

Квадратичные сортировки

10 класс



Сортировка пузырьком

По времени **худшая**: $O(N^2)$

По времени **средняя**: $O(N^2)$

По **памяти**: $O(1)$

Обменная сортировка — основана на обмене элементов местами.

Устойчивая сортировка — не меняет порядок элементов с одинаковыми ключами.

Пример сортировки одномерного массива:


[10, -5, 45, 60, 25]  [-5, 10, 25, 45, 60]

Перебор значения

	<div>4</div> <div>9</div> <div>7</div> <div>6</div> <div>2</div> <div>3</div>	<div><u>2</u></div> <div>4</div> <div>9</div> <div>7</div> <div>6</div> <div>3</div>	<div><u>2</u></div> <div><u>3</u></div> <div>4</div> <div>9</div> <div>7</div> <div>6</div>	<div><u>2</u></div> <div><u>3</u></div> <div><u>4</u></div> <div>9</div> <div>7</div> <div>6</div>	<div><u>2</u></div> <div><u>3</u></div> <div><u>4</u></div> <div><u>6</u></div> <div>9</div> <div>7</div>	<div><u>2</u></div> <div><u>3</u></div> <div><u>4</u></div> <div><u>6</u></div> <div><u>7</u></div> <div>9</div>
номер прохода	i=0	i=1	i=2	i=3	i=4	i=5



Сортировка пузырьком(Bubble Sort)



```
int main() {
    int mas[6] = { 10, -5, 45, 60, 25 };
    int len = 5;
    for (int i = 0; i < len; i++) {
        for (int j = 0; j < len - i - 1; j++) {
            if (mas[j] > mas[j + 1]) {
                int t = mas[j];
                mas[j] = mas[j + 1];
                mas[j + 1] = t;
            }
        }
    }

    for (int i = 0; i < len; i++) {
        cout << mas[i] << " ";
    }
}
```

Задание 1. Имеется информация о времени (в секундах) прохождения трассы 25 спортсменов, участвовавших в лыжной гонке. Выведите результат спортсмена-победителя гонки.

Решение без использования сортировки пузырьком

```
#include <iostream>

int main() {
    double times[25] = {27, 42, 89, 16, 4, 70, 73, 94, 61, 18, 6, 60, 52, 2, 50, 33, 64, 1, 84, 39, 47, 79, 81, 20, 57}, ti_me;
    int player = 0;
    for (int i = 1; i <= 24; i++){
        if (times[i] < times[player]){
            player = i;
            ti_me = times[i];
        }
    }
    std::cout << "Спортсмен-победитель гонки - " << ti_me << " секунд." << std::endl;
    std::cout<<player;

    return 0;
}
```

Решение с сортировкой

```
#include <iostream>

int main() {
    double times[25];
    for (int i = 0; i < 25; i++) {
        std::cout << (i + 1);
        std::cin >> times[i];
    }
    // Сортировка пузырьком
    for (int i = 0; i < 25 - 1; i++) {
        for (int j = 0; j < 25 - i - 1; j++) {
            if (times[j] > times[j + 1]) {
                // Обмен элементов
                double temp = times[j];
                times[j] = times[j + 1];
                times[j + 1] = temp;
            }
        }
    }
    std::cout << "участник #1" << times[0] << std::endl;
    return 0;
}
```



