReadLine());
```

Рисунок 1.1 — Упр №1: Метод Main, заранее заданный массив

Результат работы программы можно увидеть на рисунке 1.2.



C:\Windows\system32\cmd.exe

```
0 1 0 3 0 25 0 17 0 99
n =
```

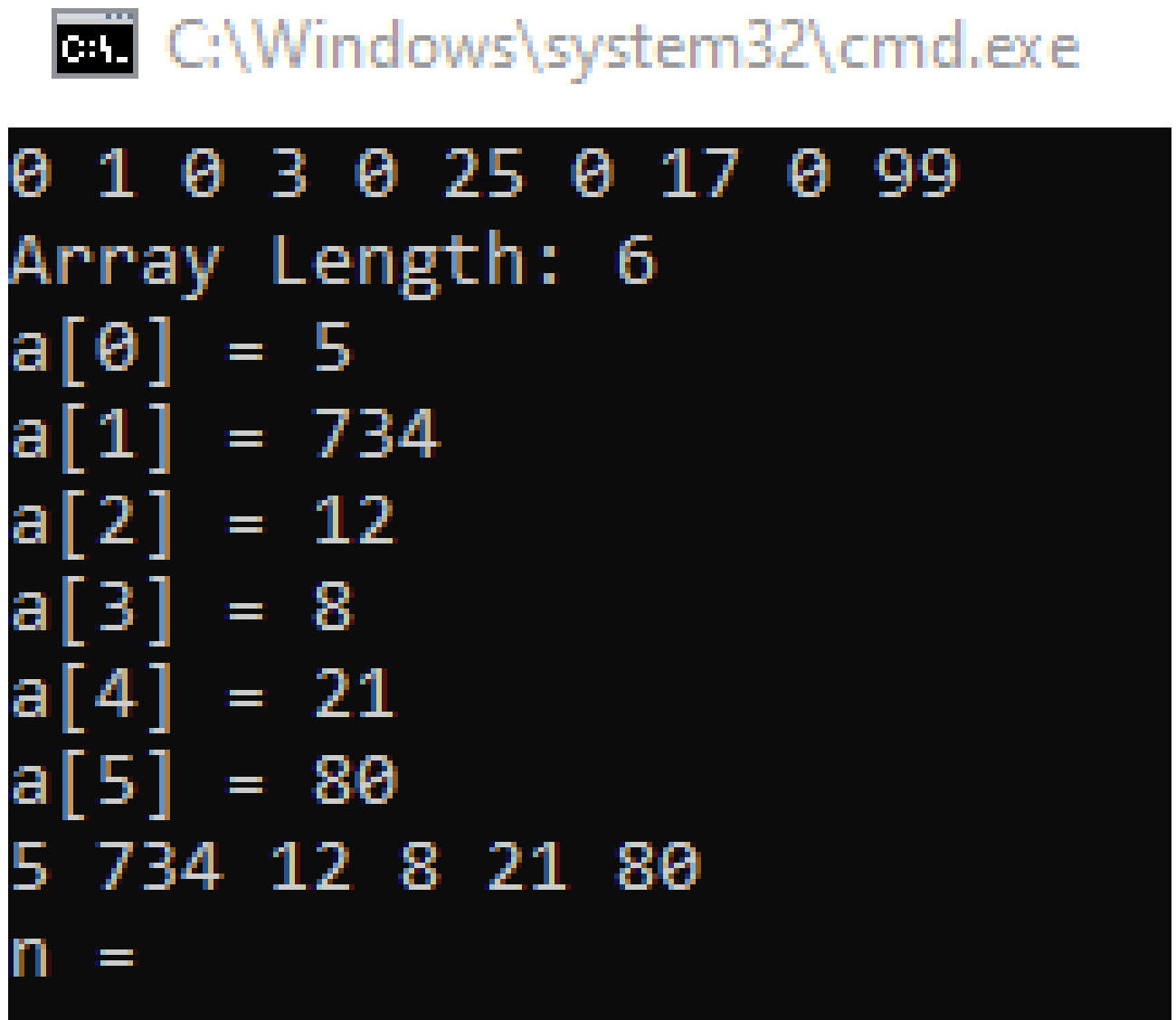
Рисунок 1.2 — Упр №1: Вывод программы с заранее заданным массивом

Был создан функционал ввода пользователем массива с кастомной длиной. После создания массива он выводится циклом foreach. Эта реализация представлена на рисунке 1.3.

