**Федеральное государственное автономное образовательное учреждение высшего образования   
"Национальный исследовательский университет   
"Высшая школа экономики"**

*Факультет социально-экономических и компьютерных наук*

Панфилов Даниил Сергеевич

**ЛАБОРАТОРНАЯ РАБОТА №4**

*Работа с одномерными массивами*

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

Руководитель

ученая степень, ученое

звание, должность

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

И.О. Найданов

Пермь, 2023 год

# 1. Постановка задачи

## 1.1. Общая постановка задачи

1. Сформировать массив из n элементов:

а) с помощью датчика случайных чисел (количество элементов массива задается пользователем с клавиатуры);

б) пользователь вводит элементы с клавиатуры (количество элементов массива задается пользователем с клавиатуры);

1. Распечатать массив.
2. Выполнить удаление указанных элементов из массива (задание 1).
3. Выполнить добавление указанных элементов в массив (задание 2).
4. Выполнить перестановку элементов в массиве (задание 3).
5. Выполнить поиск указанных в массиве элементов и подсчитать количество сравнений, необходимых для поиска нужного элемента.
6. Выполнить сортировку массива указанным методом.
7. Выполнить поиск элемента, который вводит пользователь с клавиатуры, в отсортированном массиве (бинарный поиск) и подсчитать количество сравнений, необходимых для поиска нужного элемента.

## 1.2. Частная постановка задачи

Вариант №2:

1. Сформировать массив из n элементов:

а) с помощью датчика случайных чисел (количество элементов массива задается пользователем с клавиатуры);

б) пользователь вводит элементы с клавиатуры (количество элементов массива задается пользователем с клавиатуры);

1. Распечатать массив.
2. Выполнить удаление минимального элемента массива.
3. Выполнить добавление K элементов в конец массива (K – число, вводимое пользователем).
4. Сдвинуть циклически на M элементов вправо (M – число, вводимое пользователем).
5. Выполнить поиск первого отрицательного элемента в массиве и подсчитать количество сравнений, необходимых для поиска нужного элемента.
6. Выполнить сортировку массива простым выбором.
7. Выполнить поиск элемента, который вводит пользователь с клавиатуры, в отсортированном массиве и подсчитать количество сравнений, необходимых для поиска нужного элемента.
8. Завершение работы

# 2. Анализ задачи

В начале и на протяжении всей работы с программой требуется выбрать пункт в меню, для этого понадобится переменная ans.

Таблица 1 – Классы входных и выходных данных для переменной ans

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| ans – число, которое меньше 11 и больше 0 (int) | Программа выполняется | ans – число, которое меньше 11 и больше 0 (int) |
| ans – строка | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| ans – число, больше 11 или меньше 0 (int) | Ошибка | Сообщение об ошибке, просьба ввести повторно |

## 2.1. Анализ данных для 1 задачи

Для ввода массива вручную понадобиться знать его длину, за неё отвечает переменная len. Та же самая переменная принимает значение длины массива, у которого все элементы генерируются случайно (от -100 до 100, в задании не написан промежуток, так что было принято взять такой).

Таблица 2 – Классы входных и выходных данных для переменной len

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| len – положительное число (int) | Программа выполняется | len – положительное число (int) |
| len – отрицательное число (int) | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| len = 0 | Программа выполняется | len = 0 |
| len – нецелое число | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| len – строка | Ошибка | Сообщение об ошибке, просьба ввести повторно |

Далее происходит ввод элементов массива, переменная element принимает значение каждого элемента по очереди.

Таблица 3 – Классы входных и выходных данных для переменной element

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| element – положительное число (int) | Программа выполняется | element – положительное число (int) |
| element – отрицательное число (int) | Программа выполняется | element – отрицательное число (int) |
| element = 0 | Программа выполняется | element = 0 |
| element – нецелое число | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| element – строка | Ошибка | Сообщение об ошибке, просьба ввести повторно |

## 2.2. Анализ для задачи 2

Для задачи 2 не требуется ввод, она служит для того, чтобы выводить на экран полученный массив.

## 2.3. Анализ для задачи 3

Для задачи 3 также не требуется ввод, программа работает автономно.

## 2.4. Анализ для задачи 4

Для ввода массива вручную понадобиться знать его длину, за неё отвечает переменная count.

Таблица 4 – Классы входных и выходных данных для переменной count

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| count – положительное число (int) | Программа выполняется | count – положительное число (int) |
| count – отрицательное число (int) | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| count = 0 | Программа выполняется | count = 0 |
| count – нецелое число | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| count – строка | Ошибка | Сообщение об ошибке, просьба ввести повторно |

Далее происходит ввод элементов массива, переменная element принимает значение каждого элемента по очереди.

