**МИНОБРНАУКИ РОССИИ**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

Институт среднего профессионального образования

**ОТЧЕТ**

**по лабораторной работе №2**

по профессиональному модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем»

Специальность09.02.07 «Информационные системы и программирование»

Выполнили студенты 3 курса 32919/1 группы

Зориков Антон Алексеевич, Лейко Матвей Александрович

Преподаватель: Ильин Ю. П.

Санкт-Петербург

2023

**Cодержание**

[**Часть 1.** 3](#_Toc151118833)

[**Цель работы:** 3](#_Toc151118834)

[**Задание:** 3](#_Toc151118835)

[**Математическая формулировка методов:** 4](#_Toc151118836)

[**Текст программных модулей** 10](#_Toc151118837)

[**Протокол испытаний:** 16](#_Toc151118838)

[**Результаты выполнения работы программы** 22](#_Toc151118839)

[Часть 2. 24](#_Toc151118840)

[**Задание:** 24](#_Toc151118841)

[**Текст программных модулей:** 24](#_Toc151118842)

[**Форма:** 33](#_Toc151118843)

**Тема: «Разработка класса по работе с массивами»**

**Вариант №9**

**Часть 1.**

# **Цель работы:**

Изучение простейших классов, содержащих скрытые поля, конструкторы с параметрами и без параметров, методы, свойства, научиться работе с массивами и WindowsForms.

# **Задание:**

Создать библиотечный модуль с именем ClassLibraryForArray с описанием класса IntArray по работе с одномерными целочисленными массивами. Для тестирования библиотечного модуля создать консольное приложение (проект) с именем ConsoleAppTest. Для информирования пользователя о том, какой метод отработал, добавить обработчик события Notify.

Создать приложение Windows Form с графическим интерфейсом. Подключить к нему созданный в 1-й части задания библиотечный модуль. Основная программа должна обрабатывать 2 массива разной длины. Длина каждого массива запрашивается у пользователя. Для работы с массивами надо предусмотреть на форме 2 таблицы (компонент DataGridVew), размеры которых задаются в процессе выполнения программы. Заполнение каждого массива может происходить при помощи генератора случайных чисел, вручную или из текстового файла (в последнем случае размер массива должен определяться по количеству чисел в файле). Для выбора режима заполнения массива используйте переключатель «зависимая кнопка выбора».

Для каждого массива вычисляется: сумма элементов, количество элементов, кратных заданному числу, а также выполняется сложение и вычитание двух массивов. Результат сложения или вычитания вывести в дополнительную таблицу. Добавить функцию для обработки вашего варианта. Пользовательский интерфейс может быть оформлен в виде системы меню (MenuStrip) или панели с кнопками (панели инструментов ToolStrip).

Программа должна выполнять задание с первым или вторым массивом по выбору пользователя. Если в самом массиве происходят изменения (например, перестановка элементов), результат вывести в дополнительную таблицу. Если же необходимо из двух массивов получить новый массив, то результат выводите также в дополнительную таблицу.

# **Математическая формулировка методов:**

1. Конструктор по умолчанию

public IntArray(){}

Данный конструктор является конструктором по умолчанию и следовательно не принимает никаких параметров.

1. Конструктор включающий в себя инициализацию поля длины массива

public IntArray(int length)

{

this.Length = length;

}

Данный конструктор принимает значения длины массива и устанавливает значение соответствующего поля. Параметр конструктора задан целочисленным типом int.

1. Конструктор включающий в себя инициализацию массива

public IntArray(params int[] arr)

{

this.a = arr;

}

Данный конструктор принимает массив как аргумент и заполняет массив a значениями массива arr. Параметр конструктора задан целочисленным массивом типа int.

1. Индексатор

public int this[int i]

{

get { return a[i]; }

set{ a[i] = value; }

}

Позволяют индексировать объект и обращаться к данным по индексу. Фактически с помощью данного индексатора мы можем работать с объектами как с массивами. С его помощью мы можем получить элемент массива по индексу, или изменить значение элемента по индексу.

1. Метод для создания массива, заполненного случайными элементами

public static IntArray RandomIntArray(int length, int a, int b)

{

IntArray intArray = new IntArray(length);

Random random = new Random();

for(int i = 0; i < length; i++)

{

intArray[i] = random.Next(a, b);

}

return intArray;

}

Данный метод принимает значения длины генерируемого массива. После чего создается экземпляр класса и заполняется случайными числами диапазона от a до b. Параметры метода заданы целочисленным типом int.

1. Метод для создания массива, заполненного данными из файла.

public static IntArray ArrayFromTextFile(string fileName)

{

IntArray intArray = new IntArray();

try

{

StreamReader f = new StreamReader(fileName);

LinkedList<int> list = new LinkedList<int>();

string s;

int i = 0;

int len = 0;

while ((s = f.ReadLine()) != null)

{

list.AddLast(int.Parse(s));

len++;

}

intArray.Length = len;

while (i < list.Count)

{

intArray[i] = list.ElementAt(i);

i++;

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

return intArray;

}

Данный метод принимает имя файла. После чего создается экземпляр класса и заполняется числами, находящимися в файле. Параметр метода задан типом string.

1. Метод для записи данных массива в файл

public static void ArrayToTextFile(IntArray arr, string fileName)

{

try

{

using (StreamWriter streamWriter = new StreamWriter(fileName))

{

for (int i = 0; i < arr.Length; i++)

{

streamWriter.WriteLine(arr[i]);

}

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

Данный метод принимает экземпляр класса и имя создаваемого файла. После чего создается файл в который записываются значения из массива. Параметр метода задан типом IntArray.

1. Метод для получения суммы массива

public static int SumArray(IntArray arr)

{

int sum = 0;

for(int i = 0; i < arr.Length; i++)

{

sum += arr[i];

}

return sum;

}

Данный метод принимает экземпляр. После чего суммирует все элементы массива. Параметр метода задан типом IntArray.

1. Оператор инкремента класса

public static IntArray operator ++(IntArray arr)

{

for (int i = 0; i < arr.Length; i++)

{

arr[i]++;

}

return arr;

}

Данный оператор позволяет увеличить каждый элемент массива на единицу. Параметр метода задан типом IntArray.

1. Оператор сложение массива со скаляром

public static IntArray operator +(IntArray x, int y)

{

for (int i = 0; i < x.Length; i++)

{

x[i] += y;

}

return x;

}

Данный оператор позволяет увеличить каждый элемент массива на заданное число. Параметр метода задан типом IntArray и целочисленным типом int.

1. Оператор сложение скаляра с массивом

public static IntArray operator +(int x, IntArray y)

{

for (int i = 0; i < y.Length; i++)

{

y[i] += x;

}

return y;

}

Данный оператор позволяет увеличить каждый элемент массива на заданное число. Параметр метода задан типом IntArray и целочисленным типом int.

1. Оператор сложения двух массивов

public static IntArray operator +(IntArray x, IntArray y)

{

if (y.Length > x.Length)

{

for (int i = 0; i < y.Length; i++)

{

y[i] += x[i];

}

return y;

}

else

{

for (int i = 0; i < x.Length; i++)

{

x[i] += y[i];

}

return x;

}

}

Данный оператор позволяет сложить элементы двух массивов между собой. Если размер массивов не одинаков, то выбрасывается исключение. Параметры методы задан типом IntArray.

1. Оператор декремента класса.

public static IntArray operator --(IntArray arr)

{

for (int i = 0; i < arr.Length; i++)

{

arr[i]--;

}

return arr;

}

Данный оператор позволяет уменьшить каждый элемент массива на единицу. Параметр метода задан типом IntArray.

1. Оператор вычитания скаляра из массива

public static IntArray operator -(IntArray x, int y)

{

for (int i = 0; i < x.Length; i++)

{

x[i] -= y;

}

return x;

}

Данный оператор позволяет уменьшить каждый элемент массива на заданное число. Параметр метода задан типом IntArray и целочисленным типом int.

1. Оператор вычитания массива из скаляра

public static IntArray operator -(int x, IntArray y)

{

for (int i = 0; i < y.Length; i++)

{

y[i] -= x;

}

return y;

}

Данный оператор позволяет уменьшить каждый элемент массива на заданное число. Параметр метода задан типом IntArray и целочисленным типом int.