Сортировка выбором (Selection Sort) -

Неустойчивая сортировка



По времени **худшая**: $O(N^2)$

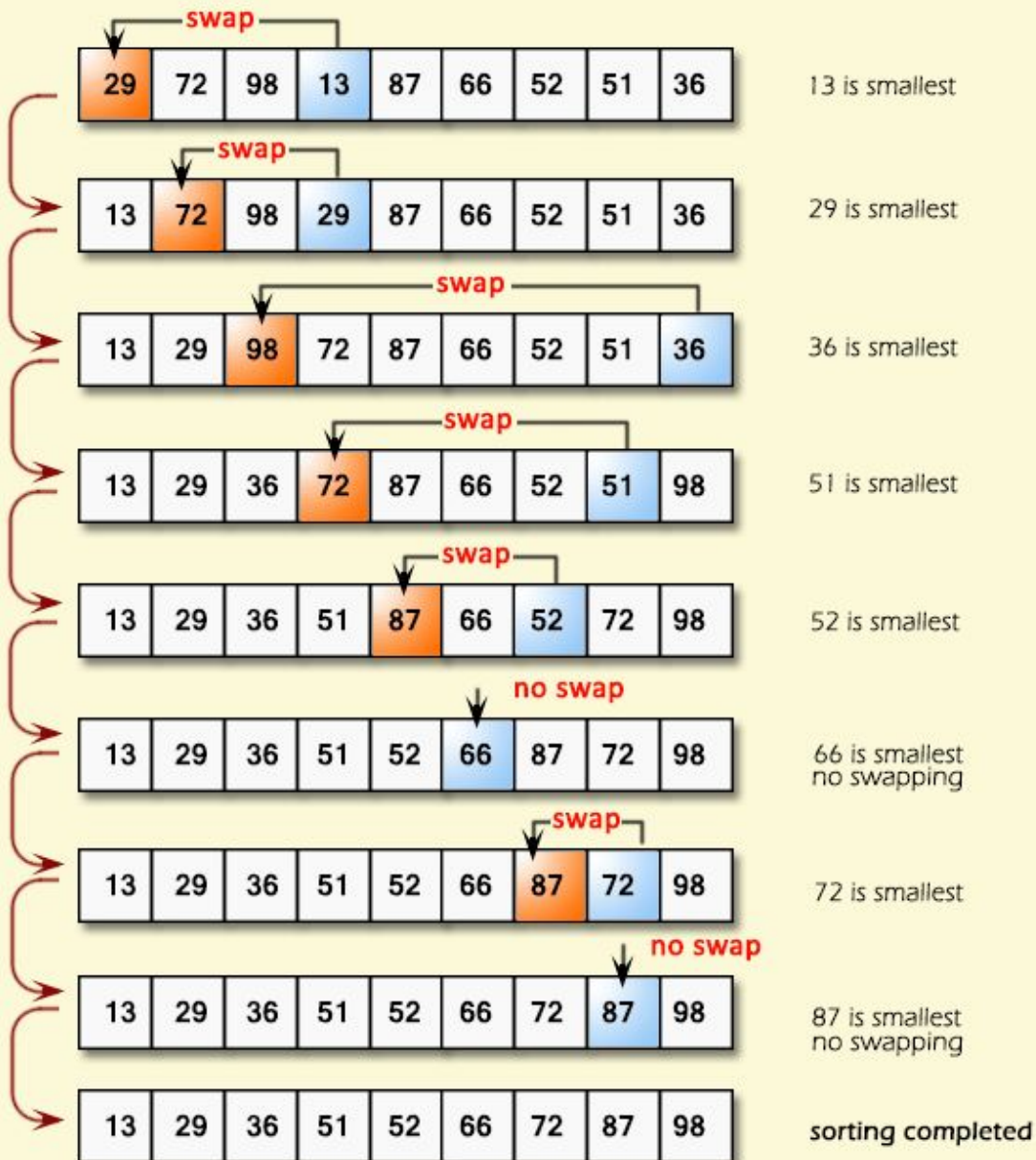
По времени **средняя**: $O(N^2)$

По **памяти**: $O(1)$

Шаги алгоритма:

1. Находим номер минимального значения
2. Производим обмен этого значения со значением первой неотсортированной позиции
3. Теперь сортируем хвост списка, исключив из рассмотрения уже отсортированные элементы.

Selection Sort



(Selection Sort) - это простой алгоритм сортировки, который на каждом шаге выбирает наименьший элемент из оставшихся и перемещает его на своё место.