Таблица 5 – Классы входных и выходных данных для переменной element

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| element – положительное число (int) | Программа выполняется | element – положительное число (int) |
| element – отрицательное число (int) | Программа выполняется | element – отрицательное число (int) |
| element = 0 | Программа выполняется | element = 0 |
| element – нецелое число | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| element – строка | Ошибка | Сообщение об ошибке, просьба ввести повторно |

## 2.5. Анализ для задачи 5

Для ввода массива вручную понадобиться знать количество элементов, которое мы хотим ввести, за него отвечает переменная num.

Таблица 6 – Классы входных и выходных данных для переменной num

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| num – положительное число (int) | Программа выполняется | num – положительное число (int) |
| num – отрицательное число (int) | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| num = 0 | Программа выполняется | num = 0 |
| num – нецелое число | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| num – строка | Ошибка | Сообщение об ошибке, просьба ввести повторно |

Далее программа работает автономно, после выполнения появляется сообщение об окончании работы.

## 2.6. Анализ для задачи 6

Для задачи 6 также не требуется ввод, программа работает автономно.

## 2.7. Анализ для задачи 7

Для задачи 7 также не требуется ввод, программа работает автономно.

## 2.8. Анализ для задачи 8

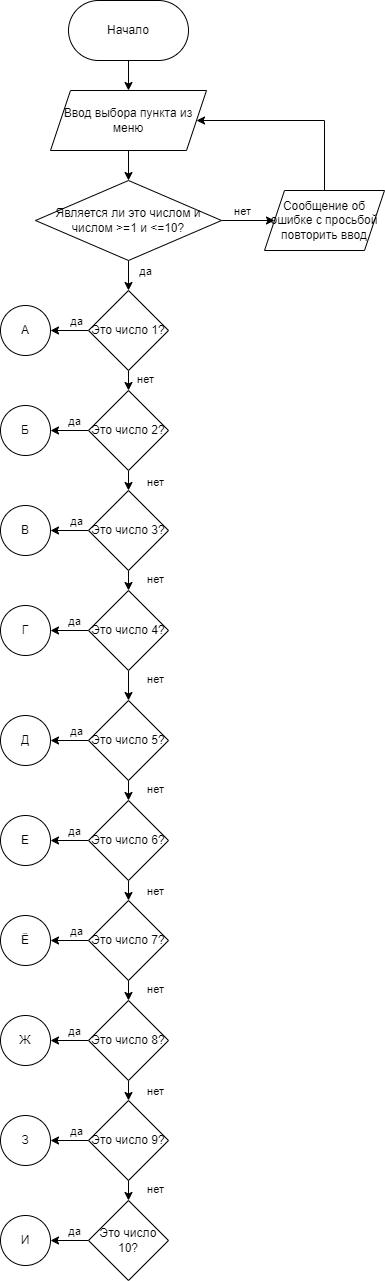
Для хранения значения конкретного элемента, который необходимо найти, была создана переменная find.

Таблица 6 – Классы входных и выходных данных для переменной find.

|  |  |  |
| --- | --- | --- |
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| find – число (int), которое меньше или равно максимальному числу в отсортированном массиве и больше или равно минимальному числу в отсортированном массиве | Программа выполняется | find – число (int), которое меньше или равно максимальному числу в отсортированном массиве и больше или равно минимальному числу в отсортированном массиве |
| find – строка (string) | Ошибка | Сообщение об ошибке, просьба ввести повторно |
| find – нецелое число | Ошибка | Сообщение об ошибке, просьба ввести повторно |

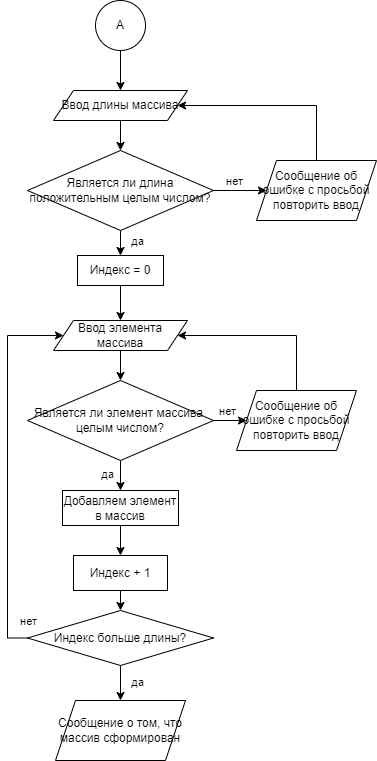
# 3. Проектирование

Для дальнейшей реализации проектирования необходимо составить блок схему для каждой задачи, она представлена ниже. Для удобства она будет поделена на каждые рисунка в соответствии с задачами.



