

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ

*Институт Принтмедиа и информационных технологий
Кафедра Информатики и информационных технологий*

направление подготовки

09.03.02 «Информационные системы и технологии»

ЛАБОРАТОРНАЯ РАБОТА № 4-5

Дисциплина: Основы алгоритмизации и программирования.

Тема: Алгоритм сортировки «пузырёк»

Выполнила:

студентка группы 201-723

Круглова А.М.

19.10.20

(Дата)

(Подпись)

Проверил: преп. Хуснулина Д.Р.

(Дата)

(Подпись)

Замечания: _____

Москва

2020

Оглавление

Цель:	3
Постановка задачи:	3
Идея алгоритма:.....	3
Словесное представление алгоритма:	4
Блок-схема с использованием элемента модификации:	5
Листинг программы с использованием параметрического цикла:	6
Результат работы программы:.....	7
Блок-схема без использования элемента модификации:	8
Листинг программы с использованием цикла с предусловием:	9
Результат работы программы:.....	10

Цель:

Получить практические навыки разработке алгоритмов и их программной реализации.

Постановка задачи:

Необходимо выполнить и оформить описание следующих пунктов:

1. Сформулировать идею алгоритма
2. Выполнить словесное представление алгоритма
3. Выполнить полнить представление алгоритма с помощью блок схем с использованием элемента модификации и без него.
4. Выполнить программную реализацию алгоритмов на языке С с использованием параметрического цикла и цикла с предусловием.

Идея алгоритма:

Алгоритм основан на повторяющихся проходах по сортируемому массиву. За каждый проход последовательно сравниваются соседние элементы. Если порядок в паре неверный, то происходит обмен значений элементов. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырьёк в воде — отсюда и название алгоритма).

Проходы по массиву повторяются $n-1$ (где n – размер массива) раз или до тех пор, пока не будет перестановок, т.е. когда массив окажется отсортированным.

Словесное представление алгоритма:

arr – массив, n – длина массива

1. Начало алгоритма
2. Параметр $i=0$
3. Если $i < 10$, то п.4, иначе п.6
4. Ввод с клавиатуры arr[i]
5. $i++$, п.3
6. Параметр внешнего цикла $i = 0$
7. Если $i < n - 1$, то п.8, иначе п.14
8. Параметр внутреннего цикла $j = n-1$
9. Если $j > i$, то п.10, иначе п.13
10. Если $arr[j-1] > arr[j]$, то п.11, иначе п.12
11. Обмен значениями arr[j-1] и arr[j]
12. $j--$, п.4
13. $i++$, п.2
14. Параметр $i=0$
15. Если $i < 10$, то п.16, иначе п.18
16. Вывод на экран arr[i]
17. $i++$, п.15
18. Конец алгоритма

Блок-схема с использованием элемента модификации:

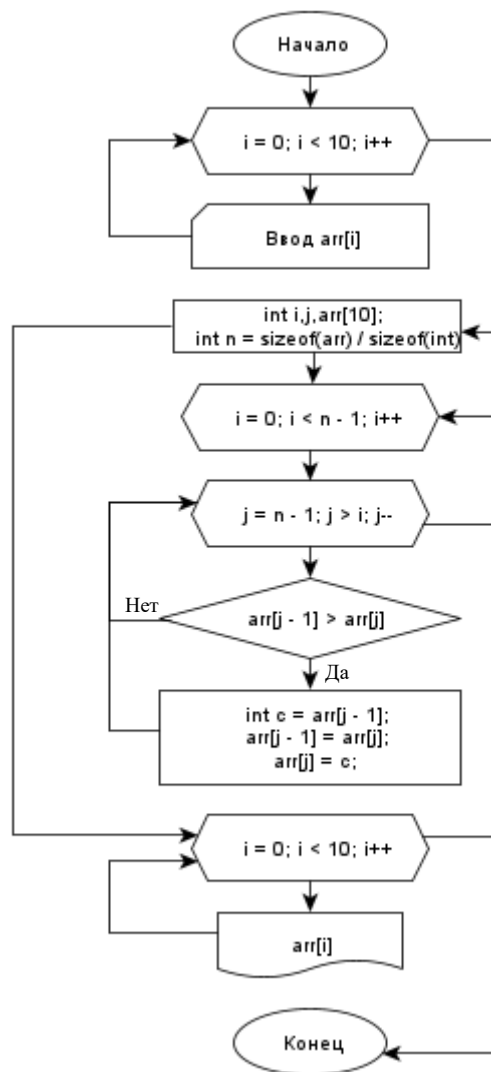


Рисунок 1 — Блок-схема 1 программы

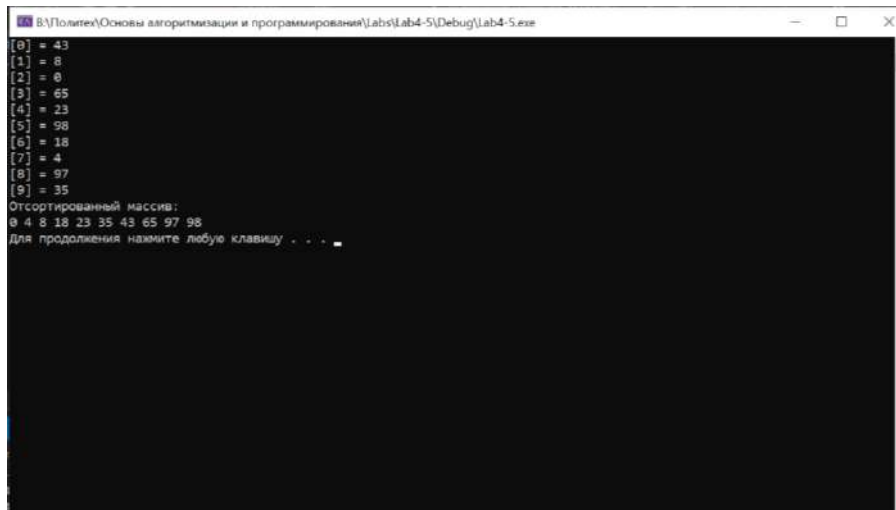
Листинг программы с использованием параметрического цикла:

Листинг 1 — Исходный код 1 программы

```
#include <iostream>
#include <stdlib.h>

int main()
{
    setlocale(LC_ALL, "Russian"); //установка русского языка
    int i, j;
    int arr[10]; // Объявляем массив из 10 элементов
    for (i = 0; i < 10; i++) // Вводим значения элементов массива через
клавиатуру
    {
        printf("[%d] = ", i);
        scanf_s("%d", &arr[i]);
    }
    int n = sizeof(arr) / sizeof(int); //размер массива
    for (i = 0; i < n - 1; i++) //начинаем внешний цикл
    {
        for (j = n - 1; j > i; j--) //внутренний цикл
        {
            if (arr[j - 1] > arr[j]) // условие, если текущий элемент меньше
предыдущего
            {
                int c = arr[j - 1]; // меняем их местами
                arr[j - 1] = arr[j];
                arr[j] = c;
            }
        }
    }
    // Выводим отсортированные элементы массива
    printf("Отсортированный массив:\n");
    for (i = 0; i < 10; i++)
        printf("%d ", arr[i]);
    printf("\n");
    system("pause");
    return 0;
}
```

