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

*Факультет экономики, менеджмента и бизнес-информатики*

Шульжик Кирилл Владимирович

**Отчёт по практической работе №4 (Вариант 16)**

студента образовательной программы бакалавриата «Программная инженерия» по направлению подготовки 09.03.04 Программная инженерия

Преподаватель

В.Д Марквирер

**Оглавление**

[Постановка задачи 3](#_Toc89620280)

[Анализ 3](#_Toc89620281)

[Алгоритм 9](#_Toc89620282)

[Листинг 30](#_Toc89620283)

Постановка задачи

Постановка задачи выглядит следующим образом:

1. Сформировать массив из n элементов с помощью датчика случайных чисел (n задается пользователем с клавиатуры).
2. Распечатать массив.
3. Выполнить удаление всех нечетных элементов из массива.
4. Выполнить добавление элемента с номером K в массив.
5. Выполнить циклическую перестановку на M элементов вправо в массиве.
6. Выполнить поиск элемента, равному среднему арифметическому элементов в массиве, и подсчитать количество сравнений, необходимых для поиска нужного элемента.
7. Выполнить сортировку массива простым обменом.
8. Выполнить **поиск элемента, равному среднему арифметическому элементов в массиве, в отсортированном массиве** и подсчитать количество сравнений, необходимых для поиска нужного элемента.

Анализ

Меню управляется с помощью ввода целых чисел от 0 до 8, где 1-8 – соответствующие функции программы из постановки задачи, 0 – выход из программы.

z – выбор пользователя (1 из 9 функций)

Таблица 1 – Анализ исходных данных для меню

|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| z | z = 0 | Выход из программы |
| z = 1 | Формирование массива |
| z = 2 | Печать массива |
| z = 3 | Массив, состоящий только из четных элементов |
| z = 4 | Добавление элемента с номеров k в массив |
| z = 5 | Массив, элементы которого сдвинуты на M пунктов вправо |
| z = 6 | 1. Поиск элемента, равного среднему арифметическому элементов в массиве и количество сравнений, необходимых для поиска нужного элемента. |
| z = 7 | Массив, отсортированный простым обменом |
| z = 8 | Индекс элемента, равного среднему арифметическому элементов в отсортированном массиве; количество сравнений, необходимых для поиска нужного элемента |
| z != [0;8] | Ошибка ввода. Введите число от 0 до 8. |

1) Формирование массива

Исходные данные n – длина массива

string g – выбор пользователя (Число 1 или 2, где 1 – формирование массива, состоящего из n целых случайных чисел, а число 2 – формирование массива, состоящего из n целых чисел, введённых пользователем. a1…an – вводимые элементы массива.

Таблица 2 – Анализ исходных данных для формирования массива

|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| n | int32, n >= 1 | Формирование соответствующего массива |
| n = 0 | Пустой массив |
| n < 0 | Ошибка. Длина массива не может быть отрицательным числом. |
| g | g = 1 | Формирование массива, состоящего из n целых случайный чисел |
| g = 2 | Формирование массива, состоящего из n целых чисел, введённых пользователем |
| g – другое | Ошибка, введите 1 или 2. |

2) Распечатать массив

Если массив сформирован, он распечатается. Если длина массива 0, то на вывод выйдет следующее сообщение: «Массив пустой. Дальнейшие действия не будут иметь смысла сформируйте массив заново». Иначе пользователю будет предложено сформировать массив.

3) Удаление всех нечетных элементов из массива

Таблица 3 – Анализ исходных данных для функции по удалению всех нечетных элементов из массива

|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| int [] arr | int32 | Массив, состоящий только из четных элементов |

4) Добавление элемента с номером K в массив

Таблица 4 – Анализ исходных данных для функции по добавлению элемента с номером K в массив

|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| K | k = int32[0;n] | Добавление элемента с номером k в массив |
| k <0 | Ошибка. Номер элемента массива не может быть отрицательным |
| k != int 32 | Ошибка. Введите целое, положительное число. |
| znach | znach = int32 | Добавление элемента znach с номером k в массив |
| znach != int32 | Ошибка! Элементы массива – целые числа. |

5) Выполнить циклическую перестановку на M элементов вправо в массиве.

Таблица 5 – Анализ исходных данных для функции по выполнению циклической перестановки на M элементов вправо

|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| M | M>=0 | Массив, элементы которого сдвинуты на M пунктов вправо. |
| M < 0 | Ошибка! Перестановка может осуществляться только на неотрицательное количество элементов вправо. |
| M != int32 | Ошибка! Введите целое число. |

6) Выполнить поиск элемента, равного среднему арифметическому элементов в массиве, и подсчитать количество сравнений, необходимых для поиска нужного элемента.

Таблица 6 – Анализ исходных данных для функции по поиску среднего арифметического и нахождения количества сравнений в массиве

|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| srznach | double | Среднее арифметическое элементов массиве. |
| kolvo | int32 | Количество сравнений, необходимых для поиска нужного элемента |
| index | int32 | Номер элемента, равного среднему арифметическому элементов в массиве. |

7) Отсортировать массив простым обменом

Таблица 7 – Анализ исходных данных для функции по сортировке простым обменом

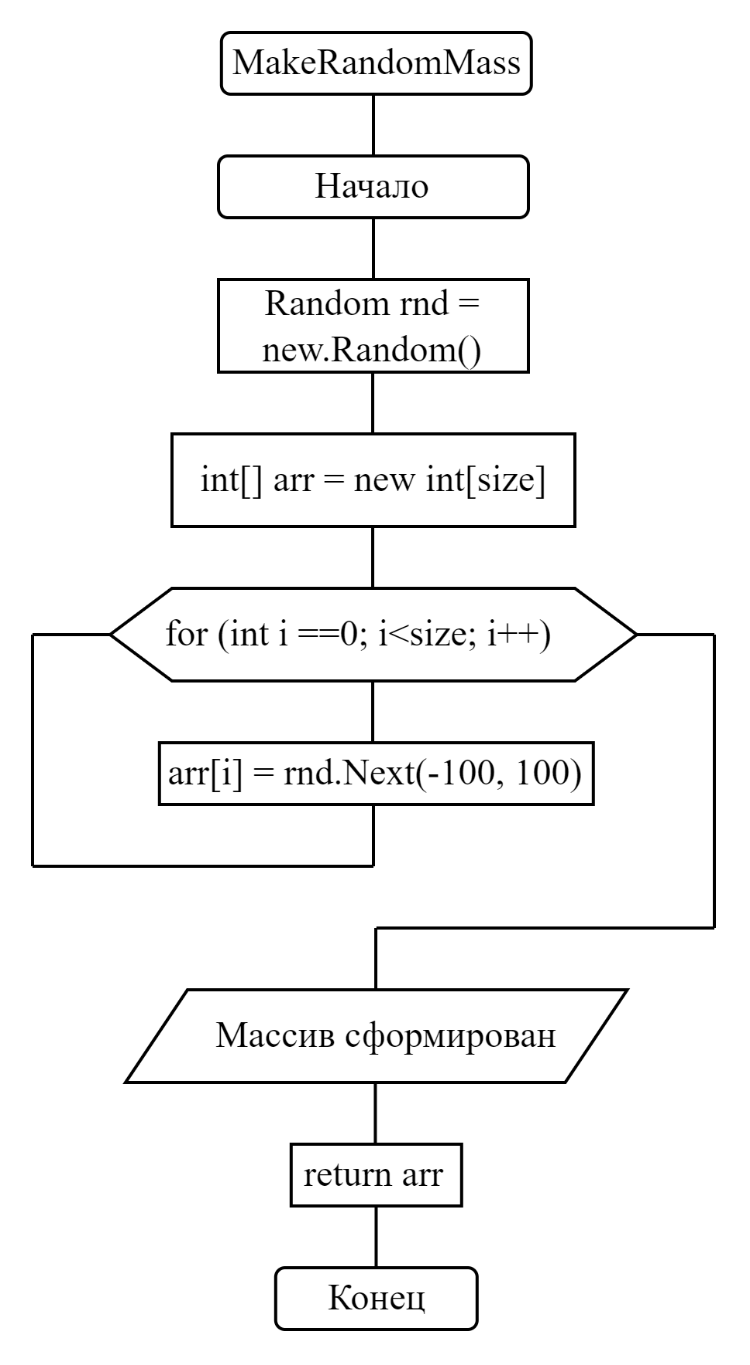
|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| temp | int32 | Массив, отсортированный простым обменом. |
| size | int32 |