```
Program.cs  x
C# Loop Loop.Program Main(string[] args)
7
8 namespace Loop
9 {
10     Ссылка 0
11     internal class Program
12     {
13         Ссылка 0
14         static void Main(string[] args)
15         {
16             // lab05
17             int i;
18             int[] myArray = { 100, 1, 32, 3, 14, 25, 6, 17, 8, 99 };
19             for (i = 0; i < myArray.Length; i++)
20             {
21                 if (myArray[i] % 2 == 0) myArray[i] = 0;
22                 Console.Write(myArray[i] + " ");
23             }
24             // user input
25             Console.WriteLine("\nArray Length: ");
26             int[] MyArray;
27             int n = int.Parse(Console.ReadLine());
28             MyArray = new int[n];
29             for (i = 0; i < MyArray.Length; ++i)
30             {
31                 Console.Write("a[{0}] = ", i);
32                 MyArray[i] = int.Parse(Console.ReadLine());
33             }
34             foreach (int elem in MyArray) Console.Write("{0} ", elem);
35             // lab03
36             Console.WriteLine("\nn = ");
37             n = int.Parse(Console.ReadLine());
```

Рисунок 1.3 — Упр №1: Метод Main, пользовательский ввод

Вывод конечной версии программы виден на рисунке 1.4.



```
C:\Windows\system32\cmd.exe
0 1 0 3 0 25 0 17 0 99
Array Length: 6
a[0] = 5
a[1] = 734
a[2] = 12
a[3] = 8
a[4] = 21
a[5] = 80
5 734 12 8 21 80
n =
```

Рисунок 1.4 — Упр №1: Вывод программы с пользовательским вводом

## 2 УПРАЖНЕНИЕ №2. ПЕРЕМНОЖЕНИЕ МАТРИЦ

В данном упражнении было необходимо создать программу, вычисляющую произведение двух матриц 2x2 по формуле из рисунка 2.1.

$$\begin{array}{cc} A1 & A2 \\ A3 & A4 \end{array} \times \begin{array}{cc} B1 & B2 \\ B3 & B4 \end{array} = \begin{array}{cc} A1.B1 + A2.B3 & A1.B2 + A2.B4 \\ A3.B1 + A4.B3 & A3.B2 + A4.B4 \end{array}$$

Рисунок 2.1 — Упр №2: Произведение матриц 2x2

Был создан новый проект MatrixMultiply. В нём была создана тестовая версия программы с заранее заданными матрицами a и b и простым выводом результирующей третьей матрицы в консоль. Код этой версии можно увидеть на рисунке 2.2.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace MatrixMultiply
8  {
9      internal class MatrixMultiply
10     {
11         static void Main(string[] args)
12         {
13             int[,] a = new int[2, 2];
14             a[0, 0] = 1; a[0, 1] = 2;
15             a[1, 0] = 3; a[1, 1] = 4;
16
17             int[,] b = new int[2, 2];
18             b[0, 0] = 5; b[0, 1] = 6;
19             b[1, 0] = 7; b[1, 1] = 8;
20
21             int[,] result = new int[2, 2];
22             result[0, 0] = a[0, 0] * b[0, 0] + a[0, 1] * b[1, 0];
23             result[0, 1] = a[0, 0] * b[0, 1] + a[0, 1] * b[1, 1];
24             result[1, 0] = a[1, 0] * b[0, 0] + a[1, 1] * b[1, 0];
25             result[1, 1] = a[1, 0] * b[0, 1] + a[1, 1] * b[1, 1];
26
27             Console.WriteLine(result[0,0]);
28             Console.WriteLine(result[0,1]);
29             Console.WriteLine(result[1,0]);
30             Console.WriteLine(result[1,1]);
31         }
32     }
33 }
34
```

Рисунок 2.2 — Упр №2: Код тестовой версии

Вывод этой версии показан на рисунке 2.3.



