**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 №3**

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**

[**Цель работы** 3](#_Toc118149827)

[**Задание** 3](#_Toc118149828)

[**Математическая формулировка методов** 3](#_Toc118149829)

[**Текст программных модулей** 7](#_Toc118149830)

[**Протокол испытаний** 11](#_Toc118149831)

[**Результаты выполнения работы программы** 13](#_Toc118149832)

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

**Вариант №9**

**Часть 1.**

# **Цель работы:**

Изучение наследования, абстрактных классов, виртуальных методов и тд и применение их на практики

# **Задание:**

Создать базовый класс Note (запись), содержащий поля: ФИО человека, номер телефона. Создать производный класс Friend (друг), содержащий дополнительные поля: электронная почта, дата рождения. Описать методы для вывода на экран записей, получения и установки значений некоторых полей. Описать класс AdressBook (телефонная книга), содержащий включаемое поле массив записей Note. Создать метод добавления записи в телефонную книгу, метод удаления записи из телефонной книги, а также метод вывода всего списка на экран. В основной программе продемонстрировать создание телефонной книги.

# **Математическая формулировка методов:**

**Класс Note**

1. Конструктор по умолчанию

public Note() { }

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

1. Конструктор включающий в себя инициализацию полей ФИО и номера телефона

public Note(string fio, string phoneNumber\_)

{

FIO\_ = fio;

PhoneNumber = phoneNumber\_;

}

Данный конструктор принимает значения ФИО и номера телефона и устанавливает значение соответствующего поля. Параметры конструктора заданы типом string.

1. Метод для вывода ФИО и номера телефона

public virtual void display()

{

Console.WriteLine(FIO + " " + PhoneNumber);

}

Данный виртуальный метод выводит информацию об экземпляре класса на экран .

1. Метод Equals

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

Note otherNote = (Note)obj;

return this.GetHashCode() == obj.GetHashCode();

}

Данный переопределенный метод позволяет проверить на равенство два объекта.

1. Метод GetHashCode

public override int GetHashCode()

{

unchecked

{

var result = 1;

result = (result \* FIO.GetHashCode()) ^ 10;

result = (result \* PhoneNumber.GetHashCode()) ^ 100;

return result;

}

}

Данный переопределенный метод позволяет сгенерировать хэшкод для объекта по нескольким полям.

**Класс Friend**

1. Конструктор по умолчанию

public Friend() { }

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

1. Конструктор включающий в себя инициализацию полей ФИО и номера телефона, почты и даты рождения

public Friend(string fio, string phoneNumber\_, string email\_, DateTime birthdayDate\_) : base(fio, phoneNumber\_)

{

Email = email\_;

BirthdayDate = birthdayDate\_;

}

Данный конструктор принимает значения ФИО и номера телефона , почты и даты рождения и устанавливает значение соответствующего поля, так же вызывая конструктор базового класса. Параметры конструктора заданы типом string.

1. Метод для вывода информации об объекте

public override void display()

{

Console.WriteLine(FIO + " " + PhoneNumber + " " + Email + " " + BirthdayDate.ToString("dd.MM.yyyy"));

}

Данный переопределенный метод выводит информацию об экземпляре класса на экран .

1. Метод Equals

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

if (!base.Equals(obj))

{

return false;

}

Friend otherFriend = (Friend)obj;

return this.GetHashCode() == obj.GetHashCode();

}

Данный переопределенный метод позволяет проверить на равенство два объекта.

1. Метод GetHashCode

public override int GetHashCode()

{

unchecked

{

var result = 1;

result = (result \* FIO.GetHashCode()) ^ 10;

result = (result \* PhoneNumber.GetHashCode()) ^ 100;

result = (result \* BirthdayDate.GetHashCode()) ^ 1000;

result = (result \* Email.GetHashCode()) ^ 10000;

return result;

}

}

Данный переопределенный метод позволяет сгенерировать хэшкод для объекта по нескольким полям.

**Класс AddressBook**

1. Конструктор по умолчанию

public AddressBook()

{

notes = new List<Note>();

}

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

1. Конструктор включающий в себя инициализацию полей ФИО и номера телефона, почты и даты рождения

public Friend(string fio, string phoneNumber\_, string email\_, DateTime birthdayDate\_) : base(fio, phoneNumber\_)

{

Email = email\_;

BirthdayDate = birthdayDate\_;

}

Данный конструктор принимает значения ФИО и номера телефона , почты и даты рождения и устанавливает значение соответствующего поля, так же вызывая конструктор базового класса. Параметры конструктора заданы типом string.