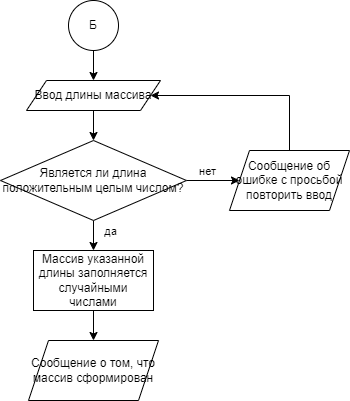
***Рисунок 1 – Схема алгоритма для пункт меню***

Ниже представлена блок-схема работы задачи 1а).



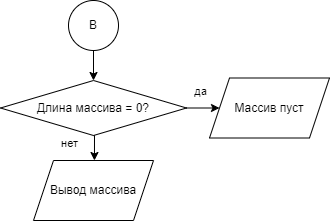
***Рисунок 1.1. – Схема алгоритма для задачи 1а)***

Ниже представлена блок-схема для задачи 1б).



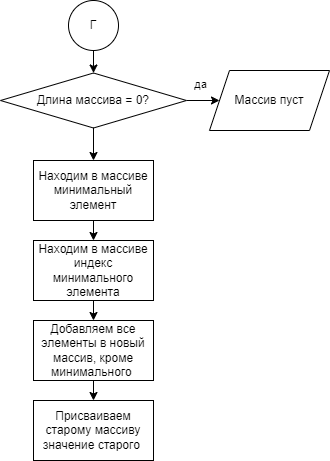
***Рисунок 1.2. – Схема алгоритма для задачи 1б)***

Ниже представлена блок-схема для задачи 2.



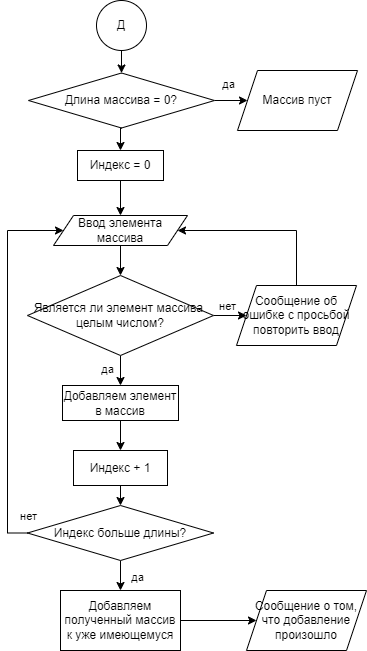
***Рисунок 2 – Схема алгоритма для задачи 2***

Ниже представлена блок-схема для задачи 3.



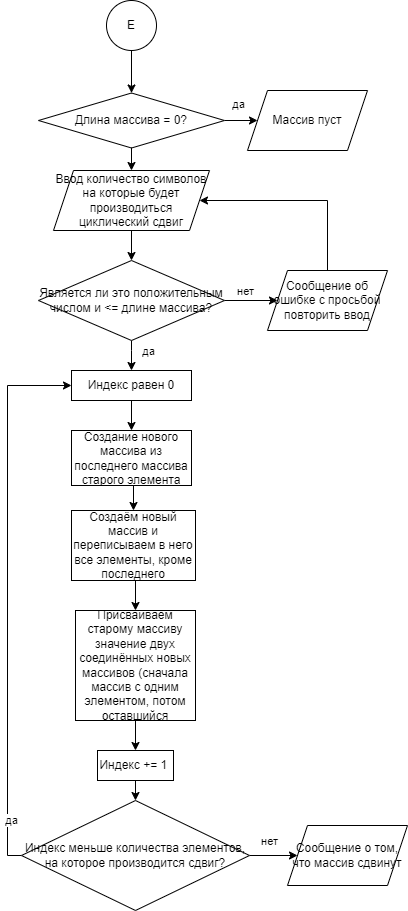
***Рисунок 3 – Схема алгоритма для задачи 3***

Ниже представлена блок-схема для задачи 4.