8)

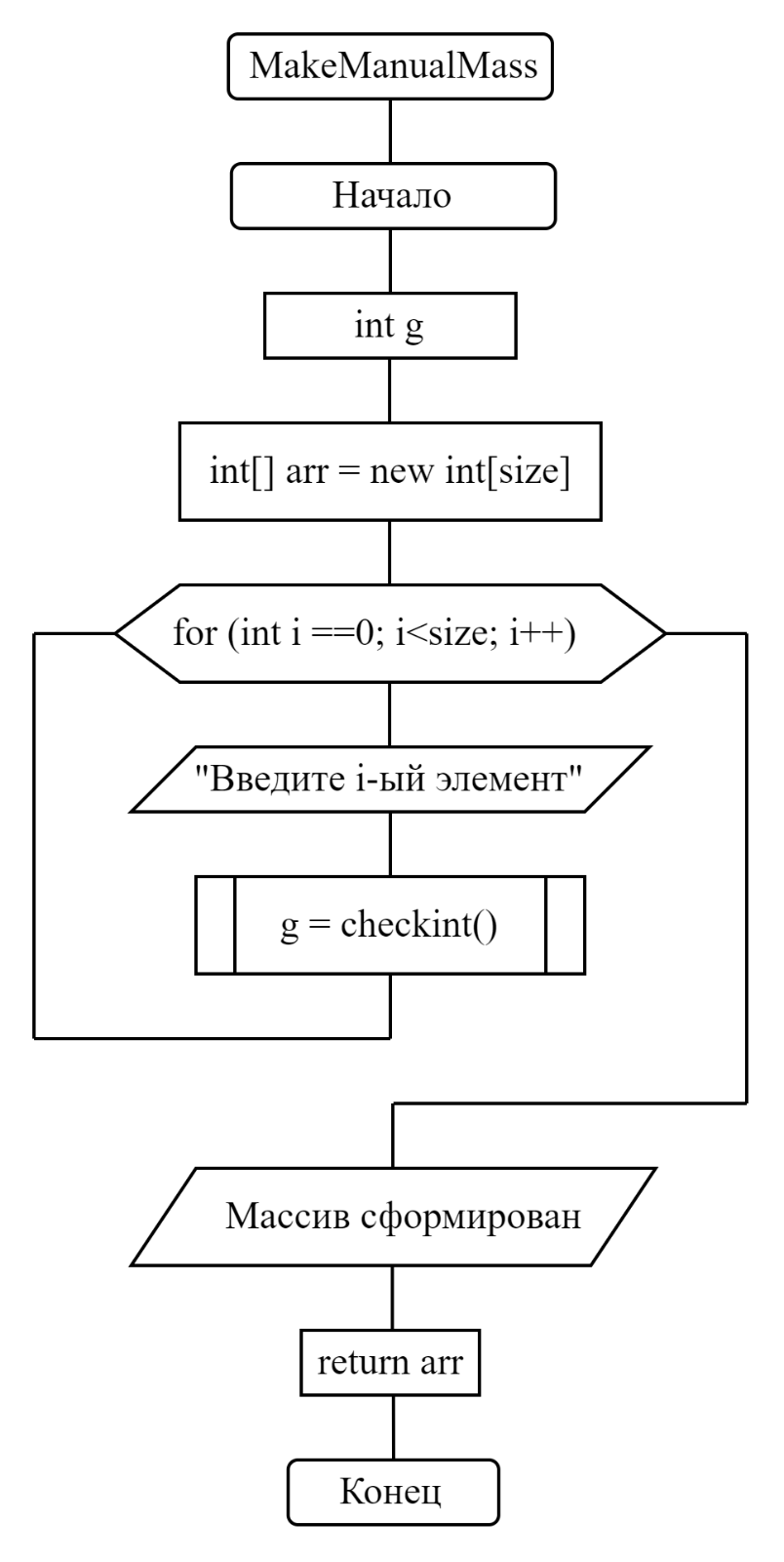
Таблица 8 – Анализ исходных данных для функции по поиску среднего арифметического и нахождения количества сравнений в отсортированном массиве

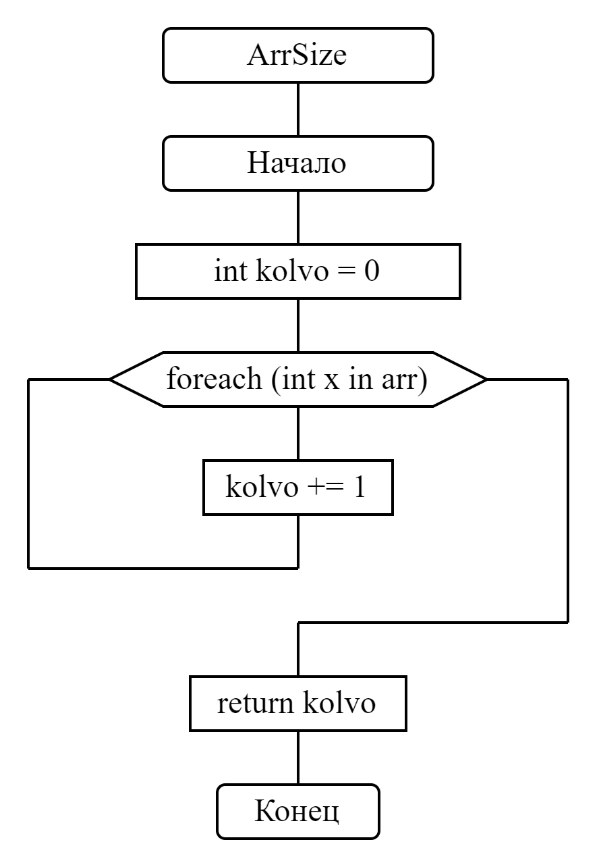
|  |  |  |
| --- | --- | --- |
| Исходные данные | Классы исходных данных | Ожидаемый результат |
| srednee | double | среднее арифметическое элементов в массиве. |
| sravn | int32 | Количество сравнений, необходимых для поиска нужного элемента |
| size | int32 | Длина массива |
| right | int32 | Правая граница массива |
| left | int32 | Левая граница массива |

Алгоритм

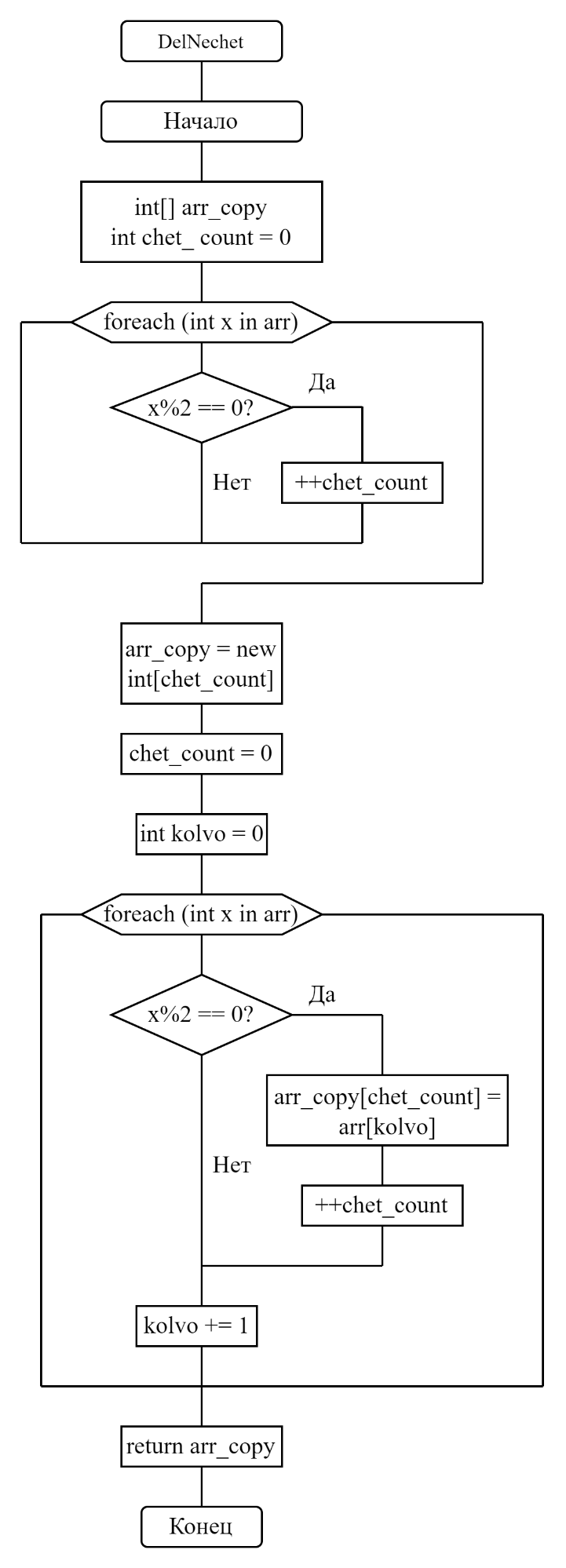
**Рисунок 1 – Алгоритм функции для формирования текстового меню**

