Минобрнауки России

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт

(технический университет)»

­­­­

**Дисциплина «Разработка ПС»**

Отчёт по лабораторной работе №1.

Использование массивов

Разработка программной системы для работы с массивом

Преподаватели: Корниенко Иван Григорьевич

Федин Алексей Константинович

Исполнил студент 494 группы: Гусев Антон Александрович

Санкт-Петербург

2021

Постановка задачи

Необходимо определить число отрицательных элементов, расположенных перед наибольшим положительным элементом одномерного массива, размер которого М.

Необходимо выделить массив требуемого размера, запросив его у пользователя в процессе работы программы или считав из файла. В программе должны быть предусмотрены три варианта заполнения исходного массива: пользователем с клавиатуры, из файла и случайными числами. В работе должна присутствовать возможность сохранения исходных данных и сохранения результата работы программы, а также модульного тестирования.

Исходные данные

В качестве исходных данных программа использует: пользовательский ввод чисел в специальное поле для ввода; текстовые файлы формата «txt», в которых хранится массив поэлементно.

Массив, который ввел пользователь, является набором целочисленных переменных (тип int в C#). Результат представляет собой целочисленную переменную (тип int в C#).

Особые ситуации

Необходимо рассмотреть следующие особые ситуации:

1. Некорректный ввод элементов массива (символы, буквы, лишних пробелов)
2. Отсутствие ожидаемых программой файлов на чтение или содержание некорректных данных внутри существующих файлов.
3. Запись работы программы в уже существующий файл или создание недопустимого файла, а также проверка атрибутов существующего файла (Атрибут «Только для чтения»).

Математические методы и алгоритмы решения задач

Поставленная задача не требует особых математических методов.

Для подсчёта количества отрицательных чисел перед наибольшим положительным используется функция, состоящая из поиска максимального положительного числа среди элементов массива и затем, производящая подсчет элементов меньших нуля при помощи постфиксного инкремента переменной.

Форматы представления данных

Формат внешних файлов, из которых производится ввод данных – строго текстовый формат txt. Внутри читаемого txt файла для корректной работы должны присутствовать цифры и числа разделённые знаком пробела.

Формат файлов для сохранения результатов работы программы - только текстовый формат txt.

Структура программы

Программа разбита на 4 класса, также отдельно подключен проект для тестирования.

Основная последовательность работы программы – ожидания решения пользователя. Программа ожидает пользовательские нажатия на доступные в тот или иной момент элементы управления. После ввода корректный данных и нажатии кнопки «Запуск» программа запускает главный алгоритм. После чего результат появляется в специальном поле внизу окна программы. Затем пользователь может сохранить исходные данные и результат работы программы или же ввести массив заново без сохранения. Кнопка «Справка» открывает информацию о программе. Комплекс продолжает свою работу до тех пор, пока его не закроет пользователь в правом верхнем углу или с помощью средств операционной системы.

Таблица 1 – Основные переменные программы

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Описание** |
| mainArray | int[] | Массив, вводимый пользователем |
| pathOpen | string | Путь к файлу, который необходимо открыть для ввода данных |
| smthGoesWrong | string | Код ошибки, возвращаемый программой |
| result | string | Результат работы программы |
| size | int | Размер массива при случайном заполнении |
| max | int | Максимальный элемент в массиве |
| countofNegative | int | Количество отрицательных элементов |

Таблица 2 – Классы, используемые в программе

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Описание** | **Методы** | **Описание** |
| Counting | Подсчет отрицательных элементов | public int CountNegative(int[] array) | Нахождение максимального положительного и подсчет отрицательных элементов перед ним |
| SaveData | Сохранение данных программы | public void SaveEnteredData(string path, int[] arrayToSave) | Сохранение исходных данных |
| public void SaveResults(string path, int[] arrayToSave, string result) | Сохранение результатов |
| InputData | Ввод данных | public int[] ManualInputting(string arr) | Ручной ввод с клавиатуры |
| public int[] RandomFilling(int size) | Заполнение случайными значениями |
| public int[] FromFileFilling(string path) | Заполнение массива из файла |