1. Оператор вычитания массива из массива

public static IntArray operator -(IntArray x, IntArray y)

{

if (y.Length > x.Length)

{

for (int i = 0; i < y.Length; i++)

{

y[i] -= x[i];

}

return y;

}

else

{

for (int i = 0; i < x.Length; i++)

{

x[i] -= y[i];

}

return x;

}

}

Данный оператор позволяет вычесть элементы двух массивов друг из друга. . Если размер массивов не одинаков, то выбрасывается исключение. Параметры методы задан типом IntArray.

1. Переопределённый метод для вывода информации об объекте.

public override string ToString()

{

string s = "";

for (int i = 0; i < Length; i++)

{

s += a[i] + " ";

}

return s;

}

Данный метод возвращает информацию об объекте (все элементы массива)

**Текст программных модулей:**

// библиотечный модуль \*.dll

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClassLibraryForArray

{

public class IntArray

{

private int[] a;

private int length;

public int Length

{

get { return length; }

set{

length = value;

a = new int[length];

}

}

public int this[int i]

{

get { return a[i]; }

set{ a[i] = value; }

}

public static IntArray RandomIntArray(int length, int a, int b)

{

IntArray intArray = new IntArray(length);

Random random = new Random();

for(int i = 0; i < length; i++)

{

intArray[i] = random.Next(a, b);

}

return intArray;

}

public static IntArray ArrayFromTextFile(string fileName)

{

IntArray intArray = new IntArray();

try

{

StreamReader f = new StreamReader(fileName);

LinkedList<int> list = new LinkedList<int>();

string s;

int i = 0;

int len = 0;

while ((s = f.ReadLine()) != null)

{

list.AddLast(int.Parse(s));

len++;

}

intArray.Length = len;

while (i < list.Count)

{

intArray[i] = list.ElementAt(i);

i++;

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

return intArray;

}

public static void ArrayToTextFile(IntArray arr, string fileName)

{

try

{

using (StreamWriter streamWriter = new StreamWriter(fileName))

{

for (int i = 0; i < arr.Length; i++)

{

streamWriter.WriteLine(arr[i]);

}

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

public static int SumArray(IntArray arr)

{

int sum = 0;

for(int i = 0; i < arr.Length; i++)

{

sum += arr[i];

}

return sum;

}

public static IntArray operator ++(IntArray arr)

{

for (int i = 0; i < arr.Length; i++)

{

arr[i]++;

}

return arr;

}

public static IntArray operator +(IntArray x, int y)

{

for (int i = 0; i < x.Length; i++)

{

x[i] += y;

}

return x;

}

public static IntArray operator +(int x, IntArray y)

{

for (int i = 0; i < y.Length; i++)

{

y[i] += x;

}

return y;

}

public static IntArray operator +(IntArray x, IntArray y)

{

if (y.Length > x.Length)

{

for (int i = 0; i < y.Length; i++)

{

y[i] += x[i];

}

return y;

}

else

{

for (int i = 0; i < x.Length; i++)

{

x[i] += y[i];

}

return x;

}

}

public static IntArray operator --(IntArray arr)

{

for (int i = 0; i < arr.Length; i++)

{

arr[i]--;

}

return arr;

}

public static IntArray operator -(IntArray x, int y)

{

for (int i = 0; i < x.Length; i++)

{

x[i] -= y;

}

return x;

}

public static IntArray operator -(int x, IntArray y)

{

for (int i = 0; i < y.Length; i++)

{

y[i] -= x;

}

return y;

}

public static IntArray operator -(IntArray x, IntArray y)

{

if (y.Length > x.Length)

{

for (int i = 0; i < y.Length; i++)

{

y[i] -= x[i];

}

return y;

}

else

{

for (int i = 0; i < x.Length; i++)

{

x[i] -= y[i];

}

return x;

}

}

public static IntArray Method(IntArray A, IntArray B)

{

IntArray C = new IntArray(A.Length - 1);

for (int i = 0; i < A.Length - 1; i++)

{

C[i] = A[i] \* A[i + 1] + B[i] \* B[i + 1];

}

return C;

}

public override string ToString()

{

string s = "";

for (int i = 0; i < Length; i++)

{

s += a[i] + " ";

}

return s;

}

public IntArray(){}

public IntArray(int length)

{

this.Length = length;

}

public IntArray(params int[] arr)

{

this.a = arr;

}

}

}

// код программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using ClassLibraryForArray;

namespace ConsoleAppTest

{

class Program

{

static void Main(string[] args)

{

string fileName = "data.txt";

IntArray arr = new IntArray();

try

{

Console.Write("Введите длину массива: ");

int len = int.Parse(Console.ReadLine());

arr.Length = len;

for (int i = 0; i < arr.Length; i++)

{

Console.Write("Введите число " + i + ": ");

arr[i] = int.Parse(Console.ReadLine());

}

Console.WriteLine(arr);

}

catch (Exception e)

{

Console.WriteLine("Ошибка: неверное значение для длины массива");

}

IntArray randomArr = IntArray.RandomIntArray(arr.Length, 0, 10);

Console.Write("Случайный массив: ");

Console.WriteLine(randomArr);

IntArray.ArrayToTextFile(randomArr, "test.txt");

Console.WriteLine("Метод ArrayToTextFile выполнен." + '\n');

int sum = IntArray.SumArray(arr);

Console.WriteLine("Сумма элементов массива: " + sum);

IntArray arr2 = IntArray.ArrayFromTextFile(fileName);

Console.WriteLine(arr2);

Console.WriteLine("Проверка инкремента");

Console.WriteLine(arr);

arr++;

Console.WriteLine(arr);

Console.WriteLine("Проверка сложения массива со скаляром");

Console.WriteLine(arr);

arr += 10;

Console.WriteLine(arr);

Console.WriteLine("Проверка сложения скаляра с массивом");

Console.WriteLine(arr);

arr = 10 + arr;

Console.WriteLine(arr);

Console.WriteLine("Проверка сложения массивов");

IntArray arr3 = IntArray.RandomIntArray(3, 0, 10);

Console.WriteLine("Массив 1: " + arr);

Console.WriteLine("Массив 2: " + arr3);

arr += arr3;

Console.WriteLine(arr);

Console.WriteLine("Проверка вычетания скаляра из массива");

Console.WriteLine(arr);

arr -= 10;

Console.WriteLine(arr);

Console.WriteLine("Проверка вычетания массива из скаляра");

Console.WriteLine(arr);

arr = 10 - arr;

Console.WriteLine(arr);

Console.WriteLine("Проверка вачетания массивов");

arr3 = IntArray.RandomIntArray(3, 0, 10);

Console.WriteLine("Массив 1: " + arr);

Console.WriteLine("Массив 2: " + arr3);

arr -= arr3;

Console.WriteLine(arr);

Console.ReadLine();

