

Санкт-Петербургский Национальный Исследовательский  
Университет Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий

**Лабораторная работа №5**  
**Создание и использование массивов**

Выполнил  
Стафеев И.А.

Группа  
К3221

Проверил  
Иванов С.Е.

Санкт-Петербург,  
2024

## СОДЕРЖАНИЕ

	Стр.
<b>ВВЕДЕНИЕ .....</b>	<b>3</b>
<b>1 Упражнение 1.....</b>	<b>4</b>
<b>2 Упражнение 2.....</b>	<b>6</b>
<b>3 Упражнение 3.....</b>	<b>10</b>
<b>ЗАКЛЮЧЕНИЕ .....</b>	<b>16</b>

## ВВЕДЕНИЕ

Цель работы: изучение массивов в языке C# и приобретение навыков работы с ними.

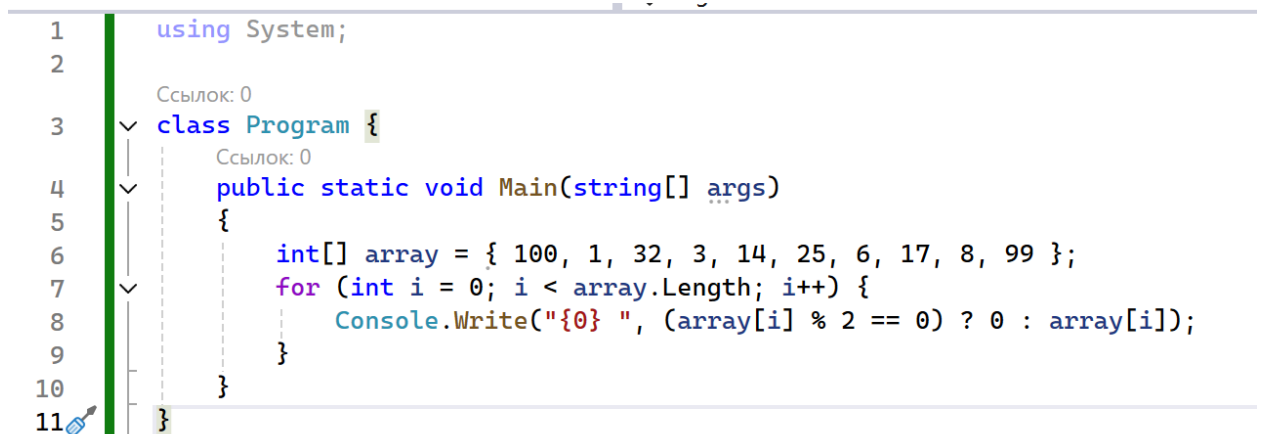
Для достижения цели необходимо выполнить следующие упражнения:

1. Работа с массивом размерного типа данных
2. Перемножение матриц
3. Обработка данных массива

## 1 Упражнение 1

Задача: реализовать вывод нечетных чисел в массиве, используя сначала массив заданного размера, затем с пользовательским вводом верхней границы.

