Пермский филиал федерального государственного автономного  
образовательного учреждения высшего образования  
«Национальный исследовательский университет  
«Высшая школа экономики»

*Факультет социально-экономических и компьютерных наук*

Мальшаков Александр Аркадьевич

**ЛАБОРАТОРНАЯ РАБОТА №4**

*Работа с одномерными массивами*

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки 38.03.05-Бизнес - информатика

Руководитель

Старший преподаватель кафедры ИТБ \_\_\_\_\_\_\_\_\_\_\_\_

Волков С. А.

Пермь, 2024 год

**Оглавление**

[1. Постановка задачи 4](#_Toc183992753)

[2. Анализ 5](#_Toc183992754)

[Формирование массива 5](#_Toc183992755)

[Подзадача №1 5](#_Toc183992756)

[Подзадача №2 5](#_Toc183992757)

[Подзадача №3 6](#_Toc183992758)

[Подзадача №4 6](#_Toc183992759)

[Подзадача №5 6](#_Toc183992760)

[Подзадача №6 6](#_Toc183992761)

[3. Проектирование и разработка 7](#_Toc183992762)

[Подзадача №1 7](#_Toc183992763)

[Подзадача №2 8](#_Toc183992764)

[Подзадача №3 9](#_Toc183992765)

[Подзадача №4 10](#_Toc183992766)

[Подзадача №5 11](#_Toc183992767)

[Подзадача №6 12](#_Toc183992768)

[4. Тестирование 13](#_Toc183992769)

[Формирование массива 13](#_Toc183992770)

[Подзадача №1 14](#_Toc183992771)

[Подзадача №2 14](#_Toc183992772)

[Подзадача №3 15](#_Toc183992773)

[Подзадача №4 16](#_Toc183992774)

[Подзадача №5 17](#_Toc183992775)

[Подзадача №6 18](#_Toc183992776)

[Приложение A 20](#_Toc183992777)

[Приложение B 26](#_Toc183992778)

[Приложение C 27](#_Toc183992779)

# Постановка задачи

1. Напечатать интерфейс для программы с выбором задачи.
2. Сформировать массив с помощью ручного ввода или случайными числами.
3. Выполнить удаление всех элементов больших среднего арифметического всего массива (Подзадача №1).
4. Выполнить вставку K элементов в начало массива (Подзадача №2).
5. Сдвинуть массив на M элементов вправо (Подзадача №3).
6. Найти индекс первого четного элемента (Подзадача №4).
7. Выполнить сортировку простым включением (Подзадача №5).
8. Выполнить бинарный поиск элемента (Подзадача №6).

При изменении массива выводить его на печать.

# Анализ

Все числа, используемые в программе, будут типа int. Они могут быть и положительными, и отрицательными, таблица контроля int приведена ниже (см. Таблица 1).

Таблица 1 – Контроль ввода int

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Условие** | **Выходные данные** |
| Любые символы, не являющиеся целым числом | Невозможно вычислить | Сообщение об ошибке о том, что введенная строка не является целым числом |
| Целое число | Программа продолжается | Стандартные |

## Формирование массива

Для формирования массива понадобиться переменная n типа int для хранения длины массива. Для n недопустимо отрицательное значение, таблица для n ниже (см. Таблица 2).

Таблица 2 – Контроль ввода n

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Условие** | **Выходные данные** |
| Любые символы, не являющиеся целым числом | Невозможно вычислить | Сообщение об ошибке о том, что введенная строка не является целым числом |
| Целое отрицательное | Невозможно вычислить | Сообщение об ошибке о том, что длина не может быть отрицательной |
| Целое не отрицательное | Программа продолжается | Стандартные |

## Подзадача №1

Для удаления определённых элементов нужно создать новый массив и положить в него только нужные элементы и сделать его текущим.

Можно заметить, что исходя из условия, по которому удаляем элементы, которые больше среднего арифметического, массив после удаления не может остаться пустым.

## Подзадача №2

Для добавления K элементов в начало массива, нужно создать новый с размером на K элементов больше старого, после с K-го индекса поместить в новый массив элементы старого, а с 0 по K-й новые элементы.

Переменная K не может иметь отрицательное значение, таблица для этой переменной ниже (см. Таблица 3).