}

}

}

# **Протокол испытаний:**

*Таблица 1.* Протокол испытаний

|  |  |  |  |
| --- | --- | --- | --- |
| **Проверяемые требования** | **Сообщения программы и вводимые значения** | **Ожидаемые результаты** | **Фактические результаты** |
| Проверка ввода корректного значения для длины массива. Элементы массива заполнены случайно | len = 4  arr[0]=1  arr[1]=2  arr[2]=-2  arr[3]=100 | Сообщение c выводом массива. | len = 4  arr[0]=1  arr[1]=2  arr[2]=-2  arr[3]=100  1 2 -2 100 |
| Проверка ввода некорректного значения для длины. | len = -2 | Сообщение, вызванное исключением (Ошибка: неверное значение для длины массива) при вводе длины. | Введите длину массива: -2  Ошибка: неверное значение для длины массива |
| Проверка работы случайного массива. | len = 2  arr[0]=2  arr[1]=2 | Сообщение c выводом введенного вручную и случайного массива такой же же длины, что и введённый вручную массив. | Введите длину массива: 2  Введите число 0: 2  Введите число 1: 2  2 2  Случайный массив: 1 6 |
| Проверка заполнения файла массивом. | len = 2  arr[0]=2  arr[1]=2 | Сообщение ‘Метод ArrayToTextFile выполнен.’. Создание файла, в котором хранится массив. | Введите длину массива: 2  Введите число 0: 2  Введите число 1: 2  2 2  Случайный массив: 1 6  Метод ArrayToTextFile выполнен. |
| Проверка нахождения суммы элементов массива. | len = 2  arr[0]=2  arr[1]=2 | Сообщение с выводом суммы элементов массива. | Введите длину массива: 2  Введите число 0: 2  Введите число 1: 2  2 2  Случайный массив: 1 6  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 4 |
| Проверка инкремента элементов массива. | len = 2  arr[0]=2  arr[1]=2 | Сообщение с выводом, исходного и измененного массива. | Введите длину массива: 2  Введите число 0: 2  Введите число 1: 2  2 2  Случайный массив: 1 6  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 4  1 2 3 4 4  Проверка инкремента  2 2  3 3 |
| Проверка сложения элементов массива со скаляром. | len = 2  arr[0]=2  arr[1]=2 | Сообщение с выводом, исходного и измененного массива. | Введите длину массива: 2  Введите число 0: 2  Введите число 1: 2  2 2  Случайный массив: 1 6  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 4  1 2 3 4 4  Проверка инкремента  2 2  3 3  Проверка сложения массива со скаляром  3 3  13 13 |
| Проверка сложения скаляра с массивом | len = 3  arr[0]=1  arr[1]=2  arr[2]=3 | Сообщение с выводом, исходного и измененного массива. | Введите длину массива: 3  Введите число 0: 1  Введите число 1: 2  Введите число 2: 3  1 2 3  Случайный массив: 2 7 5  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 6  1 2 3 4 4  Проверка инкремента  1 2 3  2 3 4  Проверка сложения массива со скаляром  2 3 4  12 13 14  Проверка сложения скаляра с массивом  12 13 14  22 23 24 |
| Проверка сложения массивов | len = 3  arr[0]=1  arr[1]=2  arr[2]=3 | Сообщение с выводом, 2 массивов и суммы элементов этих массивов(arr1[0]+arr2[0] и т.д.). | Введите длину массива: 3  Введите число 0: 1  Введите число 1: 2  Введите число 2: 3  1 2 3  Случайный массив: 2 7 5  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 6  1 2 3 4 4  Проверка инкремента  1 2 3  2 3 4  Проверка сложения массива со скаляром  2 3 4  12 13 14  Проверка сложения скаляра с массивом  12 13 14  22 23 24  Проверка сложения массивов  Массив 1: 22 23 24  Массив 2: 6 9 7  28 32 31 |
| Проверка вычитания скаляра из массива | len = 3  arr[0]=1  arr[1]=2  arr[2]=3 | Сообщение с выводом, исходного и измененного массива. | Введите длину массива: 3  Введите число 0: 1  Введите число 1: 2  Введите число 2: 3  1 2 3  Случайный массив: 2 7 5  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 6  1 2 3 4 4  Проверка инкремента  1 2 3  2 3 4  Проверка сложения массива со скаляром  2 3 4  12 13 14  Проверка сложения скаляра с массивом  12 13 14  22 23 24  Проверка сложения массивов  Массив 1: 22 23 24  Массив 2: 6 9 7  28 32 31  Проверка выброса исключения при сложении массивов  Массив 1: 28 32 31  Массив 2: 6 9 7 2 2 9 4 4 2 5 2 4 2 7 8 3 9 3 6 7 3 1 9 3 8 3 0 0 5 9  Длина массивов должна быть одинакова  Проверка вычетания скаляра из массива  28 32 31  18 22 21 |
| Проверка вычитания массива из скаляра | len = 3  arr[0]=1  arr[1]=2  arr[2]=3 | Сообщение с выводом, исходного и измененного массива. | Введите длину массива: 3  Введите число 0: 1  Введите число 1: 2  Введите число 2: 3  1 2 3  Случайный массив: 2 7 5  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 6  1 2 3 4 4  Проверка инкремента  1 2 3  2 3 4  Проверка сложения массива со скаляром  2 3 4  12 13 14  Проверка сложения скаляра с массивом  12 13 14  22 23 24  Проверка сложения массивов  Массив 1: 22 23 24  Массив 2: 6 9 7  28 32 31  Проверка выброса исключения при сложении массивов  Массив 1: 28 32 31  Массив 2: 6 9 7 2 2 9 4 4 2 5 2 4 2 7 8 3 9 3 6 7 3 1 9 3 8 3 0 0 5 9  Длина массивов должна быть одинакова  Проверка вычетания скаляра из массива  28 32 31  18 22 21  Проверка вычетания массива из скаляра  18 22 21  8 12 11 |
| Проверка вычитания массивов | len = 3  arr[0]=1  arr[1]=2  arr[2]=3 | Сообщение с выводом, 2 массивов и разницы элементов этих массивов(arr1[0]-arr2[0] и т.д.). | Введите длину массива: 3  Введите число 0: 1  Введите число 1: 2  Введите число 2: 3  1 2 3  Случайный массив: 2 7 5  Метод ArrayToTextFile выполнен.  Сумма элементов массива: 6  1 2 3 4 4  Проверка инкремента  1 2 3  2 3 4  Проверка сложения массива со скаляром  2 3 4  12 13 14  Проверка сложения скаляра с массивом  12 13 14  22 23 24  Проверка сложения массивов  Массив 1: 22 23 24  Массив 2: 6 9 7  28 32 31  Проверка выброса исключения при сложении массивов  Массив 1: 28 32 31  Массив 2: 6 9 7 2 2 9 4 4 2 5 2 4 2 7 8 3 9 3 6 7 3 1 9 3 8 3 0 0 5 9  Длина массивов должна быть одинакова  Проверка вычетания скаляра из массива  28 32 31  18 22 21  Проверка вычетания массива из скаляра  18 22 21  8 12 11  Проверка вачетания массивов  Массив 1: 8 12 11  Массив 2: 6 9 7  -2 -3 -4 |

**Результаты выполнения работы программы:**

1. Ввод корректного значения для длины. (рис.1).

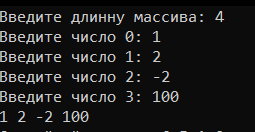


Рис.1 Вывод времени при вводе корректных значений

1. Ввод некорректного значения для длины. (рис.2)



Рис.2 Вывод исключения при некорректном вводе длины массива

1. Проверка работы случайного массива. (рис.3)

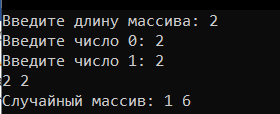


Рис.3 Вывод случайного массива

1. Проверка заполнения файла массивом (рис.4-рис.6)

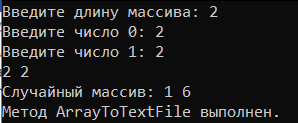


Рис.4 Вывод сообщения о выполнении метода.

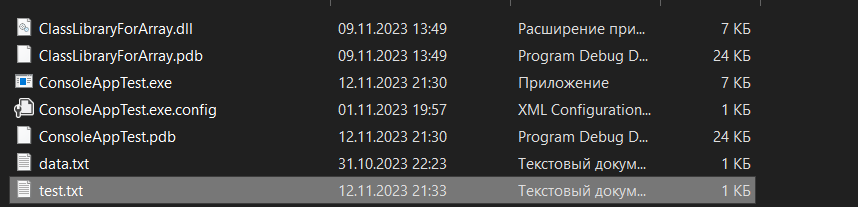


Рис.5 Создался фаил с массивом

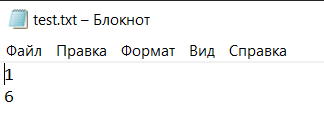


Рис.6 Содержимое созданного файла

1. Проверка нахождения суммы элементов массива



Рис.7 Сумма элементов массива

1. Проверка инкремента элементов массива.



Рис.8 инкримент элементов массива

1. Проверка сложения элементов массива со скаляром.



