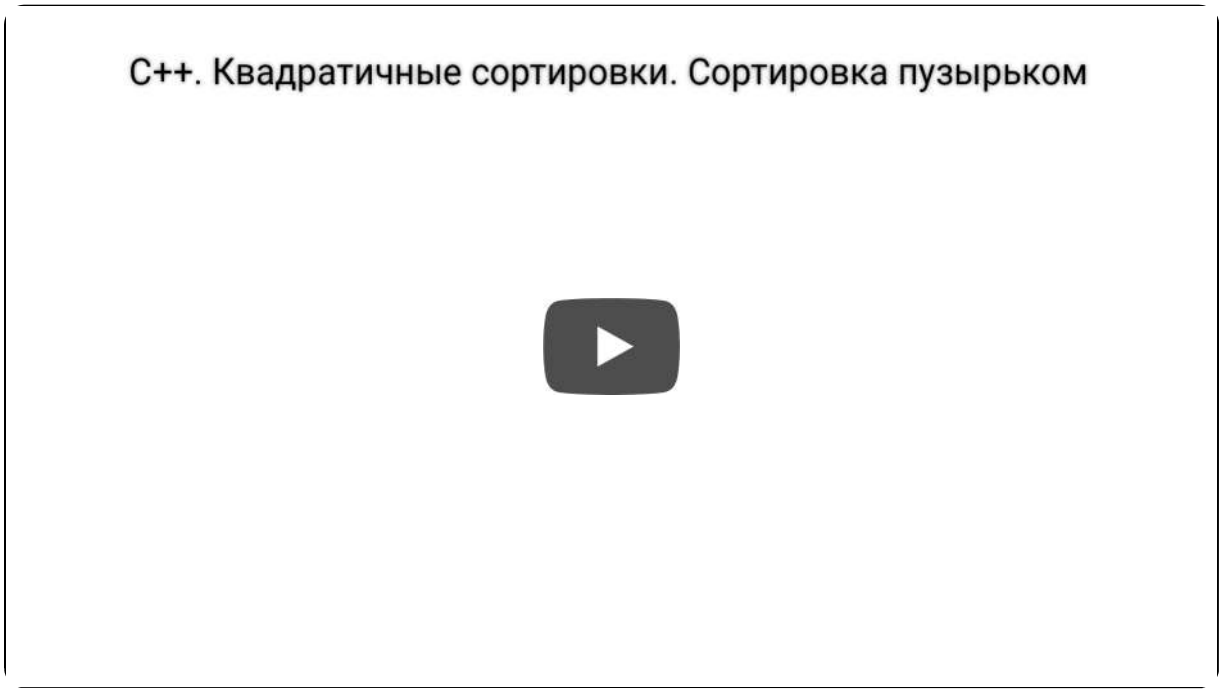
Сортировка пузырьком



Обозначим элементы исходного списка $a_0, a_1 \ldots a_{n-1}$.

Рассмотрим алгоритм **«пузырьковой» сортировки**. Суть его заключается в следующем: будем просматривать слева направо все пары соседних элементов: a_0 и a_1, a_1 и a_2, ..., a_{n-2} и a_{n-1}. Если $a_i > a_{i+1}$, то элементы меняем местами. В результате такого просмотра массива максимальный элемент окажется на крайнем справа (своём) месте. Об остальных элементах ничего определённого сказать нельзя. Будем просматривать массив снова, исключив из рассмотрения правый элемент. На своём месте теперь окажется уже второй по величине элемент. И так далее до тех пор, пока список не станет отсортирован.

Рассмотрим на примере списка:

$$[5, 2, 9, 3, 4]$$

Проиллюстрируем работу алгоритма таблицей:

Номер итерации	Рассматриваемые элементы	Элементы списка
-	-	[5, 2, 9, 3, 4]
1	a_0 и a_1	[2, 5, 9, 3, 4]

2	a_1 и a_2	[2, 5, 9, 3, 4]
3	a_2 и a_3	[2, 5, 3, 9, 4]
4	a_3 и a_4	[2, 5, 3, 4, 9]
5	a_0 и a_1	[2, 5, 3, 4, 9]
6	a_1 и a_2	[2, 3, 5, 4, 9]
7	a_2 и a_3	[2, 3, 4, 5, 9]
8	a_0 и a_1	[2, 3, 4, 5, 9]
9	a_1 и a_2	[2, 3, 4, 5, 9]
Алгоритм закончен, так как за последний проход (7 и 8 итерации) не выполнено ни одной перестановки элементов		[2, 3, 4, 5, 9]

Заметим, что если за один проход алгоритма не выполнилось ни одной операции перестановки элементов местами, то массив уже отсортирован и можно закончить выполнение алгоритма.

Оценим сложность пузырьковой сортировки. На первом шаге выполняется $n-1$ операций сравнения, на втором – $n-2$ и так далее до 1 операции сравнения на $(n-1)$-м шаге. Получим формулу:

$$(n-1)+(n-2)+\ldots+2+1=n(n-1)/2$$

Следовательно, вычислительная сложность алгоритма – $O(n^2)$.

Однако, если массив "близок к упорядоченному", то алгоритм сортировки пузырьком будет выполнять меньше сравнений: как только на каком-то шаге не будет произведено ни одного обмена, алгоритм можно завершать.

Реализация

```
void bubble_sort(vector<int> &a)
{
    int n = a.size();
    bool unordered = true;
    while (unordered){
        unordered = false;
        for (int j = 0; j < n - 1; ++j){
            if (a[j] > a[j + 1]){
```

```
                swap(a[j], a[j + 1]);
                unordered = true;
            }
        }
        --n;
    }
}
```