Таблица 3 – Контроль ввода K

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Условие** | **Выходные данные** |
| Любые символы, не являющиеся целым числом | Невозможно вычислить | Сообщение об ошибке о том, что введенная строка не является целым числом |
| Целое отрицательное | Невозможно вычислить | Сообщение об ошибке о том, что количество не может быть отрицательным |
| Целое не отрицательное | Программа продолжается | Стандартные |

## Подзадача №3

Для сдвига массива на M элементов нужно для начала взять остаток деления M на длину массива, так мы уберём цикличные сдвиги, после можно создать буфер на полученное число элементов. В буфер нужно сохранить правую часть массива, после двинуть массив и выписать элементы из буфера в начало.

Переменная M не может иметь отрицательное значение, таблица для этой переменной ниже (см. Таблица 4).

Таблица 4 – Контроль ввода M

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Условие** | **Выходные данные** |
| Любые символы, не являющиеся целым числом | Невозможно вычислить | Сообщение об ошибке о том, что введенная строка не является целым числом |
| Целое отрицательное | Невозможно вычислить | Сообщение об ошибке о том, что количество не может быть отрицательным |
| Целое не отрицательное | Программа продолжается | Стандартные |

## Подзадача №4

У этой задачи нет входных данных, вводимых пользователем, нужно лишь найти первый четный элемент. При нахождении нужного элемента возвращаем его индекс, если элемента нет, то -1;

## Подзадача №5

У этой задачи нет входных данных, вводимых пользователем, нужно лишь найти первый четный элемент. При нахождении нужного элемента возвращаем его индекс, если элемента нет, то -1;

## Подзадача №6

У этой задачи нет входных данных, вводимых пользователем, нужно лишь найти первый четный элемент. При нахождении нужного элемента возвращаем его индекс, если элемента нет, то -1;

# Проектирование и разработка

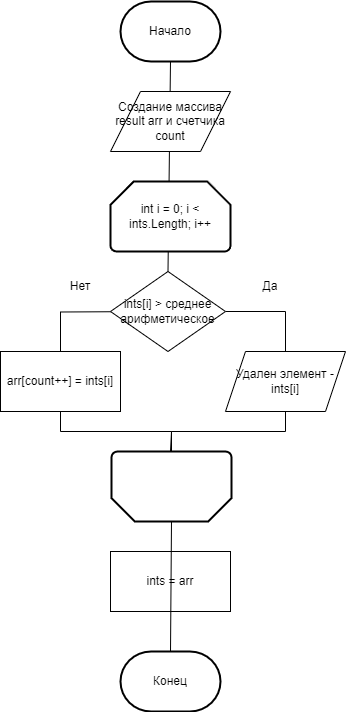
Для удобства программа будет разделена на 3 файла.

1. В первом файле будет расположен класс Program с функцией main. Здесь будет описан интерфейс.
2. Во втором файле будет класс массива MyArray, который будет хранить массив и функции для работы с ним.
3. В третьем файле будет класс InputManager, в котором будет одна статическая функция ReadVelueInt для считывания целых чисел.

Все подзадачи будут располагаться в классе MyArray, для всех будет общая переменная ints, хранящая массив целых чисел, а также переменная length, хранящая длину массива. Код находиться в приложениях (см. приложения A, B, C).

## Подзадача №1

Блок-схема для подзадачи 1 представлена ниже (см. рисунок 3.1).



***Рисунок 3.1 – Блок-схема для подзадачи 1***

## Подзадача №2

Блок-схема для подзадачи 2 представлена ниже (см. рисунок 3.2).

Изображение выглядит как текст, снимок экрана, Шрифт, черно-белый

Автоматически созданное описание

***Рисунок 3.2 – Блок-схема для подзадачи 2***

## Подзадача №3

Блок-схема для подзадачи 3 представлена ниже (см. рисунок 3.3).

Изображение выглядит как текст, снимок экрана, Шрифт, графический дизайн

Автоматически созданное описание

***Рисунок 3.3 – Блок-схема для подзадачи 3***

## Подзадача №4

Блок-схема для подзадачи 4 представлена ниже (см. рисунок 3.4).

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

***Рисунок 3.4 – Блок-схема для подзадачи 4***

## Подзадача №5

Блок-схема для подзадачи 5 представлена ниже (см. рисунок 3.5).

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

***Рисунок 3.5 – Блок-схема для подзадачи 5***

## Подзадача №6

Блок-схема для подзадачи 6 представлена ниже (см. рисунок 3.6).