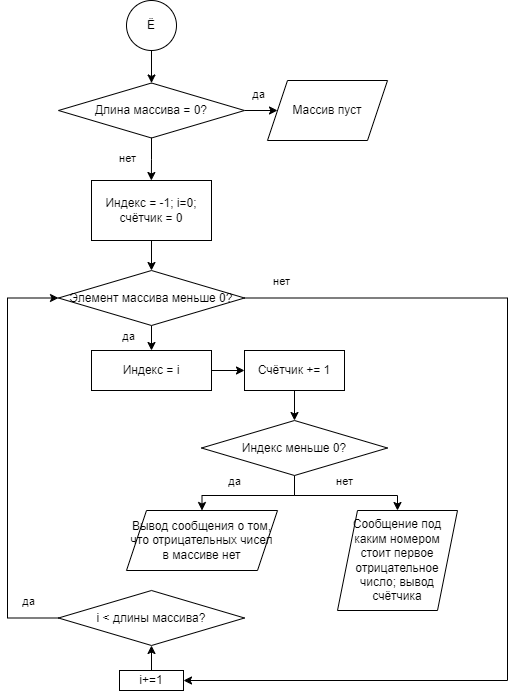
***Рисунок 4 – Схема алгоритма для задачи 4***

Ниже представлена блок-схема для задачи 5.



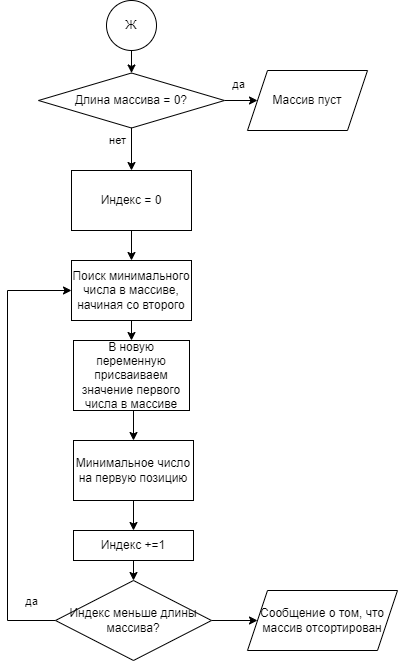
***Рисунок 5 – Схема алгоритма для задачи 5***

Ниже представлена блок-схема для задачи 6.



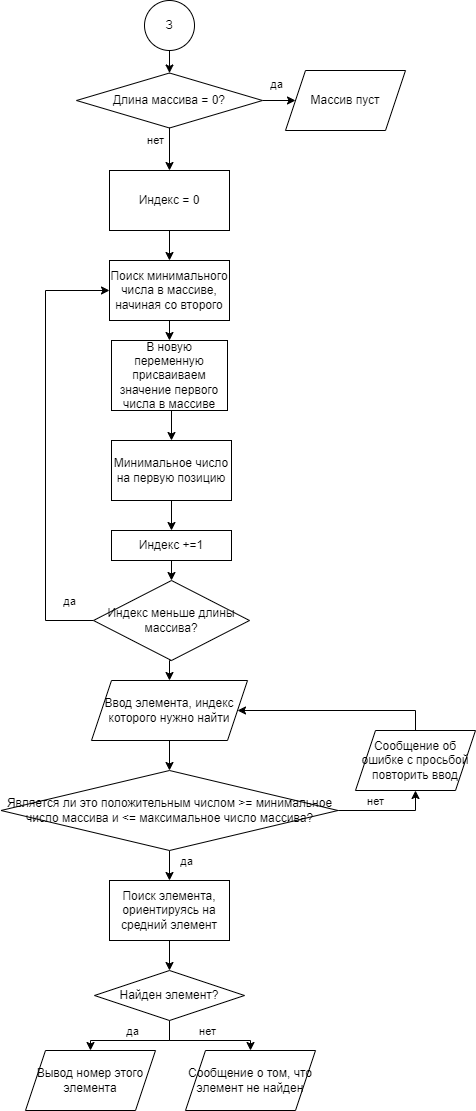
***Рисунок 6 – Схема алгоритма для задачи 6***

Ниже представлена блок-схема для задачи 7.



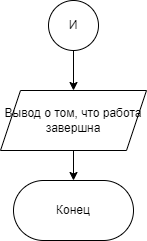
***Рисунок 7 – Схема алгоритма для задачи 7***

Ниже представлена блок-схема для задачи 8.



***Рисунок 8 – Схема алгоритма для задачи 8***

Ниже представлена блок-схема для задачи 9.



***Рисунок 9 – Схема алгоритма для задачи 9***

# 4. Разработка

using System;

using System.Collections.Generic;

using System.Collections.Specialized;

using System.Diagnostics;

using System.IO.Pipes;

using System.Linq;

using System.Linq.Expressions;

using System.Net.Sockets;

using System.Runtime.CompilerServices;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

namespace Лаба\_4

{

internal class Program

{

static void Main(string[] args)

{

int ans; // номер действия, которое выбирает пользователь

int[] arr = new int[0]; // основной массив

do

{

bool isConvertAns;

Console.WriteLine("1) Сформировать массив самостоятельно");

Console.WriteLine("2) Сформировать массив при помощи ДСЧ");

Console.WriteLine("3) Распечатать массив");