Рис.9 сложение массива со скаляром

1. Проверка сложения скаляра с массивом



Рис.10 сложение скаляра с массивом

1. Проверка сложения массивов

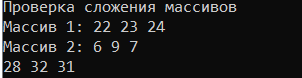


Рис.11 сложение массивов

1. Проверка вычитания скаляра из массива



Рис.12 вычитание скаляра из массива

1. Проверка вычетания массива из скаляра

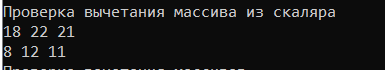


Рис 13. Вычитание массива из скаляра

1. Проверка вачетания массивов

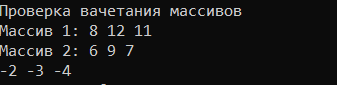


Рис.15 вычитание массивов

Часть 2.

**Задание:**

**Текст программных модулей:**

using ClassLibraryForArray;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Reflection.Emit;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class Form1 : Form

{

int n = 0;

Random r = new Random();

IntArray A = new IntArray();

IntArray B = new IntArray();

IntArray C = new IntArray();

bool isA = false;

public Form1()

{

InitializeComponent();

setInvisible();

dataGridView1.CellValueChanged += DataGridView1\_CellValueChanged;

dataGridView2.CellValueChanged += DataGridView2\_CellValueChanged;

dataGridView3.Enabled = false;

setInvisible();

dataGridView1.Visible = true;

dataGridView1.Enabled = true;

numericUpDown1.Visible = true;

groupBox1.Visible = true;

groupBox3.Visible = true;

radioButton1.Visible = true;

radioButton2.Visible = true;

radioButton5.Visible = true;

radioButton6.Visible = true;

numericUpDown3.Visible = true;

numericUpDown5.Visible = true;

textBox1.Visible = true;

button1.Visible = true;

label4.Visible = true;

label5.Visible = true;

label6.Visible = true;

}

private bool IsInt(string value)

{

return int.TryParse(value, out \_);

}

private void DataGridView1\_CellValueChanged(object sender, DataGridViewCellEventArgs e)

{

if (radioButton1.Checked)

{

DataGridViewCell cell = dataGridView1.Rows[e.RowIndex].Cells[e.ColumnIndex];

if (IsInt(cell.Value?.ToString()) && cell.Value?.ToString()[0] != '0')

A[e.ColumnIndex] = int.Parse(cell.Value.ToString());

else

{

dataGridView1.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = 0;

}

}

}

private void DataGridView2\_CellValueChanged(object sender, DataGridViewCellEventArgs e) //

{

if (radioButton1.Checked)

{

DataGridViewCell cell = dataGridView2.Rows[e.RowIndex].Cells[e.ColumnIndex];

if (IsInt(cell.Value?.ToString()))

B[e.ColumnIndex] = int.Parse(cell.Value.ToString());

else

{

dataGridView2.Rows[e.RowIndex].Cells[e.ColumnIndex].Value = 0;

}

}

}

private void массивАToolStripMenuItem\_Click(object sender, EventArgs e) //меню ToolStrip для мааасива А и отображаемые в нем элементы

{

setInvisible();

dataGridView1.Visible = true;

dataGridView1.Enabled = true;

numericUpDown1.Visible = true;

groupBox1.Visible = true;

groupBox3.Visible = true;

radioButton1.Visible = true;

radioButton2.Visible = true;

radioButton5.Visible = true;

radioButton6.Visible = true;

numericUpDown3.Visible = true;

numericUpDown5.Visible = true;

textBox1.Visible = true;

button1.Visible = true;

label4.Visible = true;

label5.Visible = true;

label6.Visible = true;

}

private void массивБToolStripMenuItem\_Click(object sender, EventArgs e) //меню ToolStrip для массива Б и отображаемые в нем элементы

{

setInvisible();

dataGridView2.Visible = true;

dataGridView2.Enabled = true;

numericUpDown2.Visible = true;

groupBox1.Visible = true;

groupBox3.Visible = true;

radioButton1.Visible = true;

radioButton2.Visible = true;

radioButton5.Visible = true;

radioButton6.Visible = true;

numericUpDown4.Visible = true;

numericUpDown6.Visible = true;

textBox2.Visible = true;

button2.Visible = true;

label7.Visible = true;

label5.Visible = true;

label6.Visible = true;

}

private void setInvisible() // метод для включения режима неотображения для пользователя

{

dataGridView1.Visible = false;

dataGridView1.Enabled = false;

int n = Convert.ToInt32(numericUpDown1.Value);

dataGridView1.SetBounds(12, 217, 100 \* n + n, 60);

dataGridView2.Visible = false;

dataGridView2.Enabled = false;

n = Convert.ToInt32(numericUpDown2.Value);

dataGridView2.SetBounds(12, 217, 100 \* n + n, 60);

dataGridView3.Visible = false;

dataGridView3.SetBounds(12, 217, 100 \* C.Length + C.Length, 60);

numericUpDown1.Visible = false;

numericUpDown2.Visible = false;

numericUpDown3.Visible = false;

numericUpDown4.Visible = false;

numericUpDown5.Visible = false;

numericUpDown6.Visible = false;

groupBox1.Visible = false;

groupBox3.Visible = false;

radioButton1.Visible = false;

radioButton2.Visible = false;

radioButton5.Visible = false;

radioButton6.Visible = false;

textBox1.Visible = false;

textBox2.Visible = false;

button1.Visible = false;

button2.Visible = false;

button3.Visible = false;

button4.Visible = false;

button5.Visible = false;

button6.Visible = false;

button7.Visible = false;

button8.Visible = false;

button9.Visible = false;

label1.Visible = false;

label2.Visible = false;

label3.Visible = false;

label4.Visible = false;

label5.Visible = false;

label6.Visible = false;

label7.Visible = false;

numericUpDown7.Visible = false;

numericUpDown8.Visible = false;

}

private void numericUpDown1\_ValueChanged(object sender, EventArgs e) //заполнение массивов А при изменении его длины

{

int n = Convert.ToInt32(numericUpDown1.Value);

dataGridView1.ColumnCount = n;

dataGridView1.SetBounds(12, 217, 100 \* n + n, 60);

A.Length = n;

for (int i = 0; i < n; i++)

{

dataGridView1.Columns[i].Name = i.ToString();

if (radioButton1.Checked)

{

A[i] = 0;

dataGridView1.Rows[0].Cells[i].Value = A[i];

}

else

{

A[i] = (r.Next((int)numericUpDown3.Value, (int)numericUpDown5.Value));

dataGridView1.Rows[0].Cells[i].Value = A[i];

}

}

}

public void updateGrids() // метод для обновления значения DataGridView из массивов

{

for (int i = 0; i < A.Length; i++)

{

dataGridView1.Rows[0].Cells[i].Value = A[i];

}

for (int i = 0; i < B.Length; i++)

{

dataGridView2.Rows[0].Cells[i].Value = B[i];

}

}

private void numericUpDown2\_ValueChanged(object sender, EventArgs e) //заполнение массивов Б при изменении его длины

{

int n = Convert.ToInt32(numericUpDown2.Value);

dataGridView2.ColumnCount = n;

dataGridView2.SetBounds(12, 217, 100 \* n + n, 60);

B.Length = n;

for (int i = 0; i < n; i++)

{

dataGridView2.Columns[i].Name = i.ToString();

if (radioButton1.Checked)

{

B[i] = 0;

dataGridView2.Rows[0].Cells[i].Value = B[i];

}

else

{

B[i] = (r.Next((int) numericUpDown4.Value, (int)numericUpDown6.Value));

dataGridView2.Rows[0].Cells[i].Value = B[i];

}

}

}

private void numericUpDown6\_ValueChanged(object sender, EventArgs e) //ограничение для задания случайных чисел

{

if(Convert.ToInt32(numericUpDown4.Value) > Convert.ToInt32(numericUpDown6.Value))

{

numericUpDown6.Value = numericUpDown4.Value;

}

}

private void button2\_Click(object sender, EventArgs e) //работа с файлами

{

if (radioButton6.Checked)

{

try

{

IntArray.ArrayToTextFile(B, textBox2.Text + ".txt");

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при записи в файл: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

if (radioButton5.Checked)