Сортировка выбором

```
3  □ int main() {
4  |   int mas[6] = { 10, -5, 45, 60, 25 };
5  |   int len = 5;
6  |   □ for (int i = 0; i < len - 1; i++) {
7  |       |   int min_index = i;
8  |       |   □ for (int j = i + 1; j < len; j++) {
9  |       |       |   □ if (mas[j] < mas[min_index]) {
10 |       |       |       |   min_index = j;
11 |       |       |       |   }
12 |       |       |   }
13 |       |   int t = mas[i];
14 |       |   mas[i] = mas[min_index];
15 |       |   mas[min_index] = t;
16 |       |   }
17 |   }
18 |   □ for (int i = 0; i < len; i++) {
19 |       |   cout << mas[i] << " ";
20 |       |   }
21 |   }
```

// Найдем минимальный
элемент в оставшейся
части массива

// Обмен значениями
текущего элемента и
минимального элемента



Сортировка вставками



По времени **худшая**: $O(N^2)$

По времени **средняя**: $O(N^2)$

По времени **лучшая**: $O(N)$

По **памяти**: $O(1)$

Устойчивая сортировка — не меняет порядок элементов с одинаковыми ключами.

Сортировка вставками (*Insertion Sort*) — это простой алгоритм сортировки. Суть его заключается в том что, на каждом шаге алгоритма мы берем один из элементов массива, находим позицию для вставки и вставляем.



Сортировка вставками

```
1  #include <iostream>
2  using namespace std;
3  int main() {
4      int mas[5] = { 10, -5, 45, 60, 25 };
5      int len = 5;
6      for (int i = 1; i < len; i++) {
7          int cur_elem = mas[i];
8          int j = i - 1;
9          while (j >= 0 && mas[j] > cur_elem) {
10             mas[j + 1] = mas[j];
11             j = j - 1;
12         }
13         mas[j + 1] = cur_elem;
14     }
15
16     for (int i = 0; i < len; i++) {
17         cout << mas[i] << " ";
18     }
19 }
```

6 5 3 1 8 7 2 4

Лабораторная работа

Задание 1. Создайте массив длины 10 из элементов 25, 10, -50, 60, 45, 35, 32, 128, 99, 100 и отсортируйте его тремя различными алгоритмами. Подсчитайте количество, обменов выполненное каждый алгоритмом, и количество сравнений, которое выполнит каждый из алгоритмов.

Задание 2. Для массива из задания 1 вычислите программой время работы каждого из трех алгоритмов.

Алгоритм	Время	Кол-во обменов	Кол-во сравнений
Пузырьком			
Выбором			
Вставками		-	

Ссылка на контест

12. Квадратичные сортировки C++ (10ИТ)

8 ноя 2023, 21:34:33
начало: 8 ноя 2023, 20:40:36

Объявления жюри

Завершить

Положение участников Задачи Посылки Сообщения Участники Ответы

А. Результаты олимпиады

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

N участников олимпиады получили уникальные номера от 1 до N. В результате решения задач на олимпиаде, каждый участник получил некоторое количество баллов (целое число от 0 до 600). Известно, кто сколько баллов набрал. Требуется перечислить участников олимпиады в порядке невозрастания набранных ими баллов.

Формат ввода

Вводится сначала число N ($1 \leq N \leq 100$) - количество участников олимпиады. Далее вводится N чисел - количества набранных участниками баллов (1-е число - это баллы, набранные участником номер 1, 2-е - участником номер 2 и т.д.).

Формат вывода

Выведите N чисел - номера участников в порядке невозрастания набранных ими баллов (участники, набравшие одинаковое количество баллов, могут быть выведены в любом порядке).

Пример

Ввод



```
5
100 312 0 321 500
```

Вывод



```
5 4 2 1 3
```

А. Результаты олимпиады (20)

В. Кто там десятый? (20)

С. Количество обменов: пузырек (20)

Д. Обувной магазин (20)

Е. Числа Фибоначчи (10)

Ф. По мотивам ЕГЭ (10)



contest.yandex.ru/contest/55600