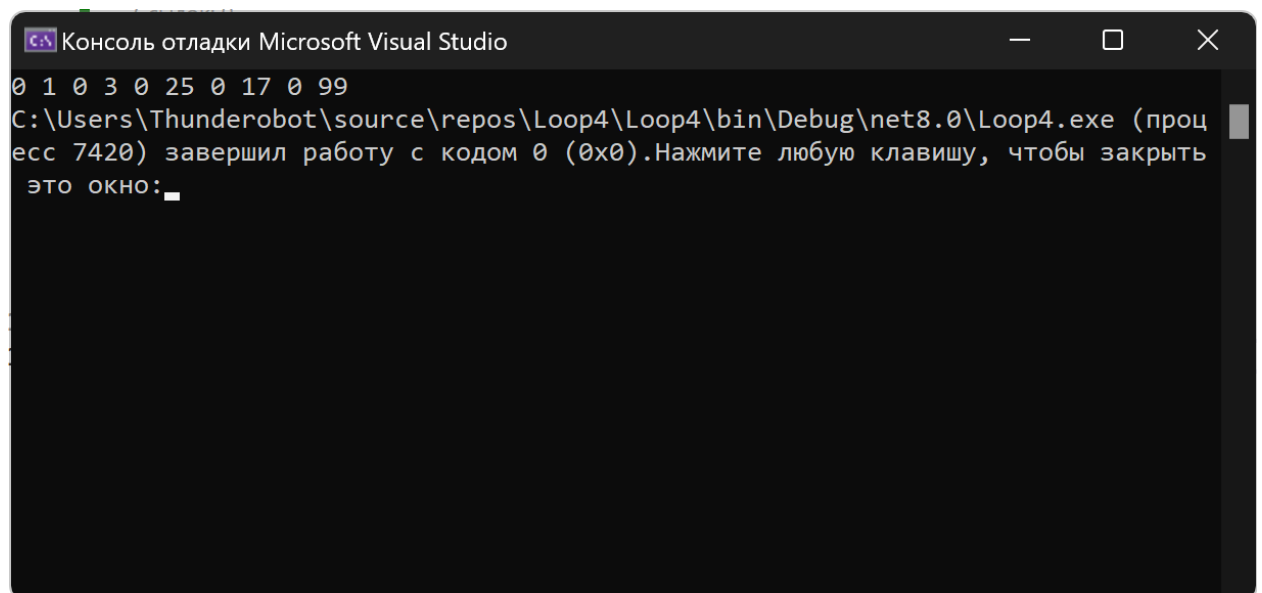
Код вывода элементов массива фиксированного размера приведен на рисунке 1. Для вывода нулей вместо четных чисел использован тернарный оператор.



```
1 using System;
2
3 class Program {
4     public static void Main(string[] args)
5     {
6         int[] array = { 100, 1, 32, 3, 14, 25, 6, 17, 8, 99 };
7         for (int i = 0; i < array.Length; i++) {
8             Console.Write("{0} ", (array[i] % 2 == 0) ? 0 : array[i]);
9         }
10    }
11 }
```

Рисунок 1 — Код для вывода нечетных элементов фиксированного массива

Результат выполнения показан на рисунке 2.



```
Консоль отладки Microsoft Visual Studio
0 1 0 3 0 25 0 17 0 99
C:\Users\Thunderobot\source\repos\Loop4\Loop4\bin\Debug\net8.0\Loop4.exe (процесс 7420) завершил работу с кодом 0 (0x0).Нажмите любую клавишу, чтобы закрыть это окно: _
```

Рисунок 2 — Пример выполнения программы

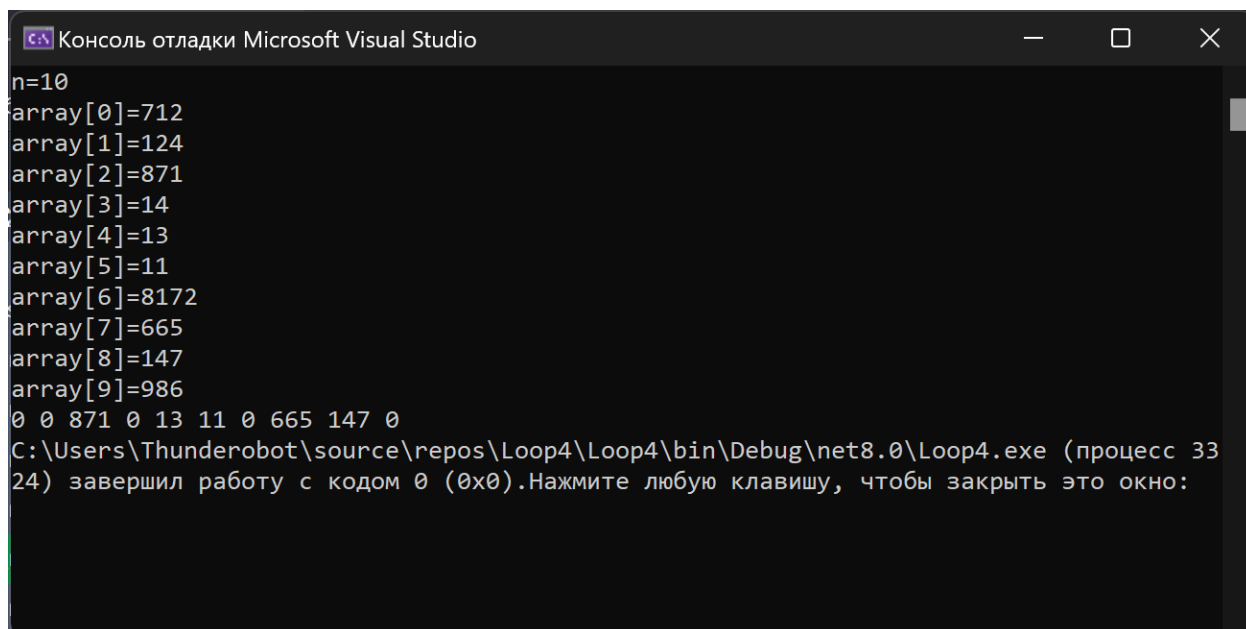
Код вывода элементов массива, длина и элементы которого задаются пользователем, можно увидеть на рисунке 3. Для создания массива с

