|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»  *Факультет социально-экономических и компьютерных наук* |
|  |
| Ханжин Александр Евгеньевич  **ЛАБОРАТОРНАЯ РАБОТА №4**  студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *09.03.04 Программная инженерия*   |  |  | | --- | --- | |  | Преподаватель кафедры ИТБ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Марквирер В.Д. |   Пермь, 2023 |

**Оглавление**

[Задача 1 3](#_Toc150525549)

[1.1 Постановка задачи 3](#_Toc150525550)

[1.2 Анализ 3](#_Toc150525551)

[1.3 Алгоритм (блок-схема) 4](#_Toc150525552)

[1.4 Программа (листинг) 7](#_Toc150525553)

[1.5 Результаты работы программы 9](#_Toc150525554)

# Задача 1

## Постановка задачи

Для начала нужно составить список задач. Программа должна выполнять следующие функции:

1. Сформировать массив из n элементов (n задается пользователем с клавиатуры) двумя способами:
   1. с помощью датчика случайных чисел,
   2. элементы вводятся пользователем с клавиатуры.
2. Распечатать массив.
3. Выполнить удаление всех чисел, больших среднего арифметического.
4. Выполнить добавление k чисел в начало массива.
5. Выполнить циклический сдвиг вправо на m элементов.
6. Выполнить поиск первого четного элемента и подсчитать количество сравнений, необходимых для поиска нужного элемента.
7. Выполнить сортировку массива простым включением.
8. Выполнить поиск указанных элементов в отсортированном массиве и подсчитать количество сравнений, необходимых для поиска нужного элемента (использовать бинарный поиск).

## Анализ

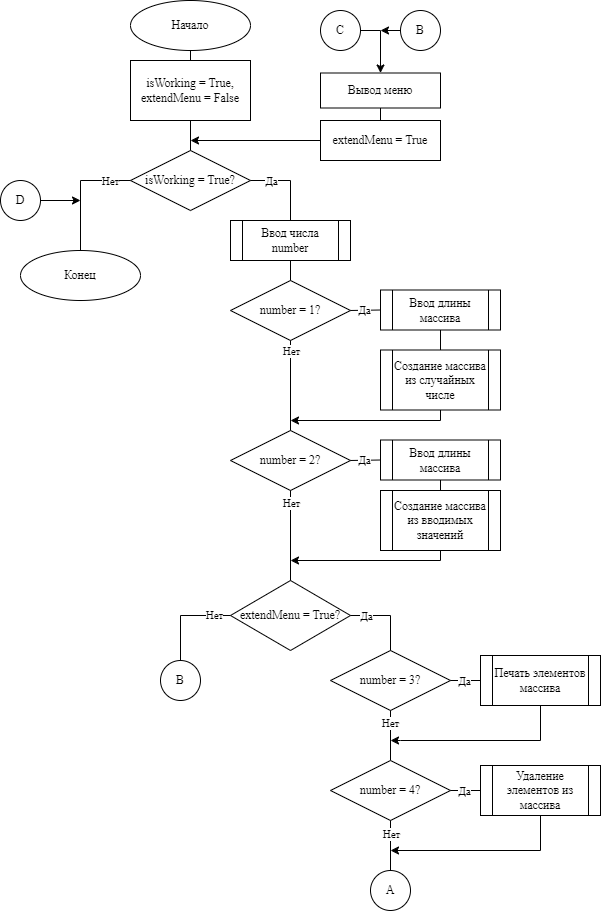
Представим анализ задачи в виде таблицы:

***Таблица 1.2.1 - Функции***

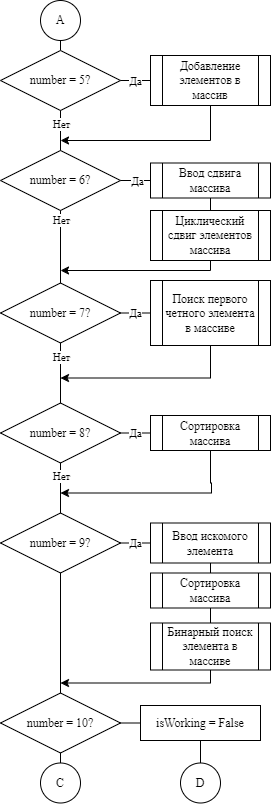
| **№** | **Описание функции** | **Входные данные** | **Классы входных данных** | **Выходные данные** |
| --- | --- | --- | --- | --- |
| **Основные функции** | | | | |
| 1.1 | Формирование массива с случайными числами | Длина массива | Натуральное число |  |
| Символ | Сообщение о ошибке |
| Вещественное число |
| Число за пределами целочисленного типа |
| 1.2 | Формирование массива с клавиатуры | Длина массива | Натуральное число |  |
| Символ | Сообщение об ошибке |
| Вещественное число |
| Число за пределами целочисленного типа |
| Элементы массива | Целое число |  |
| Символ | Сообщение об ошибке |
| Вещественное число |
| Число за пределами целочисленного типа |
| 2 | Печать массива | Массив | Массив целых чисел | Числа из массива |
| 3 | Удаление первого четного элемента из массива | Массив | Массив целых чисел | Преобразованный массив |
| 4 | Добавление элементов в начало массива | Массив | Массив целых чисел | Преобразованный массив |
| Количество добавленных элементов | Натуральное число |  |
| Символ | Сообщение об ошибке |
| Вещественное число |
| Число за пределами целочисленного типа |
| Элементы для добавления в массив | Целое число |  |
| Символ | Сообщение об ошибке |
| Вещественное число |
| Число за пределами целочисленного типа |
| 5 | Циклический сдвиг элементов массива | Сдвиг элементов в массиве | Натуральное число |  |
| Символ | Сообщение об ошибке |
| Вещественное число |
| Число за пределами целочисленного типа |
| 6 | Поиск первого четного элемента в массиве | Массив с четным элементом | Массив чисел | Индекс элемента |
| Количество итераций необходимое для поиска индекса |
| Массив без четного элемента | Массив чисел | Сообщение о отсутствии четного элемента в массиве |
| 7 | Сортировка элементов массива методом простого включения | Массив | Массив чисел | Отсортированный массив чисел |
| 8 | Поиск в отсортированном массиве (Бин. поиск) | Массив | Элемент присутствует в массиве | Индекс элемента |
| Искомый элемент массива | Количество итераций цикла |
| Массив | Элемент в массиве отсутствует | Сообщение о отсутствии элемента в массиве |
| Искомый элемент массива |
| **Вспомогательные функции** | | | | |
| 9 | Ввод целого числа с заданными границами | Целое число | Число в пределах границ | Число |
| Целое число | Строка | Сообщение об ошибке |
| Вещественное число |
| Целое число, которое не помещается в целый тип данных (происходит переполение) |
| Число выходит за пределы указанного |

## Алгоритм (блок-схема)

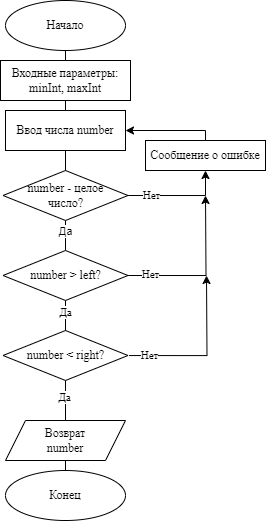
Для дальнейшей реализации проектирования необходимо составить блок схему. Для упрощения программы она разбита на функции. В каждой функции выполняется отдельная часть программы. Вызов всех функций связывает оператор выбора меню в функции main, в которой в зависимости от вводимого значения вызывается определенная функция. Так же в отдельную функцию вынесен ввод целого числа в вводимыми границами. Благодаря использованию функционального подхода к проектированию структуры программы достигается понятность в ее написании и будущем расширении возможностей программы.