{

try

{

B = IntArray.ArrayFromTextFile(textBox2.Text + ".txt");

dataGridView2.RowCount = 1;

dataGridView2.ColumnCount = A.Length;

int n = B.Length;

dataGridView2.SetBounds(12, 217, 100 \* n + n, 60);

for (int i = 0; i < B.Length; i++)

{

dataGridView2.Columns[i].Name = i.ToString();

dataGridView2.Rows[0].Cells[i].Value = B[i];

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при чтении из файла: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void button1\_Click(object sender, EventArgs e) //работа с файлами

{

if (radioButton6.Checked)

{

try

{

IntArray.ArrayToTextFile(A, textBox1.Text + ".txt");

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при записи в файл: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

if (radioButton5.Checked)

{

try

{

A = IntArray.ArrayFromTextFile(textBox1.Text + ".txt");

dataGridView1.RowCount = 1;

dataGridView1.ColumnCount = A.Length;

int n = A.Length;

dataGridView1.SetBounds(12, 217, 100 \* n + n, 60);

for (int i = 0; i < A.Length; i++)

{

dataGridView1.Columns[i].Name = i.ToString();

dataGridView1.Rows[0].Cells[i].Value = A[i];

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при чтении из файла: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void numericUpDown4\_ValueChanged(object sender, EventArgs e) //нижняя граница случайного числа

{

if (Convert.ToInt32(numericUpDown4.Value) > Convert.ToInt32(numericUpDown6.Value))

{

numericUpDown4.Value = numericUpDown6.Value;

}

}

private void массивАToolStripMenuItem1\_Click(object sender, EventArgs e)//открытие вкладки для массива А

{

setInvisible();

isA = true;

dataGridView1.Visible = true;

button3.Visible = true;

button4.Visible = true;

button5.Visible = true;

button6.Visible = true;

label1.Visible = true;

label2.Visible = true;

numericUpDown7.Visible = true;

numericUpDown8.Visible = true;

label3.Visible = true;

}

private void массивБToolStripMenuItem1\_Click(object sender, EventArgs e) //открытие вкладки для массива Б

{

setInvisible();

isA = false;

dataGridView2.Visible = true;

button3.Visible = true;

button4.Visible = true;

button5.Visible = true;

button6.Visible = true;

label1.Visible = true;

label2.Visible = true;

numericUpDown7.Visible = true;

numericUpDown8.Visible = true;

label3.Visible = true;

}

private void button3\_Click(object sender, EventArgs e) //инкримент массивов

{

if(isA)

{

A++;

}

else

{

B++;

}

updateGrids();

}

private void button4\_Click(object sender, EventArgs e) //декримент массивов

{

if (isA)

{

A--;

}

else

{

B--;

}

updateGrids();

}

private void button5\_Click(object sender, EventArgs e) // сложение массива со скаляром

{

if (isA)

{

A += Convert.ToInt32(numericUpDown8.Value);

}

else

{

B += Convert.ToInt32(numericUpDown8.Value);

}

updateGrids();

}

private void button6\_Click(object sender, EventArgs e) // вывод информации по массиву(сумма элементов, размер и количество элементов кратных заданному числу

{

IntArray N;

if (isA)

{

N = A;

}

else

{

N = B;

}

int sum = 0, count = N.Length, crat = 0, num = Convert.ToInt32(numericUpDown7.Value);

if(num == 0)

{

label3.Text = "Числа нельзя проверить на кратность нулю";

return;

}

for (int i = 0; i < N.Length; i++)

{

sum += N[i]; ;

if (N[i] % num == 0)

crat++;

}

label3.Text = "Сумма элементов массива = " + sum + " , количество элементов = " + count + " , количество элементов кратных " + num + " = " + crat;

}

private void действиеНадОбоимиМассивамиToolStripMenuItem\_Click(object sender, EventArgs e) //открытие вкладки для действий с обоими массивами

{

setInvisible();

dataGridView1.Visible = true;

int n = Convert.ToInt32(numericUpDown1.Value);

dataGridView1.SetBounds(12, 148, 100 \* n + n, 60);

dataGridView2.Visible = true;

n = Convert.ToInt32(numericUpDown2.Value);

dataGridView2.SetBounds(12, 210, 100 \* n + n, 60);

dataGridView3.Visible = true;

dataGridView3.SetBounds(12, 280, 100 \* C.Length + C.Length, 60);

button7.Visible = true;

button8.Visible = true;

button9.Visible = true;

}

private void button9\_Click(object sender, EventArgs e) // индивидуальное задание

{

if (A.Length != 0 && B.Length != 0)

{

C = IntArray.Method(A, B);

dataGridView3.RowCount = 1;

dataGridView3.ColumnCount = C.Length;

int n = C.Length;

dataGridView3.SetBounds(12, 280, 100 \* n + n, 60);

for (int i = 0; i < C.Length; i++)

{

dataGridView3.Columns[i].Name = i.ToString();

dataGridView3.Rows[0].Cells[i].Value = C[i];

}

}

}

private void button7\_Click(object sender, EventArgs e) // сложение массивов

{

dataGridView3.RowCount = 1;

dataGridView3.ColumnCount = A.Length;

int n = A.Length;

C.Length = Math.Max(A.Length, B.Length);

dataGridView3.SetBounds(12, 280, 100 \* n + n, 60);

for (int i = 0; i < C.Length; i++)

{

if(A.Length > i)

C[i] += A[i];

if (B.Length > i)

C[i] += B[i];

dataGridView3.Columns[i].Name = i.ToString();

dataGridView3.Rows[0].Cells[i].Value = C[i];

}

}

private void button8\_Click(object sender, EventArgs e) //вычитание массивов

{

dataGridView3.RowCount = 1;

dataGridView3.ColumnCount = A.Length;

int n = A.Length;

C.Length = Math.Max(A.Length, B.Length);

dataGridView3.SetBounds(12, 280, 100 \* n + n, 60);

for (int i = 0; i < C.Length; i++)

{

if (A.Length > i)

C[i] = A[i];

if (B.Length > i)

C[i] -= B[i];

dataGridView3.Columns[i].Name = i.ToString();

dataGridView3.Rows[0].Cells[i].Value = C[i];

}

}

private void numericUpDown3\_ValueChanged(object sender, EventArgs e) // нижняя граница для случайного числа

{

if (Convert.ToInt32(numericUpDown3.Value) > Convert.ToInt32(numericUpDown5.Value))

{

numericUpDown3.Value = numericUpDown5.Value;

}

}

private void numericUpDown5\_ValueChanged(object sender, EventArgs e)// верхняя граница случайного числа

{

if (Convert.ToInt32(numericUpDown3.Value) > Convert.ToInt32(numericUpDown5.Value))

{

numericUpDown5.Value = numericUpDown3.Value;

