



Урок 21. C++: Сортировка выбором

Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 21 из 34 · 45 минут · Модуль «Сортировка»

Уровень: начинающий программист; C++17.

Программа: <https://schoolotzyv.ru/blog/python-cpp-9-klass-programma>

Сортировка выбором ищет минимум всей оставшейся части, а затем делает один обмен. Малое число обменов ещё не означает малое число сравнений.

Результат урока: Реализовать сортировку выбором, проверить готовый префикс и объяснить различие числа сравнений, обменов и устойчивости.

01. Проблемная ситуация

0–5 минут. Для [4,2,3,1] найдите минимум и обменяйте его с первой позицией. Сколько сравнений понадобилось до обмена? Повторите для оставшейся части.

02. Понятия и разобранный пример

5–14 минут: 4 минуты на понятия, 3 на пример, 2 на объяснение способа партнёру.

Минимум суффикса и готовый префикс

Перед шагом i позиции $0...i-1$ уже содержат наименьшие элементы в правильном порядке. Индекс $best$ начинается с i ; просмотр $j=i+1...n-1$ сохраняет в $best$ позицию минимума просмотренной части. Обмен $a[i]$ и $a[best]$ расширяет готовый префикс на один.

Для $n \geq 1$ число сравнений значений всегда $(n-1)+(n-2)+...+1=n(n-1)/2$, независимо от исходного порядка. Код не делает обмен с самим собой, поэтому фактических обменов не более $n-1$. Пустой вектор не входит в цикл благодаря условию $i+1 < a.size()$.

Время $O(n^2)$, дополнительная память $O(1)$. Большой обмен может перескочить через равный ключ и нарушить устойчивость. Это не мешает сортировать голые числа, но важно для записей с одинаковыми баллами. Контракт: $0...100$ целых от -1000 до 1000 .

Отступы набирайте обычными пробелами: 4 на уровень. При копировании с веб-страницы замените неразрывные пробелы обычными; полный код с обычными пробелами есть в PDF.

Для [4,2,3,1] на шаге $i=0$ $best$ становится 1, затем 3; меняем 4 и 1 и получаем [1,2,3,4]. Следующие шаги не обменивают элементы, но сравнения продолжаются. Всего $s=3+2+1=6$, $s=1$. Для уже готового [1,2,3,4] с тоже 6, $s=0$.

```
#include <iostream>
#include <vector>
#include <utility>
std::pair<int,int> selection(std::vector<int>& a) {
    int c=0, s=0;
    for(std::size_t i=0; i+1<a.size(); ++i) {
        std::size_t best=i;
        for(std::size_t j=i+1; j<a.size(); ++j) {
            ++c;
            if(a[j]<a[best]) best=j;
        }
        if(best!=i) {
            std::swap(a[i],a[best]); ++s;
        }
    }
    return {c,s};
}
int main() {
    std::vector<int> a{4,2,3,1};
    auto [c,s]=selection(a);
    for(int x:a) std::cout<<x<<' ';
    std::cout<<"\ncomparisons="<<c<<" swaps="<<s<<"\n";
}
```



Вход	4 2 3 1
i=0; best=3	1 2 3 4
i=1; best=1	1 2 3 4
i=2; best=2	1 2 3 4

Готовый префикс растёт; даже без обмена поиск минимума выполняется.

03. Практика 1. Проверяем материал

Проверяем два счётчика

14–25 минут. В журнале разделяйте сравнения значений и обмены, а не называйте всё одним словом «шаг».

Вход	Результат; c,s
{4,2,3,1}	1 2 3 4; 6,1
{1,2,3,4}	1 2 3 4; 6,0
{ }; {-2}	Пусто; 0,0 / -2; 0,0
{2,1,2}; {3,2,1}	1 2 2; 3,1 / 1 2 3; 3,1

2 минуты — прогноз, 4 — шесть запусков, 3 — ручная проверка best на первом входе, 2 — вывод. Критерии: верные c,s, сохранены повторы, объяснён случай готового ряда.

04. Практика 2. Перенос способа

Проверяем устойчивость на карточках

25–33 минуты. Карточки (балл,метка): (2,A),(2,B),(1,C). Сравнивайте только баллы, как в коде, и перемещайте карточку целиком. За 2 минуты предскажите, за 4 выполните шаги на бумаге, за 2 предложите способ сохранить порядок A,B. Критерии: точная трасса, вывод об устойчивости и рабочая альтернатива.