Изображение выглядит как текст, снимок экрана, Шрифт, графический дизайн

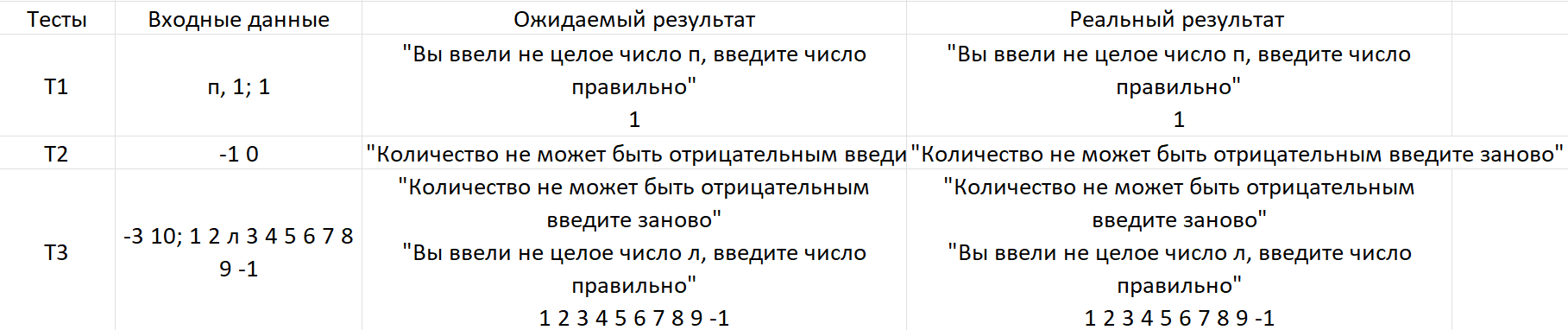
Автоматически созданное описание

***Рисунок 3.5 – Блок-схема для подзадачи 5***

# Тестирование

## Формирование массива

Тесты для формирования массива представлены на рисунке ниже (см. рисунок 4.1). В тестах входные данные для длины и чисел разделяются точкой с запятой, а сами числа друг от друга пробелом, в программе все вводиться последовательно через Enter.



***Рисунок 4.1 – Тесты для формирования массива***

Критерии черного ящика для формирования массива, представлены ниже (см. рисунок 4.2).

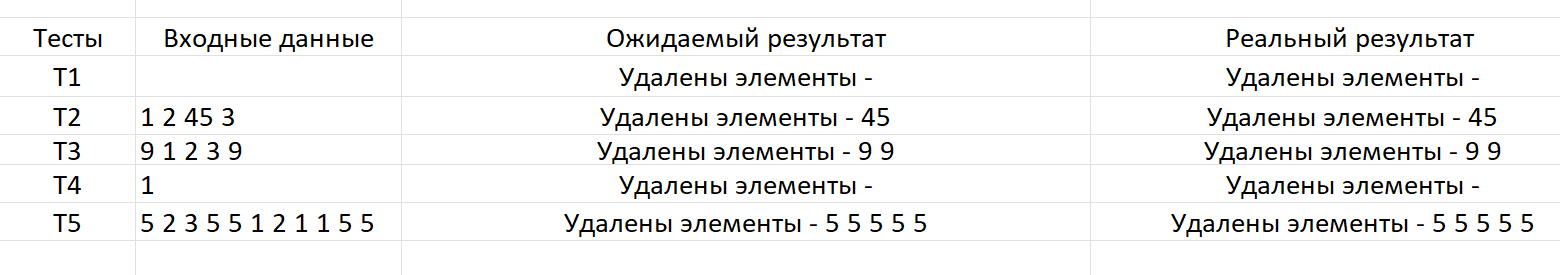


***Рисунок 4.2 – Оценка полноты тестирования по черному ящику для формирования массива***

Для формирования массива с помощью случайных чисел тесты будут те же, но без ввода элементов массива.

## Подзадача №1

Тесты для подзадачи №1 представлены на рисунке ниже (см. рисунок 4.3). В тестах входные данные для длины и чисел разделяются точкой с запятой, а сами числа друг от друга пробелом, в программе все вводиться последовательно через Enter.



***Рисунок 4.3 – Тесты для подзадачи №1***

Критерии черного ящика для подзадачи №1, представлены ниже (см. рисунок 4.4).