}

}

}

}

**Форма:**

1.Запуск Формы

При запуске формы на экране появилась возможность взаимодействия с массивом А

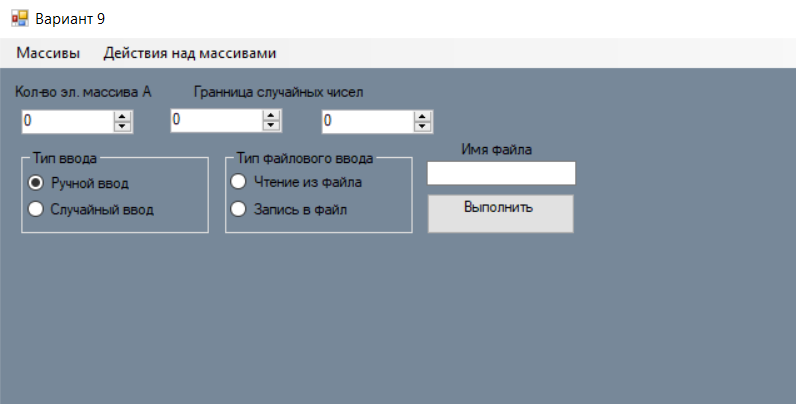


Рис.16 запуск формы

2.Переключение между массивами

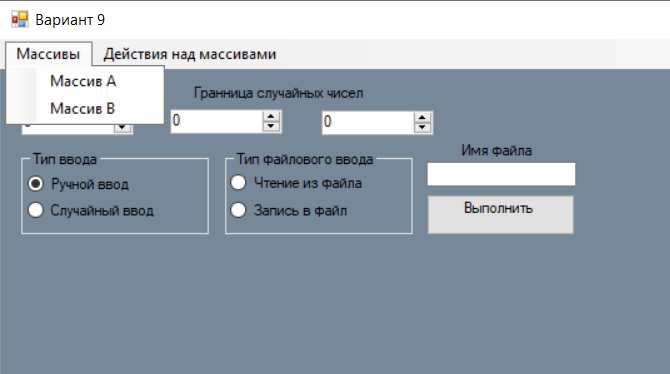


Рис.17 выбор массива

3.Меню для выбора действий над одним из массивов или сразу над двумя

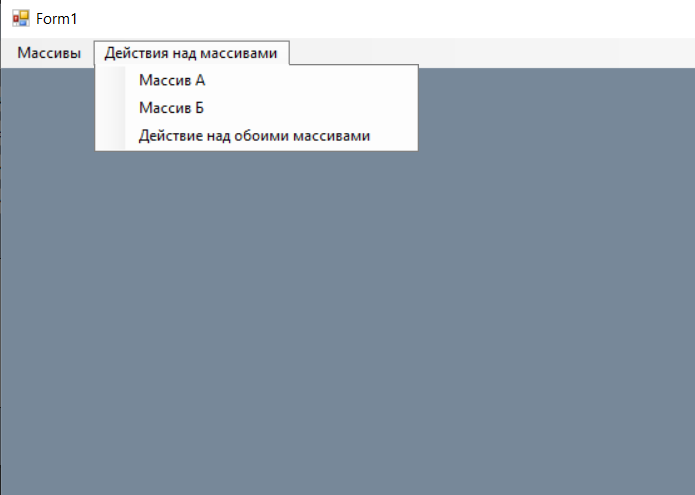
****

Рис.18 выбор одного или сразу 2 массивов для работы с ними.

4.Работа ручного режима ввода

Сначала надо выбрать режим заполнения (по умолчанию стоит ручной ввод), затем задать количество элементов в массиве. При выборе типа ввода “ручной” создаться массив, в котором все элементы будут равны 0. Если нажать мышкой на ячейку с элементом массива, то ег0 можно будет перезаписать с клавиатуры.

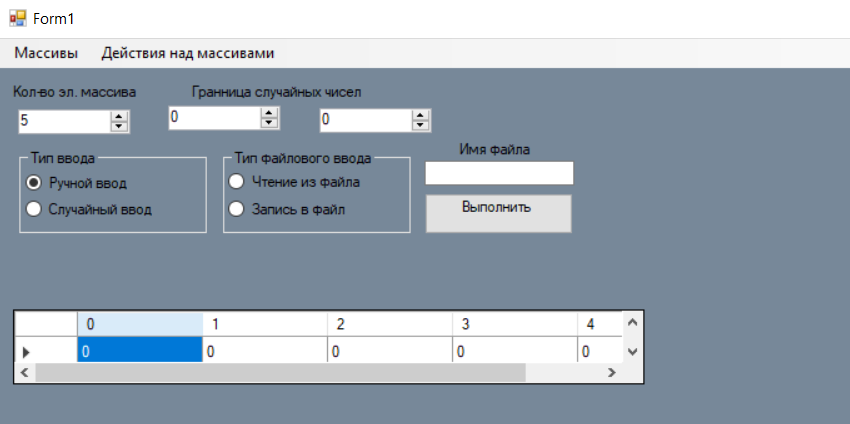


Рис.19 ручной ввод элементов массива

5.Работа случайного заполнения

Сначала надо выбрать режим заполнения (по умолчанию стоит ручной ввод), затем задать количество элементов в массиве. При выборе типа ввода “случайный” необходимо так же указать диапазон случайных чисел.

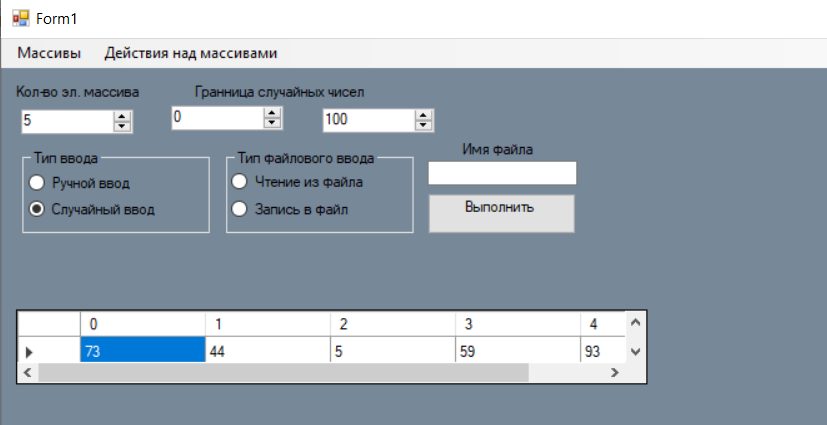


Рис.20 заполнения массива случайными числами.

6.Чтение из файла

Для чтения массива из файла необходимо выбрать тип файлового ввода – чтение из файла. Затем в поля для ввода имени файла написать название файла (без расширения). Файл должен находится в папке Debug приложения (путь к папке: \WindowsFormsApp1\bin\Debug).

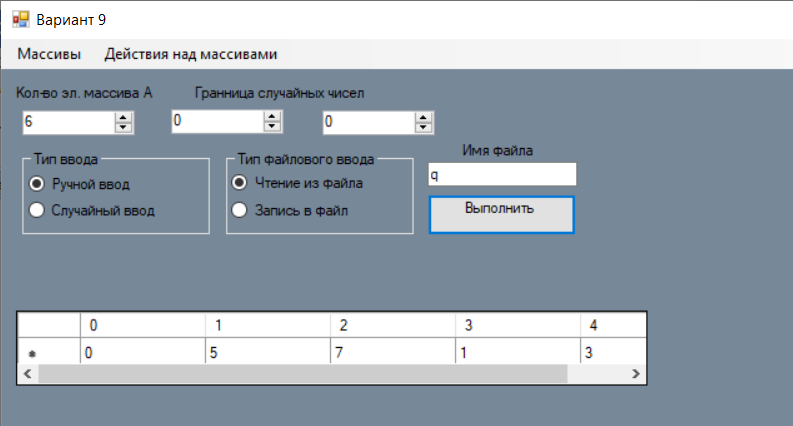


Рис.21 ввод массива из файла

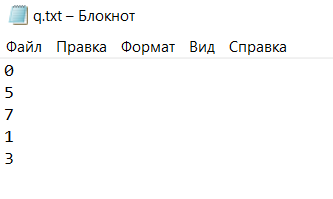


Рис.22 содержимое файла q.

7.Сохранение в файл

Для сохранения массива в файл необходимо сначала создать массив любым способом. Затем выбрать тип файлового ввода-запись в файл, после чего ввести название файла в поле “Имя файла” и нажать кнопку выполнить.

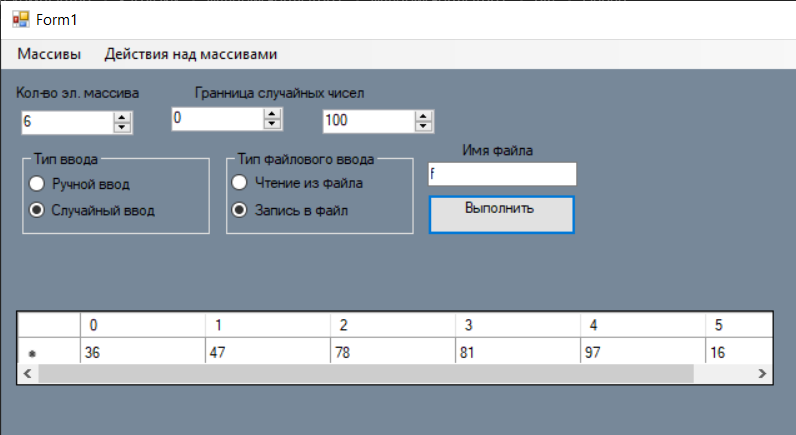


Рис.23 Запись массива в файл.