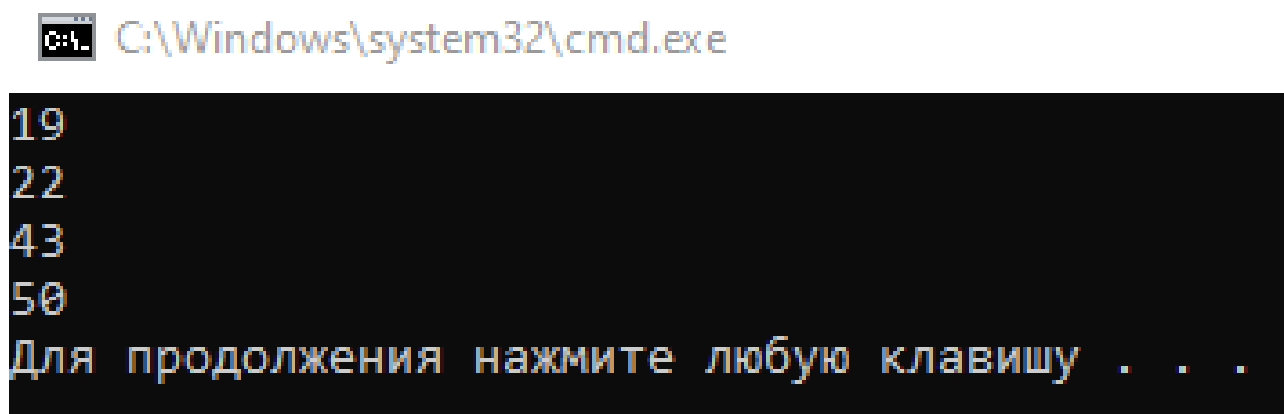


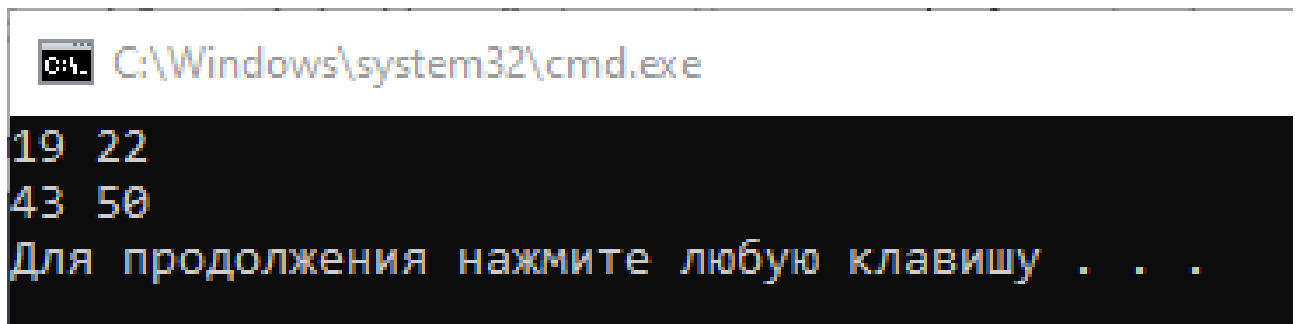
Рисунок 2.3 — Упр №2: Вывод тестовой версии

Затем из метода Main был выделен новый метод Output, в котором были объявлены 2 цикла, которые можно увидеть на рисунке 2.4. Они отвечают за вывод значений result в виде матрицы.

```
7 namespace MatrixMultiply
8 {
9     internal class MatrixMultiply
10    {
11        static void Main(string[] args)
12        {
13            int[,] a = new int[2, 2];
14            a[0, 0] = 1; a[0, 1] = 2;
15            a[1, 0] = 3; a[1, 1] = 4;
16
17            int[,] b = new int[2, 2];
18            b[0, 0] = 5; b[0, 1] = 6;
19            b[1, 0] = 7; b[1, 1] = 8;
20
21            int[,] result = new int[2, 2];
22            result[0, 0] = a[0, 0] * b[0, 0] + a[0, 1] * b[1, 0];
23            result[0, 1] = a[0, 0] * b[0, 1] + a[0, 1] * b[1, 1];
24            result[1, 0] = a[1, 0] * b[0, 0] + a[1, 1] * b[1, 0];
25            result[1, 1] = a[1, 0] * b[0, 1] + a[1, 1] * b[1, 1];
26
27            Output(result);
28        }
29
30        private static void Output(int[,] result)
31        {
32            for (int r = 0; r < result.GetLength(0); r++)
33            {
34                for (int c = 0; c < result.GetLength(1); c++)
35                {
36                    Console.Write("{0} ", result[r, c]);
37                }
38                Console.WriteLine();
39            }
40        }
41    }
42 }
43
```

Рисунок 2.4 — Упр №2: Метод вывода в матричной форме

Данный вывод можно увидеть на рисунке 2.5.