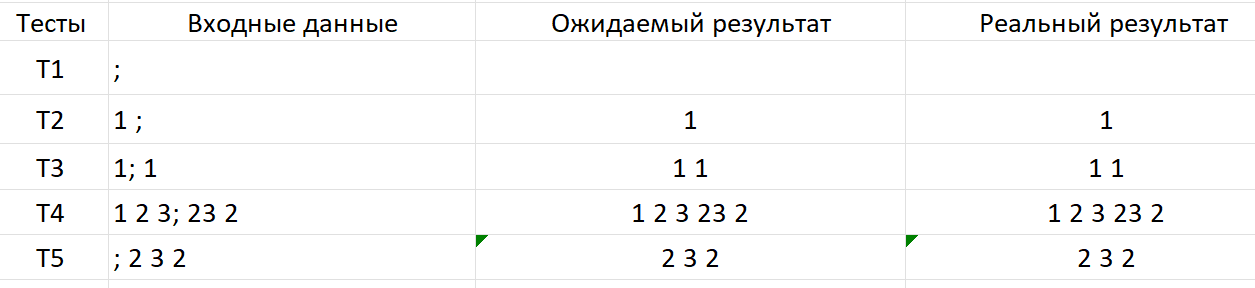
Изображение выглядит как текст, число, Шрифт, снимок экрана

Автоматически созданное описание

***Рисунок 4.4 – Оценка полноты тестирования по черному ящику для подзадачи №1***

## Подзадача №2

Тесты для подзадачи №2 представлены на рисунке ниже (см. рисунок 4.5). Во водных данных 2 массива, первый массив добавляемых элементов, получается также как и формируется основной массив, и второй это основной массив.



Критерии черного ящика для подзадачи №2, представлены ниже (см. рисунок 4.6).

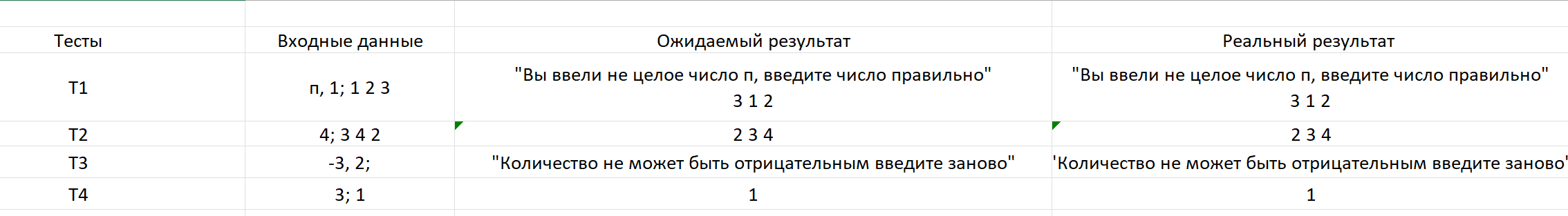
***Рисунок 4.5 – Тесты для подзадачи №2***



***Рисунок 4.6 – Оценка полноты тестирования по черному ящику для подзадачи №2***

## Подзадача №3

Тесты для подзадачи №3 представлены на рисунке ниже (см. рисунок 4.7). Во входных данных сначала указано число M, а потом до этого сформированный массив.



***Рисунок 4.7 – Тесты для подзадачи №3***

Критерии черного ящика для подзадачи №3, представлены ниже (см. рисунок 4.8).

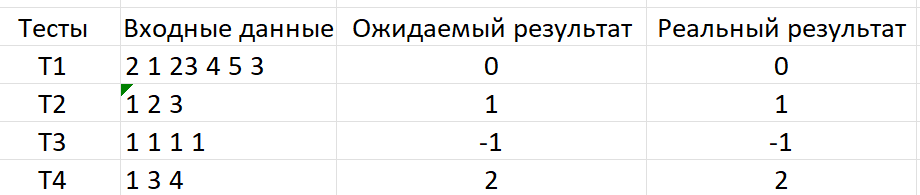
Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

***Рисунок 4.8 – Оценка полноты тестирования по черному ящику для подзадачи №3***

## Подзадача №4

Тесты для подзадачи №4 представлены на рисунке ниже (см. рисунок 4.9). Во входных данных массив, сформированный ранее.



***Рисунок 4.9 – Тесты для подзадачи №4***

Критерии черного ящика для подзадачи №4, представлены ниже (см. рисунок 4.10).