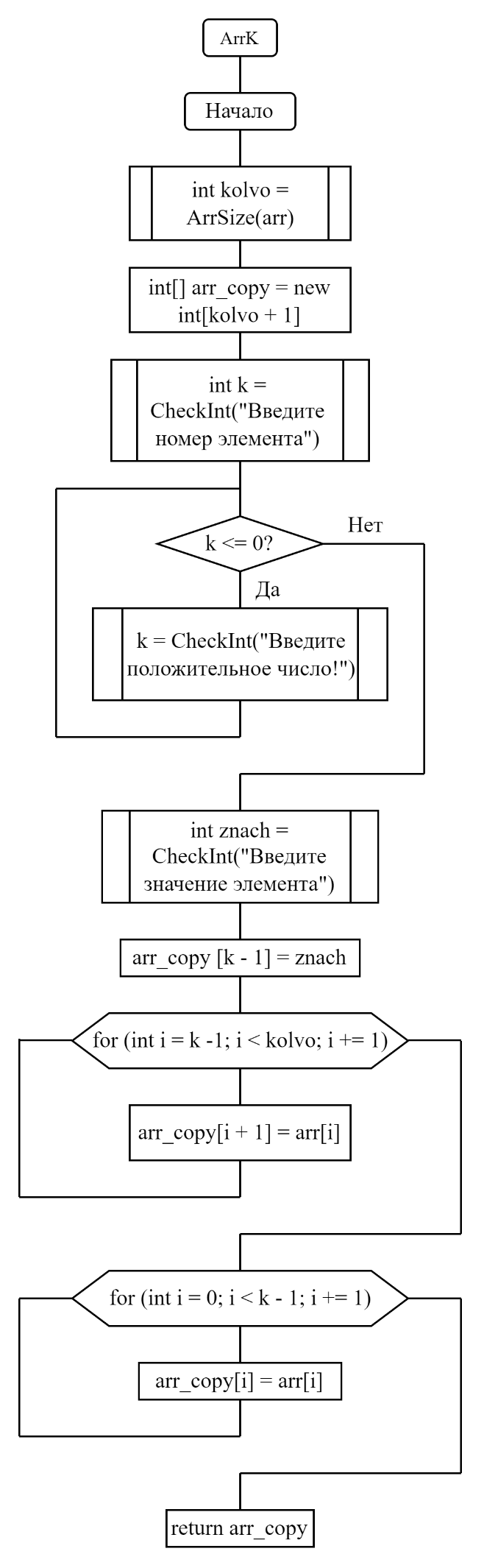
**Рисунок 2 – Алгоритм функции для создания массива длины n, заполненного случайными числами.**

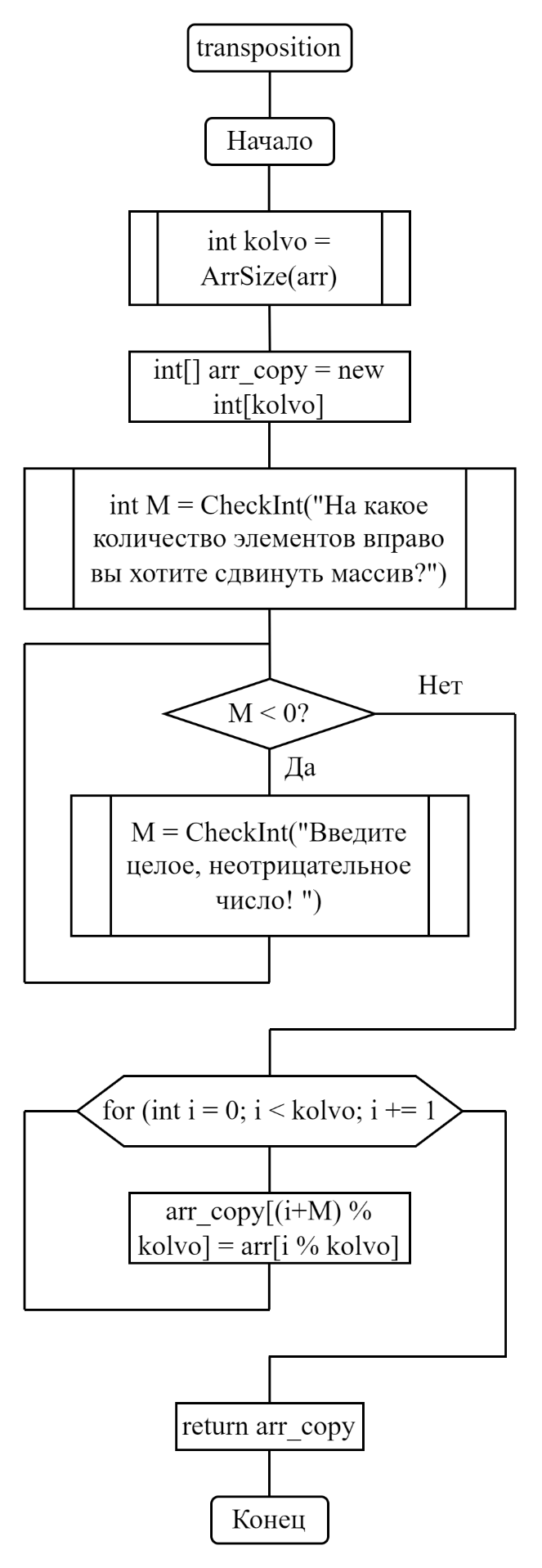
**Рисунок 3 – Алгоритм функции для создания массива длины n, заполненного числами, введёнными с клавиатуры**

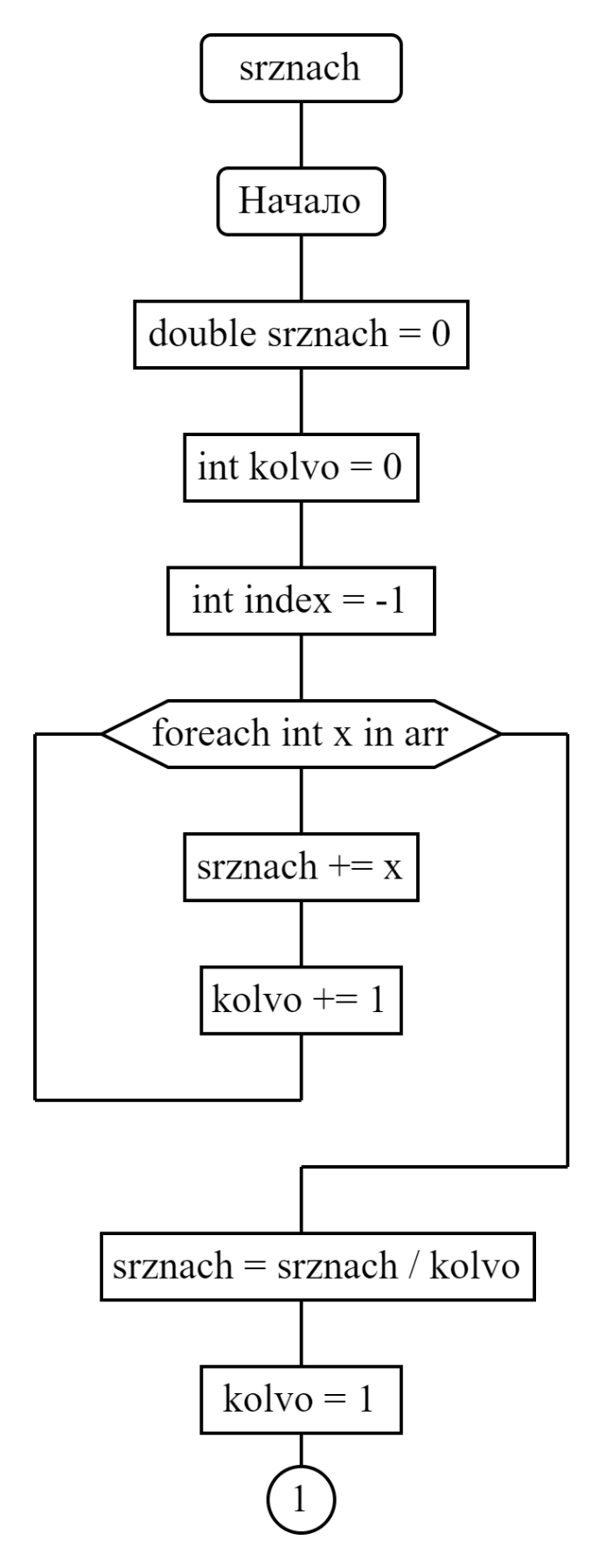
**Рисунок 4 – Алгоритм функции для печати массива**

