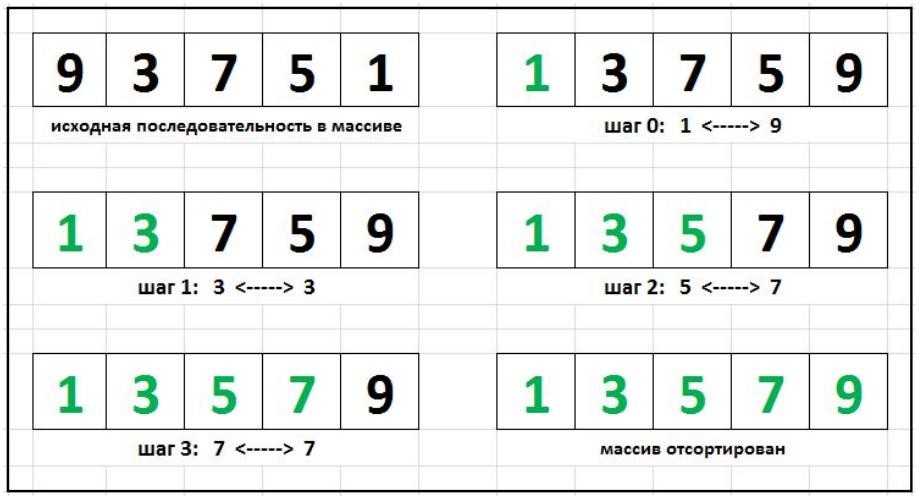
# Отчет №8. Алгоритмы сортировки

## Сортировка выбором

Смысл **Сортировки выбором (Selection Sort)** состоит в поиске минимального значения элемента в массиве, и перемещения этого значения в начало массива. Нужно сразу оговориться, что в данном случае можно назвать “началом” массива (куда перемещается найденное минимальное значение).

“Начало” в алгоритме **Сортировка выбором** с каждым шагом цикла смещается в сторону хвоста массива. Поэтому на первой итерации

цикла, найденное минимальное значение меняется местами со значением в нулевой ячейке массива. На второй итерации “начало” уже будет указывать на следующую (первую) ячейку и так далее.



По факту получается простой обмен местами значений ячеек [массива](https://purecodecpp.com/archives/1). При таком обмене значениями не нужен сдвиг (перезапись) всех элементов массива, чтобы записать минимальное значение в соответствующую ячейку.

То есть алгоритм **Сортировка выбором** не требует использования дополнительной памяти. Перезапись значений происходит сразу после нахождения минимального значения в массиве.

## Код:

1. #include <iostream>
2. using namespace std; 3.

4. int main()

**5. {**

6. const int N = 10;

7. int a[N] = { 1, 25, 6, 32, 43, 5, 96, 23, 4, 55 };

8.

1. int min = 0; // для записи минимального значения
2. **int buf = 0; // для обмена значениями**

12. */\*\*\*\*\*\*\*\*\*\*\* Начало сортировки \*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

13. for (int i = 0; i < N; i++)

14. {

1. **min = i; // запомним номер текущей ячейки, как ячейки с минимальным значением**
2. // в цикле найдем реальный номер ячейки с минимальным значением
3. for (int j = i + 1; j < N; j++)
4. min = ( a[j] < a[min] ) ? j : min;
5. // cделаем перестановку этого элемента, поменяв его местами с текущим
6. **if (i != min)**

21. {

1. buf = a[i];
2. a[i] = a[min];
3. a[min] = buf;

**25. }**

26. }

27. */\*\*\*\*\*\*\*\*\*\*\* Конец сортировки \*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

28.

29. for (int i = 0; i < N; i++) //Вывод отсортированного массива

**30. cout << a[i] << '\t';**

31. cout << endl;

32. }

# Пузырьковая сортировка

Допустим у нас есть неотсортированный [массив](https://purecodecpp.com/archives/1) чисел из 5-ти элементов и нам предстоит разместить значения по возрастанию. Для наглядности, на рисунке изобразим этот массив вертикально “Исходный массив”.



Алгоритм сортировки “пузырьком” состоит в повторении проходов по массиву с помощью [вложенных циклов](https://purecodecpp.com/archives/770). При каждом проходе по массиву

сравниваются между собой пары “соседних” элементов. Если числа какой-то из сравниваемых пар расположены в неправильном порядке – происходит обмен (перезапись) значений ячеек массива.