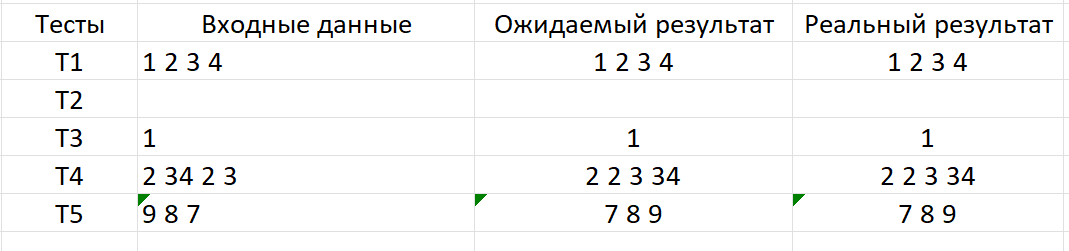
Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

***Рисунок 4.10 – Оценка полноты тестирования по черному ящику для подзадачи №4***

## Подзадача №5

Тесты для подзадачи №5 представлены на рисунке ниже (см. рисунок 4.11). Во входных данных массив, сформированный ранее.



***Рисунок 4.11 – Тесты для подзадачи №5***

Критерии черного ящика для подзадачи №5, представлены ниже (см. рисунок 4.12).

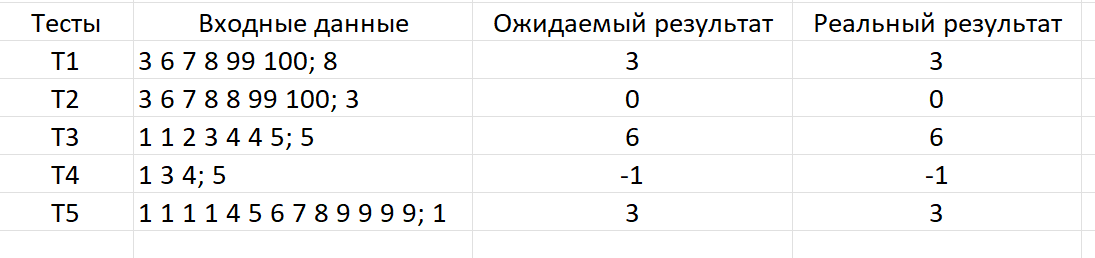
Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание

***Рисунок 4.12 – Оценка полноты тестирования по черному ящику для подзадачи №5***

## Подзадача №6

Тесты для подзадачи №6 представлены на рисунке ниже (см. рисунок 4.13). Во входных данных массив, сформированный ранее, и число, которое надо найти бинарным поиском.



***Рисунок 4.13 – Тесты для подзадачи №1***

Критерии черного ящика для подзадачи №6, представлены ниже (см. рисунок 4.14).

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

***Рисунок 4.14 – Оценка полноты тестирования по черному ящику для подзадачи №6***

# Приложение A

sing MyTypes;

using System;

namespace lab\_4

{

internal class Program

{

public static int ReadValueInt()

{

do

{

string s = Console.ReadLine();

if (int.TryParse(s, out int a))

return a;

else

Console.WriteLine($"Вы ввели не целое число {s}, введите число правильно");

}

while (true);

}

static void Main(string[] args)

{

Console.WriteLine("Начать работу - любое число кроме -1 выйти из программы - -1");

int a = ReadValueInt();

while (a != -1)

{

Console.Clear();

Console.WriteLine("Введите количество элементов массива");

int n = 0;

do

{

if (n < 0)

Console.WriteLine("Количество не может быть отрицательным введите заново");

n = ReadValueInt();

}

while (n < 0);

var CurrentArray = new MyArray(n);

Console.Clear();

Console.WriteLine("Выберите способ заполнения 1 - ввод с консоли 2 - автоматически");

a = ReadValueInt();

switch (a)

{

case 1:

CurrentArray.Fill(false);

break;

case 2:

CurrentArray.Fill(true);

break;

}

while (true)

{

Console.Clear();

CurrentArray.WriteArray();

Console.WriteLine("Действия которые можно выполнить с массивом:");

Console.WriteLine("-1 Выйти из программы");

Console.WriteLine("1 Вставить K элементов в начало");

Console.WriteLine("2 Удалить все элементы которые больше среднего арифметического всего массива");

Console.WriteLine("3 Циклически сдвинуть на M элементов вправо");

Console.WriteLine("4 Найти первый четный");

Console.WriteLine("5 Сортировать массив");

a = ReadValueInt();

if (a == -1)

break;

switch (a)

{

case 1:

Console.Clear();

CurrentArray.WriteArray();

Console.WriteLine("Введите количество элементов");

do

{

if (n < 0)

Console.WriteLine("Количество не может быть отрицательным введите заново");

n = ReadValueInt();

}

while (n < 0);

Console.WriteLine("Вставить с консоли - 2, вставить автоматически (рандомно) - 3");

while (a != 2 && a != 3)