1. Метод для проверки наличия записи по ФИО

private bool NoteExistsFIO(string fio)

{

return notes.Any(note => note.FIO == fio);

}

Данный метод принимает на вход параметр в виде строки и проверяет существует ли запись к таким именем в списке

1. Метод для проверки наличия записи по Номеру телефона

private bool NoteExistsFIO(string fio)

{

return notes.Any(note => note.FIO == fio);

}

Данный метод принимает на вход параметр в виде строки и проверяет существует ли запись к таким именем в списке

1. Метод добавления записи

public void AddNote(Note note)

{

try

{

if (NoteExistsFIO(note.FIO) || NoteExistsPhoneNum(note.PhoneNumber))

{

throw new Exception("Запись с таким именем или номером телефона уже существует.");

}

if (!IsValidEmail(note.Email))

{

throw new Exception("Адресс электронной почты не валиден.");

}

notes.Add(note);

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

Данный метод принимает на вход запись которую надо добавить, проверяет ее на корректность и добавляет ее.

1. Метод удаления записи

public void RemoveNote(Note note)

{

try

{

if (!NoteExistsFIO(note.FIO))

{

throw new Exception("Запись не найдена.");

}

notes.Remove(note);

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

Данный метод принимает на вход запись которую надо удалить и удаляет ее.

1. Метод для вывода всех записей на экран

public void DisplayAllNotes()

{

foreach (Note note in notes)

{

note.display();

}

}

Данный метод выводит все записи на экран

1. Метод для проверки на корректность

static bool IsValidEmail(string email)

{

string pattern = @"^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$";

return Regex.IsMatch(email, pattern);

}

Данный метод проверяет почту на корректность с помощью регулярного выражения. Принимает на вход строку.

**Текст программных модулей:**

// библиотечный модуль \*.dll

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClassLibrary1

{

public class Note

{

private string FIO\_;

private string phoneNumber;

public string FIO

{

get { return FIO\_; }

set { FIO\_ = value; }

}

public string PhoneNumber

{

get { return phoneNumber; }

set { phoneNumber = value; }

}

public virtual void display()

{

Console.WriteLine(FIO + " " + PhoneNumber);

}

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

Note otherNote = (Note)obj;

return FIO == otherNote.FIO && PhoneNumber == otherNote.PhoneNumber;

}

public Note() { }

public Note(string fio, string phoneNumber\_)

{

FIO\_ = fio;

PhoneNumber = phoneNumber\_;

}

}

public class Friend : Note

{

private string email;

private DateTime birthdayDate;

public string Email

{

get { return email; }

set { email = value; }

}

public DateTime BirthdayDate

{

get { return birthdayDate; }

set { birthdayDate = value; }

}

public override void display()

{

Console.WriteLine(FIO + " " + PhoneNumber + " " + Email + " " + BirthdayDate.ToString("dd.MM.yyyy"));

}

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

if (!base.Equals(obj))

{

return false;

}

Friend otherFriend = (Friend)obj;

return Email == otherFriend.Email && BirthdayDate == otherFriend.BirthdayDate;

}

public Friend() { }

public Friend(string fio, string phoneNumber\_, string email\_, DateTime birthdayDate\_) : base(fio, phoneNumber\_)

{

Email = email\_;

BirthdayDate = birthdayDate\_;

}

}

public class AddressBook

{

private List<Note> notes;

private bool NoteExists(string fio)

{

return notes.Any(note => note.FIO == fio);

}

public void AddNote(Note note)

{

try

{

if (NoteExists(note.FIO))

{

throw new InvalidOperationException("Запись с таким именем уже существует.");

}

notes.Add(note);

}

catch (InvalidOperationException ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

public void RemoveNote(Note note)

{

try

{

if (!NoteExists(note.FIO))

{

throw new InvalidOperationException("Запись не найдена.");

}

notes.Remove(note);

}

catch (InvalidOperationException ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

public void DisplayAllNotes()

{

foreach (Note note in notes)

{

note.display();

}

}

public AddressBook()

{

notes = new List<Note>();

}

}

}

// код программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

using ClassLibrary1;

namespace ConsoleApp1

{

internal class Program

{

static void Main(string[] args)

{

AddressBook addressBook = new AddressBook();

Friend friend1 = new Friend("Antony Zorikov", "123456789", "Antony@example.com", new DateTime(2005, 9, 12));

Friend friend2 = new Friend("Matvey Leiko", "987654321", "Matvey@example.com", new DateTime(2005, 11, 17));

Friend friend3 = new Friend("Antony Zorikov", "123456789", "Antony@example.com", new DateTime(2005, 9, 12));

addressBook.AddNote(friend1);

addressBook.AddNote(friend2);

addressBook.AddNote(friend3);

addressBook.DisplayAllNotes();

addressBook.RemoveNote(friend1);

Console.WriteLine(" ");

addressBook.DisplayAllNotes();

Console.WriteLine(" ");

addressBook.RemoveNote(friend1);

Console.ReadLine();