Console.WriteLine("4) Удалить минимальный элемент массива");

Console.WriteLine("5) Переставить K элементов в конец массива");

Console.WriteLine("6) Циклически сдвинуть массив на М клеток вправо");

Console.WriteLine("7) Найти первый отрицательный элемент в массиве и посчитать, сколько нужно было сделать сравнений, чтобы его найти");

Console.WriteLine("8) Выполнить сортировку массива простым выбором");

Console.WriteLine("9) Выполнить поиск элемента, который вводит пользователь с клавиатуры, в отсортированном массиве и подсчитать количество сравнений, необходимых для поиска нужного элемента.");

Console.WriteLine("10) Выход");

do

{

isConvertAns = int.TryParse(Console.ReadLine(), out ans);

if (!isConvertAns)

{

Console.WriteLine("Неправильно введено число, \nпопробуйте ещё раз");

}

} while (!isConvertAns);

switch (ans)

{

case 1: //формирование массива вручную

{

int len;

int element;

bool isLenCorrest;

bool isElCorrect;

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

do

{

isLenCorrest = int.TryParse(Console.ReadLine(), out len);

if (!isLenCorrest | len < 0)

{

Console.WriteLine("Введите, пожалуйста, положительное число (длину):");

}

} while (!isLenCorrest | len < 0);

arr = new int[len];

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

{

do

{

Console.WriteLine($"Введите элемент массива {i + 1}: ");

isElCorrect = int.TryParse(Console.ReadLine(), out element);

if (!isElCorrect)

{

Console.WriteLine("Введите корректный элемент последовательности:");

}

} while (!isElCorrect);

arr[i] = element;

}

Console.WriteLine("Массив сформирован");

break;

}

case 2: //формирование массива при помощи ДСЧ

{

int len;

bool isLenCorrest;

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

do

{

isLenCorrest = int.TryParse(Console.ReadLine(), out len);

if (!isLenCorrest | len <= 0)

{

Console.WriteLine("Введите, пожалуйста, положительное число (длину):");

}

} while (!isLenCorrest | len <= 0);

Random rnd = new Random();

int[] temp = new int[len];

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

{

temp[i] = rnd.Next(-100, 100);

}

Console.WriteLine("Массив сформирован");

arr = temp;

break;

}

case 3: //печать массива

{

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

foreach (int item in arr)

Console.Write($"{item} ");

Console.WriteLine();

break;

}

case 4: //удаление минимального элемента

{

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

int j = 0; //индекс минимального элемента

int l = 0; //счётчик в temp

int min = 1000; //значение минимума

int[] temp = new int[arr.Length - 1];

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

if (arr[i] < min)

{

min = arr[i];

}

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

{

if (arr[i] == min)

{

j = i; break;

}

}

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

{

if (i != j)

{

temp[l] = arr[i];

l++;

}

}

arr = temp;

Console.WriteLine($"Минимальный элемент удалён");

break;

}

case 5: //добавление элементов в массив

{

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

int count; //добавляемое количество элементов

int element;

bool isCountCorrect;

bool isElCorrect;

int[] temp = new int[0]; //вспомогательный массив

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

Console.WriteLine("Введите К (количество элементов, которое хотите добавить):");

do

{

isCountCorrect = int.TryParse(Console.ReadLine(), out count);

if (!isCountCorrect | count < 0)

{

Console.WriteLine("Введите, пожалуйста, положительное число (количество элементов):");

}

} while (!isCountCorrect | count < 0);

temp = new int[count];

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

{

do

{

Console.WriteLine($"Введите элемент массива {i + 1}: ");

isElCorrect = int.TryParse(Console.ReadLine(), out element);

if (!isElCorrect)

{

Console.WriteLine("Введите корректный элемент последовательности:");

}

} while (!isElCorrect);

temp[i] = element;

}

arr = arr.Concat(temp).ToArray(); //добавление чисел в конец массива

Console.WriteLine("Добавление произошло");

break;

}

case 6://сдвиг циклически на М элементов

{

int l = 0;

int j = arr.Length - 1; //индекс последнего элемента

int[] temp = new int[arr.Length-1];

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

int num; //количество элементов, на которое циклически сдвигается массив

bool isNumCorrect;

Console.WriteLine("Введите количство элементов, на которое хотите сдвинуть массив:");

do

{

isNumCorrect = int.TryParse(Console.ReadLine(), out num);

if (!isNumCorrect | num < 0 | num > arr.Length)

{

Console.WriteLine($"Введите, пожалуйста, положительное число (количество элементов), меньше или равно: {arr.Length} (длине массива):");

}

} while (!isNumCorrect | num < 0 | num > arr.Length);

for (int t = 0; t < num; t++)