заданным пользователем размером используется ключевое слово **new**, создающее экземпляр данного класса.

```
1  using System;
2
3  class Program {
4      public static void Main(string[] args)
5      {
6          int[] array;
7          Console.Write("n=");
8          int n = int.Parse(Console.ReadLine());
9          array = new int[n];
10         for (int i = 0; i < array.Length; i++) {
11             Console.Write("array[{0}]=", i);
12             array[i] = int.Parse(Console.ReadLine());
13         }
14         foreach (int x in array) Console.Write("{0} ", (x % 2 == 0) ? 0 : x);
15     }
16 }
```

Рисунок 3 — Код для вывода нечетных элементов заданного пользователем массива

Пример выполнения программы показан на рисунке 4.



```
Консоль отладки Microsoft Visual Studio
n=10
array[0]=712
array[1]=124
array[2]=871
array[3]=14
array[4]=13
array[5]=11
array[6]=8172
array[7]=665
array[8]=147
array[9]=986
0 0 871 0 13 11 0 665 147 0
C:\Users\Thunderobot\source\repos\Loop4\Loop4\bin\Debug\net8.0\Loop4.exe (процесс 3324) завершил работу с кодом 0 (0x0).Нажмите любую клавишу, чтобы закрыть это окно:
```

Рисунок 4 — Пример выполнения программы

## 2 Упражнение 2

Задача: В этом упражнении вы напишите программу, в которой массивы будут использоваться для перемножения матриц. Программа будет считывать с консоли 4 целых числа и сохранять их в матрице размером 2x2, затем будут считываться еще 4 целых числа и сохраняться еще в одной матрице размером 2x2. Далее эти матрицы будут перемножаться, а результат сохранится в третьей матрице тех же размеров. Результат перемножения матриц выведется на экран консоли.

Для решения задачи было реализовано несколько методов основного класса программы.

Метод для ввода значений матрицы представлен на рисунке 5. Благодаря реализации метод позволяет вводить элементы матрицы любого размера, поскольку использует метод `GetLength`.

Ссылка: 2

```
private static void Input(int[,] a)
{ // ввод элементов матрицы
    for (int row = 0; row < a.GetLength(0); row++)
    {
        for (int col = 0; col < a.GetLength(1); col++)
        {
            Console.Write("Enter value for [{0},{1}] : ", row, col);
            a[row, col] = int.Parse(Console.ReadLine());
        }
    }
    Console.WriteLine();
}
```

Рисунок 5 — Код метода Input

Метод для вывода результирующей матрицы показан на рисунке 6. Реализация у него почти такая же, как у метода для ввода матрицы.