}

}

}

# **Протокол испытаний:**

*Таблица 1.* Протокол испытаний

|  |  |  |  |
| --- | --- | --- | --- |
| **Проверяемые требования** | **Сообщения программы и вводимые значения** | **Ожидаемые результаты** | **Фактические результаты** |
| Проверка создания записей | Friend friend1 = new Friend("Antony Zorikov", "123456789", "Antony@example.com", new DateTime(2005, 9, 12));  Friend friend2 = new Friend("Matvey Leiko", "987654321", "Matvey@example.com", new DateTime(2005, 11, 17));  Friend friend3 = new Friend("Karl Zorikov", "223456789", "Karl@example.com", new DateTime(2005, 9, 12)); | Сообщение c выводом всех записей. | Antony Zorikov 123456789 Antony@example.com 12.09.2005  Matvey Leiko 987654321 Matvey@example.com 17.11.2005  Karl Zorikov 323456789 Antony@example.com 12.09.2005 |
| Проверка создания записей с одинаковыми именами. | friend1 = new Friend("Antony Zorikov", "123456789", "Antony@example.com", new DateTime(2005, 9, 12));  friend2 = new Friend("Matvey Leiko", "987654321", "Matvey@example.com", new DateTime(2005, 11, 17));  friend3 = new Friend("Antony Zorikov", "223456789", "Karl@example.com", new DateTime(2005, 9, 12)); | Сообщение, с выводом 2 записей и 1 ошибки(“Запись с таким именем уже существует.”). | Error: Запись с таким именем уже существует.  Antony Zorikov 123456789 Antony@example.com 12.09.2005  Matvey Leiko 987654321 Matvey@example.com 17.11.2005 |
| Проверка создания записей с одинаковыми номерами. | friend1 = new Friend("Antony Zorikov", "123456789", "Antony@example.com", new DateTime(2005, 9, 12));  friend2 = new Friend("Matvey Leiko", "987654321", "Matvey@example.com", new DateTime(2005, 11, 17));  friend3 = new Friend("Karl Zorikov", "123456789", "Karl@example.com", new DateTime(2005, 9, 12)); | Сообщение, с выводом 2 записей и 1 ошибки(“Запись с таким номером телефона уже существует.”). | Error: Запись с таким номером телефона уже существует.  Antony Zorikov 123456789 Antony@example.com 12.09.2005  Matvey Leiko 987654321 Matvey@example.com 17.11.2005 |
| Проверка создания записей с одинаковыми email. | friend1 = new Friend("Antony Zorikov", "123456789", "Antony@example.com", new DateTime(2005, 9, 12));  friend2 = new Friend("Matvey Leiko", "987654321", "Matvey@example.com", new DateTime(2005, 11, 17));  friend3 = new Friend("Karl Zorikov", "223456789", "Antony@example.com", new DateTime(2005, 9, 12)); | Сообщение, с выводом 2 записей и 1 ошибки(“Запись с таким email уже существует.”). | 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).

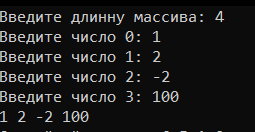


рис.1 Вывод времени при вводе корректных значений

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



рис.2 Вывод исключения при некорректном вводе длины массива

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

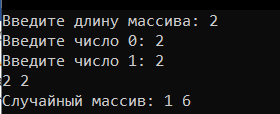


рис.3 Вывод случайного массива

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

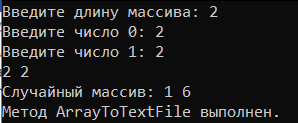


рис.4 Вывод сообщения о выполнении метода.

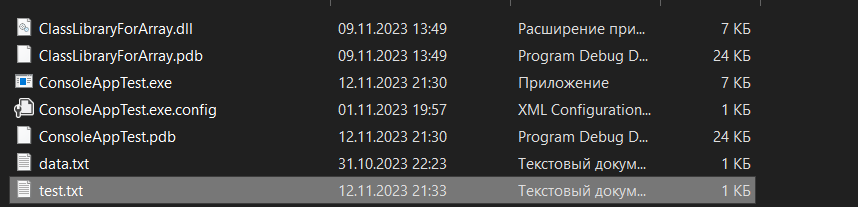


рис.5 Создался фаил с массивом

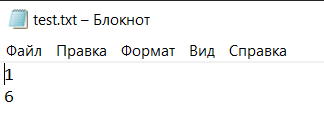


рис.6 Содержимое созданного файла

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

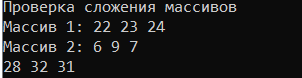


рис.11 сложение массивов

1. Проверка вычитания скаляра из массива



рис.12 вычитание скаляра из массива

1. Проверка вычетания массива из скаляра

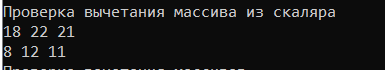


рис 13. Вычитание массива из скаляра

1. Проверка вачетания массивов

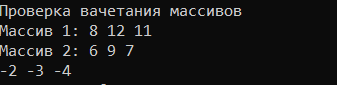


рис.15 вычитание массивов

**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;

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 Form1\_Load(object sender, EventArgs e)

{

}

private void contextMenuStrip1\_Opening(object sender, CancelEventArgs e)

{

}

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 dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

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 dataGridView2\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