******

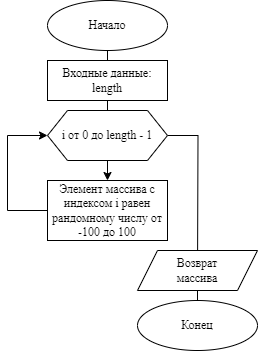
***Рис. 1.3.1. Блок-схема операции Main (Часть 1)***

******

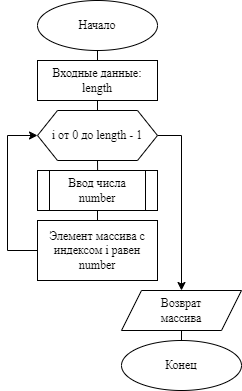
***Рис. 1.3.2. Блок-схема операции Main (Часть 2)”***

******

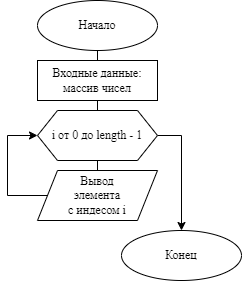
***Рис. 1.3.3. Блок-схема операции “Ввод числа”***

******

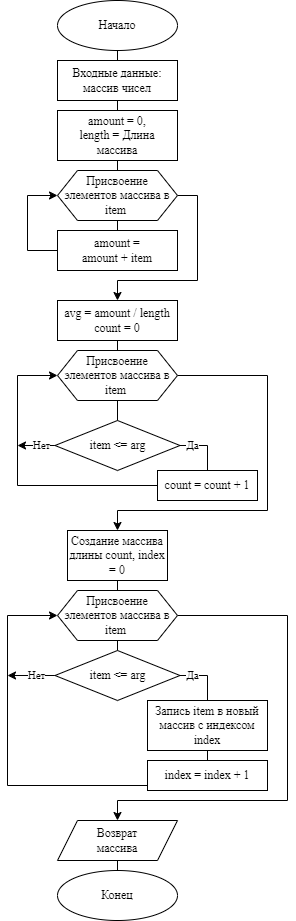
***Рис. 1.3.4. Блок-схема операции “Формирование массива из рандомных чисел”***



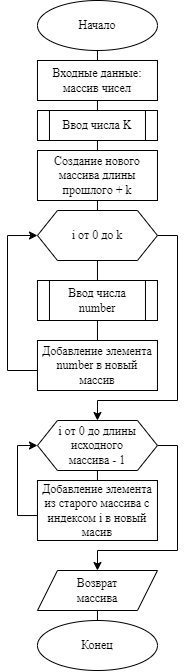
***Рис. 1.3.5. Блок-схема операции “Формирование массива из входных чисел***

***”***

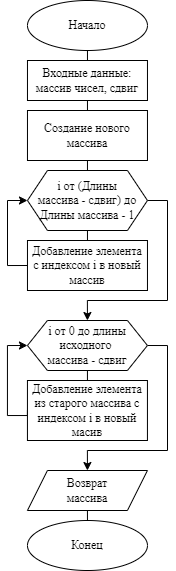
***Рис. 1.3.6. Блок-схема операции “Вывод массива”***



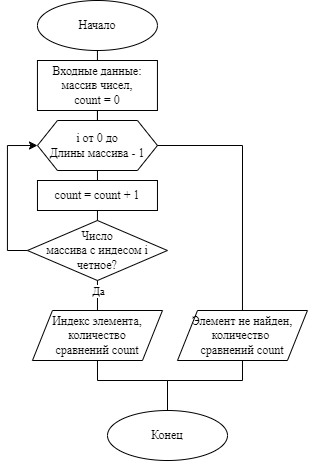
***Рис. 1.3.7. Блок-схема операции “Удаление элементов из массива”***



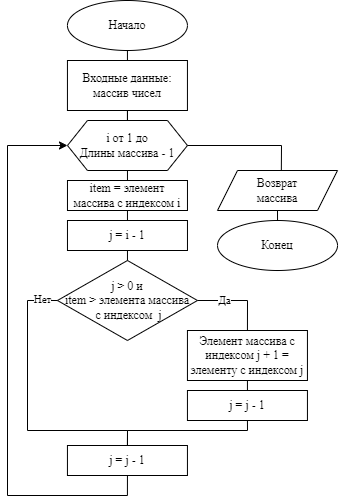
***Рис. 1.3.8. Блок-схема операции “Добавление элементов в массив”***



