

Тема лекции:

Алгоритмы сортировки данных

Сортировка – процесс упорядочивания данных по заданному правилу.

Обычно с упорядоченными элементами проще работать, чем с произвольно расположенными: легче найти необходимые элементы, исключить или добавить новые.

В общем, методы сортировок разделяют на два типа: ***сортировку массивов*** и ***сортировку файлов***.

На практике массивы хранятся в оперативной памяти устройства, а файлы в более «медленных» внешних устройствах хранения но более вместительных.



К методам сортировки применяются два основных требования:

- 1) **Математическое** – алгоритм должен быть корректен математически;
- 2) **Бизнес-требование** – алгоритм должен удовлетворять требованиям конкретной бизнес-задачи.





Математическая корректность может не удовлетворять бизнес-требование, но бизнес-требование должно быть математически корректным.



Пример математической корректности алгоритма

Сортировка
от дешевых к дорогим ▾

Новинка

 <p>Bravis NB70 Black</p> <p>999 грн</p> <p>★★★★★ 15 отзывов</p> <p>Купить</p> <p>IPS 4 ЯДРА</p>	 <p>Bravis NB701 Black</p> <p>999 грн</p> <p>★★★★★ 73 отзыва</p> <p>Купить</p> <p>КРЕДИТ 0% 3 МЕС. 4 ЯДРА</p>	 <p>Elenberg TAB716 Black</p> <p>999 грн</p> <p>★★★★★ 2 отзыва</p> <p>Купить</p> <p>КРЕДИТ 0% 3 МЕС. 4 ЯДРА</p>	 <p>Ainol Novo7 White</p> <p>1 099 грн</p> <p>★ 1 отзыв</p> <p>Купить</p> <p>КРЕДИТ 0% 3 МЕС. IPS 4 ЯДРА</p>
---	--	--	---

Пример бизнес-требования к алгоритму

Сортировка
популярные

Топ продаж



ROZETKA

Lenovo TAB 2 A7-10 7" 8GB WiFi Black (59434747)

1 999 грн ★★★★★ 301 отзыв



Топ продаж



Bravis NB102 Black

1 999 грн ★★★★★ 62 отзыва



Эксклюзивно



Товар находится в Польше, срок поставки до 7 рабочих дней. Доставка возможна только курьерской службой Мист Экспресс.

Xiaomi MiPad 64GB White (Международная версия)

3 999 грн ★★★★★ 44 отзыва



Топ продаж



ROZETKA

Lenovo TAB 2 A7-10 7" 8GB WiFi Black (59434747) - Уценка

1 849 грн ★★★★★ 16 отзывов



Сортировка массивов

Главное требование к разработке алгоритмов сортировки массивов – экономное использование доступной оперативной памяти.

Хорошая мера эффективности – число необходимых сравнений и перестановок элементов.

Хорошие методы сортировки требуют порядка $n \cdot \log(n)$ сравнений, простые же – n^2 .

Характеристики простых методов:

- 1) Хороши для понимания основных принципов сортировок.
- 2) Легки для понимания.
- 3) Обычно более быстры для малых n но нельзя использовать для больших n .

Пузырьковая сортировка $O(n^2)$

Дано: массив значений $\{ A[1], A[2], \dots, A[n] \}$
 n – число элементов

Псевдокод:

Для i от 1 до $n-1$ выполнять

Для k от $i+1$ до n выполнять

Если $A[k] < A[i]$ то

tmp:=A[k];

A[k]:=A[i];

A[i]:=tmp;

Пузырьковая сортировка $O(n^2)$

6 5 3 1 8 7 2 4

Сортировка вставками $O(n^2)$

Дано: массив значений $\{ A[1], A[2], \dots, A[n] \}$

n – число элементов

Псевдокод:

Для i от 1 до n выполнять

key = $A[i]$;

$k=i-1$;

Пока ($k>0$) и ($A[j]>key$) выполнять

$A[k+1]:=A[k]$;

$k:=k-1$;

$A[k+1]:=key$;

Сортировка вставками $O(n^2)$

6 5 3 1 8 7 2 4

Сортировка слиянием $O(n \log_2(n))$

1. Сортируемый массив разбивается на две части примерно одинакового размера;
2. Каждая из получившихся частей сортируется отдельно, например — этим же самым алгоритмом;
3. Два упорядоченных массива половинного размера соединяются в один.