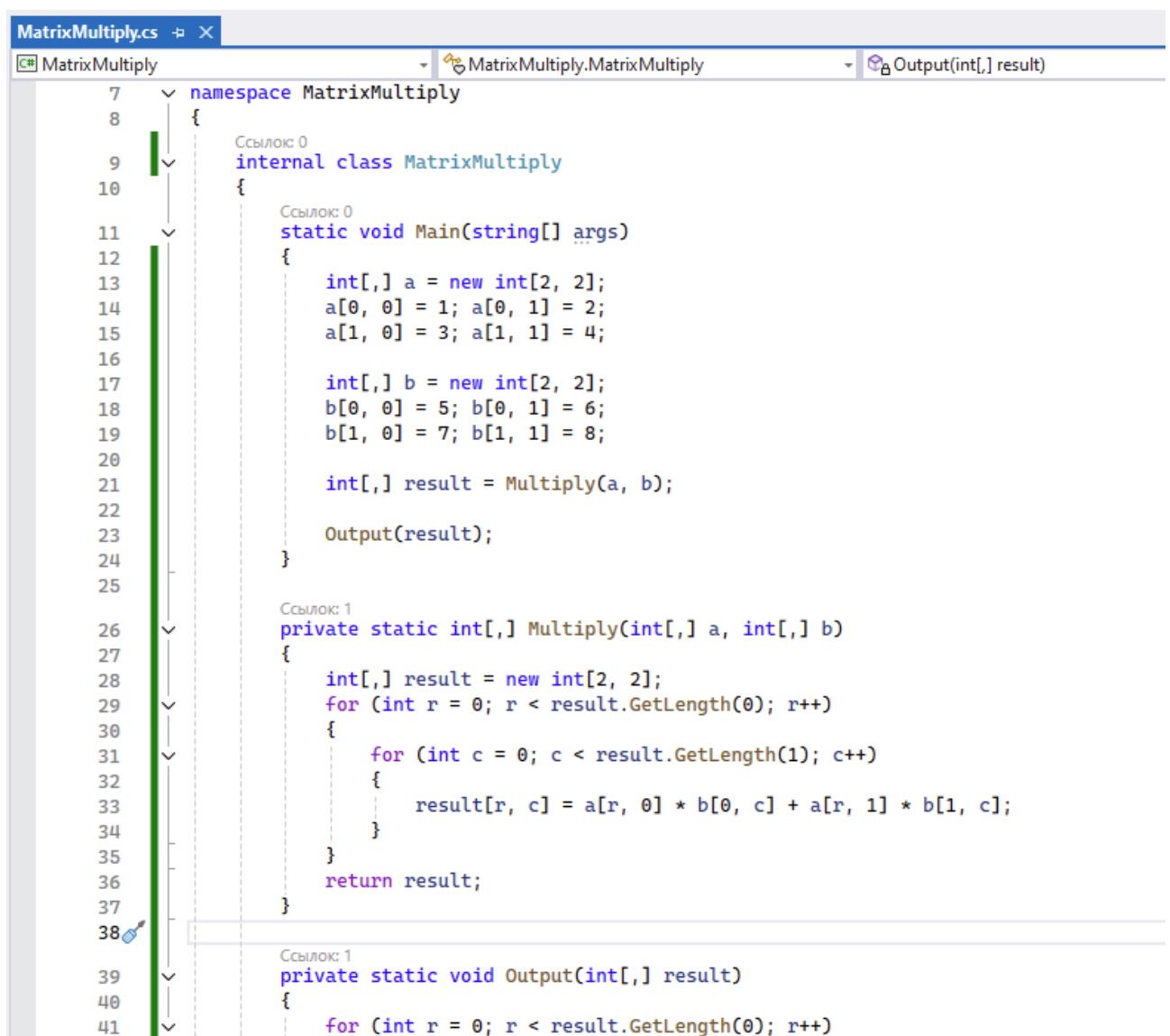


```
C:\Windows\system32\cmd.exe

19 22
43 50
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.5 — Упр №2: Вывод в матричной форме

Из метода Main был выделен новый метод Multiply, чей функционал был также расписан с помощью циклов for. Его можно увидеть на рисунке 2.6.



```
MatrixMultiply.cs
MatrixMultiply
namespace MatrixMultiply
{
    Ссылка: 0
    internal class MatrixMultiply
    {
        Ссылка: 0
        static void Main(string[] args)
        {
            int[,] a = new int[2, 2];
            a[0, 0] = 1; a[0, 1] = 2;
            a[1, 0] = 3; a[1, 1] = 4;

            int[,] b = new int[2, 2];
            b[0, 0] = 5; b[0, 1] = 6;
            b[1, 0] = 7; b[1, 1] = 8;

            int[,] result = Multiply(a, b);

            Output(result);
        }

        Ссылка: 1
        private static int[,] Multiply(int[,] a, int[,] b)
        {
            int[,] result = new int[2, 2];
            for (int r = 0; r < result.GetLength(0); r++)
            {
                for (int c = 0; c < result.GetLength(1); c++)
                {
                    result[r, c] = a[r, 0] * b[0, c] + a[r, 1] * b[1, c];
                }
            }
            return result;
        }