{

int[] p = new int[1] { arr[arr.Length - 1] }; //массив из последнего элемента

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

{

temp[i] = arr[i];

}

arr = p.Concat(temp).ToArray();

}

Console.WriteLine("Массив сдвинут");

break;

}

case 7: //поиск первого отрицательного элемента

{

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

int count = 0;

int index = -1;

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

{

if (arr[i] < 0)

{

index = i;

break;

}

count++;

}

if (index < 0)

{

Console.WriteLine("Отрицательных элементов в массиве нет");

}

else

{

Console.WriteLine($"Первое отрицательное число находится в массиве под номером: {index + 1}");

Console.WriteLine($"Количество сравнений: {count + 1}");

}

break;

}

case 8: //сортировка массива простым выбором

{

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

int min; //индекс минимального числа

int temp; //вспомогательная переменная

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

{

min = i;

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

{

if (arr[j] < arr[min])

{

min = j;

}

}

temp = arr[min];

arr[min] = arr[i];

arr[i] = temp;

}

Console.WriteLine("Сортировка массива простым выбором выполнена");

break;

}

case 9: //поиск элемента в массиве

{

if (arr.Length == 0)

{

Console.WriteLine("Массив пуст");

break;

}

int min; //сортировка массива

int temp;

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

{

min = i;

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

{

if (arr[j] < arr[min])

{

min = j;

}

}

temp = arr[min];

arr[min] = arr[i];

arr[i] = temp;

}

int left = 0; //индекс нижней границы массива

int mid = 0; //индекс элемента в середине массива

int right = arr.Length - 1; //индекс верхней границы массива

int find; //перменная, которую надо найти

int count = 0; //счётчик количества операций

bool isItemCorrect;

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

do

{

isItemCorrect = int.TryParse(Console.ReadLine(), out find);

if (!isItemCorrect | find > arr[arr.Length-1] | find < arr[0])

{

Console.WriteLine($"Введите, пожалуйста, число. Оно должно быть >= {arr[0]} и <= {arr[arr.Length-1]}");

}

} while (!isItemCorrect | find > arr[arr.Length - 1] | find < arr[0]);

do

{

mid = (left + right) / 2;

if (arr[mid] < find) left = mid + 1;

else right = mid;

count++;

} while (left != right);

if (arr[left] == find)

{

Console.WriteLine($"Номер элемента {find} равен: {left + 1}");

Console.WriteLine($"Количество операций: {count}");

}

else Console.WriteLine("Элемент не найден");

break;

}

case 10: //конец работы

{

Console.WriteLine("Работа завершена");

break;

}

default:

{

Console.WriteLine("Неправильно задан пункт меню");

break;

}

}

} while (ans != 10);

}

}

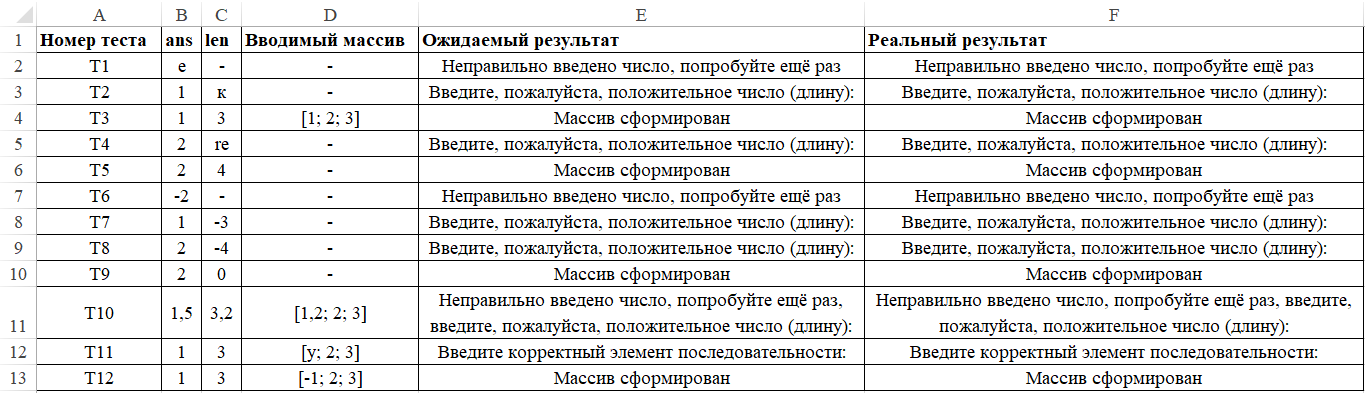
}

# 5. Тестирование

В данном пункте следует рассматривать только задачи с входными и выходными данными (номера: 1, 4, 5, 8).

