



Санкт-Петербургский
государственный
университет

Сортировки

Сортировка вставкой, сортировка выборкой

Никита Гребень

Факультет прикладной математики - процессов управления



Санкт-Петербургский
государственный
университет

- ❶ Немного C++
- ❷ Сортировка вставкой
- ❸ Сортировка выборкой
- ❹ Задачи

Обзор



Санкт-Петербургский
государственный
университет

1 Немного C++

- Задача
- Решение 1
- Решение 2
- Решение 3

2 Сортировка вставкой

3 Сортировка выборкой

4 Задачи

Задача



Санкт-Петербургский
государственный
университет

- ▶ В первой строке дано число $n \leq 1000$, длина массива.
- ▶ Во второй строке целые числа, разделенные пробелами ($a_i \leq 10^9$)
- ▶ Вывести отсортированный массив.

Решение 1

```
1 #include <iostream>
2 using namespace std;
3 const int N = 1001;
4 int a[N];
5 int main() {
6     int n;
7     cin >> n;
8     for(int i = 0; i < n; ++i)
9         cin >> a[i];
10    for(int i = 0; i < n - 1; ++i)
11        for(int j = 0; j < n - 1; ++j)
12            if (a[j] > a[j + 1])
13                swap(a[j], a[j + 1]);
14    for(int i = 0; i < n; ++i)
15        cout << a[i] << " ";
16    return 0;
17 }
```

Решение 2

```
1  #include <iostream>
2  using namespace std;
3  const int N = 1001;
4  int a[N];
5  int main() {
6      ios_base::sync_with_stdio(0);
7      cin.tie(0);
8      int n;
9      cin >> n;
10     for(int i = 0; i < n; ++i)
11         cin >> a[i];
12     for(int i = 0; i < n - 1; ++i)
13         for(int j = 0; j < n - i - 1; ++j)
14             if (a[j] > a[j + 1])
15                 swap(a[j], a[j + 1]);
16     for(int i = 0; i < n; ++i)
17         cout << a[i] << " ";
18     return 0;
19 }
```

Решение 3

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 int main() {
5     ios_base::sync_with_stdio(0);
6     cin.tie(0);
7     int n;
8     cin >> n;
9     vector<int> a(n);
10    for(int& x: a)
11        cin >> x;
12    for(int i = 0; i < n - 1; ++i)
13        for(int j = 0; j < n - i - 1; ++j)
14            if (a[j] > a[j + 1])
15                swap(a[j], a[j + 1]);
16    for(auto x: a)
17        cout << x << " ";
18    return 0;
19 }
```

Обзор



Санкт-Петербургский
государственный
университет

1 Немного C++

2 Сортировка вставкой

- Алгоритм
- Процедура вставки
- Реализация
- Оптимизация
- Сокращенный вариант

3 Сортировка выборкой

4 Задачи

Сортировка вставкой



Санкт-Петербургский
государственный
университет

- ▶ Одна из простейших квадратичных сортировок, являющейся устойчивой и адаптивной.
- ▶ Хорошо показывает себя по сравнению с другими квадратичными сортировками.
- ▶ Эффективна на небольших объемах данных. Выигрывает у **quicksort** и **mergesort** на небольших массивах.

Алгоритм



Санкт-Петербургский
государственный
университет

Сортировка коротко может быть описана следующим образом:

1. Предполагаем, что некоторая последовательность из i первых элементов уже отсортирована.
2. Для $i + 1$ элемента пытаемся *засунуть* его на свое место в уже отсортированном массиве.
3. Изначально предполагаем, что самый первый элемент последовательности является отсортированным.

Процедура вставки



Санкт-Петербургский
государственный
университет

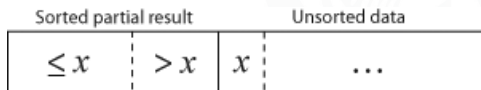


Рис. 1: До вставки элемента

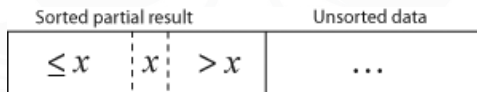


Рис. 2: После вставки элемента

Реализация



Санкт-Петербургский
государственный
университет