Результат работы программы:



```
В:\Политех\Основы алгоритмизации и программирования\labs\lab4-5\Debug\lab4-5.exe
[0] = 43
[1] = 8
[2] = 0
[3] = 65
[4] = 23
[5] = 98
[6] = 18
[7] = 4
[8] = 97
[9] = 35
Отсортированный массив:
0 4 8 18 23 35 43 65 97 98
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2 — Результат работы 1 программы

Блок-схема без использования элемента модификации:

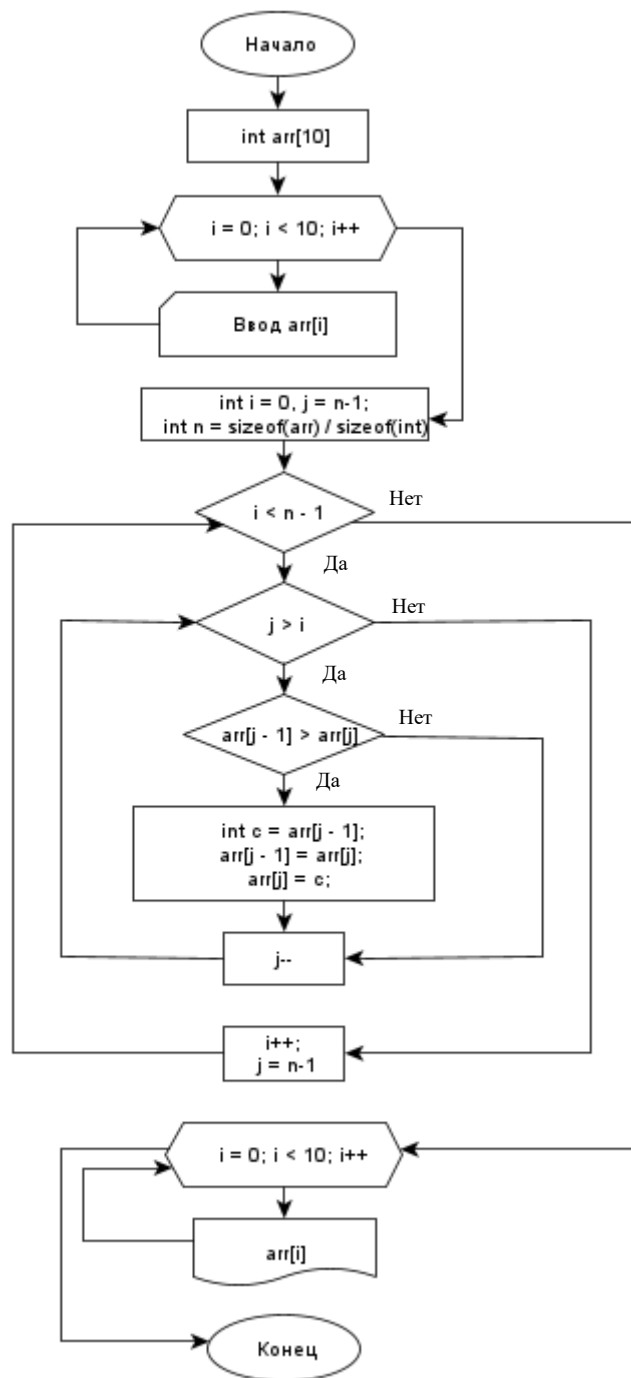


Рисунок 3 — Блок-схема 2 программы

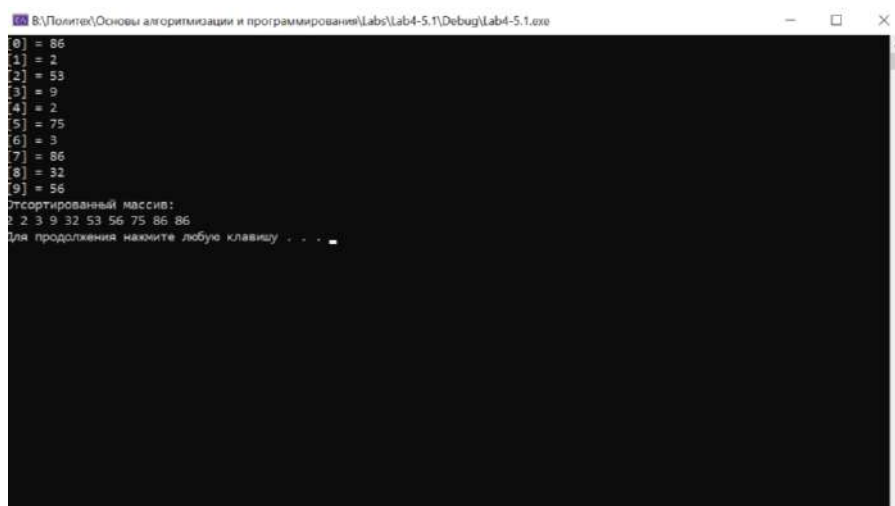
Листинг программы с использованием цикла с предусловием:

Листинг 2 — Исходный код 2 программы

```
#include <iostream>
#include <stdlib.h>

int main()
{
    setlocale(LC_ALL, "Russian"); //установка русского языка
    int arr[10]; // Объявляем массив из 10 элементов
    int n = sizeof(arr) / sizeof(int); //размер массива
    for (int i = 0; i < 10; i++) // Вводим значения элементов массива через
клавиатуру
    {
        printf("[%d] = ", i);
        scanf_s("%d", &arr[i]);
    }
    int i = 0, j = n-1; //начальное значение
    while (i < n - 1) //внешний цикл
    {
        while (j > i) //внутренний цикл
        {
            if (arr[j - 1] > arr[j]) // условие, если текущий элемент меньше
предыдущего
            {
                int c = arr[j - 1]; // меняем их местами
                arr[j - 1] = arr[j];
                arr[j] = c;
            }
            j--;
        }
        i++;
        j = n-1;
    }
    // Выводим отсортированные элементы массива
    printf("Отсортированный массив:\n");
    for (i = 0; i < 10; i++)
        printf("%d ", arr[i]);
    printf("\n");
    system("pause");
    return 0;
}
```

Результат работы программы:



```
8:\Политех\Основы алгоритмизации и программирования\Lab5\Lab4-5.1\Debug\Lab4-5.1.exe
[0] = 86
[1] = 1
[2] = 53
[3] = 9
[4] = 2
[5] = 75
[6] = 3
[7] = 86
[8] = 32
[9] = 56
Отсортированный массив:
1 2 3 9 32 53 56 75 86 86
для продолжения нажмите любую клавишу . . .
```

Рисунок 4 — Результат работы 2 программы