}

private void radioButton2\_CheckedChanged(object sender, EventArgs e)

{

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

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 radioButton4\_CheckedChanged(object sender, EventArgs e)

{

}

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 radioButton6\_CheckedChanged(object sender, EventArgs e)

{

}

private void radioButton7\_CheckedChanged(object sender, EventArgs e)

{

}

private void radioButton8\_CheckedChanged(object sender, EventArgs e)

{

}

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 menuStrip1\_ItemClicked(object sender, ToolStripItemClickedEventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

}

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 numericUpDown8\_ValueChanged(object sender, EventArgs e)

{

}

private void label3\_Click(object sender, EventArgs e)

{

}

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 dataGridView3\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

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 radioButton5\_CheckedChanged(object sender, EventArgs e)

{

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

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;

}

}

private void label6\_Click(object sender, EventArgs e)

{

}

private void label4\_Click(object sender, EventArgs e)

{

}

private void label7\_Click(object sender, EventArgs e)

{

}

}

}

**Форма:**

1.Запуск Формы

При запуске формы на экране появилась возможность взаимодействия с массивом А

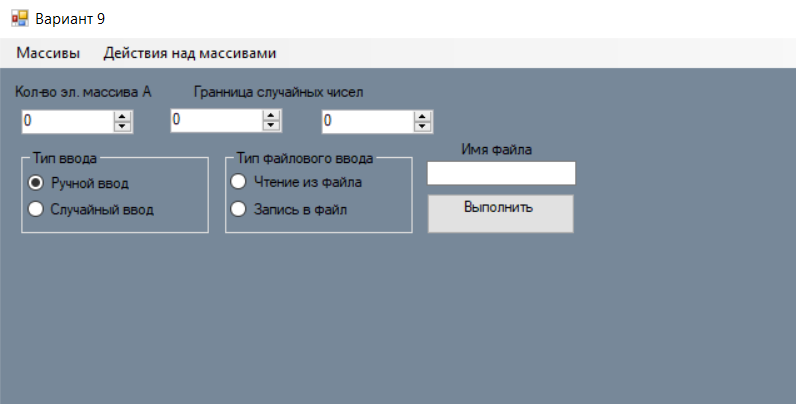


Рис.16 запуск формы

2.Переключение между массивами

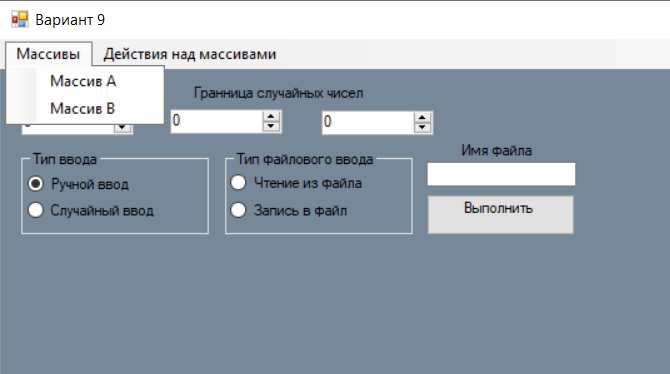


Рис.17 выбор массива

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

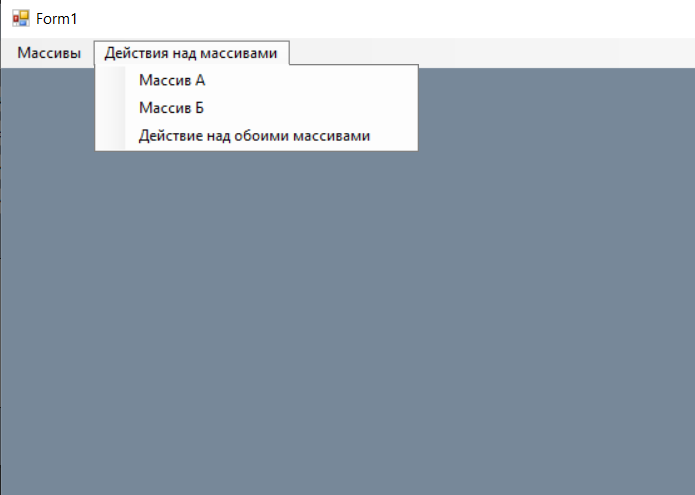
****

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.

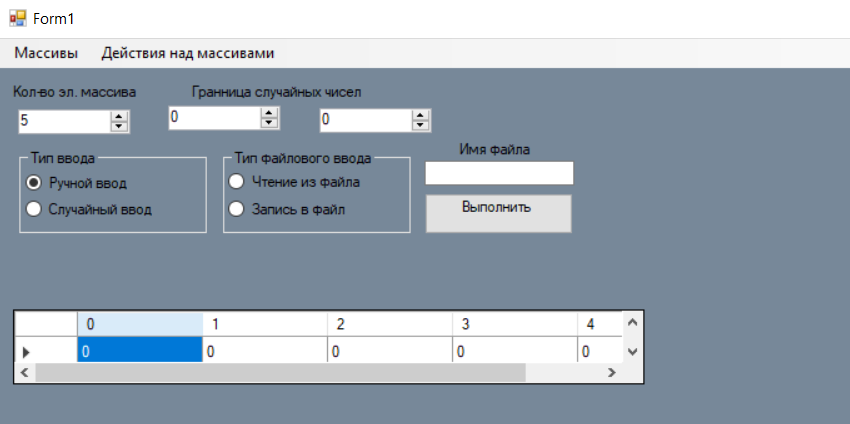


Рис.19 ручной ввод элементов массива

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.

