

Алгоритмы и структуры данных на Python. Интерактивный  
курс

## Урок 7



# Алгоритмы сортировки

Сортировка пузырьком. Быстрая  
сортировка (Quick sort).  
Сортировка Шелла.

# План

- Понятие сортировки
- Способы оценки сортировки
- Сортировка пузырьком



# Понятие сортировки

**Сортировка** - это упорядочивание элементов любой структуры хранения данных в соответствии с выбранным критерием.





# Алгоритм сортировки

Обычно под **алгоритмом сортировки** подразумевают упорядочивание множества элементов по возрастанию или убыванию одного из «ключей»



# Практически каждый алгоритм сортировки можно разбить на 3 части:

1. Сравнение, определяющее упорядоченность пары элементов;
2. Перестановка, меняющая местами пару элементов;
3. Сортирующий алгоритм, который сравнивает и переставляет элементы до тех пор, пока все они не будут упорядочены.



# Алгоритм сортировки пузырьком

- Сложность:  $O(n^2)$
- Устойчивость (стабильность): Устойчивая
- Тип (категория): Обменная
- Потребление памяти: Не требует доп. памяти



# Итоги:

## Теория

- Сортировки и способы (критерии) оценки сортировки

## Практика

- Сортировка пузырьком





# План

- Сортировка выбором



# Алгоритм сортировки выбором

- Сложность:  $O(n^2)$
- Устойчивость (стабильность): Устойчивая / неустойчивая
- Тип (категория): Выбором
- Потребление памяти: Не требует доп. памяти



# Алгоритм сортировки выбором

- Найти наименьший элемент в неотсортированной части массива
- Поменять его местами с первым элементом в неотсортированной части массива
- Продолжать эти действия, пока весь массив не будет отсортирован.



# Итоги:

## Теория и Практика

- Сортировка выбором



# План

- Сортировка вставками



# Алгоритм сортировки вставками

- Сложность:  $O(n^2)$  / лучшее время  $O(n)$
- Устойчивость (стабильность): Устойчивая
- Тип (категория): Вставками
- Потребление памяти: Не требует доп. памяти



# Алгоритм сортировки вставками

- Из массива последовательно берется каждый элемент, кроме первого (`index == 0`).
- И вставляется в отсортированную часть массива.



# Итоги:

## Теория и Практика

- Сортировка вставками





# План

- Сортировка Шелла



# Алгоритм сортировки Шелла

- Сложность:  $O(n^2)$  /  $O(n (\log n)^2)$  или  $O(n^{3/2})$
- Устойчивость (стабильность): Неустойчивая
- Тип (категория): Вставками
- Потребление памяти: Не требует доп. памяти



# Алгоритм сортировки Шелла

- Выбираем шаг для сравнения элементов (increment).
- Сравниваем последовательно элементы массива находящиеся один от другого на расстоянии шага.
- Уменьшаем шаг и повторяем пункт два.



War (increment)

1, 4, 10, 23, 57, 132, 301, 701, 1750



# Итоги:

## Теория и Практика

- Сортировка Шелла

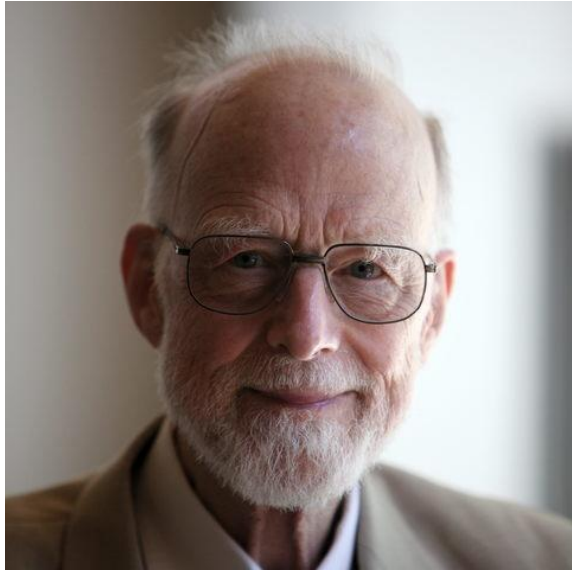


# План

- Быстрая сортировка. Сортировка Хоара



# Сэр Чарльз Энтони Ричард Хоар



Чарльз Хоар - автор алгоритма Быстрой сортировки.

Алгоритм был разработан в 1960 во время обучения в МГУ.



# Алгоритм быстрой сортировки

- Сложность:  $O(n^2)$  /  $O(n \log n)$
- Устойчивость (стабильность): Неустойчивая
- Тип (категория): Обменная
- Потребление памяти:  $O(n)$  / Не требует доп. памяти





# Алгоритм быстрой сортировки

- Выбираем опорный элемент (pivot).
- Сравниваем элементы массива с опорным и переставляем их так, чтобы разбить массив на три непрерывных отрезка: «меньшие опорного», «равные» и «большие».
- Для отрезков «меньше» и «больше» рекурсивно выполнить сортировку.



# Итоги:

## Теория и Практика

- Сортировка Хоара. Быстрая сортировка.



# План

- Разворот массива
- Сортировка по умолчанию в Python
- Сортировка сложных структур с использованием ключа



# Алгоритм сортировки Timsort

- Сложность:  $O(n \log n)$
- Устойчивость (стабильность): Устойчивая
- Тип (категория): Гибридная  
(Вставками + Слиянием)
- Потребление памяти:  $O(n)$



# Итоги:

## Теория и Практика

- Разворот массива
- Сортировка по умолчанию в Python
- Сортировка сложных структур с использованием ключа



# Домашнее задание

1. Отсортировать по убыванию методом «пузырька» одномерный целочисленный массив, заданный случайными числами на промежутке  $[-100; 100)$ . Вывести на экран исходный и отсортированный массивы.
2. Отсортируйте по возрастанию методом слияния одномерный вещественный массив, заданный случайными числами на промежутке  $[0; 50)$ . Выведите на экран исходный и отсортированный массивы.



# Домашнее задание

3. Массив размером  $2m + 1$ , где  $m$  – натуральное число, заполнен случайным образом. Найти в массиве медиану – элемент ряда, делящий его на две равные части: в одной находятся элементы, которые не меньше медианы, в другой – не больше медианы.

Примечание: Задачу можно решить без сортировки исходного массива. Но если это слишком сложно, то используйте метод сортировки, который не рассматривался на уроках.



# План

- Разбор домашнего задания