        Ссылка: 1
        private static void Output(int[,] result)
        {
            for (int r = 0; r < result.GetLength(0); r++)
```

Рисунок 2.6 — Упр №2: Метод умножения матриц

Затем был выделен новый метод Input, где пользователь вводит свои данные для новой матрицы. Он тоже был реализован через циклы for. Код метода представлен на рисунке 2.8.

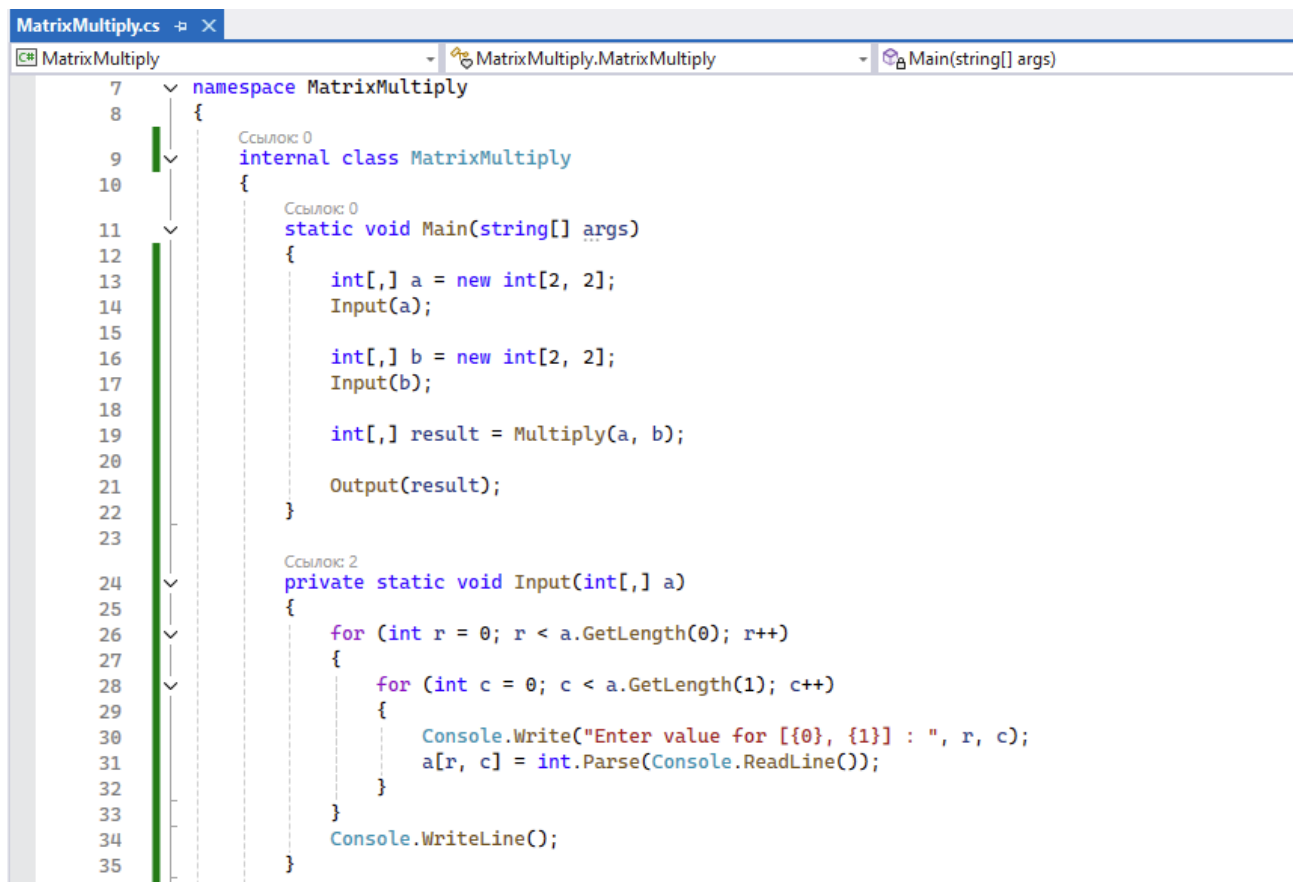


Рисунок 2.7 — Упр №3: Метод Input

Вывод финальной версии программы с прежними данными можно увидеть на рисунке 2.8.

C:\Windows\system32\cmd.exe

```
Enter value for [0, 0] : 1
Enter value for [0, 1] : 2
Enter value for [1, 0] : 3
Enter value for [1, 1] : 4
```

```
Enter value for [0, 0] : 5
Enter value for [0, 1] : 6
Enter value for [1, 0] : 7
Enter value for [1, 1] : 8
```

```
19 22
```

```
43 50
```

```
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.8 — Упр №2: Вывод программы с прежними данными

Вывод финальной версии программы с пользовательскими данными можно увидеть на рисунке 2.9.