Файл f появился в Debug(путь к этой папке: \WindowsFormsApp1\bin\Debug) приложения.

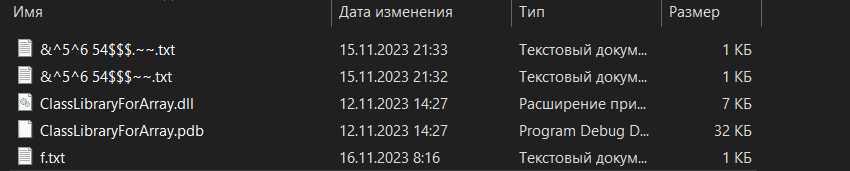


Рис.23 содержимое папки Debug.

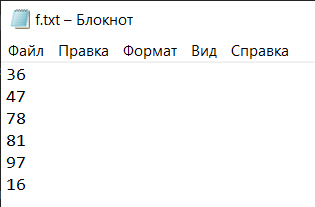


Рис.24 Содержимое файла f

8.Действие над одним массивом

При выборе действия над одним из массивов, на экране появится варианты работы с этим массивом и сам массив.

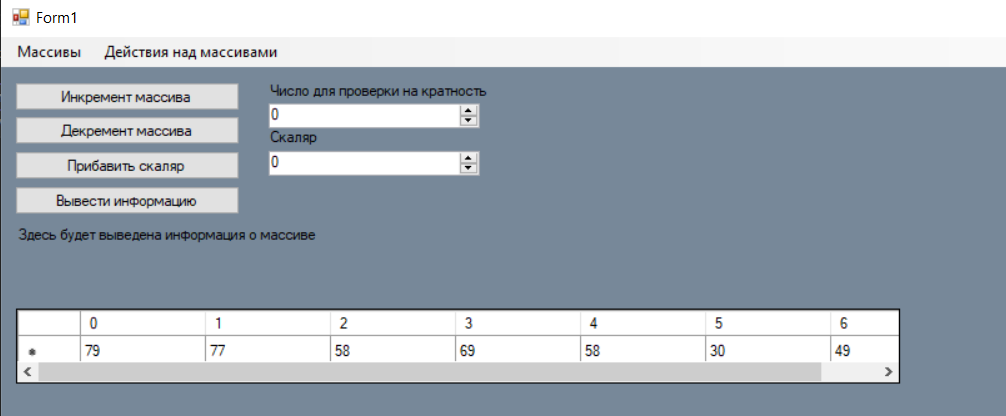


Рис.25 форма действий над одним массивом

9.Инкримент массива

При нажатии на кнопку “Инкремент массива” каждый из элементов массива увеличиться на 1.

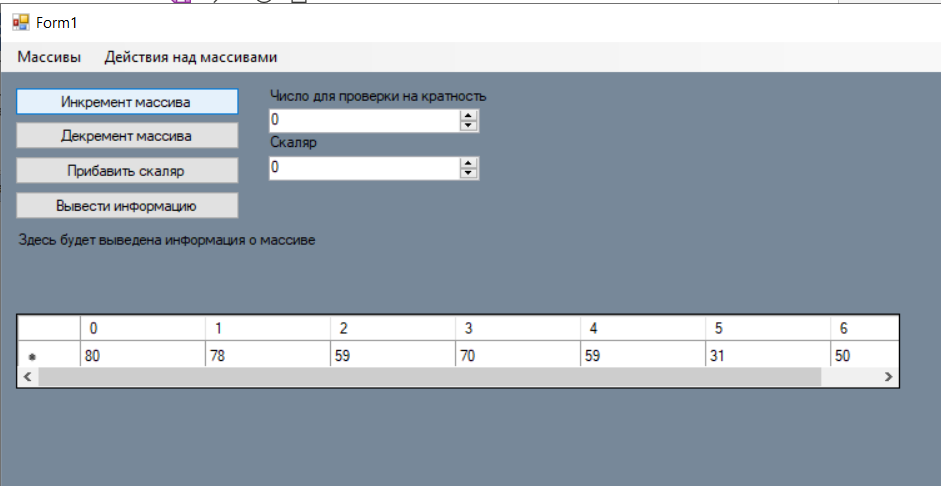


Рис.26 работа инкремента элементов массива.

10.Декримент массива

При нажатии на кнопку “Декремент массива” все элементы текущего массива уменьшаться на 1.

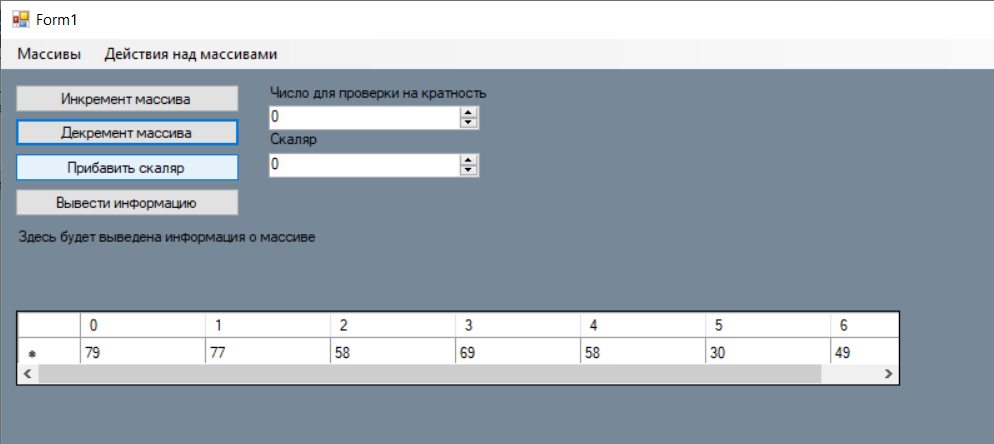


Рис.27 Декремент элементов массива

11.Сложение со скаляром

При нажатии на кнопку “Сложение со скаляром” все элементы текущего массива уменьшаться на заданное число в поле “Скаляр”.

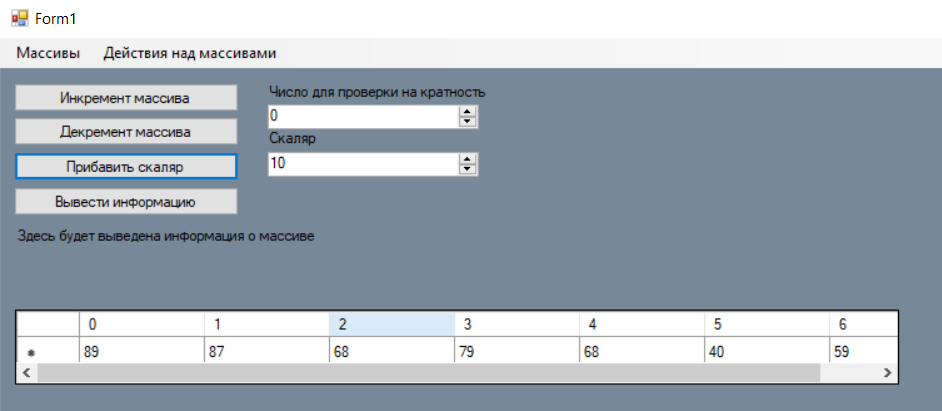


Рис.28 прибавление скаляра к элементам массива.

12.Вывод информации по массиву

При нажатии на кнопку “Вывести информацию” на экране появится: сумма элементов массива, количество элементов и количество чисел кратных заданному (это число задается в поле “Число для проверки кратности”).