{

a = ReadValueInt();

if (a != 2 && a != 3)

Console.WriteLine("Введите либо 2 либо 3");

}

switch (a)

{

case 2:

Console.WriteLine("Введите " + n.ToString() + " элементов");

int[] b = new int[n];

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

b[i] = ReadValueInt();

CurrentArray.Insert(0, new MyArray(b));

CurrentArray.WriteArray();

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

case 3:

var arr = new MyArray(n);

arr.Fill();

arr.WriteArray();

CurrentArray.Insert(0, arr);

CurrentArray.WriteArray();

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

}

break;

case 2:

Console.Clear();

CurrentArray.WriteArray();

CurrentArray.DeleteElem(x => x > CurrentArray.Average);

CurrentArray.WriteArray();

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

case 3:

Console.Clear();

CurrentArray.WriteArray();

Console.WriteLine("Введите на какое количество нужно сдвинуть");

int m = ReadValueInt();

CurrentArray.Move(m);

CurrentArray.WriteArray();

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

case 4:

Console.Clear();

CurrentArray.WriteArray();

int elem = CurrentArray.FindFirst(x => x % 2 == 0);

Console.WriteLine("Первый четный элемент " + elem.ToString());

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

case 5:

Console.Clear();

CurrentArray.WriteArray();

CurrentArray.SortInsets();

Console.WriteLine("Отсортированный массив");

CurrentArray.WriteArray();

Console.WriteLine("любая цифра кроме 1 - продолжить работу 1 - Выполнить бинарный поиск элемента");

a = ReadValueInt();

if (a == 1)

{

Console.WriteLine("Введите число которое нужно найти");

n = ReadValueInt();

var num = CurrentArray.GetIndexBinary(n);

if (num != -1)

Console.WriteLine("Ваше число находиться под индексом - " + num.ToString());

}

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

default:

Console.WriteLine("Вы ввели неправильное a - " + a.ToString());

Console.WriteLine("Для продолжения нажмите enter");

Console.Read();

break;

}

}

}

}

}

}

# Приложение B

using System;

namespace DataManage

{

static class InputManager

{

public static int ReadValueInt()

{

do

{

string s = Console.ReadLine();

if (int.TryParse(s, out int a))

return a;

else

Console.WriteLine($"Вы ввели не целое число {s}, введите число правильно");

}

while (true);

}

}

}

# Приложение C

using lab\_4;

using System;

using System.Collections;

namespace MyTypes

{

delegate T Criterion<T, J>(J a);

class MyArray : IEnumerable, IEnumerator

{

private int[] ints;

private int length;

private Random rand = new Random();

private int start = 0;

private int end = 10;

public void SetBorderRand(int start, int end)

{

this.start = start;

this.end = end;

}

public int Sum

{

get

{

int sum = 0;

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

{

sum += ints[i];

}

return sum;

}

}

public double Average

{

get

{

return (double)Sum / length;

}

}

public bool isEmpty

{

get

{

return length == 0 || ints == null;

}

}

public int Count

{

get { return length; }

}

public int Max

{

get

{

if (isEmpty)

{

Console.WriteLine("В массиве нет максимума массив пуст");

return -1;

}

int max = int.MinValue;

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

max = max > ints[i] ? max : ints[i];

return max;

}

}

public int Min

{

get

{

if (isEmpty)

{

Console.WriteLine("В массиве нет минимума массив пуст");

return -1;

}

int max = int.MaxValue;

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

max = max < ints[i] ? max : ints[i];

return max;

}

}

private int[] Ints

{

set

{

ints = value;

//WriteArray();

}

}

public MyArray(int size)

{

int p = 2;

while (p < size) p \*= 2;

ints = new int[p];

length = size;

}

public MyArray(int[] numbers)

{

int p = 2;

while (p < numbers.Length) p \*= 2;

ints = new int[p];

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

ints[i] = numbers[i];

length = numbers.Length;

}

/// <summary>

/// if random is true then array will fill random numbers

/// else elements of array will be entered from console

/// </summary>

/// <param name="mode"></param>

public void Fill(bool random = true)

{

if (random)

{

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

ints[i] = rand.Next(start, end);

}

else

{

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

ints[i] = Program.ReadValueInt();

}

}

public void DeleteElem(int ind)

{

if (!inBorder(ind))

{

Console.WriteLine("Удаление элемента невозможно элемент находиться вне границ массива");

return;

}

int j = 0;

Console.WriteLine("Удален элемент - " + ints[ind].ToString());