Блок-схемы алгоритмов программы

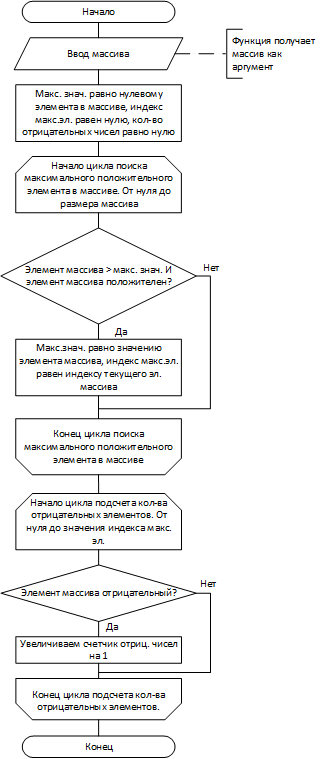


Рисунок 1 - Блок-схема алгоритма

Описание хода выполнения лабораторной работы

В ходе лабораторной работы было создано решение (Solution) интегрированной среде разработки Microsoft Visual Studio C# 2019. В нём был создан проект.

Программа была разбита на 4 условные части: модуль работы с файлами, модуль обработки полученных данных, модуль пользовательского интерфейса и основной модуль с циклом работы программы.

После написания основного цикла работы программы, были созданы функции пользовательского ввода, чтения данных из файла и случайного заполнения массива. Далее были созданы методы класса работы с массивом. После был создан алгоритм печати полученных данных на экран и сохранения во внешний файл как исходных данных, так и результатов.

В ходе работы над проектом были учтены и обработаны ошибки ввода некорректных данных, некорректное чтение файлов, а также обработаны возникающие исключения.

Также в код программы были включены модульные тесты, предназначенные для тестирования основного модуля программы. Данные для тестирования берутся из заранее подготовленных файлов. В случае удачного прохождения тестов на экран выводится сообщение об успешном их выполнении, иначе - сообщение о неудаче в ходе тестирования.

Результаты работы программы

При первом запуске программы пользователю предлагается выбрать вариант ввода массива: ручной ввод, заполнение случайными числами или ввод из файла.

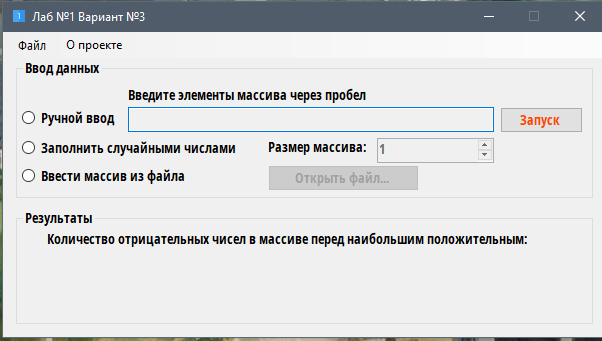


Рисунок 2 – Первый запуск программы

При выборе пункта «Ручной ввод» поле для ввода становиться доступным для редактирования и после ввода данных и нажатии кнопки «Запуск» пользователь получит результат работы программы. Стоит также отметить, что размер введенного с клавиатуры массива выводится специальное поле.

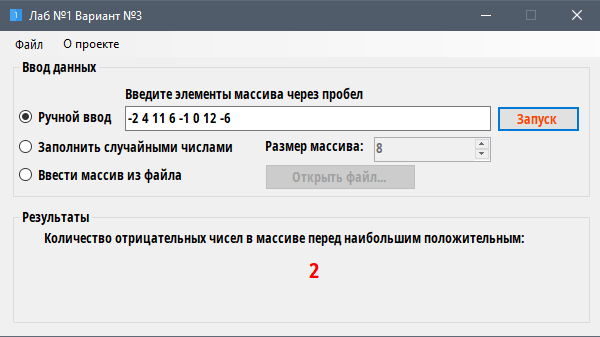


Рисунок 3 – Пример работы программы при ручном вводе массива

При выборе варианта заполнения случайными числами пользователю предлагается ввести размер массива в специальном поле, которое становиться доступно для редактирования. По нажатии кнопки «Запуск» программа также выведет решение для данного массива. Стоит отметить, что сгенерированный массив появляется в поле для ввода с клавиатуры, но редактировать его можно только после перехода в режим ручного ввода данных.