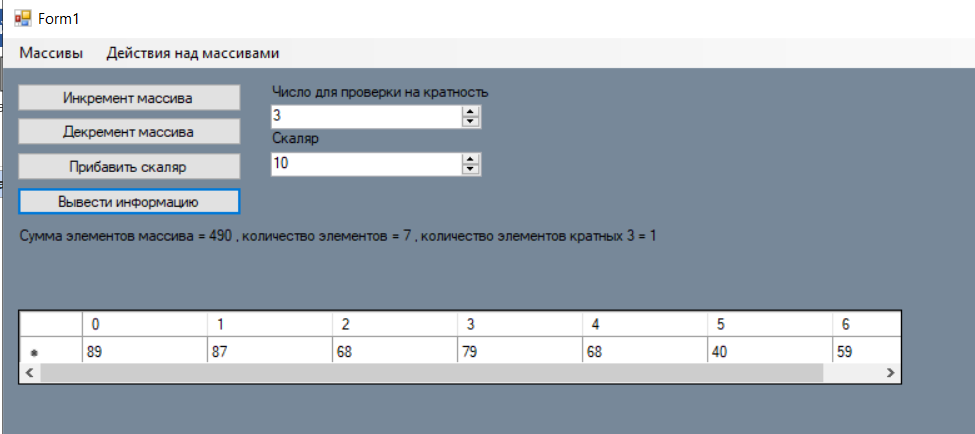


Рис.29 вывод информации по массиву.

13.Действия над обоими массивами

При выборе действия над обоими массивами, на экране появится варианты работы с этим массивами и сам массивы.

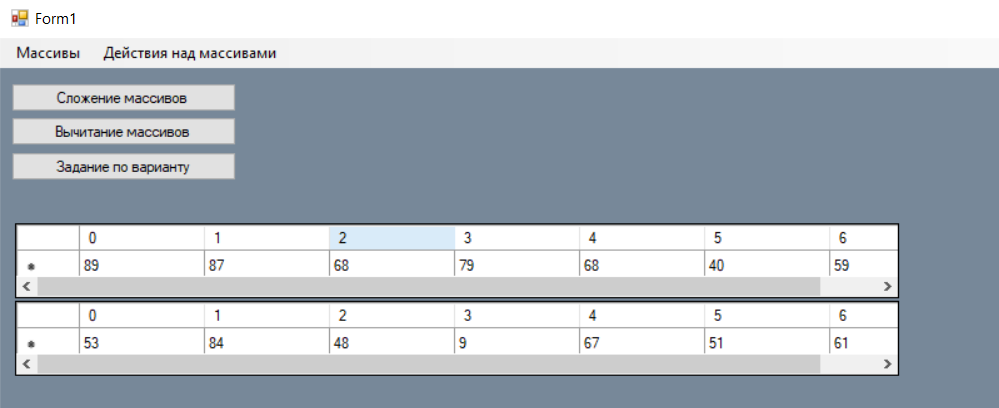


Рис.30 действия над 2 массивами.

14.Сложение массивов

При нажатии на кнопку “Сложение массивов ” элементы 2 массивов сложатся и создадут новый массив.

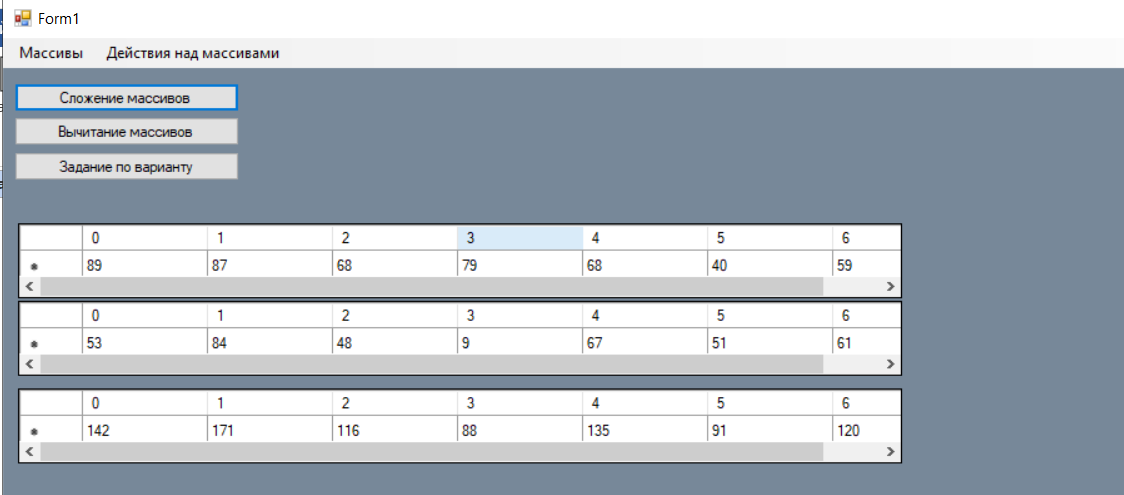


Рис.31 сложение массивов

15.Вычитание массивов

При нажатии на кнопку “Вычитание массивов ” элементы 2 массивов будут вычитаться и создадут новый массив.

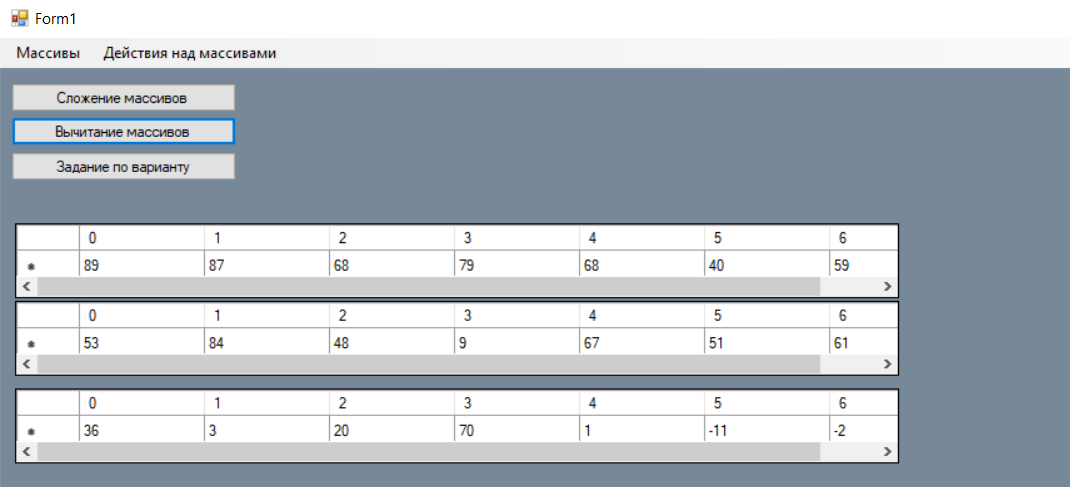


Рис.32 разность массивов

16.Задание оп варианту

При нажатии на кнопку “Задание по варианту” новый массив будет состоять из сумма перемноженных пар соседних элементов двух одномерных массивов А и В.

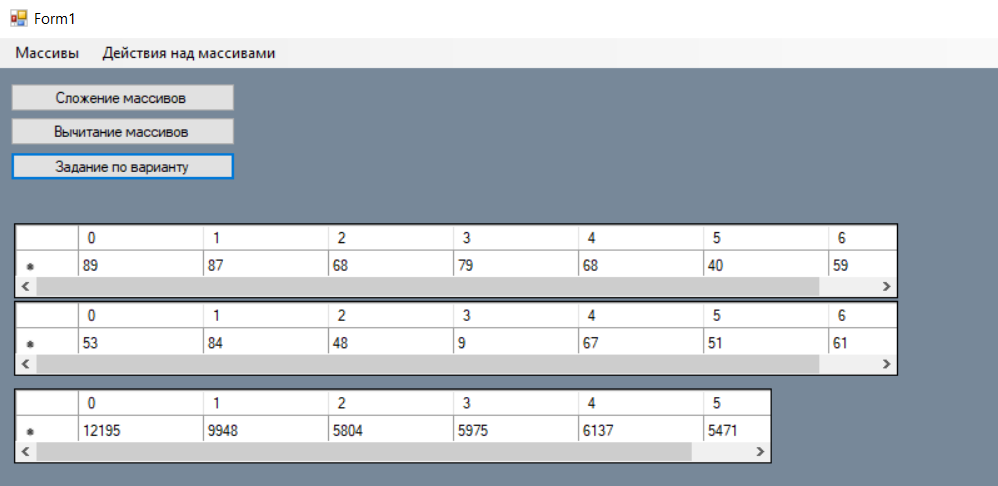


Рис. 33 задание по варианту