C:\Windows\system32\cmd.exe

```
Enter value for [0, 0] : 2
Enter value for [0, 1] : 4
Enter value for [1, 0] : 6
Enter value for [1, 1] : 8
```

```
Enter value for [0, 0] : 1
Enter value for [0, 1] : 1
Enter value for [1, 0] : 1
Enter value for [1, 1] : 2
```

```
6 10
```

```
14 22
```

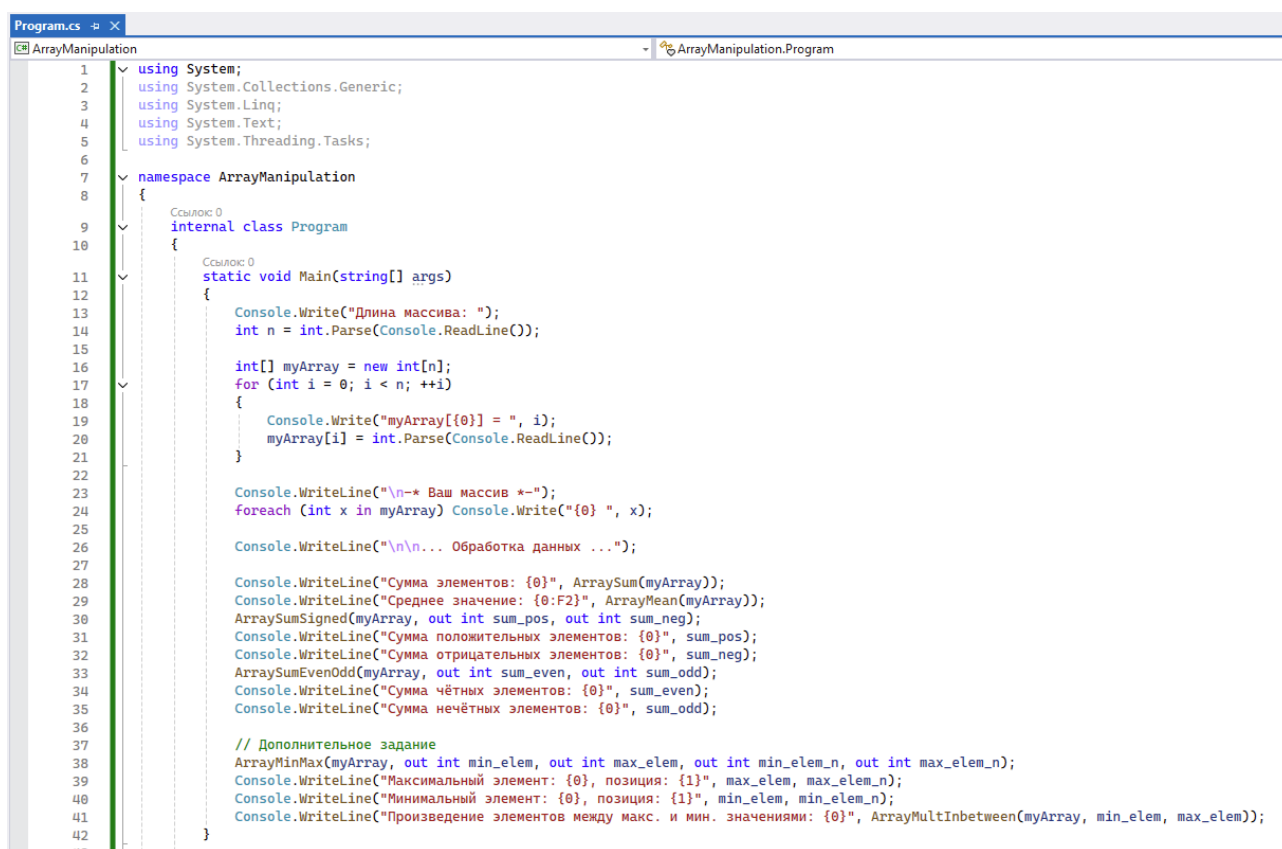
```
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.9 — Упр №3: Вывод программы с пользовательскими данными

### 3 УПРАЖНЕНИЕ №3. ОБРАБОТКА ДАННЫХ МАССИВА

В данном упражнении необходимо сделать несколько различных методов обработки данных массива.

Был создан новый проект ArrayManipulation. В методе Main был реализован принятие ввода пользователя, определяющего длину массива, а также определяющего сами элементы массива. Затем в консоль отображаются результаты написанных далее методов. Код метода Main можно увидеть на рисунке 3.1.



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace ArrayManipulation
8 {
9     internal class Program
10     {
11         static void Main(string[] args)
12         {
13             Console.WriteLine("Длина массива: ");
14             int n = int.Parse(Console.ReadLine());
15
16             int[] myArray = new int[n];
17             for (int i = 0; i < n; ++i)
18             {
19                 Console.WriteLine("myArray[{0}] = ", i);
20                 myArray[i] = int.Parse(Console.ReadLine());
21             }
22
23             Console.WriteLine("\n-* Ваш массив *-");
24             foreach (int x in myArray) Console.WriteLine("{0} ", x);
25
26             Console.WriteLine("\n\n... Обработка данных ...");
27
28             Console.WriteLine("Сумма элементов: {0}", ArraySum(myArray));
29             Console.WriteLine("Среднее значение: {0:F2}", ArrayMean(myArray));
30             ArraySumSigned(myArray, out int sum_pos, out int sum_neg);
31             Console.WriteLine("Сумма положительных элементов: {0}", sum_pos);
32             Console.WriteLine("Сумма отрицательных элементов: {0}", sum_neg);
33             ArraySumEvenOdd(myArray, out int sum_even, out int sum_odd);
34             Console.WriteLine("Сумма чётных элементов: {0}", sum_even);
35             Console.WriteLine("Сумма нечётных элементов: {0}", sum_odd);
36
37             // Дополнительное задание
38             ArrayMinMax(myArray, out int min_elem, out int max_elem, out int min_elem_n, out int max_elem_n);
39             Console.WriteLine("Максимальный элемент: {0}, позиция: {1}", max_elem, max_elem_n);
40             Console.WriteLine("Минимальный элемент: {0}, позиция: {1}", min_elem, min_elem_n);
41             Console.WriteLine("Произведение элементов между макс. и мин. значениями: {0}", ArrayMultInbetween(myArray, min_elem, max_elem));
42         }
43     }
44 }
```