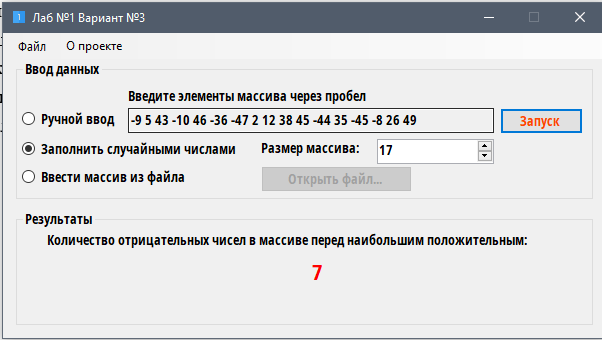


Рисунок 4 – Пример работы программы при заполнении случайными числами

При выборе пункта «Ввести массив из файла» для пользователя становиться доступна кнопка «Открыть файл…», при нажатии которой открывается окно с предупреждением о том, что ждёт программа в файле, который выберет пользователь. После выбора корректного файла, программа выведет результат, а также сам массив в поле для ввода с клавиатуры и его размер.

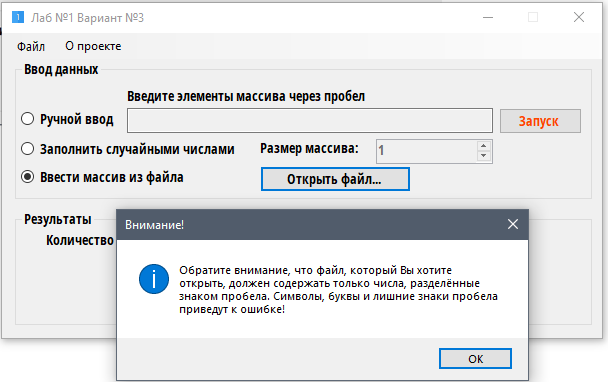


Рисунок 5 – Пример работы программе при выборе ввода из файла

Изначально кнопки меню «Файл» недосутпны для нажатий пользователя ввиду того, что исходные данные и результаты отсутсвтуют.

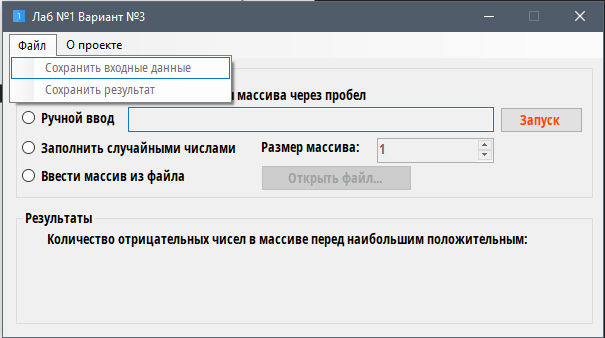


Рисунок 6 – Меню «Файл» при первом запуске программы

Однако после того, как был введён массив любым из доступных способов и получены результаты, пользователь может сохранить все данные, которые использовались в программе.

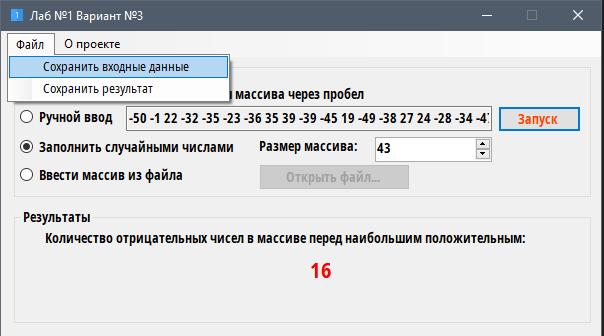


Рисунок 7 – Доступные пункты меню «Файл»

Входные данные, то есть изначально введённый массив, сохраняется так же, как того требует программа при открытии файлов. Таким образом сохраненные данные могут быть использованны повторно в данной программе. Результат сохраняется вместе с введённым массивом, его размером и числом отрицательных чисел перед наибольшим положительным в массиве.

При нажатии кнопки меню «О проекте» появляется окно с информацией об авторе данной программы и ее задаче.

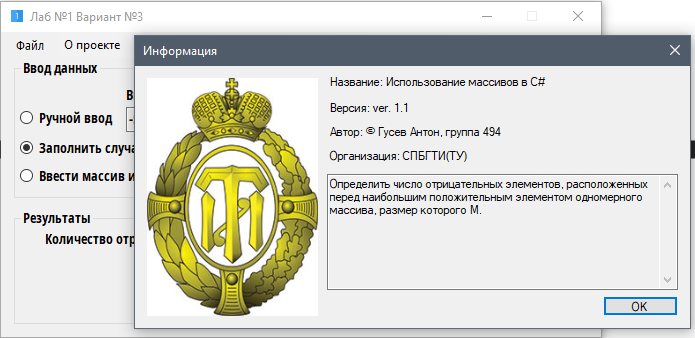


Рисунок 8 – Окно «Информация»