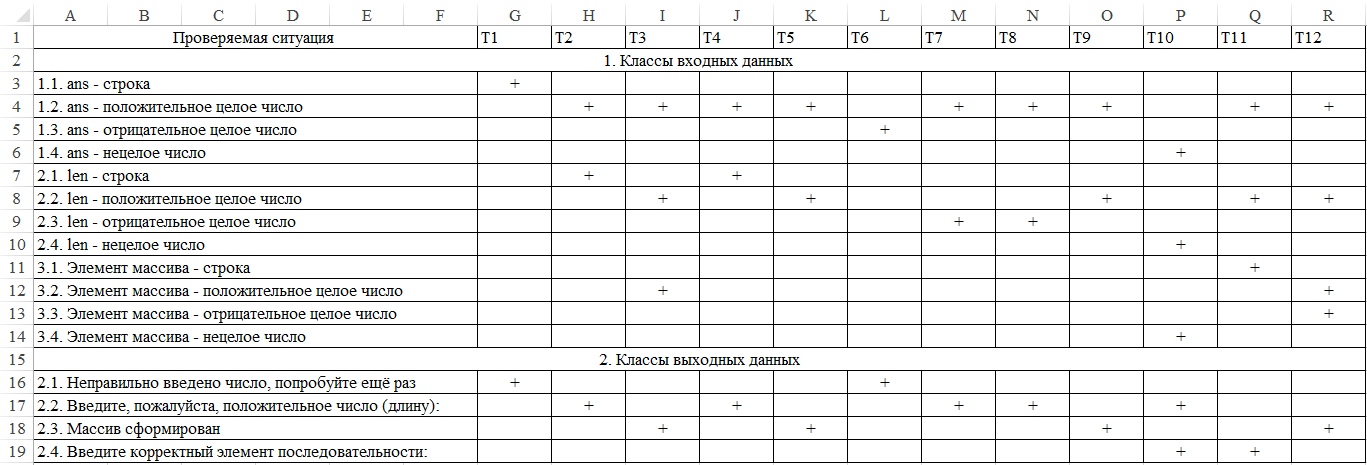
## 5.1. Тестирование задачи 1.

Результаты тестирования представлены на рисунке 5.1.1. По рисунку видно, что все ожидаемые результаты совпадают с реальными.



***Рисунок 5.1.1 – Тесты задачи 1***

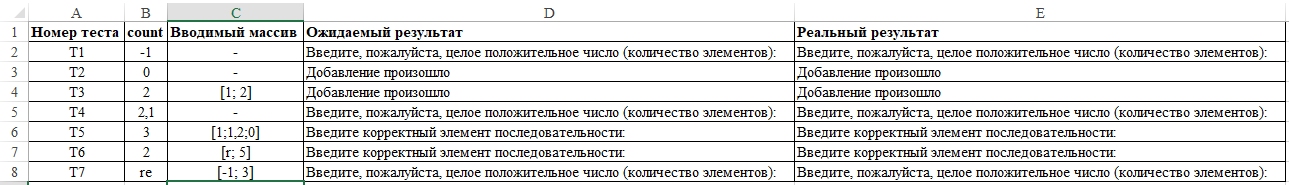
На рисунке 5.1.2. представлено тестирование по критериям черного ящика.



***Рисунок 5.1.2 – Черный ящик по задаче 1***

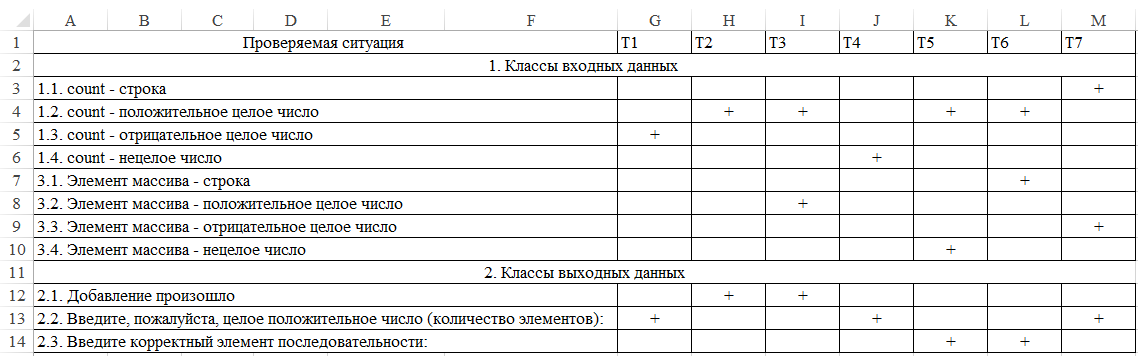
## 5.2. Тестирование задачи 4.