```
for(int i = 1; i < n; ++i) {  
    int j = i;  
    while (j > 0 && a[j - 1] > a[j]) {  
        swap(a[j - 1], a[j]);  
        j--;  
    }  
}
```

Оптимизация



Санкт-Петербургский
государственный
университет

```
for(int i = 1; i < n; ++i) {  
    int j = i;  
    int x = a[i];  
    while (j > 0 && a[j - 1] > x) {  
        a[j] = a[j - 1];  
        j--;  
    }  
    a[j] = x;  
}
```

```
for (int i = 1; i < n; ++i) {  
    for (int j = i; j > 0; --j) {  
        if (a[j - 1] < a[j])  
            break;  
        swap(a[j], a[j - 1]);  
    }  
}
```

Обзор



Санкт-Петербургский
государственный
университет

1 Немного C++


2 Сортировка вставкой

3 Сортировка выборкой

- Алгоритм
- Реализация
- Оптимизация
- Сравнение с сортировкой вставками

4 Задачи

Сортировка выборкой



Санкт-Петербургский
государственный
университет

- ▶ Самая очевидная идея.
- ▶ Совершает всего $O(n)$ свопов по сравнению с другими алгоритмами, в $n/2$ раз меньше по сравнению с пузырьковой.
- ▶ Также эффективна на небольших массивах.

Алгоритм

- ▶ Алгоритм предельно прост, аналогично идеи сортировки вставками, разбиваем массив на две части: отсортированного и ожидающего сортировки.
- ▶ Если i первых элементов уже отсортировано, выполняем поиск минимума в промежутке $[i, n - 1]$ и обновляем значение в позиции $i + 1$

6	3	7	2	8	1*
1	3	7	2*	8	6
1	2	7	3*	8	6
1	2	3	7	8	6*
1	2	3	6	8	7*
1	2	3	6	7	8

Рис. 3: Процесс выбора нового элемента

Реализация



Санкт-Петербургский
государственный
университет

```
for (i = 0; i < n - 1; ++i) {  
    for (j = i + 1; j < n; ++j) {  
        if (a[i] > a[j])  
            swap(a[j], a[i]);  
    }  
}
```

```
#include <algorithm>
...
...
for(int i = 0; i < n - 1; ++i)
    swap(a[i], *min_element(a + i, a + n));
```

Сравнение с сортировкой вставками



Санкт-Петербургский
государственный
университет

Таблица 1: Асимптотическая сложность

Случаи	Выбором	Вставками
Наихудшее кл-во сравнений/свапов	$O(N^2)/O(N)$	$O(N^2)/O(N^2)$
Наилучшее кл-во сравнений/свапов	$O(N^2)/O(1)$	$O(N)/O(1)$
Среднее кл-во сравнений/свапов	$O(N^2)/O(N)$	$O(N^2)/O(N^2)$

Обзор



Санкт-Петербургский
государственный
университет

- 1 Немного C++
- 2 Сортировка вставкой
- 3 Сортировка выборкой
- 4 Задачи**
 - Интересная сортировка
 - Хорошая сортировка?

Интересная сортировка



Санкт-Петербургский
государственный
университет

- ▶ Имеется массив a длины n .
- ▶ Массив является интересным, если выполняются следующие два условия:
 1. $a_i \geq a_{i-1}$, где i - четные
 2. $a_i \leq a_{i-1}$, где i - нечетные и больше 1.
- ▶ Необходимо отсортировать массив a так, чтобы сделать его интересным.
- ▶ Считать $n \leq 1000$, $a_i \leq 10^9$

Хорошая сортировка?



Санкт-Петербургский
государственный
университет

- ▶ Имеется массив a длины $n \leq 1000$.
- ▶ Необходимо отсортировать массив по неубыванию, для этого дается следующий алгоритм:

```
for(int i = 0; i < n - 1; ++i)
    for(int j = i; j < n - 1; ++j)
        if (a[j] > a[j + 1]) swap(a[j], a[j + 1]);
```

- ▶ Считать, что на вход программе подается число n . Сгенерировать массив a длины n , который будет являться контрпримером для данного алгоритма сортировки, или сообщить, что для заданной длины n сортировка работает верно при любых значениях a_i .