Ссылка: 1

```
private static void Output(int[,] result)
{ // вывод матрицы
    for (int row = 0; row < result.GetLength(0); row++) {
        for (int col = 0; col < result.GetLength(1); col++) {
            Console.Write("{0} ", result[row, col]);
        }
        Console.WriteLine();
    }
}
```

---

Рисунок 6 — Код метода Output

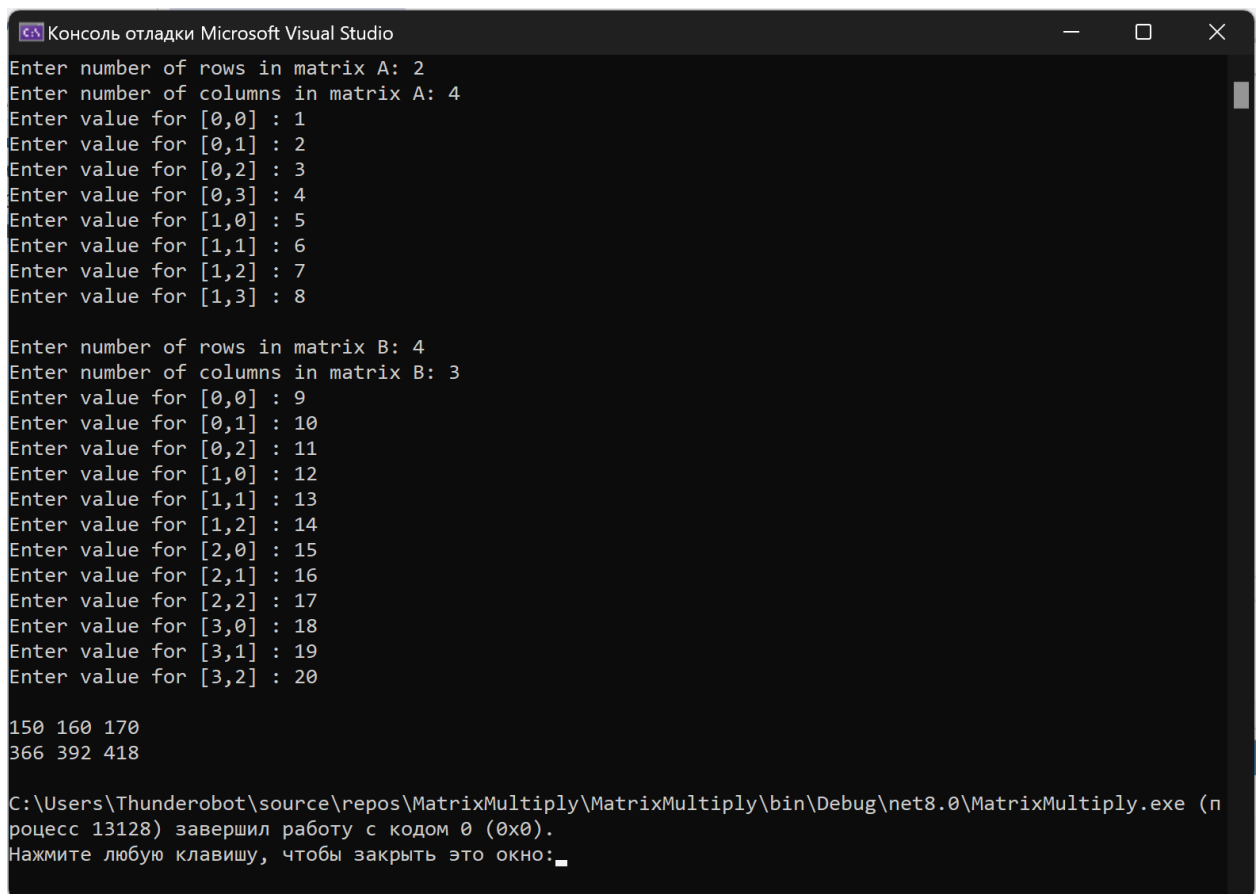
Метод для перемножения матриц можно увидеть на рисунке 7. Он был реализован с двумя вложенными циклами и использованием метода массива `GetLength`, чтобы перемножить две матрицы любых размеров (а не только 2 на 2, как в задании). Если матрицы нельзя перемножить, вызывается исключение.