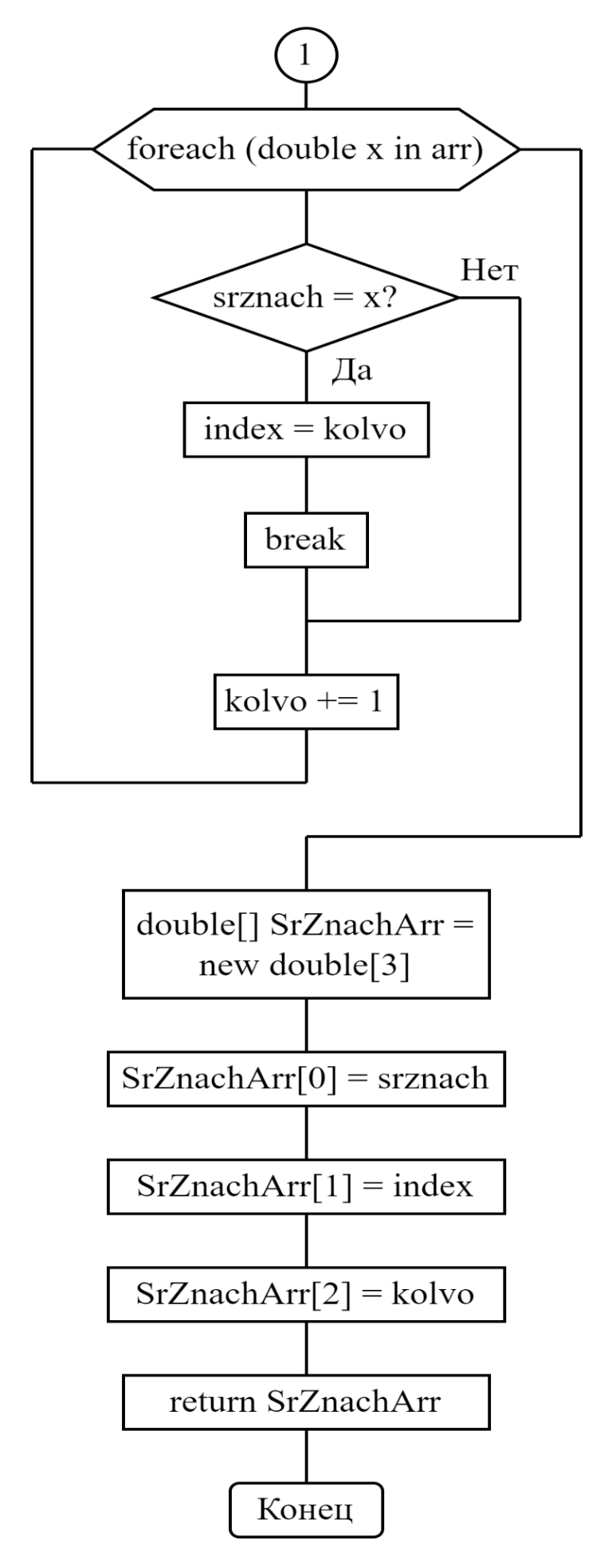
**Рисунок 5 – Алгоритм функция для нахождения длины массива**

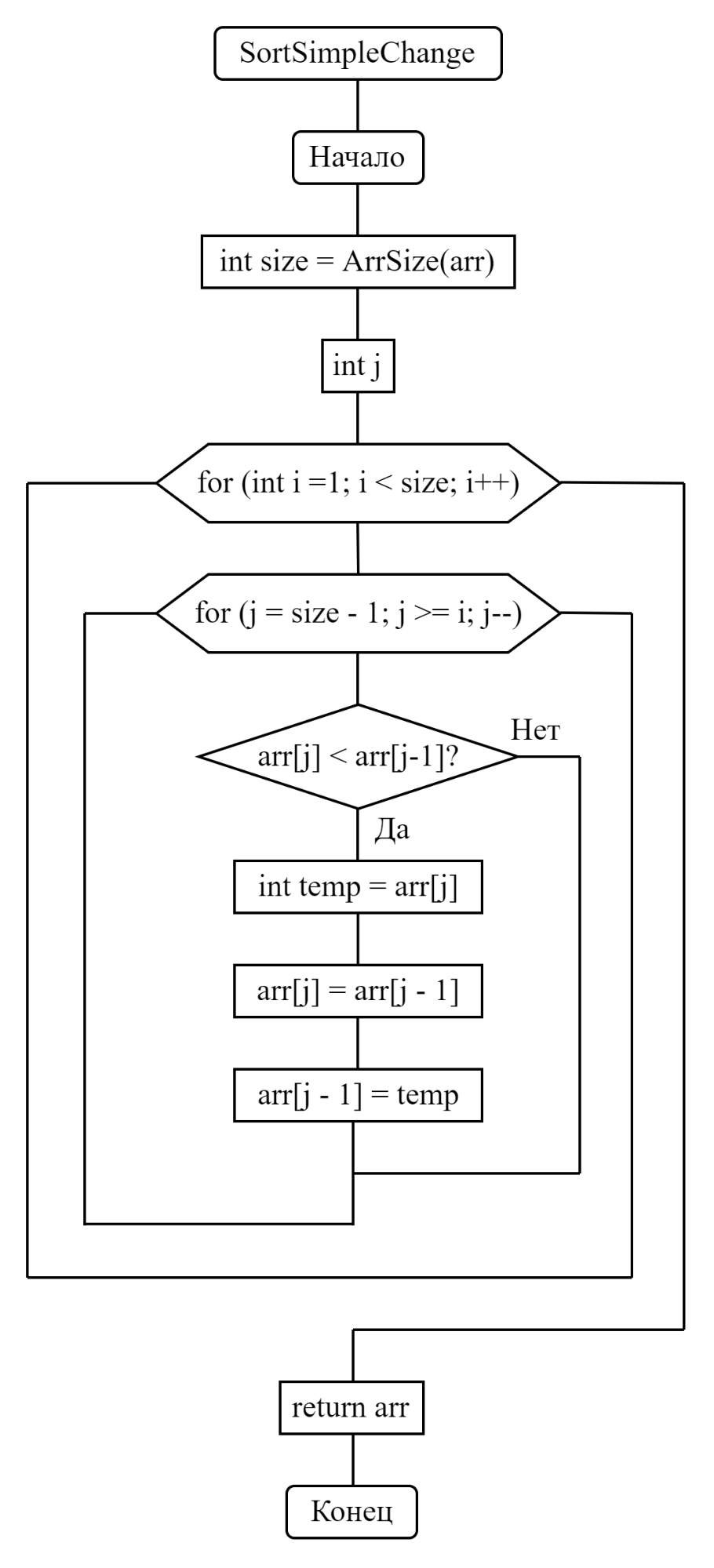
**Рисунок 6 – Алгоритм функции для удаления нечетных элементов из массива**