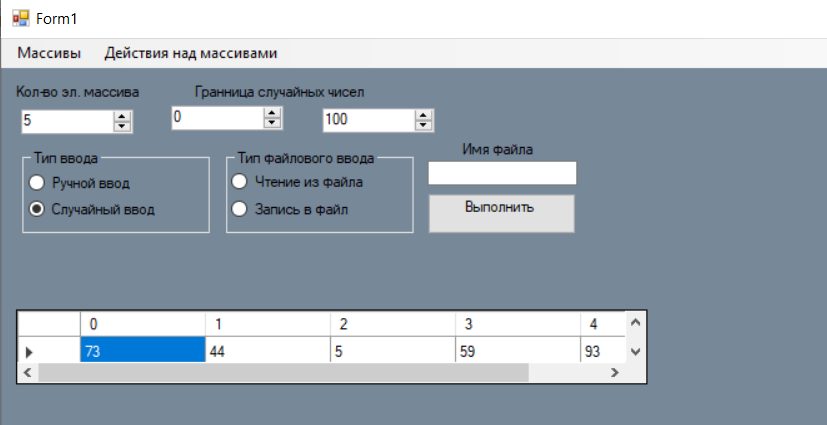


Рис.20 заполнения массива случайными числами.

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).

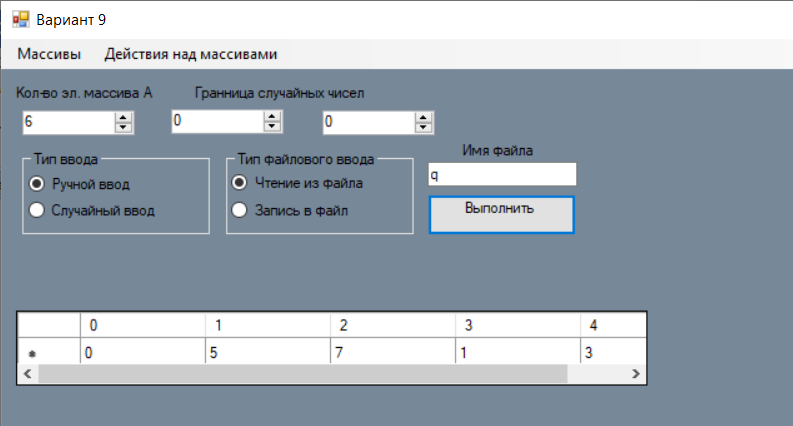


Рис.21 ввод массива из файла

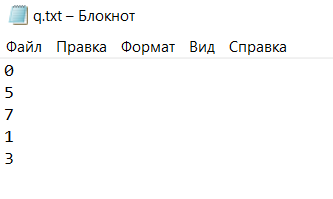


Рис.22 содержимое файла q.

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.

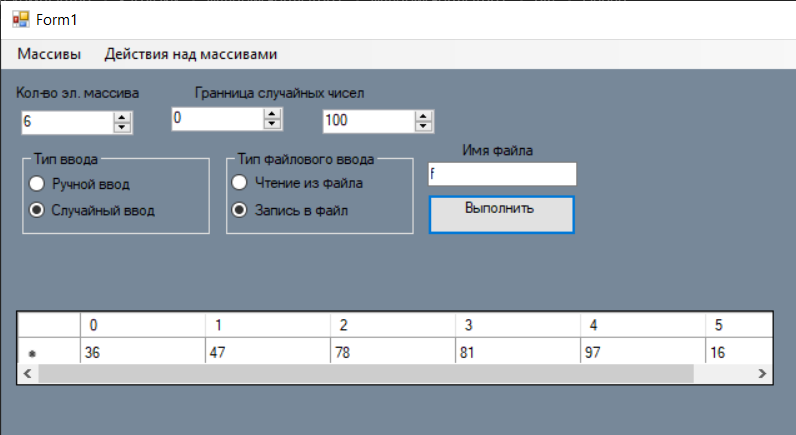


Рис.23 Запись массива в файл.

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

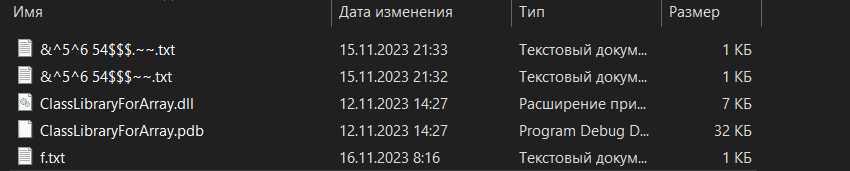


Рис.23 содержимое папки Debug.