Ссылка: 1

```
private static int[,] Multiply(int[,] a, int[,] b)
{ // перемножение матриц
    if (a.GetLength(1) != b.GetLength(0)) throw new ArgumentException();
    int[,] result = new int[a.GetLength(0), b.GetLength(1)];
    for (int row = 0; row < a.GetLength(0); row++) {
        for (int col = 0; col < b.GetLength(1); col++) {
            for (int i = 0; i < a.GetLength(1); i++) {
                result[row, col] += a[row, i] * b[i, col];
            }
        }
    }
    return result;
}
```

Рисунок 7 — Код метода Multiply

Пример выполнения программы приведен на рисунке 8.



```
Консоль отладки Microsoft Visual Studio
Enter number of rows in matrix A: 2
Enter number of columns in matrix A: 4
Enter value for [0,0] : 1
Enter value for [0,1] : 2
Enter value for [0,2] : 3
Enter value for [0,3] : 4
Enter value for [1,0] : 5
Enter value for [1,1] : 6
Enter value for [1,2] : 7
Enter value for [1,3] : 8

Enter number of rows in matrix B: 4
Enter number of columns in matrix B: 3
Enter value for [0,0] : 9
Enter value for [0,1] : 10
Enter value for [0,2] : 11
Enter value for [1,0] : 12
Enter value for [1,1] : 13
Enter value for [1,2] : 14
Enter value for [2,0] : 15
Enter value for [2,1] : 16
Enter value for [2,2] : 17
Enter value for [3,0] : 18
Enter value for [3,1] : 19
Enter value for [3,2] : 20

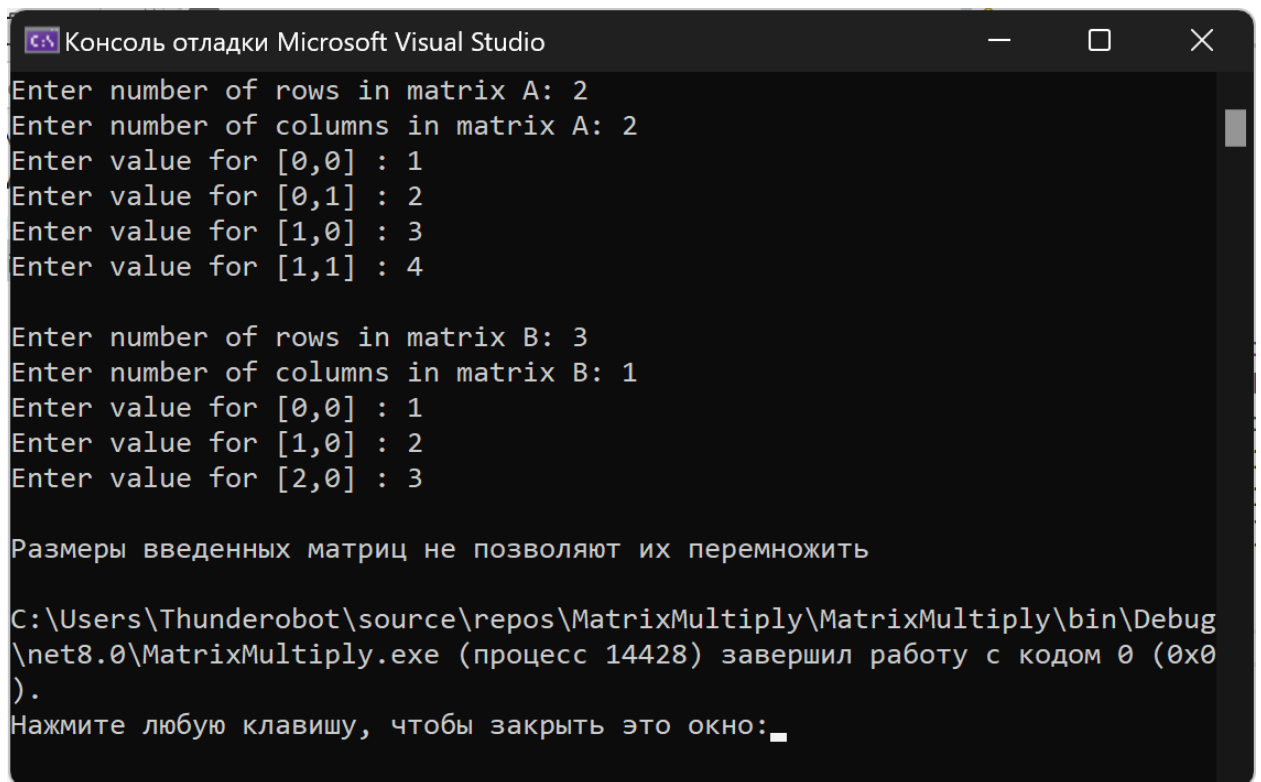
150 160 170
366 392 418

C:\Users\Thunderobot\source\repos\MatrixMultiply\MatrixMultiply\bin\Debug\net8.0\MatrixMultiply.exe (п
роцесс 13128) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: _
```