Алгоритм объединения двух упорядоченных массивов A и B (K_1 – длина A, K_2 – длина B):

$i = 1; k = 1;$

Пока ($i \leq K_1$) и ($k \leq K_2$) выполнять

если $A[i] < B[k]$ то поместить в выходной массив $A[i]$ и $i = i + 1$

иначе поместить в выходной массив $B[k]$ и $k = k + 1$

Поместить в выходной массив оставшиеся элементы массива A с i -го по K_1 и массива B с k -го по K_2 .

Сортировка слиянием $O(n \log_2(n))$

6 5 3 1 8 7 2 4

Быстрая сортировка $O(n \log_2(n))$

- 1) Выбираем в массиве некоторый элемент, который будем называть опорным элементом. Например, средний по положению.
- 2) Операция разделения массива: реорганизуем массив таким образом, чтобы все элементы, меньшие или равные опорному элементу, оказались слева от него, а все элементы, большие опорного — справа от него. Обычный алгоритм операции:
 1. Два индекса — l и r , приравниваются к минимальному и максимальному индексу разделяемого массива соответственно.
 2. Вычисляется индекс опорного элемента m . В нашем случае $m=(l+r)/2$.
 3. Индекс l последовательно увеличивается до m до тех пор, пока l -й элемент не превысит опорный.
 4. Индекс r последовательно уменьшается до m до тех пор, пока r -й элемент не окажется меньше либо равен опорному.
 5. Если $r = l$ — найдена середина массива — операция разделения закончена, оба индекса указывают на опорный элемент.
 6. Если $l < r$ — найденную пару элементов нужно поменять местами и продолжить операцию разделения с тех значений l и r , которые были достигнуты.
- 3) Рекурсивно упорядочиваем подмассивы, лежащие слева и справа от опорного элемента, пока размеры подмассивов не будут равны 1

Быстрая сортировка $O(n \log_2(n))$

Дано множество

{9,6,3,4,10,8,2,7}

Берем **9** в качестве базового элемента. Сравниваем **9** с противоположно стоящим элементом, в данном случае это **7**. **7** меньше, чем **9**, следовательно элементы меняются местами.

{7,6,3,4,10,8,2,9}

Далее начинаем последовательно сравнивать элементы с **9**, и менять их местами в зависимости от сравнения.

{7,6,3,4,10,8,2,9}

{7,6,3,4,10,8,2,9}

{7,6,3,4,10,8,2,9}

{7,6,3,4,9,8,2,10} - 9 и 10 меняем местами.

{7,6,3,4,8,9,2,10} - 9 и 8 меняем местами.

{7,6,3,4,8,2,9,10} - 2 и 9 меняем местами.

После такого перебрасывания элементов весь массив разбивается на два подмножества, разделенных элементом 9.

{7,6,3,4,8,2} и {10}

Быстрая сортировка $O(n \log_2(n))$

Далее по уже отработанному алгоритму сортируются эти подмножества. Подмножество из одного элемента естественно можно не сортировать. Выбираем в первом подмножестве базовый элемент 7.

{7,6,3,4,8,2}

{2,6,3,4,8,7} - меняем местами 2 и 7.

{2,6,3,4,8,7}

{2,6,3,4,8,7}

{2,6,3,4,8,7}

{2,6,3,4,7,8} - меняем местами 7 и 8

Получили снова два подмножества.

{2,6,3,4} и {8}

Далее алгоритм продолжается.

Быстрая сортировка $O(n \log_2(n))$