05. Обсуждение и границы вывода

33–40 минут. Почему при дорогих перемещениях может интересовать выбор, хотя пузырьрёк имеет раннюю остановку? За 2 минуты назовите разные стоимости, за 3 сопоставьте готовый и обратный ряд, за 2 сформулируйте ограниченный вывод.

06. Индивидуальная проверка

40–45 минут. Для {5,4,3,2,1} найдите общее число сравнений, обменов, ряд после первого шага и размер готового префикса после двух шагов. Четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Для четырёх чисел всегда 6 сравнений; для трёх — 3; для нуля и одного — 0. На {3,2,1} один обмен сразу даёт порядок, но оставшийся поиск всё равно сравнивает 2 и 3. Нельзя утверждать, что алгоритм делает столько сравнений, сколько обменов.

Практика 2: разбор и критерии

На первом шаге меняются (2,A) и (1,C): получается (1,C),(2,B),(2,A). На втором равные баллы не меняются, итог C,B,A. A и B поменяли относительный порядок без прямого сравнения на равенство. Устойчивая альтернатива — извлечь минимум C и сдвинуть предшествующие карточки вправо, получив C,A,B; это уже другой способ перемещения, а не текущий swap.

Обсуждение: возможный вывод

Выбор делает не более $n-1$ обменов, но всегда квадратичное число сравнений. Пузырёк на готовом ряду останавливается быстро, а на обратном делает много обменов. Решение зависит от цены сравнения, перемещения и размеров; для обычных прикладных задач сначала рассматривают стандартную сортировку, не объявляя один учебный алгоритм лучшим во всех случаях.

Индивидуальная проверка: ответ и критерии

$c=10$; $s=2$; после первого шага [1,4,3,2,5]; после второго [1,2,3,4,5], готовый гарантированный префикс длины 2. То, что остальная часть случайно готова, не меняет инвариант.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, редактор, папку с правом записи, распечатку ученической части PDF, бумагу и ручку. Компилятор G++ с поддержкой C++17 и редактор. Сборка: `g++ -std=c++17 -Wall -Wextra -pedantic main.cpp -o lesson`. Запуск `./lesson` в Linux/macOS или `./lesson.exe` в Windows (при сборке задайте `-o lesson.exe`). Установку и проверку запуска учитель выполняет заранее. Все программы, данные и ожидаемые ответы даны; регистрация детей и интернет на занятии не нужны.

Уровень — начинающий программист, 9 класс. Предпосылки: vector, индексы, вложенные циклы и ссылки. В паре автор прогнозирует и вводит код, проверяющий ведёт журнал, затем роли меняются. Без компьютера выполните трассировку полного кода на бумаге и сопоставьте с готовыми результатами; учитель демонстрирует подготовленный запуск. Бумажный вариант не считается фактическим запуском ученика. Перед практикой учитель выдаёт готовый файл разобранного примера, чтобы дети тратили время на изменение и проверку, а не длинное перепечатывание. Матрица программы предназначена учителю.

Минуты	Работа	Результат
0–5	Проблема	Вопрос и первое решение
5–14	Понятия и пример	Способ проверки
14–25	Практика 1	Готовый продукт и проверка
25–33	Практика 2	Перенос способа
33–40	Обсуждение	Аргумент и ограничение
40–45	Индивидуальная проверка	Самостоятельный ответ



07. Соответствие исходной программе

Пункт программы	Действие ученика	Проверяемый результат
Алгоритмы сортировки	Проверяем два счётчика	Реализовать сортировку выбором, проверить готовый префикс и объяснить различие числа сравнений, обменов и устойчивости.
Структурирование решения и отладка	Проверяем устойчивость на карточках	Исходник, журнал обычных и граничных тестов, самостоятельное объяснение

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-9-klass-programma>

NIST: сортировка выбором

<https://xlinux.nist.gov/dads/HTML/selectionSort.html>

Тексты, учебные данные, реплики, программы и схемы подготовлены для уроков портала «Школы России». Числовые серии — учебные модели, не результаты проведённых классом опытов. Схемы авторские, сторонние фотографии не используются. Методические ссылки приведены для учителя; ученик получает все материалы здесь. Проверка первичных источников: 18.10.2026.