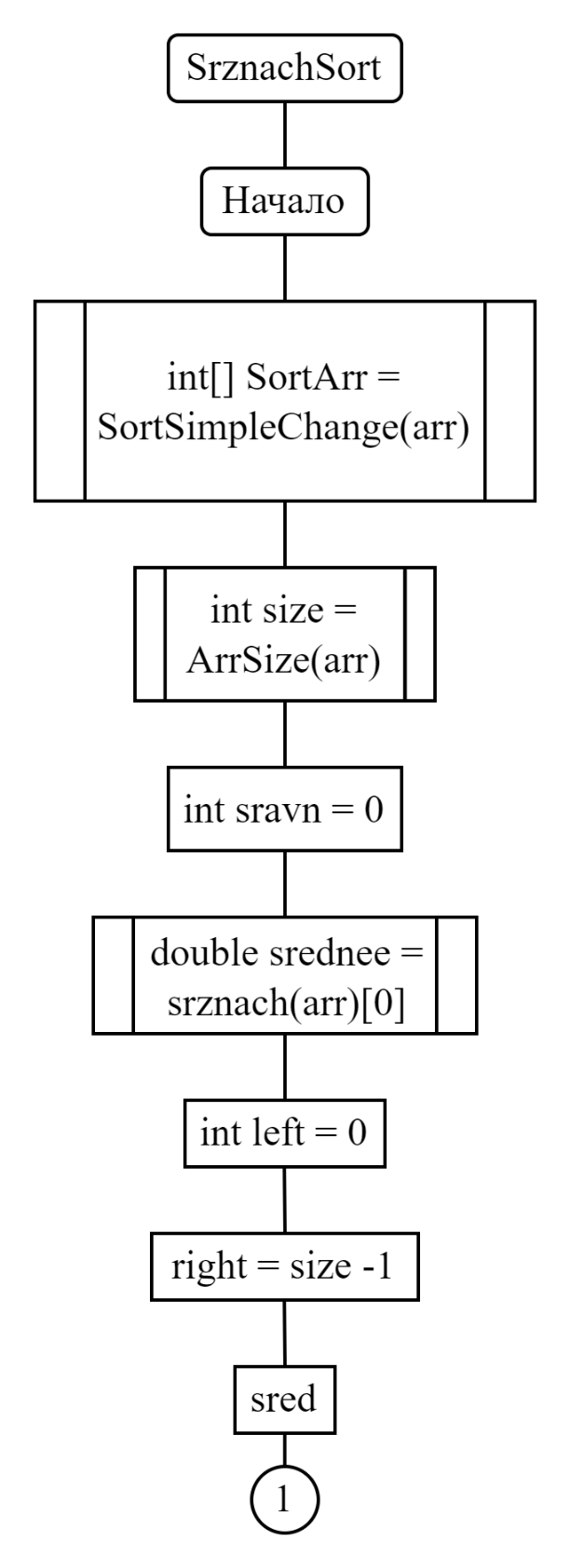
**Рисунок 7 – Алгоритм функции для добавления элемента с номером K в массив**

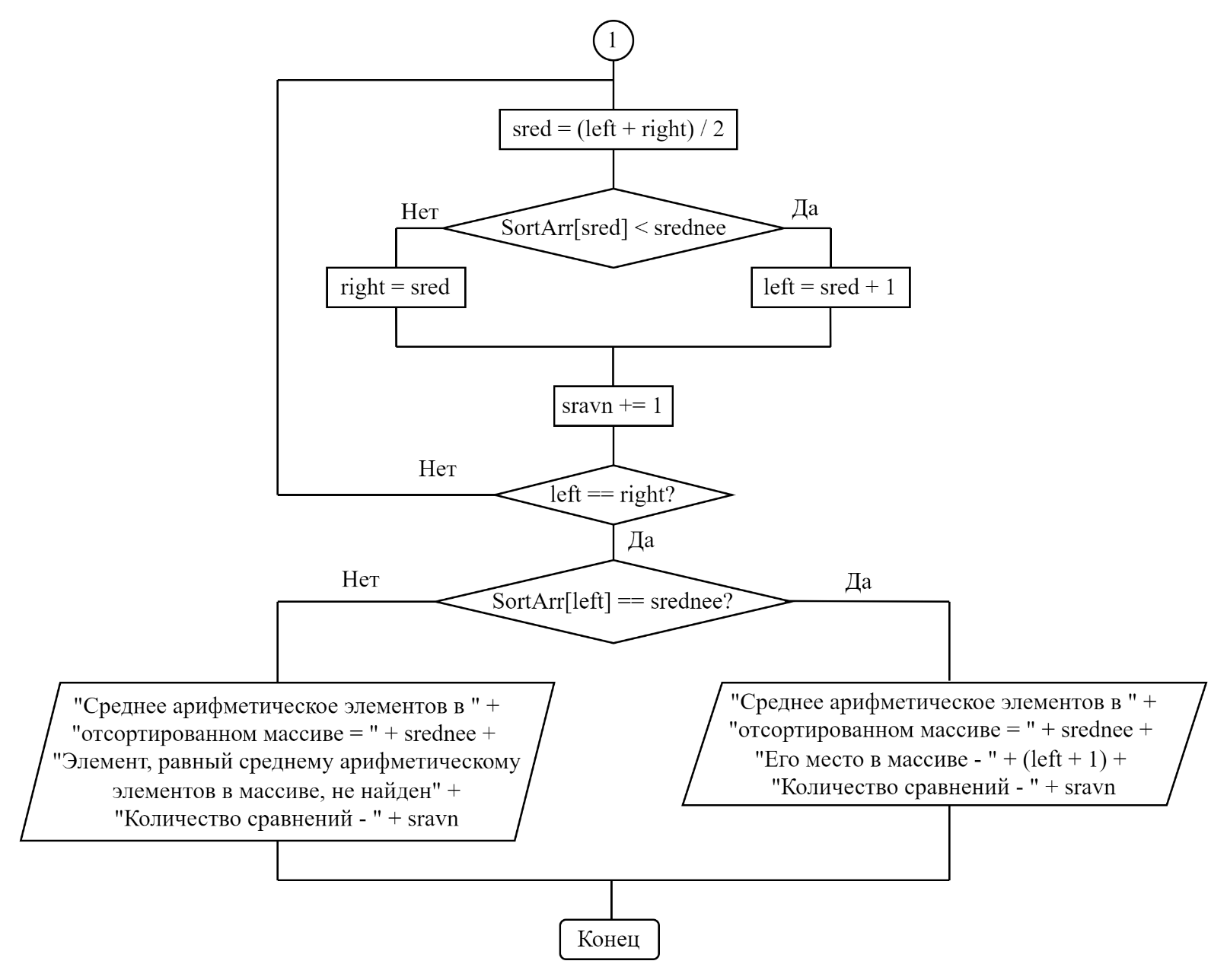
**Рисунок 8 – Алгоритм функции для циклической перестановки на М элементов вправо в массиве.**