Образно говоря в нашем примере 2 “легче” чем 3 – поэтому обмен есть, далее 2 “легче” чем 7 – снова обмен и т.д. При сравнении нулевого и

первого элемента на нашем рисунке обмена нет, так как 2 “тяжелее” чем

1. Таким образом более “легкое” значение, как пузырек в воде,

поднимается вверх до нужной позиции. Вот почему у этого алгоритма такое название.

## Код:

* 1. #include <iostream>
  2. using namespace std; 3.

1. // прототип функции, которая выполнит сортировку "пузырьком"
2. **void bubbleSort(int arrForSort[], const int SIZE);**

6.

7. void main() 8. {

1. setlocale(LC\_ALL, "rus");
2. **const int SIZE = 5;**
3. int arr[SIZE]; 12.
4. cout << "Исходный массив:**\n**";
5. for (int i = 0; i < SIZE; i++)

**15. {**

16. arr[i] = SIZE - i; // заполняем значениями по убыванию

17. cout << arr[i] << "**\n \n**";

18. }

19. cout << "**\n\n**";

**20.**

21. bubbleSort(arr, SIZE); // передаем в функцию для сортировки

22.

1. cout << "Массив после сортировки:**\n**";
2. for (int i = 0; i < SIZE; i++)

**25. {**

26. cout << arr[i] << "**\n \n**";

27. }

28. cout << "**\n\n**";

29. }

**30.**

31. void bubbleSort(int arrForSort[], const int SIZE)

32. {

33. int buff = 0; // для временного хранения значения, при перезаписи

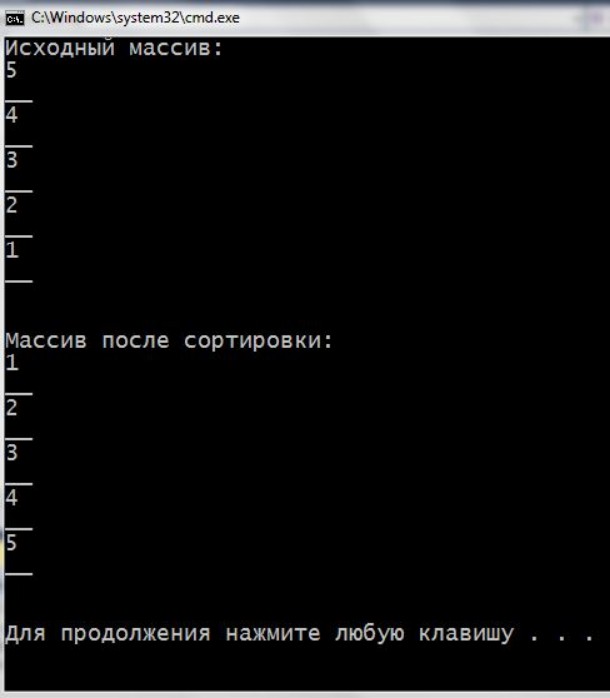
34.

**35. for (int i = 0; i < SIZE - 1; i++) //**

36. {

1. // вложенный цикл проходит от четвертого элемента
2. // до первого, находит с помощью if самое "легкое" значение,

|  |  |  |
| --- | --- | --- |
| 39. | // сравнивая соседние пары элементов, |  |
| **40.** | **// и поднимает его в нулевую ячейку массива** |
| 41. | for (int j = SIZE - 1; j > i; j--) |
| 42. | { |
| 43. | if (arrForSort[j] < arrForSort[j - 1]) |
| 44. | { |
| **45.** | **buff = arrForSort[j - 1];** |
| 46. | arrForSort[j - 1] = arrForSort[j]; |
| 47. | arrForSort[j] = buff; |
| 48. | } |
| 49. | } |
| **50.** | **// далее значение i увеличивается на 1** |
| 51. | // и внутренний цикл будет перебирать элементы |
| 52. | // от четвертого до второго. Таким образом поднимет | самое |
| 53. | // "легкое" значение из оставшихся в первую ячейку | и т.д. |
| 54. | } |  |
| **55.** | **}** |  |



Источники:

- <https://purecodecpp.com/algoritmy-v-c>

//также на сайте есть информация про другие виды сортировки (и поиска!!), в том числе про разновидности пузырьковой сортировки

//открывается только с VPN