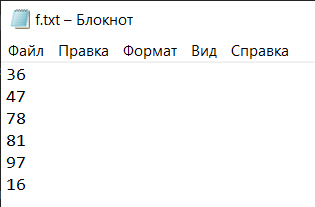


Рис.24 Содержимое файла f

8.Действие над одним массивом

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

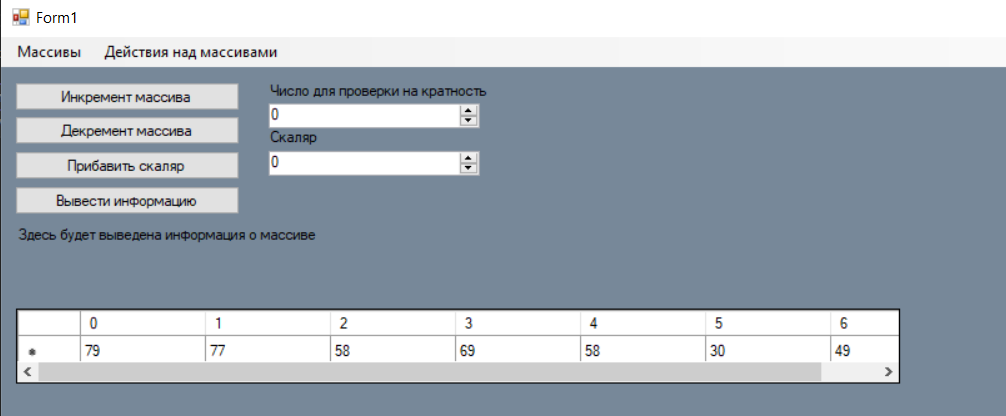


Рис.25 форма действий над одним массивом

9.Инкримент массива

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

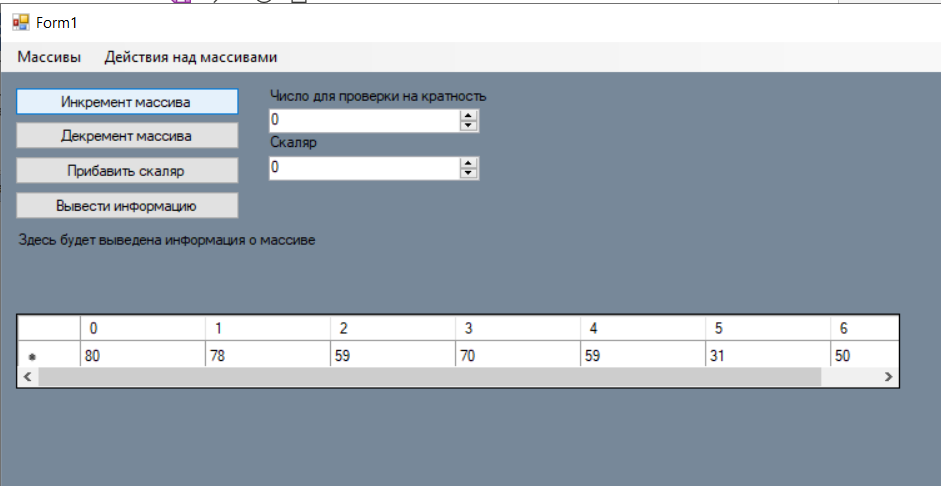


Рис.26 работа инкремента элементов массива.