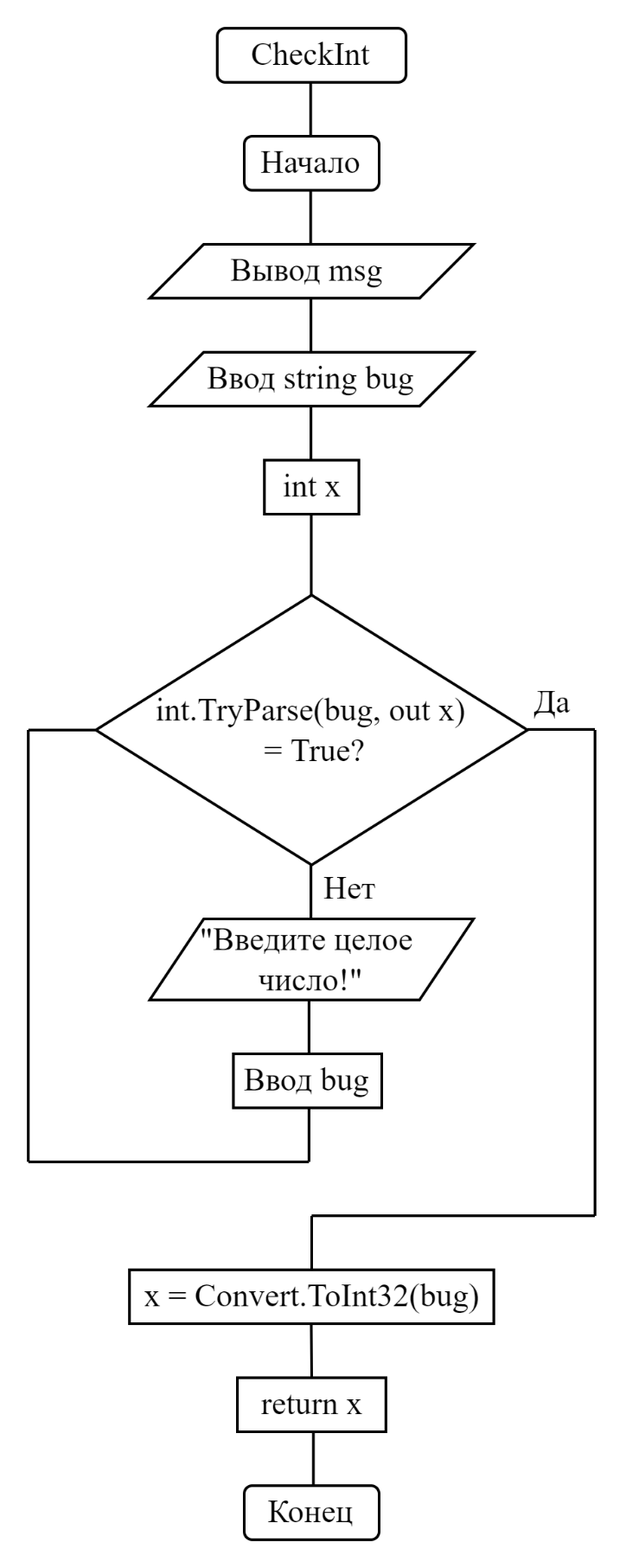
**Рисунок 9 – Алгоритм функции для выполнения поиска элемента, равного среднему арифметическому элементов в массиве, и подсчета количества сравнений, необходимых для поиска нужного элемента.**

 **Рисунок 10 – Продолжение алгоритма функции для выполнения поиска элемента, равного среднему арифметическому элементов в массиве, и подсчета количества сравнений, необходимых для поиска нужного элемента.**

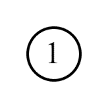
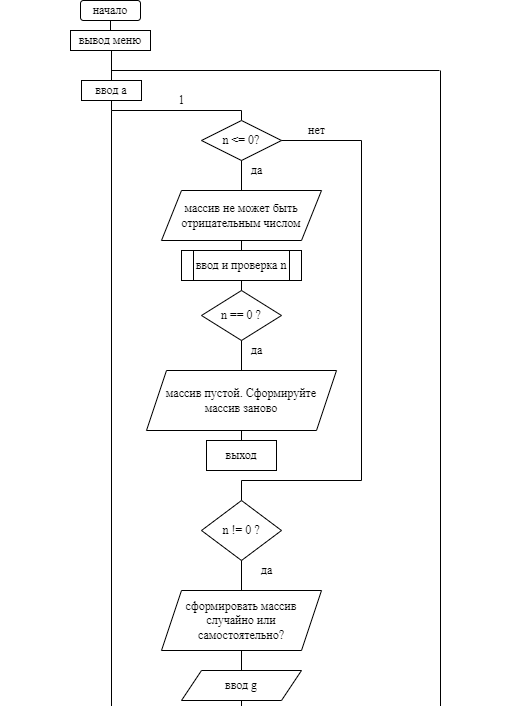
**Рисунок 11 – Алгоритм функции для сортировки простым обменом**

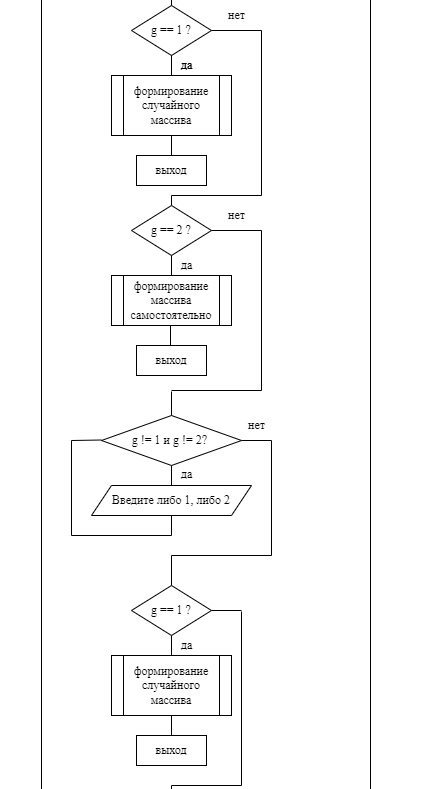
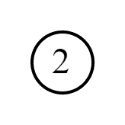
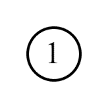
**Рисунок 11 – Алгоритм функции для выполнения поиска элемента, равного среднему арифметическому элементов в отсортированном массиве, и подсчета количество сравнений, необходимых для поиска нужного элемента **

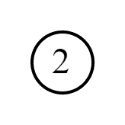
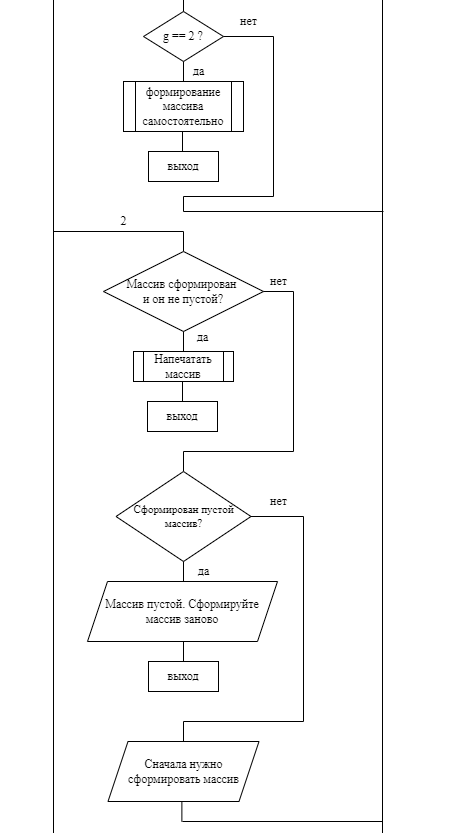
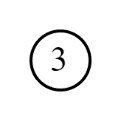
**Рисунок 12 – Продолжение алгоритма функции для выполнения поиска элемента, равного среднему арифметическому элементов в отсортированном массиве, и подсчета количество сравнений, необходимых для поиска нужного элемент**

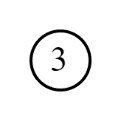
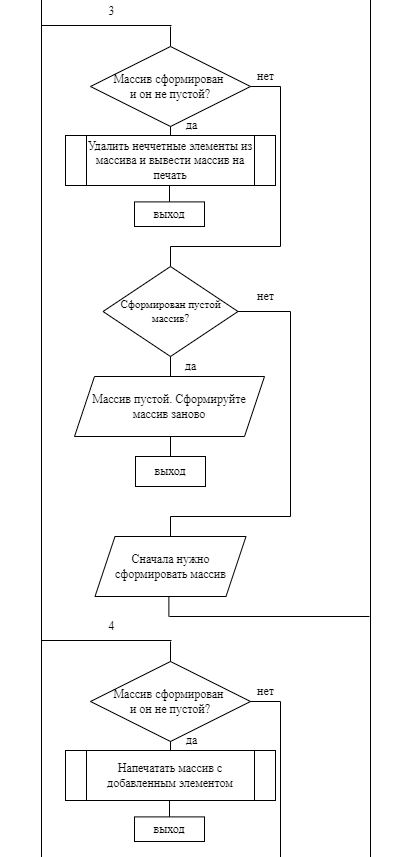
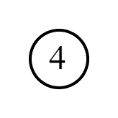