Рисунок 8 — Пример выполнения программы

Пример выполнения, когда введенные матрицы нельзя перемножить, показан на 9.





```
Консоль отладки Microsoft Visual Studio

Enter number of rows in matrix A: 2
Enter number of columns in matrix A: 2
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 number of rows in matrix B: 3
Enter number of columns in matrix B: 1
Enter value for [0,0] : 1
Enter value for [1,0] : 2
Enter value for [2,0] : 3

Размеры введенных матриц не позволяют их перемножить

C:\Users\Thunderobot\source\repos\MatrixMultiply\MatrixMultiply\bin\Debug\net8.0\MatrixMultiply.exe (процесс 14428) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно: 
```

Рисунок 9 — Пример выполнения программы с неперемножаемыми матрицами

### 3 Упражнение 3

Задача: Требуется создать массив, заполнить его числами и выполнить обработку данных. С помощью цикла (тип цикла на ваше усмотрение) реализуйте заполнение массива с клавиатуры. Добавьте в программу методы для обработки данных массива:

- определение суммы всех элементов массива
- среднего значения массива
- расчет суммы отрицательных или положительных элементов
- расчет суммы элементов с нечетными или четными номерами
- найти максимальный или минимальный элементы массива и вывести их индексы
- рассчитать произведение элементов массива, расположенных между максимальным и минимальным элементами

Методы для ввода массива, нахождения суммы всех элементов и среднего значения показаны на рисунке 10. Метод Mean внутри себя использует уже реализованный метод Sum.

```
private static void Input(int[] a)
{ // ввод элементов массива
    for (int i = 0; i < a.GetLength(0); i++)
    {
        Console.Write("a[{0}]=", i);
        a[i] = int.Parse(Console.ReadLine());
    }
}
```

Ссылка: 2

```
private static long Sum(int[] a) {
    // сумма элементов
    long sum = 0;
    foreach (int i in a) sum += i;
    return sum;
}
```

Ссылка: 0

```
private static double Mean(int[] a) {
    // среднее значение
    double sum = Sum(a);
    return sum / a.Length;
}
```

Ссылка: 2

Рисунок 10 — Методы Input, Sum и Mean

Для нахождения суммы элементов одного знака и суммы элементов на местах одной четности был перегружен метод Sum двумя другими методами. Метод, принимающий в качестве параметра булевое значение, находит сумму элементов одного знака, а метод, принимающий строку - сумму элементов на местах одной четности. Код показан на рисунке 11.

Ссылка: 2

```
private static long Sum(int[] a, bool flag) {  
    // сумма положительных (flag=true) или отрицательных (flag=false) элементов  
    long sum = 0;  
    foreach (int x in a) {  
        if (flag && x > 0 || !flag && x < 0) sum += x;  
    }  
    return sum;  
}
```

Ссылка: 2

```
private static long Sum(int[] a, string flag)  
{  
    // сумма элементов на нечетных (flag='odd') или четных (flag='even') позициях  
    long sum = 0;  
    for (int i = 0; i < a.Length; i++)  
    {  
        if (flag == "odd" && i % 2 != 0) sum += a[i];  
        else if (flag == "even" && i % 2 == 0) sum += a[i];  
    }  
    return sum;  
}
```

Рисунок 11 — Перегрузка метода Sum

Методы для нахождения индексов минимального и максимального элементов и нахождения перемножения элементов между ними представлены на рисунке 12. Метод `MultiplyBetweenMinMax` вызывает внутри себя метод `GetMaxMinIndexes`. Метод `MultiplyBetweenMinMax` использует возвращаемые параметры. В методе для перемножения использовано ключевое слово **checked**, которое вызовет исключение, если будет переполнение. Это исключением обрабатывается в методе `Main`.