10.Декримент массива

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

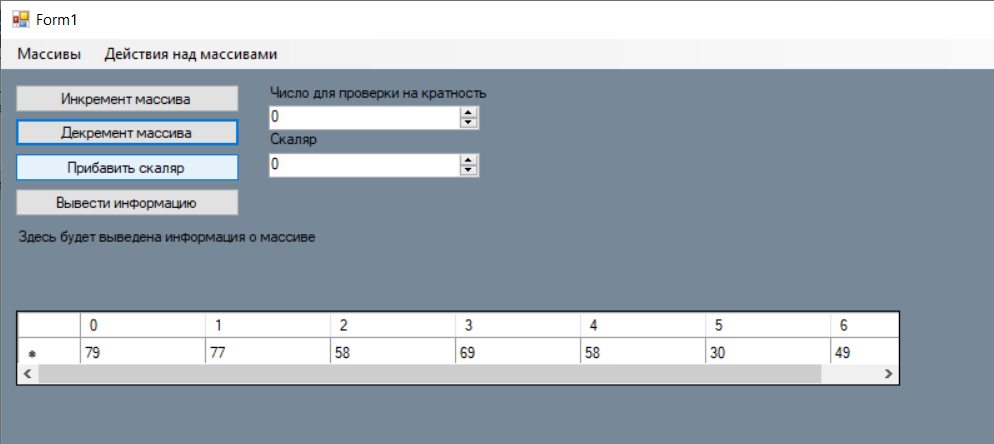


Рис.27 Декремент элементов массива

11.Сложение со скаляром

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

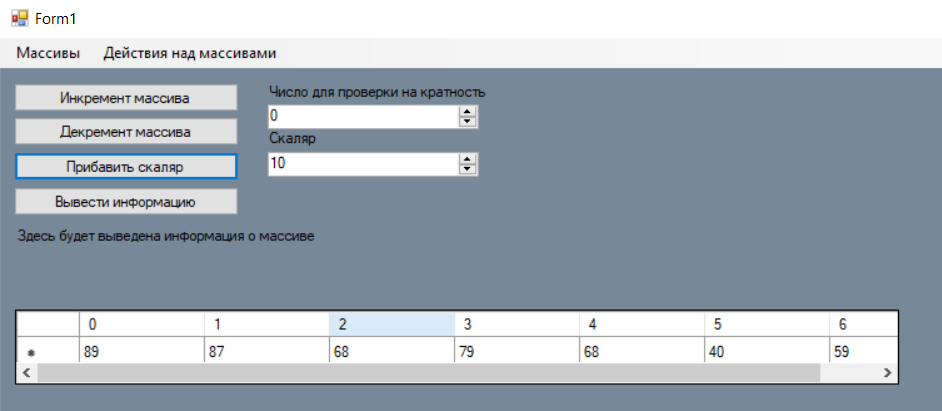


Рис.28 прибавление скаляра к элементам массива.

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”).

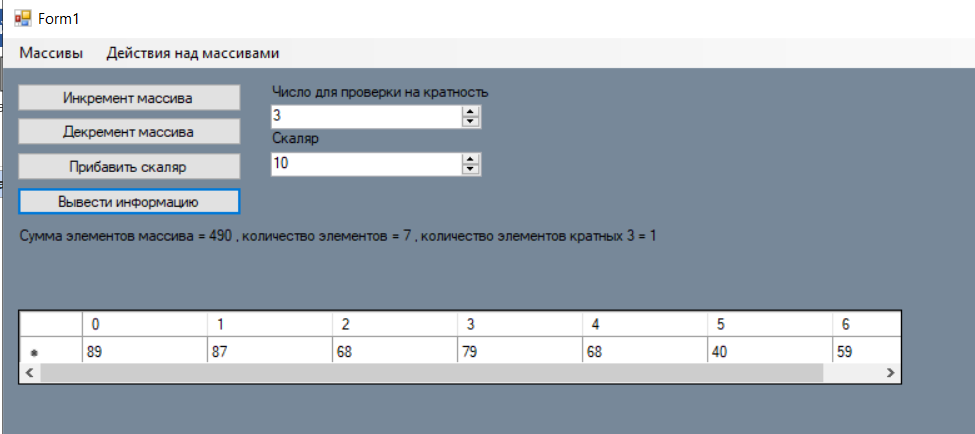


Рис.29 вывод информации по массиву.

13.Действия над обоими массивами

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

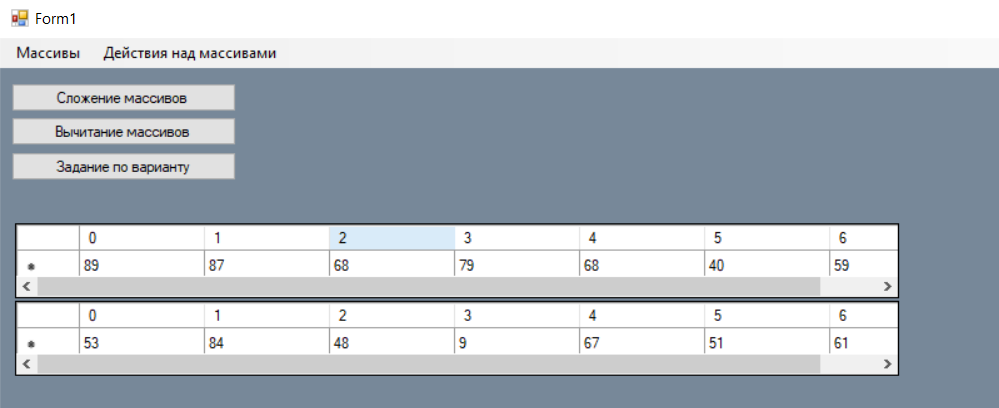


Рис.30 действия над 2 массивами.