Исходный текст программы

[Начало программы ---]

[Начало Program.cs ---]

using System;

using System.Windows.Forms;

namespace firstLab

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainWindow());

}

}

}

[Конец Program.cs --- ]

[Начало MainWindow.cs ---]

using System;

using System.Diagnostics.CodeAnalysis;

using System.Linq;

using System.Windows.Forms;

namespace firstLab

{

[SuppressMessage("ReSharper", "InconsistentNaming")]

public partial class MainWindow : Form

{

private const string SmthGoesWrong = "-1"; // код потенциальной ошибки работы

private string pathOpen = "";

private int[] mainArray;

public MainWindow()

{

InitializeComponent();

MaximizeBox = false; // запрет изменения размера окна

saveFileDialog.Filter = @"Text files(\*.txt)|\*.txt"; // фильтр файлов при сохранении

}

///<summary>

///Область описания поведения элементов интерфейса для ввода массива

///</summary>

/\* EventArgs - это класс, дающий возможность передать какую-нибудь дополнительную

\* информацию обработчику (например, текущие координаты мыши при событии MouseMove).

sender - это объект, который вызвал событие.\*/

private void RadioButtonManual\_CheckedChanged(object sender, EventArgs e) // если выбран ручной ввод

{

// включение и отключение различных ограничений в интерфейсе

InputArrayTextBox.ReadOnly = false; // поле ввода можно редактировать

UpDownSize.Enabled = false; // элемент выбора размера заблокирован

ButtonChooseFile.Enabled = false; // кнопка выбор файла заблокирована

}

private void RadioButtonRandom\_CheckedChanged(object sender, EventArgs e) // если выбран рандомное заполнение

{

UpDownSize.Enabled = true; // выбор размера разблокирован

InputArrayTextBox.ReadOnly = true; // редактирование поле вывода массива запрещено

ButtonChooseFile.Enabled = false; // кнопка выбора файла заблокирована

}

private void RadioButtonFile\_CheckedChanged(object sender, EventArgs e) // если выбран ввод из файла

{

ButtonChooseFile.Enabled = true; // кнопка выбора файла разблокирована

InputArrayTextBox.ReadOnly = true; // редактирование поле вывода массива запрещено

UpDownSize.Enabled = false; // элемент выбора размера заблокирован

}

///<summary>

///Описание поведения при нажатии кнопки Enter

///</summary>

private void ButtonEnterArray\_Click(object sender, EventArgs e) // если нажата кнопка Запуск

{

var Process = new InputData(); // создаём объект класса InputData

if (radioButtonManual.Checked) // если выбран ручной ввод

{

var inputtedArray = InputArrayTextBox.Text; // вводим массив

if (string.IsNullOrEmpty(inputtedArray)) // обработка пустой строки

{

MessageBox.Show(@"Вы пытаетесь ввести пустую строку. Попытайтесь еще раз", @"Внимание!");

}

else

{

var countNegative = new Counting(); // создаём объект класса Counting

mainArray = Process.ManualInputting(inputtedArray); // формирование массива из чисел

var result = countNegative.CountNegative(mainArray).ToString(); // формирование результата

if (result == SmthGoesWrong) // обработка возможности отсутствия результата

{

labelResult.Text = @"Нет решений";

}

else if (mainArray.Length == 1 && mainArray[0] == 0)

{

labelResult.Text = "";

}

else

{

labelResult.Text = result; // вывод результата

}

}

}

else if (radioButtonRandom.Checked) // если выбрано заполнение рандомными числами

{

var size = Convert.ToInt32(UpDownSize.Text); // получение размера будущего массива

mainArray = Process.RandomFilling(size); // формирование массива размером size

InputArrayTextBox.Text = string.Empty; // очистка области вывода массива

for (var i = 0; i < mainArray.Length; i++) // вывод сформированного массива

{

InputArrayTextBox.Text += mainArray[i];

if(i != mainArray.Length - 1)

InputArrayTextBox.Text += @" ";

}

var countNegative = new Counting(); // создаём объект класса Counting

var result = countNegative.CountNegative(mainArray).ToString(); // формирование результата

labelResult.Text = result == SmthGoesWrong ? "Нет решений" : result;

