



Урок 23. C++: Быстрая сортировка

Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 23 из 34 · 45 минут · Модуль «Сортировка»

Уровень: начинающий программист; C++17.

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

Два указателя двигаются навстречу и обменивают элементы не на своей стороне. Даже при равенстве опорному указатели должны сдвинуться, иначе цикл может застрять.

Результат урока: Реализовать быструю сортировку на месте с двумя указателями, проверить повторяющиеся значения и объяснить уменьшение участков.

01. Проблемная ситуация

0–5 минут. Ряд [3,1,3,2], $\text{pivot}=1$ из среднего индекса $(0+3)//2$. Найдите слева первый элемент не меньше pivot и справа первый не больше pivot . Какой обмен и какие сдвиги выполняются?

02. Понятия и разобранный пример

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

Два указателя и копия опорного

Здесь границы l и r включены, в отличие от полуинтервала предыдущего урока. pivot копируется по значению до обменов. i идёт вправо, пока $a[i]<\text{pivot}$, j идёт влево, пока $a[j]>\text{pivot}$. Если $i\leq j$, меняем элементы и обязательно делаем $++i$ и $--j$.

Проверки $i\leq r$ и $j\geq l$ стоят слева от $\&\&$: короткое замыкание не даёт прочесть элемент вне границ. После разбиения сортируются $[l,j]$ и $[i,r]$, только если в участке не меньше двух элементов. `wgapper quick` не вычисляет последний индекс для пустого вектора.

На сбалансированных разбиениях время $O(n \log n)$, в худшем случае $O(n^2)$, глубина стека может стать $O(n)$. Среднее положение pivot не гарантирует медиану. В этой реализации вспомогательные данные каждого вызова постоянны, но стек не бесплатен. Учебный контракт: не более 100 целых -1000...1000; сортировка меняет исходный `vector` и не гарантирует устойчивость.

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 [3,1,3,2], $l=0, r=3, \text{pivot}=1$: $i=0$, j движется к 1, обмен $a[0]$ и $a[1]$ даёт [1,3,3,2], затем $i=1, j=0$. Левая часть закончена, правая [1,3] сортируется отдельно; итог [1,2,3,3].

```
#include <iostream>
#include <vector>
#include <utility>
void qs(std::vector<int>& a, int l, int r) {
    int i=l, j=r, pivot=a[l+(r-l)/2];
    while(i<=j) {
        while(i<=r && a[i]<pivot) ++i;
        while(j>=l && a[j]>pivot) --j;
        if(i<=j) {
            std::swap(a[i],a[j]); ++i; --j;
        }
    }
    if(l<j) qs(a,l,j);
    if(i<r) qs(a,i,r);
}
void quick(std::vector<int>& a) {
    if(a.size()>1) qs(a,0,int(a.size())-1);
}
int main() {
    std::vector<int> a{3,1,3,2};
    quick(a);
    for(int x:a) std::cout<<x<<' ';
    std::cout<<'\n';
}
```



Начало	[3,1,3,2]; pivot=1
Указатели	i=0; j=1 после поиска
Обмен и сдвиг	[1,3,3,2]; i=1,j=0
Следующая задача	Только [1,3], затем итог [1,2,3,3]

Границы включены; после обмена оба указателя сдвигаются даже для равных значений.

03. Практика 1. Проверяем материал

Проверяем повторы и пустой вектор

14–25 минут. Используйте wrapper quick, не вызывайте qs с пустым вектором.

Вход	Результат
{3,1,3,2}	1 2 3 3
{}; {9}	Пусто; 9
{4,4,4,4}	4 4 4 4, завершение
{5,4,3,2,1}; {-2,0,-2}	1 2 3 4 5; -2 -2 0

2 минуты — прогноз, 4 — шесть запусков, 3 — ручная трасса равных значений, 2 — журнал границ. Критерии: сохранены повторы, все вызовы завершаются, границы не выходят за vector, объяснена копия pivot.

04. Практика 2. Перенос способа

Исследуем глубину разбиений

25–33 минуты. На бумаге проследите первую фазу для {1,2,3,4,5,6,7}, затем вариант с pivot=a[l] на свежей копии. За 2 минуты найдите pivot, за 4 выпишите размеры рекурсивных участков, за 2 оцените риск продолжения. Код прочих строк не меняйте. Критерии: обе пары размеров и объяснение худшего случая.

05. Обсуждение и границы вывода

33–40 минут. Можно ли обещать устойчивость, раз сравнения строгие? За 2 минуты возьмите карточки (2,A),(2,B), за 3 выполните обмен, за 2 объясните, когда нужен другой алгоритм.

06. Индивидуальная проверка

40–45 минут. Для {2,2} укажите pivot, индексы после первого обмена, наличие рекурсивных вызовов и причину безопасной пустоты через quick. Четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

У четырёх четвёрок i, j после двух обменов равных значений становятся 2,1; возникают два участка длины 2, затем они заканчиваются. Без $++i, --j$ первый обмен равных оставил бы состояние прежним. `pivot` должен быть отдельным `int`: ссылка на элемент могла бы изменить его значение при обмене.

Практика 2: разбор и критерии

Средний `pivot=4`: указатели встретятся на индексе 3, после обмена с собой $i=4, j=2$; участки $[0,2]$ и $[4,6]$ имеют длины 3 и 3. Первый `pivot=1` даёт после первого разбиения $i=1, j=-1$; остаётся $[1,6]$ длины 6. На возрастающем ряду выбор первого продолжает отделять по одному, вызывая квадратичную работу и линейную глубину.

Обсуждение: возможный вывод

При равных ключах оба поиска останавливаются, карточки А и В обмениваются: получится В,А. Строгие сравнения сами по себе не дают устойчивость двухуказательной сортировке. Если важен порядок поступления равных записей, используют устойчивую сортировку либо явно добавляют исходный номер к ключу.

Индивидуальная проверка: ответ и критерии

`pivot=2; i=1, j=0`; рекурсивных вызовов нет, оба проверяемых участка одноэлементные; `quick` для пустоты не вызывает `qs` и не вычисляет `size()-1`. По одному баллу.

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: рекурсия, `vector`, ссылки, индексы и обмен. 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 повторы и пустой вектор	Реализовать быструю сортировку на месте с двумя указателями, проверить повторяющиеся значения и объяснить уменьшение участков.
Структурирование решения и отладка	Исследуем глубину разбиений	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: разбиение и ограничения быстрой сортировки

<https://xlinux.nist.gov/dads/HTML/quicksort.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.