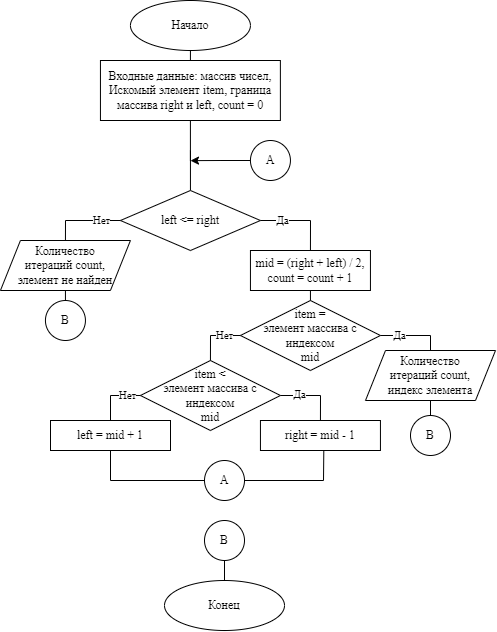
***Рис. 1.3.9. Блок-схема операции “Циклический сдвиг элементов в массиве”***



***Рис. 1.3.10. Блок-схема операции “Поиск первого четного элемента”***

******

***Рис. 1.3.11. Блок-схема операции “Сортировка элементов в массиве”***



***Рис. 1.3.12. Блок-схема операции “Бинарный поиск элемента в массиве”***

## Программа (листинг)

## Основная функции программы

/// <summary>

/// Основная функция

/// </summary>

/// <param name="number">Номер списка меню</param>

/// <param name="flag">Проверка вывода возможных пунктов меню</param>

/// <param name="arr">Основной массив</param>

/// <param name="isWork">Переменная завершения работы программы</param>

static void Main(string[] args)

{

Console.WriteLine("Введите номер того пунка, который хотите выполнить.");

Console.WriteLine("1. Сформировать массив из случайных чисел заданной длины.");

Console.WriteLine("2. Сформировать массив из входных данных заданной длины.");

int number;

bool extendMenu = false;

int[] arr = new int[1];

bool isWorking = true;

number = GetInt(1, 2, "Число не пренадлежит списку. Введите номер пункта еще раз.");

// Меню программы

while (isWorking)

{

switch (number)

{

case 1:

{

Console.Clear();

Console.WriteLine("Введите длину массива.");

int lenght = GetInt(1, 2147483591, "Длина массива должна быть натуральным числом. Повторите ввод числа.");

arr = CreateRandomArray(lenght); // Формирование рандомного массива

Console.WriteLine($"Сформирован массив длины: {arr.Length}");

break;

};

case 2:

{

Console.Clear();

Console.WriteLine("Введите длину массива.");

int lenght = GetInt(1, 2147483591, "Длина массива должна быть натуральным числом. Повторите ввод числа.");

arr = CreateArray(lenght); // Формирование массива из вводимых чисел

Console.WriteLine($"Сформирован массив длины: {arr.Length}");

break;

}

case 3 when extendMenu:

{

Console.Clear();

Console.WriteLine("Вывод элементов массива в строку через пробел: ");

PrintArray(arr); // Вывод элементов массива в консоль

break;

};

case 4 when extendMenu:

{

Console.Clear();

arr = DeliteItems(arr); // Удаление значений больших среднего арифметического

break;

};

case 5 when extendMenu:

{

Console.Clear();

arr = AddItems(arr); // Добавление элементов в массив

break;

}

case 6 when extendMenu:

{

Console.Clear();

Console.WriteLine("Введите на какое количество элементов вы хотите сдвинуть элементы массива");

int shift = GetInt(1, int.MaxValue, "Ввод отрицательных значений со сдвигом влево по уловию задачи не предусмотрен. Введите число еще раз.");

// Обработка ошибки ввода сдвига, большего длины массива

shift %= arr.Length;

arr = ShiftElements(arr, shift); // Циклический сдвиг элементов массива

break;

}

case 7 when extendMenu:

{

Console.Clear();

FindFirstEvenItem(arr); // Поиск первого четного элемента

break;

}

case 8 when extendMenu:

{

Console.Clear();

arr = SlowSort(arr); // Медленная сортировка массива O(n\*n)

break;

}

case 9 when extendMenu:

{

Console.Clear();