}

else if (radioButtonFile.Checked) // если выбран ввод из файла

{

if (pathOpen == string.Empty) // обработка пустого имени файла

{

MessageBox.Show(@"Вы не выбрали файл. Сначала укажите из какого файла ввести данные", @"Внимание!");

}

else if(mainArray == null)

{

MessageBox.Show(@"Вы не ввели массив", @"Внимание!");

}

else

{

var countNegative = new Counting(); // создаём объект класса Counting

var result = countNegative.CountNegative(mainArray).ToString(); // формирование результата задачи

labelResult.Text = result == SmthGoesWrong ? "Нет решений" : result;

}

}

pathOpen = string.Empty; // очистка имени файла

if (mainArray != null && !string.IsNullOrEmpty(InputArrayTextBox.Text)) // если массив существует

{

SaveINToolStripMenuItem.Enabled = true; // разблокировка кнопки сохранения введенного массива

}

if (!string.IsNullOrEmpty(InputArrayTextBox.Text)) // если результат существует

{

SaveOUTToolStripMenuItem.Enabled = true;// разблокировка кнопки сохранения результатов

}

try

{

if (mainArray != null)

{

UpDownSize.Value = mainArray.Length;

}

}

catch

{

labelSize.Text += @" >";

}

}

private void ButtonChooseFile\_Click(object sender, EventArgs e) // нажатие кнопки выбора файла из которого загружаем массив

{

MessageBox.Show(@"Обратите внимание, что файл, который Вы хотите открыть, должен содержать только числа, разделённые знаком пробела. Символы, буквы и лишние знаки пробела приведут к ошибке!",

@"Внимание!",

MessageBoxButtons.OK,

MessageBoxIcon.Information

); // предупреждение о формате вводимых данных

if (openFileDialog.ShowDialog() == DialogResult.Cancel)

{

pathOpen = string.Empty;

return; // обработка закрытия окна выбора файла

}

pathOpen = openFileDialog.FileName; // получение имени файла

var Process = new InputData(); // создаём объект класса InputData

mainArray = Process.FromFileFilling(pathOpen); // формирование массива из данных из файла

InputArrayTextBox.Text = string.Empty;// очистка области вывода массива

if (mainArray != null)

{

for (var i = 0; i < mainArray.Length; i++) // вывод сформированного массива

{

InputArrayTextBox.Text += mainArray[i];

if (i != mainArray.Length - 1)

InputArrayTextBox.Text += @" ";

}

}

try

{

if (mainArray != null)

{

UpDownSize.Value = mainArray.Length;

}

}

catch

{

labelSize.Text += @" >";

}

}

private void InfoToolStripMenuItem1\_Click(object sender, EventArgs e)

{

var infoWindow = new AboutBox(); // создание объекта класса AboutBox

if (!Application.OpenForms.OfType<AboutBox>().Any())

infoWindow.Show(); // открываем форму с информацией один раз

}

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

{

var saving = new SaveData(); // создание объекта класса SaveData

if (saveFileDialog.ShowDialog() == DialogResult.Cancel)

return; // обработка закрытия окна сохранения введенных данных

var filenameSaveEnteredData = saveFileDialog.FileName;// получение имени файла

saving.SaveEnteredData(filenameSaveEnteredData, mainArray); // сохранение введённых данных

}

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

{

var saving = new SaveData(); // создание объекта класса SaveData

if (saveFileDialog.ShowDialog() == DialogResult.Cancel)

return; // обработка закрытия окна сохранения результатов

var filenameSaveRes = saveFileDialog.FileName; // получение имени файла

saving.SaveResults(filenameSaveRes, mainArray, labelResult.Text); // сохранение результатов

}

}

}

[Конец MainWindow.cs --- ]

[Начало InputData.cs ---]

using System;

using System.IO;

using System.Windows.Forms;

namespace firstLab

{

public class InputData

{

public int[] ManualInputting(string arr)

{

int[] errorArr = {0};

var splittedArr = arr.Split(' ');

for (var i = 1; i < splittedArr.Length; i++)

if (splittedArr[i] == splittedArr[i - 1] && splittedArr[i] == " ")

splittedArr[i] = splittedArr[i].Remove(1);

var tempArray = new int[splittedArr.Length];

try

{

for (var i = 0; i < splittedArr.Length; i++) tempArray[i] = Convert.ToInt32(splittedArr[i]);

}

catch

{

MessageBox.Show(@"Массив содержит символы или буквы. Попробуйте еще раз",

@"Ошибка!");

return errorArr;

}

return tempArray;