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

{

ints[j++] = ints[i];

}

for (int i = ind + 1; i < length; i++)

{

ints[j++] = ints[i];

}

length--;

int newLen = 1;

while (newLen <= length)

newLen \*= 2;

SetSize(newLen);

}

public void DeleteElem(Criterion<bool, int> criterion)

{

var currarray = new int[length];

int count = 0;

Console.Write("Удалены элементы - ");

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

{

if (!criterion(ints[i]))

currarray[count++] = ints[i];

else

Console.Write(ints[i].ToString() + " ");

}

Console.WriteLine();

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

{

ints[i] = currarray[i];

}

int newLen = 1;

while (newLen <= count)

newLen \*= 2;

SetSize(newLen);

length = count;

}

public void Insert(int ind, MyArray array)

{

if (!(ind <= length && ind >= 0))

{

Console.WriteLine("Ошибка в количестве добавляемых элементов");

return;

}

int newLen = ints.Length;

while (length + array.Count > newLen)

newLen \*= 2;

SetSize(newLen);

for (int i = length + array.Count; i >= ind + array.Count; i--)

ints[i] = ints[i - array.Count];

for (int i = ind; i < array.Count; i++)

ints[i] = array[i - ind];

length += array.Count;

}

public void Insert(int ind, int[] array)

{

if (!(ind <= length && ind >= 0))

{

Console.WriteLine("Ошибка в количестве добавляемых элементов");

return;

}

int newLen = ints.Length;

while (length + array.Length > newLen)

newLen \*= 2;

SetSize(newLen);

for (int i = length + array.Length; i >= ind + array.Length; i--)

ints[i] = ints[i - array.Length];

for (int i = ind; i < array.Length; i++)

ints[i] = array[i - ind];

length += array.Length;

}

public void AddElem(int ind, int value)

{

if (!(ind <= length && ind >= 0))

{

Console.WriteLine("Ошибка в количестве добавляемых элементов");

return;

}

int newLen = ints.Length;

while (length + 1 > newLen)

newLen \*= 2;

SetSize(newLen);

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

{

ints[i + 1] = ints[i];

}

ints[ind] = value;

length++;

}

public void AddElem(int ind, int count = 1, bool fill = false, bool random = true)

{

if (!(ind <= length && ind >= 0))

{

Console.WriteLine("Ошибка в количестве добавляемых элементов");

return;

}

int newLen = ints.Length;

while (length + count > newLen)

newLen \*= 2;

SetSize(newLen);

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

{

ints[i + count] = ints[i];

}

if (fill)

for (int i = ind; i <= ind + count - 1; i++)

{

if (random)

ints[i] = rand.Next(start, end);

else

ints[i] = Program.ReadValueInt();

}

length += count;

}

public void Reverse()

{

int[] result = new int[ints.Length];

int j = length - 1;

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

result[i] = ints[j--];

Ints = result;

}

public void Move(int count, bool right = true)

{

if (length == 0)

return;

count = count % length;

int[] result = new int[ints.Length];

int k = 0;

if (right)

{

for (int i = length - count; i < length; i++)

result[k++] = ints[i];

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

result[k++] = ints[i];

}

else

{

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

result[k++] = ints[i];

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

result[k++] = ints[i];

}

ints = result;

}

public int FindFirst(Criterion<bool, int> criterion)

{

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

{

if (criterion(ints[i]))

{

return ints[i];

}

}

Console.WriteLine("Ни один элемент не подешел по критерию");

return -1;

}

public int FindLast(Func<int, bool> criterion)

{

for (int i = length; i >= 0; i--)

{

if (criterion(ints[i]))

{

return ints[i];

}

}

Console.WriteLine("Ни один элемент не подешел по критерию");

return -1;

}

public MyArray FindAll(Criterion<bool, int> criterion)

{

var currarray = new int[length];

int[] result;

int k = 0;

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

if (criterion(ints[i]))

currarray[k++] = ints[i];

result = new int[k];

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

{

result[i] = currarray[i];

}

return new MyArray(result);

}

public void SortInsets(bool rise = true)

{

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

{

int buff = ints[i];

int j = i - 1;

while (j >= 0 && (buff < ints[j] == rise))

{

ints[j + 1] = ints[j];

j--;

}

ints[j + 1] = buff;

}

}

public void SelectionSort(bool rise = true)

{

for (int i = 0; i < length - 1; i++)

{

int buff1 = ints[i];

int buff2 = !rise ? int.MinValue : int.MaxValue;

int k = i + 1;

for (int j = i + 1; j < length; j++)