Console.WriteLine("Введите элемент, который хотите найти.");

int item = GetInt(int.MinValue, int.MaxValue, "");

arr = SlowSort(arr); // Медленная сортировка массива

// Поиск элемента в отсортированном массиве (Бин поиск)

int index = BinSearch(arr, item, 0, arr.Length - 1);

if (index != -1)

{

Console.WriteLine($"Элемент {item} стоит под индексом {index} в отсортированном массиве:");

PrintArray(arr);

}

else

{

Console.WriteLine($"Элемент {item} не найден в массиве.");

}

break;

}

case 10 when extendMenu:

{

Console.Clear();

Console.WriteLine("Завершение работы.");

isWorking = false;

break;

}

}

if (!isWorking) {

break;

}

extendMenu = true;

// Вывод меню

Console.WriteLine("");

Console.WriteLine("Введите номер того пунка, который хотите выполнить.");

Console.WriteLine("1. Сформировать массив из случайных чисел заданной длины.");

Console.WriteLine("2. Сформировать массив из входных данных заданной длины.");

Console.WriteLine("3. Вывод элементов массива.");

Console.WriteLine("4. Удалить все элементы больше среднего арифметического элементов массива.");

Console.WriteLine("5. Добавить выбранное количество элементов в начало массива.");

Console.WriteLine("6. Сдвинуть циклически на указаное количество элементов вправо.");

Console.WriteLine("7. Поиск первого четного элемента массива.");

Console.WriteLine("8. Медленная сортировка массива.");

Console.WriteLine("9. Сортировка массива и бинарный поиск элемента.");

Console.WriteLine("10. Выход");

Console.WriteLine();

number = GetInt(1, 10, "Число не пренадлежит списку. Введите номер пункта еще раз.");

}

## Считывание числа

/// <summary>

/// Ввод числа

/// </summary>

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

/// <param name="isConvert">Проверка правильности ввода</param>

/// <returns>Введенное число number</returns>

static int GetInt(int minInt, int maxInt, string errorMessage)

{

bool isConvert;

int number;

// Проверка корректности ввода числа

do

{

isConvert = int.TryParse(Console.ReadLine(), out number);

if (!isConvert)

{

Console.WriteLine("Некорректный ввод. Повторите ввод числа.");

}

else if (number < minInt)

{

Console.WriteLine(errorMessage);

isConvert = false;

}

else if (number > maxInt)

{

Console.WriteLine(errorMessage);

isConvert = false;

};

} while (!isConvert);

return number;

}

## Создание массива из рандомных чисел

/// <summary>

/// Создание массива из случайных чисел

/// </summary>

/// <param name="lenght">Длина массива</param>

/// <param name="arr">Массив чисел</param>

/// <param name="rand">Объект класса random</param>

/// <returns>Созданный массив arr</returns>

static int[] CreateRandomArray(int lenght)

{

int[] arr = new int[lenght];

var rand = new Random();

// Формирование массива из рандомных элементов

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

{

arr[i] = rand.Next(-100, 100);

};

return arr;

1. }

## Создание массива из вводимых значений

/// <summary>

/// Создание массива из вводимых значений

/// </summary>

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

/// <param name="lenght">Длина массива</param>

/// <param name="arr">Массив чисел</param>

/// <returns>Созданный массив arr</returns>

static int[] CreateArray(int lenght)

{

int number;

int[] arr = new int[lenght];

// Формирование массива из введенных значений

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

{

Console.WriteLine($"Введите число для формирования массива. Осталось ввести {lenght - i}");

number = GetInt(int.MinValue, int.MaxValue, "");

arr[i] = number;

}

return arr;

}

## Печать массива

/// <summary>

/// Вывод массива

/// </summary>

/// <param name="arr">Массив чисел</param>

/// <returns>Созданный массив arr</returns>

static void PrintArray(int[] arr)

{

// Вывод массива в консоль в 1 строчку

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

{

Console.Write(arr[i] + " ");

}

Console.WriteLine();