}

public int[] RandomFilling(int size)

{

var rand = new Random();

var tempArray = new int[size];

for (var i = 0; i < size; i++) tempArray[i] = rand.Next(-50, 50);

return tempArray;

}

public int[] FromFileFilling(string path)

{

var arrayStr = File.ReadAllText(path);

var splittedArr = arrayStr.Split(' ');

var tempArray = new int[splittedArr.Length];

try

{

for (var i = 0; i < splittedArr.Length; i++) tempArray[i] = Convert.ToInt32(splittedArr[i]);

}

catch

{

MessageBox.Show(

@"Массив в файле содержит символы или буквы. Проверьте правильность данных в файле или укажите путь на другой файл ",

@"Ошибка!");

return null;

}

return tempArray;

}

}

}

[Конец InputData.cs --- ]

[Начало SaveData.cs ---]

using System;

using System.IO;

using System.Windows.Forms;

namespace firstLab

{

internal class SaveData

{

public void SaveEnteredData(string path, int[] arrayToSave)

{

try

{

File.WriteAllText(path, string.Empty);

for (var i = 0; i < arrayToSave.Length; i++)

{

File.AppendAllText(path, arrayToSave[i].ToString());

if (i != arrayToSave.Length - 1) File.AppendAllText(path, @" ");

}

MessageBox.Show(@"Данные сохранены",

@"Информация");

}

catch

{

MessageBox.Show(@"Выбранный Вами файл предназначен только для чтения",

@"Ошибка!");

}

}

public void SaveResults(string path, int[] arrayToSave, string result)

{

try

{

File.WriteAllText(path, string.Empty);

File.AppendAllText(path, @"Вы ввели массив длинной: " + arrayToSave.Length + Environment.NewLine);

File.AppendAllText(path, @"Массив: " + Environment.NewLine);

for (var i = 0; i < arrayToSave.Length; i++)

{

File.AppendAllText(path, @"[" + arrayToSave[i] + @"]");

if (i != arrayToSave.Length - 1) File.AppendAllText(path, @" ");

}

File.AppendAllText(path, Environment.NewLine);

File.AppendAllText(path,

@"Количество отрицательных чисел перед наибольшим положительным элементом в массиве: " + result +

Environment.NewLine);

MessageBox.Show(@"Результаты сохранены",

@"Информация");

}

catch

{

MessageBox.Show(@"Выбранный Вами файл предназначен только для чтения",

@"Ошибка!");

}

}

}

}

[Конец SaveData.cs --- ]

[Начало Counting.cs ---]

namespace firstLab

{

public class Counting

{

public int CountNegative(int[] array)

{

var max = array[0];

var indexOfMax = 0;

var countNegative = 0;

for (var i = 0; i < array.Length; i++)

if (array[i] > max && array[i] > 0)

{

max = array[i];

indexOfMax = i;

}

for (var i = 0; i < indexOfMax; i++)

if (array[i] < 0)

countNegative++;

if (max == array[0] && (indexOfMax != 0 || max < 0))

return -1;

return countNegative;

}

}

}

[Конец Counting.cs --- ]

[Начало UnitTest.cs --- ]

using Microsoft.VisualStudio.TestTools.UnitTesting;

using firstLab;

namespace UnitTestLab1

{

[TestClass]

public class UnitTest

{

[TestMethod]

public void TestMethodUsualArray()

{

int[] arr = { -1, 1, -2, 4, -3 };

var test = new Counting();

var res = test.CountNegative(arr);

Assert.AreEqual(2, res);

}

[TestMethod]

public void TestMethodAllNegative()

{

int[] arr = { -1, -4, -2 };

var test = new Counting();

var res = test.CountNegative(arr);

Assert.AreEqual(-1, res);

}

[TestMethod]

public void TestMethodAllPositive()

{

int[] arr = { 7, 30, 4 };

var test = new Counting();

var res = test.CountNegative(arr);

Assert.AreEqual(0, res);

}

[TestMethod]

public void TestMethodAllZero()

{

int[] arr = { 0, 0, 0 };

var test = new Counting();

var res = test.CountNegative(arr);

Assert.AreEqual(0, res);

}

[TestMethod]

public void TestMethodMaxIsFirst()

{

int[] arr = { 10, -60, -12 };

var test = new Counting();

var res = test.CountNegative(arr);

Assert.AreEqual(0, res);

}

}

}

[Конец UnitTest.cs --- ]

[Конец программы --- ]