Ссылка: 2

```
private static void GetMaxMinIndexes(int[] a, out int xmin, out int xmax) {  
    // получение индексов минимального и максимального элемента  
    xmin = 0; xmax = 0;  
    long min = int.MaxValue; long max = int.MinValue;  
    for (int i = 0; i < a.Length; i++)  
    {  
        if (a[i] < min) { min = a[i]; xmin = i; }  
        if (a[i] > max) { max = a[i]; xmax = i; }  
    }  
}
```

Ссылка: 1

```
private static long MultiplyBetweenMinMax(int[] a) {  
    // умножение всех элементов между минимальным и максимальным  
    int xmin, xmax;  
    long mult = 1;  
    GetMaxMinIndexes(a, out xmin, out xmax);  
    checked // вдруг будет переполнение  
    {  
        for (int i = Math.Min(xmin, xmax) + 1; i < Math.Max(xmin, xmax); i++)  
        {  
            mult *= a[i];  
        }  
    }  
    return mult;  
}
```

Рисунок 12 — Методы GetMaxMinIndexes и MultiplyBetweenMinMax

Код метода Main приведен на рисунке 13. Он внутри себя вызывает все созданные ранее методы и выводит результат их выполнения.

```

public static void Main(string[] args)
{
    try
    {
        Console.WriteLine("Enter massive length");
        int n = int.Parse(Console.ReadLine());
        int[] array = new int[n];
        Input(array);
        Console.WriteLine("Сумма элементов равна {0}", Sum(array));
        Console.WriteLine("Сумма положительных элементов равна {0}", Sum(array, true));
        Console.WriteLine("Сумма отрицательных элементов равна {0}", Sum(array, false));
        Console.WriteLine("Сумма элементов на четных местах равна {0}", Sum(array, "even"));
        Console.WriteLine("Сумма элементов на нечетных местах равна {0}", Sum(array, "odd"));
        int x1, x2;
        GetMaxMinIndexes(array, out x1, out x2);
        Console.WriteLine("Индексы минимального и максимального элементов - {0} и {1}", x1, x2);
        Console.WriteLine("Перемножение элементов между максимальным и минимальным равно {0}",
            MultiplyBetweenMinMax(array));
    }
    catch (FormatException e)
    {
        Console.WriteLine("Ошибка формата: {0}", e.Message);
    }
    catch (Exception e) {
        Console.WriteLine("Элементы массива слишком большие, чтобы посчитать их перемножение");
    }
}