}

## Удаление элементов массива

/// <summary>

/// Удаление элементов больших среднего арифметического

/// </summary>

/// <param name="amount">Сумма чисел массива</param>

/// <param name="avg">Среднее арифметическое</param>

/// <param name="arr">Исходный массив чисел</param>

/// <param name="count">Длина нового массива</param>

/// <param name="newArr">Преобразованный массив чисел</param>

/// <param name="index">Индекс элемента</param>

/// <returns>Преобразованный массив чисел newArr</returns>

static int[] DeliteItems(int[] arr)

{

long amount = 0;

// Нахождение суммы элементов массива для среднего арифметического

foreach (int item in arr)

{

amount += item;

}

int count = 0;

long avg = amount / arr.Length;

// Поиск длины нового массива

foreach (int item in arr)

{

if (avg >= item)

{

count += 1;

}

}

int[] newArr = new int[count];

int index = 0;

// Запись элементов меньших среднего арифметического в новый массив

foreach (int item in arr)

{

if (avg >= item)

{

newArr[index] = item;

index++;

}

}

Console.WriteLine($"Удалены все элементы, большие {avg}");

return newArr;

}

## Добавление элементов в массив

/// <summary>

/// Добавление элементов в массив

/// </summary>

/// <param name="k">Количество добавляемых элементов</param>

/// <param name="arr">Исходный массив чисел</param>

/// <param name="newArr">Преобразованный массив чисел</param>

/// <returns>Преобразованный массив чисел newArr</returns>

static int[] AddItems(int[] arr)

{

Console.WriteLine("Введите количество элементов, которое хотите добавить в массив.");

int k = GetInt(1, int.MaxValue - arr.Length, "Количество элементов должно быть натуральным числом. Повторите ввод еще раз.");

int[] newArr = new int[k + arr.Length];

// Ввод новых элементов и их запись в новый массив

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

{

Console.WriteLine($"Введите число. Осталось ввести {k - i} чисел.");

newArr[i] = GetInt(int.MinValue, int.MaxValue, "");

}

// добавление элементов исходного массива в новый

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

{

newArr[i + k] = arr[i];

}

Console.WriteLine($"Добавлено чисел в массив - {k}. Суммарная длина полученного массива - {newArr.Length}.");

return newArr;

}

## Циклический сдвиг элементов в массиве

/// <summary>

/// Циклический сдвиг элементов массива

/// </summary>

/// <param name="shift">Число сдвига</param>

/// <param name="arr">Исходный массив чисел</param>

/// <param name="newArr">Преобразованный массив чисел</param>

/// <param name="index">Индекс элемента</param>

/// <returns>Преобразованный массив чисел newArr</returns>

static int[] ShiftElements(int[] arr, int shift)

{

int[] newArr = new int[arr.Length];

int index = 0;

// запись последних shift элементов в новый массив

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

{

newArr[index] = arr[i];

index++;

}

// Запись оставшихся элементов в новый массив

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

{

newArr[index] = arr[i];

index++;

}

Console.WriteLine($"Выполен циклический сдвиг вправо на {shift} элементов.");

return newArr;

}

## Поиск первого четного элемента в массиве

/// <summary>

/// Поиск первого четного элемента в массиве

/// </summary>

/// <param name="arr">Исходный массив чисел</param>

/// <param name="count">Количество итераций</param>

/// <param name="elem">Найденный элемент</param>

/// <param name="flag">Показатель найден ли четный элемент</param>

static void FindFirstEvenItem(int[] arr)

{

int count = 0;

int elem = 0;

bool isEvenNumberFound = false;

// Поиск первого четного элемента

foreach (int item in arr)

{

count++;

if (item % 2 == 0)

{

isEvenNumberFound = true;

elem = item;

break;

};

}

if (isEvenNumberFound)

{

Console.WriteLine($"Первый четный элемент массива - {elem}. Количество сравнений - {count} из {arr.Length} возможных.");

}

else

{

Console.WriteLine($"Четных элементов в массиве нет. Количество проведнных сравнений - {count}.");

};

}

## Сортировка элементов

/// <summary>

/// Медленная сортировка

/// </summary>

/// <param name="arr">Исходный массив чисел</param>

/// <param name="j">Индекс элемента</param>

/// <param name="item">Буфер для сохранения числа</param>

/// <returns>Отсортированный массив чисел newArr</returns>

static int[] SlowSort(int[] arr)