{

if (buff2 > ints[j] == rise)

{

buff2 = ints[j];

k = j;

}

}

if (buff1 > buff2 == rise)

{

ints[i] = buff2;

ints[k] = buff1;

}

}

}

public void BubbleSort(bool rise = true)

{

for (int i = 0; i < length - 1; i++)

{

for (int j = i + 1; j < length; j++)

{

int buff = ints[j];

if (ints[i] > ints[j] == rise)

{

ints[j] = ints[i];

ints[i] = buff;

}

}

}

}

public void Clear()

{

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

ints[i] = 0;

}

/// <summary>

///

/// </summary>

/// <param name="value"></param>

/// <returns></returns>

public int GetIndex(int value)

{

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

{

if (ints[i] == value)

return i;

}

Console.WriteLine("Такого элемента нет, индекс вернуть невозможно");

return -1;

}

/// <summary>

/// For binary search, your array should be sorted

/// </summary>

/// <param name="number"></param>

/// <returns></returns>

public int GetIndexBinary(int value)

{

int offsetLeft = 0;

int offsetRight = length / 2;

while (offsetRight != 0)

{

if (ints[offsetLeft + offsetRight] == value)

return offsetLeft + offsetRight;

else if (ints[offsetLeft + offsetRight] > value)

{

offsetRight /= 2;

}

else if (ints[offsetLeft + offsetRight] < value)

{

offsetLeft += offsetRight;

offsetRight /= 2;

}

}

Console.WriteLine("Такого элемента нет, индекс вернуть невозможно");

return -1;

}

public bool Contains(int value)

{

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

{

if (ints[i] == value)

return true;

}

return false;

}

/// <summary>

/// For binary search, your array should be sorted

/// </summary>

/// <param name="number"></param>

/// <returns></returns>

public bool ContainsBinary(int value)

{

int offsetLeft = 0;

int offsetRight = length / 2;

while (offsetRight != 0)

{

if (ints[offsetLeft + offsetRight] == value)

return true;

else if (ints[offsetLeft + offsetRight] > value)

{

offsetRight /= 2;

}

else if (ints[offsetLeft + offsetRight] < value)

{

offsetLeft += offsetRight;

offsetRight /= 2;

}

}

return false;

}

public void WriteArray()

{

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

{

if (i % 10 != 0)

Console.ForegroundColor = (ConsoleColor)(i % 10);

Console.Write(ints[i].ToString() + " ");

}

Console.WriteLine();

Console.ForegroundColor = ConsoleColor.White;

}

public bool inBorder(int ind)

{

return ind < length && ind >= 0;

}

private void SetSize(int size)

{

if (size == ints.Length) return;

var result = new int[size];

for (int i = 0; i < (length > size ? size : length); i++)

result[i] = ints[i];

ints = result;

}

public int this[int key]

{

get

{

if (!inBorder(key))

{

Console.WriteLine("Выход за границы массива");

return 0;

}

return ints[key];

}

set

{

if (!inBorder(key))

{

Console.WriteLine("Выход за границы массива");

return;

}

ints[key] = value;

}

}

public MyArray Copy(int indS)

{

if (!inBorder(indS))

{

Console.WriteLine("Выход за границы массива");

return new MyArray(0);

}

MyArray result = new MyArray(Count - indS);

for (int i = indS; i < Count; i++)

result[i - indS] = ints[i];

return result;

}

public MyArray Copy(int indS, int indE)

{

if (!inBorder(indS) || !inBorder(indE))

{

Console.WriteLine("Выход за границы массива");

return new MyArray(0);

}

MyArray result = new MyArray(indE - indS);

for (int i = indS; i < indE; i++)

result[i - indS] = ints[i];

return result;

}

public static MyArray operator +(MyArray arr1, MyArray arr2)

{

MyArray result = new MyArray(arr1.Count + arr2.Count);

int count = 0;

for (int i = 0; i < arr1.Count; i++)

result[count++] = arr1[i];

for (int i = 0; i < arr2.Count; i++)

result[count++] = arr2[i];

return result;

}

public IEnumerator GetEnumerator()

{

return this;

}

public object Current

{

get

{

if (!inBorder(index))

{

Console.WriteLine("Сначала сдвиньте указатель");

}

return ints[index];

}

}

int index = -1;

public bool MoveNext()

{

index++;

if (index < length && index > -1)

return true;

else

return false;

}

public void Reset()

{

index = -1;

}

}

}