Рисунок 3.1 — Упр №3: Метод Main

На рисунке 3.2 представлены методы нахождения суммы всех элементов, положительных и отрицательных, чётных и нечётных элементов массива; а также произведения всех элементов, находящихся между максимальным и минимальным элементов массива.

```
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85
```

```
Console.WriteLine("Произведение элементов между макс. и мин. значениями: {0}", ArrayManipulation.  
    }  
  
    Ссылка: 1  
    private static int ArrayMultInbetween(int[] myArray, int min_elem, int max_elem)  
    {  
        int res = 1;  
        foreach (int elem in myArray)  
        {  
            if (elem != min_elem && elem != max_elem) res *= elem;  
        }  
  
        return res;  
    }  
  
    Ссылка: 1  
    private static int ArraySum(int[] myArray)  
    {  
        int sum = 0;  
        foreach (int x in myArray) sum += x;  
        return sum;  
    }  
  
    Ссылка: 1  
    private static void ArraySumSigned(int[] myArray, out int sum_pos, out int sum_neg)  
    {  
        sum_pos = 0;  
        sum_neg = 0;  
  
        foreach (int x in myArray)  
        {  
            if (x > 0) sum_pos += x;  
            if (x < 0) sum_neg += x;  
        }  
    }  
  
    Ссылка: 1  
    private static void ArraySumEvenOdd(int[] myArray, out int sum_even, out int sum_odd)  
    {  
        sum_even = 0;  
        sum_odd = 0;  
  
        foreach (int x in myArray)  
        {  
            if (x % 2 == 0) sum_even += x;  
            if (x % 2 != 0) sum_odd += x;  
        }  
    }  
}
```

Рисунок 3.2 — Упр №3: Методы произведения и сумм элементов

На рисунке 3.3 показаны методы нахождения минимального и максимального значений массива и их индексов, среднего значения массива.