14.Сложение массивов

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

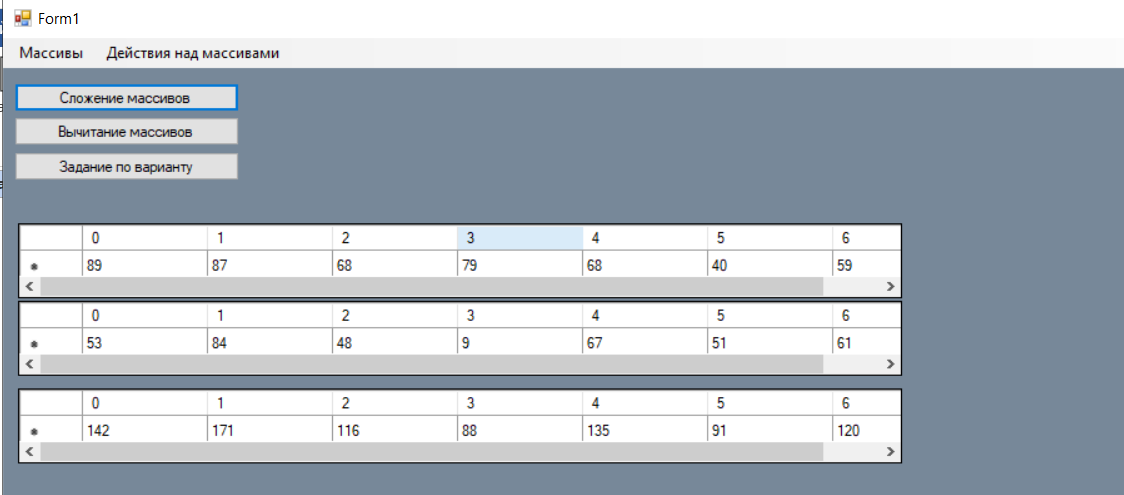


Рис.31 сложение массивов

15.Вычитание массивов

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

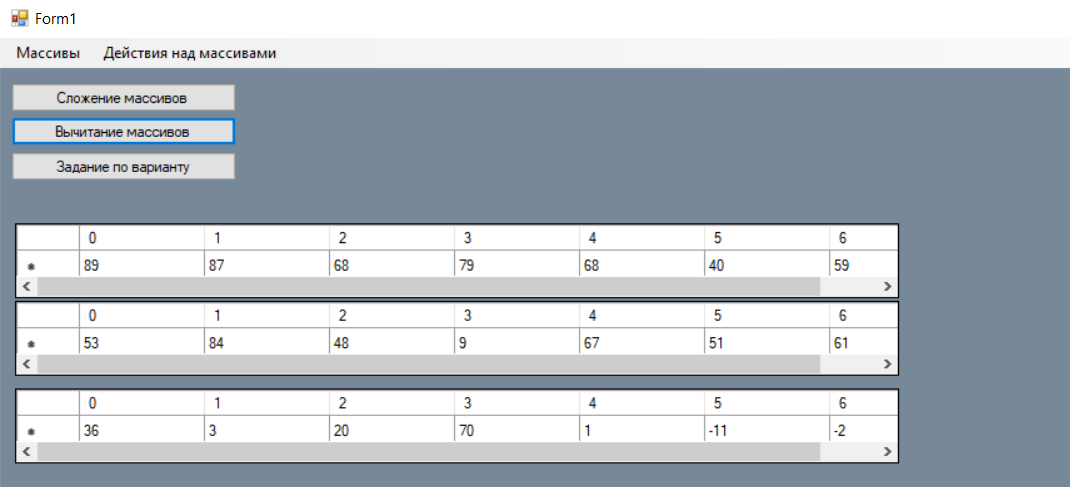


Рис.32 разность массивов

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.

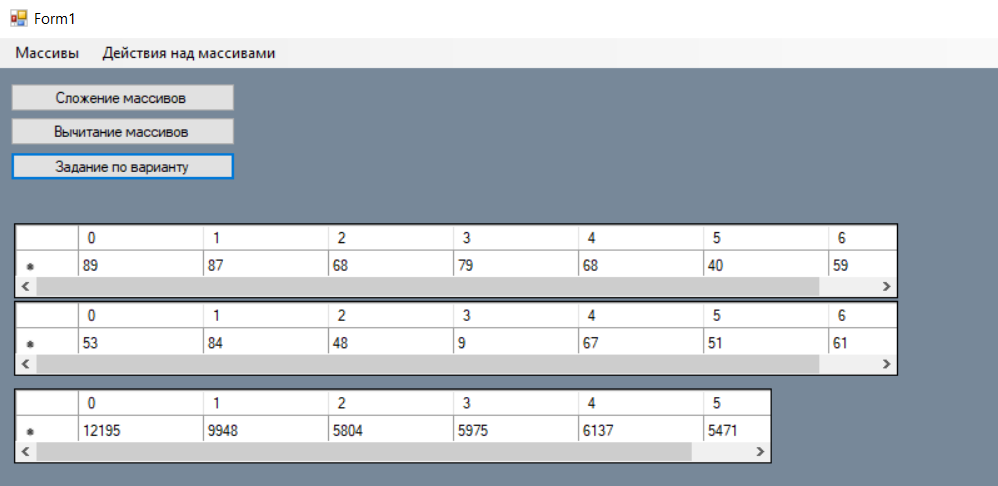


Рис. 33 задание по варианту