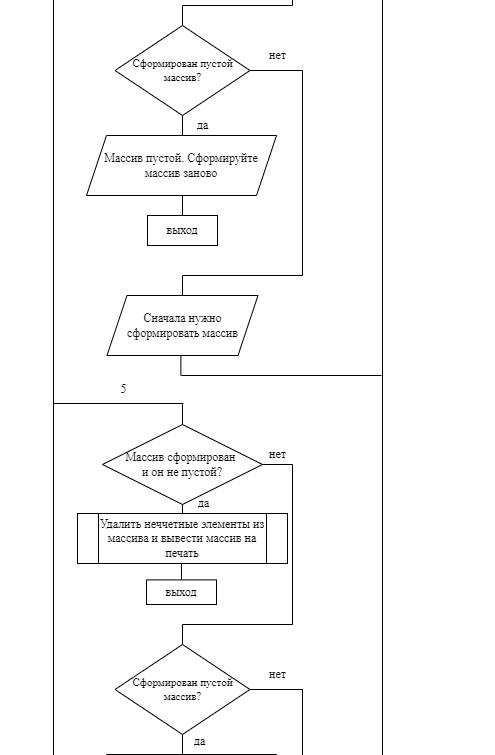
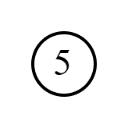
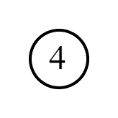
**Рисунок 13 – Функция для проверки вводимых чисел и преобразования их к типу int**

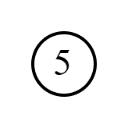
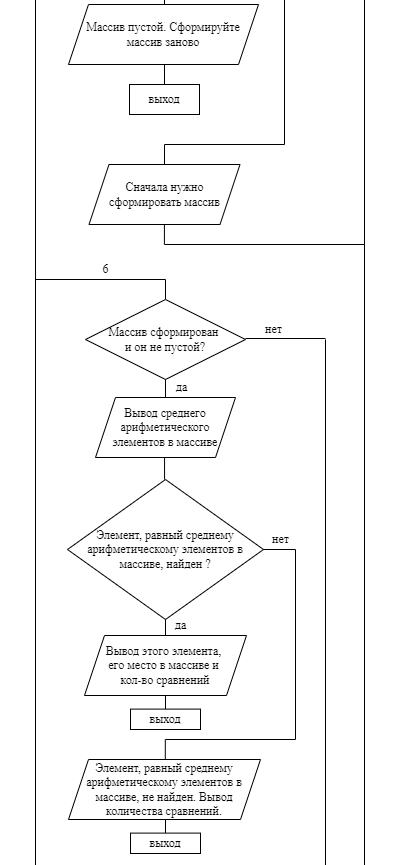
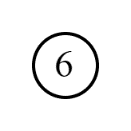
**Рисунок 14 – Алгоритм функции Mai****n**

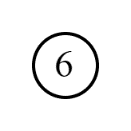
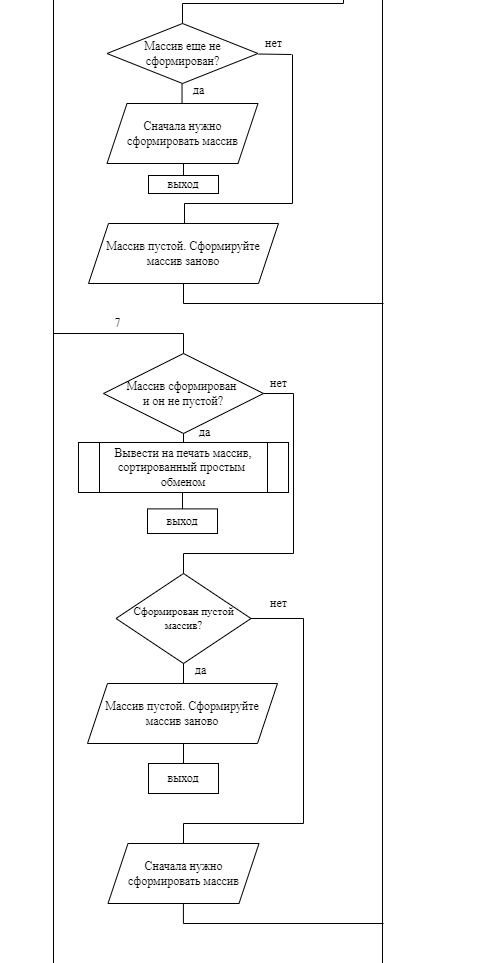
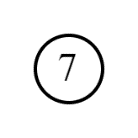
**Рисунок 15 – Продолжение алгоритма функции Main** 

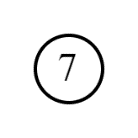
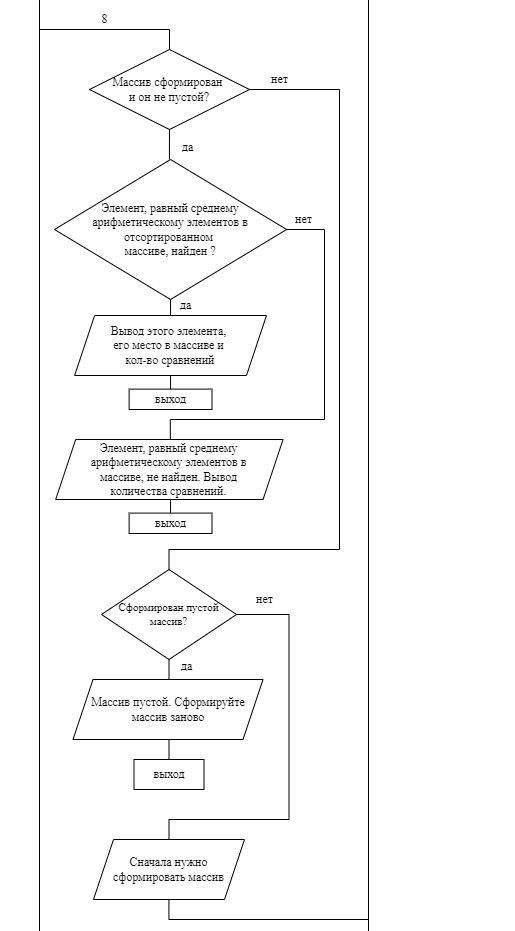
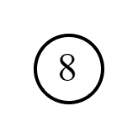
**Рисунок 16 – Продолжение алгоритма функции Main** 

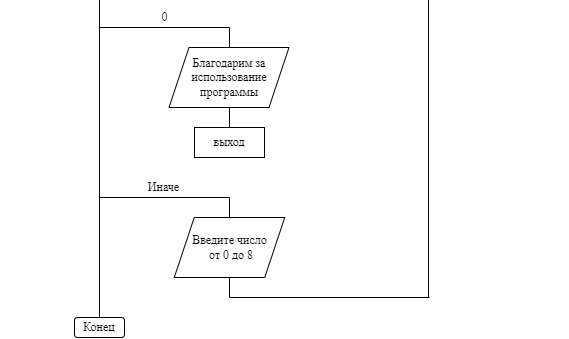
**Рисунок 17 – Продолжение алгоритма функции Main** 

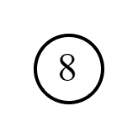
**Рисунок 18 – Продолжение алгоритма функции Main** 

**Рисунок 19 – Продолжение алгоритма функции Main** 

**Рисунок 20 – Продолжение алгоритма функции Main** 

**Рисунок 21 – Продолжение алгоритма функции Main** 



**Рисунок 18 – Продолжение алгоритма функции Main**

Листинг

Using System;

namespace Лаба\_4\_доп

{

class Program

{

static void PrintMenu()

{

Console.WriteLine("1. Формирование массива");

Console.WriteLine("2. Печать массива");

Console.WriteLine("3. Удаление нечетных элементов из массива");

Console.WriteLine("4. Добавление элемента с номером K в массив");

Console.WriteLine("5. Циклическая перестановка на M элементов вправо в массиве");

Console.WriteLine("6. Поиск элемента, равного среднему арифметическому элементов" +

" массива");

Console.WriteLine("7. Сортировка массива простым обменом");

Console.WriteLine("8. Поиск элемента, равного среднему арифметическому элементов " +

"в отсортированном массиве");

Console.WriteLine("0. Выход из программы");

}

static int[] MakeRandomMas(int size)

{

Random rnd = new Random();

int[] arr = new int[size];

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

{

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

}

Console.WriteLine("Массив сформирован");

Console.WriteLine();

return arr;

}

static int[] MakeManualMas(int size)

{

int[] arr = new int[size];

int g;

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

{

Console.WriteLine("Введите {0}-ый элемент", i + 1);

g = CheckInt();

arr[i] = g;