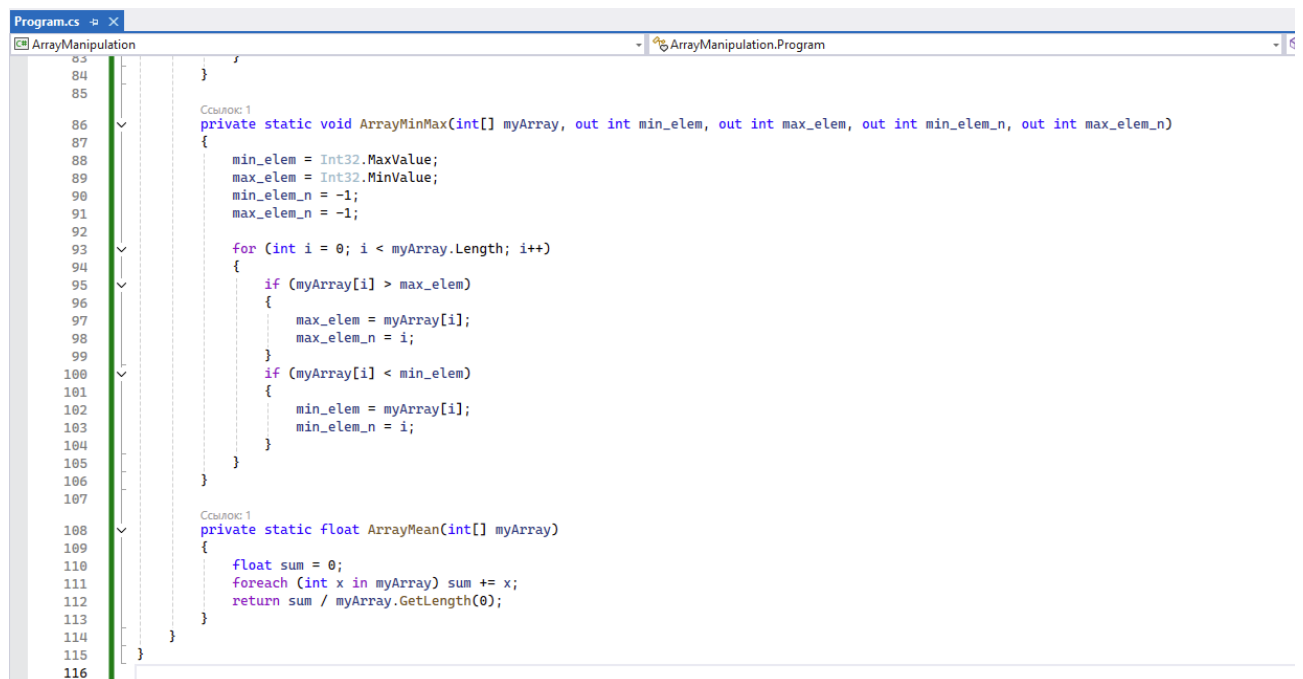


Рисунок 3.3 — Упр №3: Методы нахождения мин., макс. и среднего значений

На рисунках 3.4 и 3.5 можно увидеть примеры работы программы.

C:\Windows\system32\cmd.exe

```
Длина массива: 5
myArray[0] = 4
myArray[1] = 1
myArray[2] = 2
myArray[3] = 5
myArray[4] = 3

-* Ваш массив *-
4 1 2 5 3

... Обработка данных ...
Сумма элементов: 15
Среднее значение: 3,00
Сумма положительных элементов: 15
Сумма отрицательных элементов: 0
Сумма чётных элементов: 6
Сумма нечётных элементов: 9
Максимальный элемент: 5, позиция: 3
Минимальный элемент: 1, позиция: 1
Произведение элементов между макс. и мин. значениями: 24
Для продолжения нажмите любую клавишу . . .
```

Рисунок 3.4 — Упр №3: Первый вариант вывода программы

C:\Windows\system32\cmd.exe

```
Длина массива: 10
myArray[0] = 1
myArray[1] = 2
myArray[2] = -3
myArray[3] = -4
myArray[4] = 5
myArray[5] = 6
myArray[6] = -7
myArray[7] = -8
myArray[8] = 9
myArray[9] = 10

-* Ваш массив *-
1 2 -3 -4 5 6 -7 -8 9 10

... Обработка данных ...
Сумма элементов: 11
Среднее значение: 1,10
Сумма положительных элементов: 33
Сумма отрицательных элементов: -22
Сумма чётных элементов: 6
Сумма нечётных элементов: 5
Максимальный элемент: 10, позиция: 9
Минимальный элемент: -8, позиция: 7
Произведение элементов между макс. и мин. значениями: -45360
Для продолжения нажмите любую клавишу . . .
```

Рисунок 3.5 — Упр №3: Второй вариант вывода программы

## **ЗАКЛЮЧЕНИЕ**

При выполнении лабораторной работы были созданы различные методы работы над массивами, их создания и вывода.

Цель изучения и приобретения навыков работы с массивами была выполнена.