{

int j, item;

for (int i = 1; i < arr.Length; i++)

{

item = arr[i];

j = i - 1;

// сравнение элемента со всеми последующими до тех пор, пока он больше

while (j >= 0 && item < arr[j])

{

arr[j + 1] = arr[j];

j--;

}

arr[j + 1] = item;

}

return arr;

}

## Бинарный поиск

/// <summary>

/// Поиск элемента (Бин. Поиск)

/// </summary>

/// <param name="arr">Исходный массив чисел</param>

/// <param name="item">Искомое значение</param>

/// <param name="left">Левая граница</param>

/// <param name="right">Правая граница</param>

/// <param name="mid">Центральное значение индекса left и right</param>

/// <returns>Индекс искомого элемента в массиве</returns>

static int BinSearch(int[] arr, int item, int left, int right)

{

int count = 0;

while (left <= right)

{

int mid = (left + right) / 2;

count++;

if (item == arr[mid]) // сравнение элемента с элементом посередине между границами left и right

{

Console.WriteLine($"Количество проведенных интераций {count}.");

return ++mid;

}

else if (item < arr[mid])

{

right = mid - 1; // смещение границ

}

else

{

left = mid + 1; // смещение границ

}

}

Console.WriteLine($"Количество проведенных интераций {count}.");

return -1;

}

## Тестирование программы

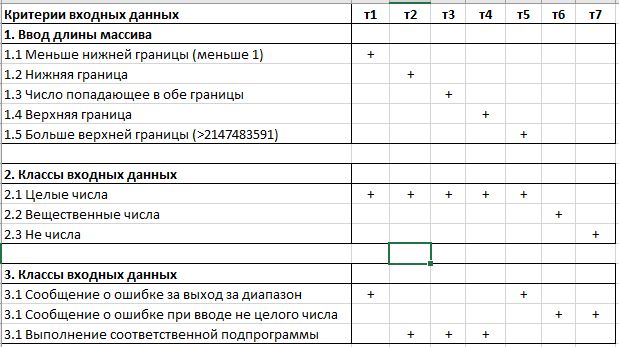
Дли упрощения тестирования программы будут тестироваться отдельные части ввода. Для ввода данных в программе создана отдельная функция ввода числа с заданными границами, поэтому ввод различных значений будет отличаться только установленными границами. Все вводимые значения можно разделить на: номер меню, длина массива и целое число. Границы ввода номера меню зависит места. При первоначальном создании массива предоставленных для пользователя номеров меню всего 2, то есть границы от 1 до 2х. В дальнейшем пользователю предлагается выбирать из 10ти разных функций программы, то есть границы становятся от 1 до 10ти. При вводе длины массива нужно учесть, что длина максимальная длина массива может быть 2147483591. При вводе целого числа, ограничения ставятся в соответствии размерности используемых типов данных, то есть от -2147483648 до 2147483647. Тестировании ввода эти значений представлены в таблицах ниже.



***Рис. 1.5.1. Критерии тестирования ввода выбора меню.***



***Рис. 1.5.2. Тесты ввода выбора меню.***



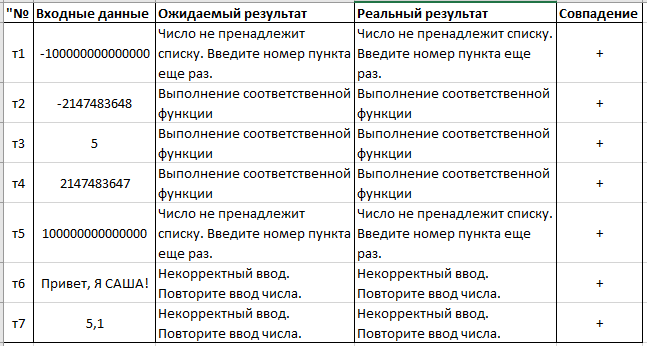
***Рис. 1.5.3. Критерии тестирования ввода длины последовательности.***



***Рис. 1.5.4. Тесты ввода длины последовательности.***

******

***Рис. 1.5.5. Критерии тестирования ввода целого числа.***

******

***Рис. 1.5.6. Тесты ввода целого числа.***