Результаты тестирования представлены на рисунке 5.2.1. По рисунку видно, что все ожидаемые результаты совпадают с реальными.



***Рисунок 5.2.1 – Тесты задачи 4***

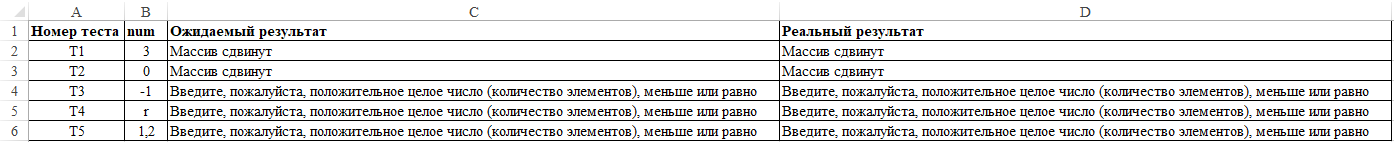
На рисунке 5.2.2. представлено тестирование по критериям черного ящика.



***Рисунок 5.2.2 – Черный ящик по задаче 4***

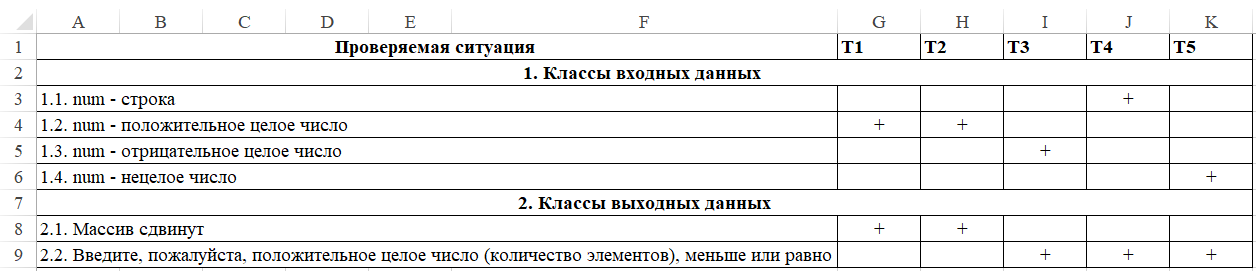
## 5.3. Тестирование задачи 5.

Результаты тестирования представлены на рисунке 5.3.1. По рисунку видно, что все ожидаемые результаты совпадают с реальными.



***Рисунок 5.3.1 – Тесты задачи 5***

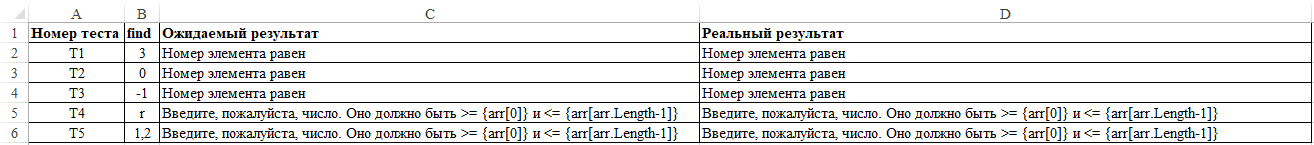
На рисунке 5.3.2. представлено тестирование по критериям черного ящика.



***Рисунок 5.3.2 – Черный ящик по задаче 5***

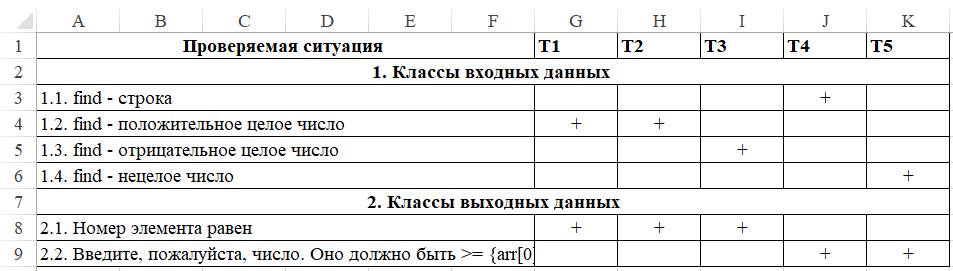
## 5.4. Тестирование задачи 8.

Результаты тестирования представлены на рисунке 5.4.1. По рисунку видно, что все ожидаемые результаты совпадают с реальными.



***Рисунок 5.4.1 – Тесты задачи 8***

На рисунке 5.4.2. представлено тестирование по критериям черного ящика.



***Рисунок 5.4.2 – Черный ящик по задаче 8***