```

Рисунок 13 — Метод Main

Пример выполнения программы приведен на рисунке 14.

```

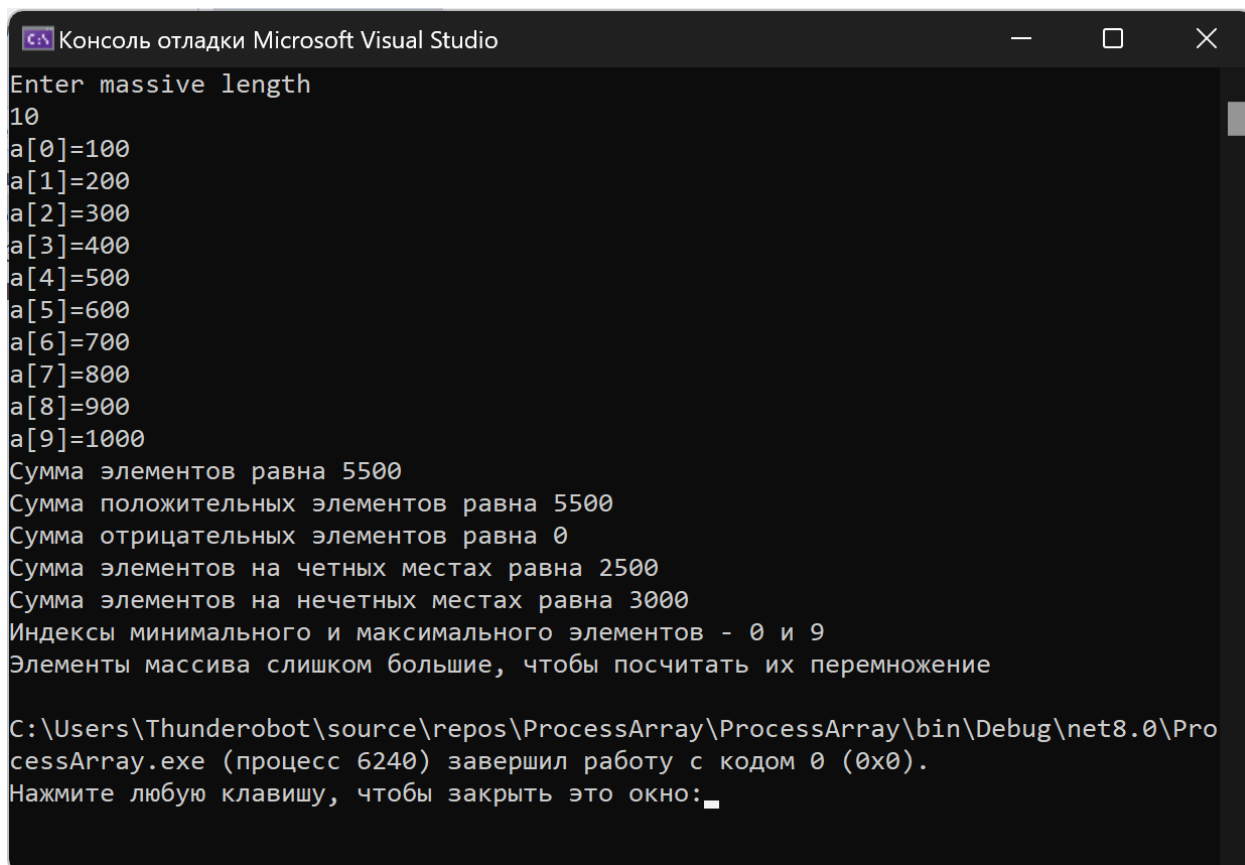
Консоль отладки Microsoft Visual Studio
Enter massive length
10
a[0]=5
a[1]=4
a[2]=3
a[3]=200
a[4]=1
a[5]=2
a[6]=3
a[7]=4
a[8]=-10
a[9]=6
Сумма элементов равна 218
Сумма положительных элементов равна 228
Сумма отрицательных элементов равна -10
Сумма элементов на четных местах равна 2
Сумма элементов на нечетных местах равна 216
Индексы минимального и максимального элементов - 8 и 3
Перемножение элементов между максимальным и минимальным равно 24

C:\Users\Thunderobot\source\repos\ProcessArray\ProcessArray\bin\Debug\net8.0\ProcessArray.exe (
процесс 14040) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:

```

Рисунок 14 — Пример выполнения программы

Пример выполнения с переполнением показан на 15.



```
Консоль отладки Microsoft Visual Studio
Enter massive length
10
a[0]=100
a[1]=200
a[2]=300
a[3]=400
a[4]=500
a[5]=600
a[6]=700
a[7]=800
a[8]=900
a[9]=1000
Сумма элементов равна 5500
Сумма положительных элементов равна 5500
Сумма отрицательных элементов равна 0
Сумма элементов на четных местах равна 2500
Сумма элементов на нечетных местах равна 3000
Индексы минимального и максимального элементов - 0 и 9
Элементы массива слишком большие, чтобы посчитать их перемножение

C:\Users\Thunderobot\source\repos\ProcessArray\ProcessArray\bin\Debug\net8.0\ProcessArray.exe (процесс 6240) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:■
```

Рисунок 15 — Пример выполнения с переполнением

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторной работы были выполнены все требуемые упражнения. Цель работы достигнута. Получены знания и приобретены навыки работы с массивами в языке C#.