}

Console.WriteLine("Массив сформирован");

Console.WriteLine();

return arr;

}

static void PrintMas(int[] arr)

{

Console.WriteLine("Массив представлен ниже");

foreach (int x in arr)

{

Console.Write(x + " ");

}

Console.WriteLine("\n");

}

static int ArrSize(int[] arr)

{

int kolvo = 0;

foreach (int x in arr)

kolvo += 1;

return kolvo;

}

static int[] DelNechet(int[] arr)

{

int[] arr\_copy;

int chet\_count = 0;

foreach (int x in arr)

{

if (x % 2 == 0)

++chet\_count;

}

arr\_copy = new int[chet\_count];

chet\_count = 0;

int kolvo = 0;

foreach (int x in arr)

{

if (x % 2 == 0)

{

arr\_copy[chet\_count] = arr[kolvo];

++chet\_count;

}

kolvo += 1;

}

return arr\_copy;

}

static int[] ArrK(int[] arr)

{

int kolvo = ArrSize(arr);

int[] arr\_copy = new int[kolvo + 1];

int k = CheckInt("Введите номер элемента\n");

while (k <= 0 || k > kolvo+1)

k = CheckInt("Введите положительное число, не превышающее длину массива, увеличенную на 1!\n");

int znach = CheckInt("Введите значение элемента\n");

arr\_copy[k - 1] = znach;

for (int i = k - 1; i < kolvo; i += 1)

arr\_copy[i + 1] = arr[i];

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

arr\_copy[i] = arr[i];

return arr\_copy;

}

static int[] transposition(int[] arr)

{

int kolvo = ArrSize(arr);

int[] arr\_copy = new int[kolvo];

int M = CheckInt("На какое количество элементов вправо " +

"вы хотите сдвинуть массив?\n");

while (!(M >= 0))

M = CheckInt("Введите целое, неотрицательное число! ");

for (int i = 0; i < kolvo; i += 1)

arr\_copy[(i + M) % kolvo] = arr[i % kolvo];

return arr\_copy;

}

static double[] srznach(int[] arr)

{

double srznach = 0;

int kolvo = 0;

int index = -1;

foreach (int x in arr)

{

srznach += x;

kolvo += 1;

}

srznach = srznach / kolvo;

kolvo = 1;

foreach (double x in arr)

{

if (srznach == x)

{

index = kolvo;

break;

}

kolvo += 1;

}

double[] SrZnachArr = new double[3];

SrZnachArr[0] = srznach;

SrZnachArr[1] = index;

SrZnachArr[2] = kolvo;

return SrZnachArr;

}

static int[] SortSimpleChange(int[] arr)

{

int size = ArrSize(arr);

int j;

for (int i = 1; i < size; i++)

for (j = size - 1; j >= i; j--)

{

if (arr[j] < arr[j - 1])

{

int temp = arr[j];

arr[j] = arr[j - 1];

arr[j - 1] = temp;

}

}

return arr;

}

static void SrznachSort(int[] arr)

{

int[] SortArr = SortSimpleChange(arr);

int size = ArrSize(arr);

int sravn = 0;

double srednee = srznach(arr)[0];

int left = 0, right = size - 1, sred;

do

{

sred = (left + right) / 2;

if (SortArr[sred] < srednee)

left = sred + 1;

else

right = sred;

sravn += 1;

} while (left != right);

if (SortArr[left] == srednee)

Console.WriteLine("Среднее арифметическое элементов в " +

"отсортированном массиве = " + srednee + "\nЕго место в массиве - " + (left + 1) + "\nКоличество " +

"сравнений - " + sravn + "\n");

else

Console.WriteLine("Среднее арифметическое элементов " +

"в отсортированном массиве = " + srednee + "\nЭлемент, равный среднему арифметическому элементов в массиве, не найден\n" +

"Количество сравнений - " + sravn + "\n");

}

static int CheckInt(string msg = "")

{

Console.Write(msg);

string bug = Console.ReadLine();

int x;

while (!int.TryParse(bug, out x))

{

Console.WriteLine("Введите целое число!");

bug = Console.ReadLine();

}

x = Convert.ToInt32(bug);

return x;

}

static void Main(string[] args)

{

string a; int n; int[] arr = null; int ll = 0; bool isfirst = false; bool massive = true;

do

{

PrintMenu();

Console.WriteLine();

a = Console.ReadLine();

switch (a)

{

case "1":

massive = true;

isfirst = true;

n = CheckInt("Размер массива ");

if (n == 0)

{

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" Сформируйте массив заново\n");

massive = false;

}

else if (n > 0)

{

Console.WriteLine("1. Сформировать случайный массив длины " + n);

Console.WriteLine("2. Сформировать массив длины " + n + " Самостоятельно\n");

string g = Console.ReadLine();

while (g != "1" & g != "2")

{

Console.WriteLine("Введите либо 1, либо 2");

g = Console.ReadLine();

}

if (g == "1")

{

arr = MakeRandomMas(n);

}

else

{

arr = MakeManualMas(n);

}

}

else

{

while (!(n > 0))

{

Console.WriteLine("Размер массива не может быть отрицательным числом! Введите целое, неотрицательное число\n");

n = CheckInt("Размер массива ");

if (n == 0)

{

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

massive = false;

break;

}

}

if (n != 0)

{

Console.WriteLine("1. Сформировать случайный массив длины " + n);

Console.WriteLine("2. Сформировать массив длины " + n + " Самостоятельно\n");

string g = Console.ReadLine();

if (g == "1")

{

arr = MakeRandomMas(n);

}

else if (g == "2")

{

arr = MakeManualMas(n);

}

else

{

while (g != "1" & g != "2")

{

Console.WriteLine("Введите либо 1, либо 2");

g = Console.ReadLine();

}

if (g == "1")

{

arr = MakeRandomMas(n);

}

else

{

arr = MakeManualMas(n);

}

}

}

}

break;

case "2":

if (isfirst && massive)

{

PrintMas(arr);

}

else if (!massive)

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла" +

" сформируйте массив заново\n");

else

Console.WriteLine("Сначала нужно сформировать массив!\n");

break;

case "3":

if (isfirst && massive)

if (ArrSize(DelNechet(arr)) != 0)

PrintMas(DelNechet(arr));

else

Console.WriteLine("В результате удаления нечетных элементов получился пустой массив\n");

else if (!massive)

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

else

Console.WriteLine("Сначало нужно сформировать массив!\n");

break;

case "4":

if (isfirst && massive)

PrintMas(ArrK(arr));

else if (!massive)

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

else

Console.WriteLine("Сначало нужно сформировать массив!\n");

break;

case "5":

if (isfirst && massive)

PrintMas(transposition(arr));

else if (!massive)

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

else

Console.WriteLine("Сначало нужно сформировать массив!\n");

break;

case "6":

if (isfirst && massive)

{

double[] SrZnachArr = srznach(arr);

Console.WriteLine("Среднее арифметическое элементов" +

" в массиве = " + Math.Round(SrZnachArr[0], 2));

if (SrZnachArr[1] != -1)

{

Console.WriteLine("Элемент, равный среднему " +

"арифметическому элементов в массиве," +

" найден\nМесто элемента в массиве - " + SrZnachArr[1]);

Console.WriteLine($"Понадобилось сравнений – {SrZnachArr[1]}\n");

}

else

Console.WriteLine("Элемент, равный среднему арифметическому" +

" элементов в массиве, не найден\nКоличество сравнений - " + SrZnachArr[2] + "\n");

}

else if (!isfirst)

Console.WriteLine("Сначала нужно сформировать массив!\n");

else

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

break;

case "7":

if (isfirst && massive)

PrintMas(SortSimpleChange(arr));

else if (!massive)

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

else

Console.WriteLine("Сначало нужно сформировать массив!\n");

break;

case "8":

if (isfirst && massive)

SrznachSort(arr);

else if (!massive)

Console.WriteLine("Массив пустой. Дальнейшие действия не будут иметь смысла." +

" сформируйте массив заново\n");

else

Console.WriteLine("Сначало нужно сформировать массив!\n");

break;

case "0":

Console.WriteLine("Благодарим за использование программы!\n");

ll = -4;

break;

default:

Console.WriteLine("Введите число от 0 до 8!\n");

break;

}

} while (ll != -4);

}

}

}