Знакомство с сортировками

Для вопросов по курсу:

- Почта дисциплины - alg@moevm.pro

Префикс в теме письма [ALG_53XX]

Напоминание

- Сложность по времени - зависимость числа операций от объема входных данных. Сложность по памяти - зависимость объема необходимой памяти от объема входных данных.
- Если алгоритм не затрачивает дополнительную память, то он является in-place, иначе out-of-place (или не in-place).
- Устойчивая (стабильная) сортировка — сортировка, которая не меняет относительный порядок сортируемых элементов, имеющих одинаковые ключи, по которым происходит сортировка.

Виды сортировок

- Не основанные на сравнении элементов

Сортировка подсчетом

- Основанные на сравнение элементов

Сортировки пузырьком, вставками, слиянием, быстрая, timsort

Сортировка подсчетом

➢ Основная идея алгоритма: произвести сопоставление:

число – количество вхождений этого числа;

выписать по возрастанию все вхождения чисел.

	Время	Память	Устойчивая
Худший случай	O(n+k)	O(n+k)	Нет
Средний случай	O(n+k)		
Лучший случай	O(n+k)		

n - объем входных данных, k - количество различных элементов

Почему такая сложность?

Время:

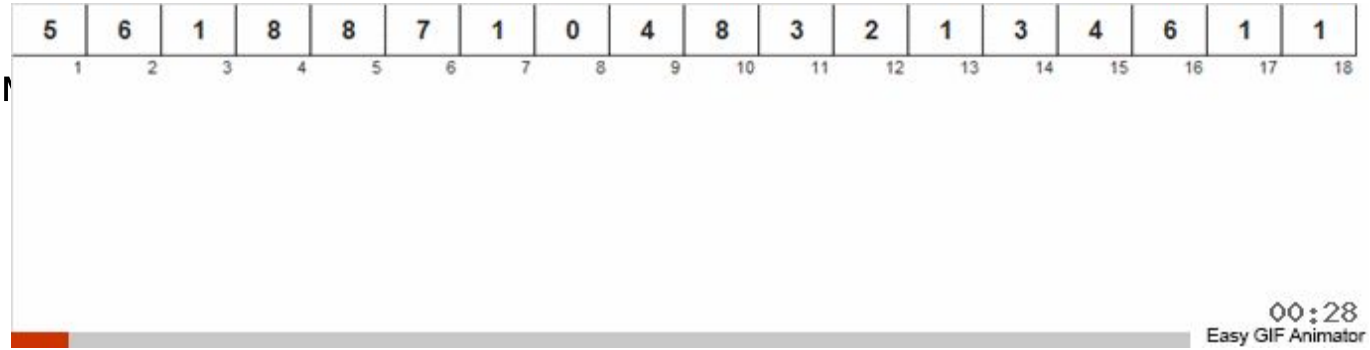
Необходимо пройтись по всем элементам (n), посчитав их количество (k).

Память:

Необходимо хранить количество всех различных элементов (k).

Необходимо сформировать сортированное множество из n элементов (n).

n - объем



Сортировка вставками

1. Массив разбивается на отсортированную и неотсортированную части
2. Берем элемент в неотсортированной части массива
3. Находим место элемента в отсорованной части массива и вставляем

	Время	Память	Устойчивая
Худший случай	n^2	in-place	Да
Средний случай	n^2		
Лучший случай	n		

n - объем входных данных

6 5 3 1 8 7 2 4

Почему такая сложность?

Время:

В лучшем случае массив отсортирован (n).

В худшем случаем искать место среди k элемента придется среди (k-1) элементов. O(k*(k-1)) == O(k^2) или же O(n^2).

Память:

in-place

n - объем входных данных

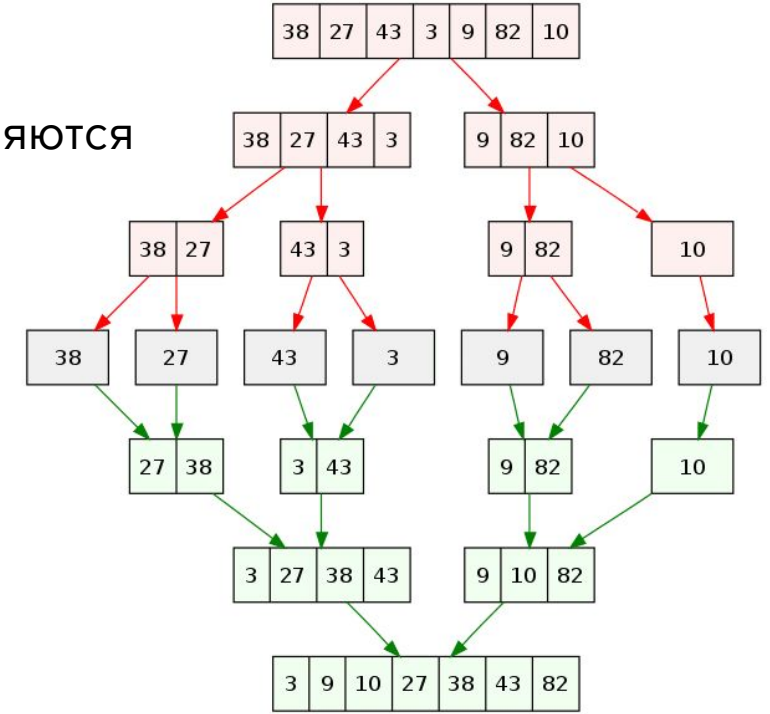


Сортировка слиянием

1. Рекурсивно массив разбивается надвое
2. Если в массиве 1 элемент => массив отсортирован
3. Отсортированные подмассивы объединяются

	Время
Худший случай	n*log(n)
Средний случай	n*log(n)
Лучший случай	n*log(n)
Память	O(n)
Устойчивая	Да

n - объем входных данных



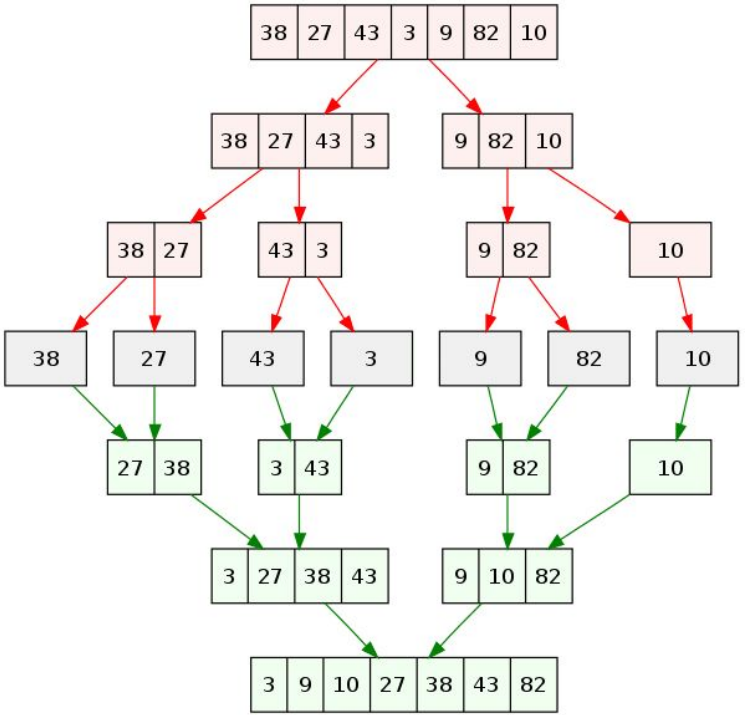
Почему такая сложность?

Время:

В любом (как в лучшем, так и в худшем) случае весь массив (n) рекурсивно дробится надвое. Рекурсия приводит к логарифмической сложности O(n*log(n)). Основание логарифма традиционно не учитывается, но оно 2.

Память:

Необходимо дополнительно сохранить каждый элемент (n).

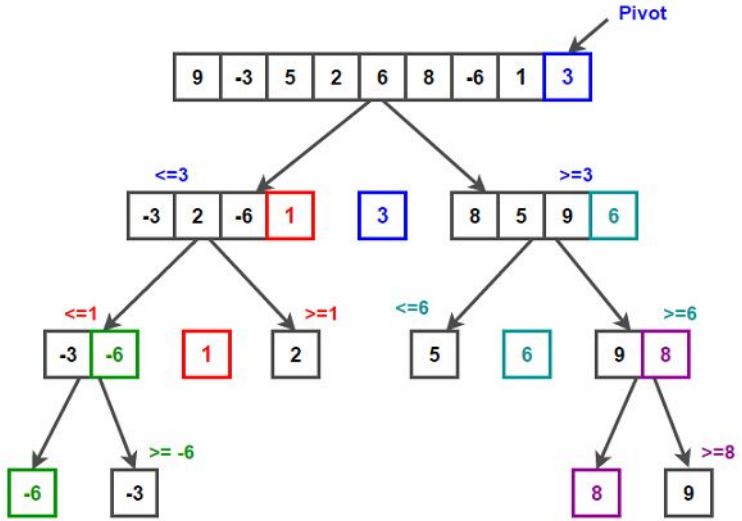


n - объем входных данных

Быстрая сортировка

Выбирается опорный элемент. Все значения меньше него перемещаются влево от него, а все ключи больше либо равные - вправо.
Далее алгоритм рекурсивно применяется к каждой из частей.

	Время
Худший случай	n^2
Средний случай	n*log(n)
Лучший случай	n*log(n)
Память	O(n)*
Устойчивая	Нет**



n - объем входных данных * / ** - модификации исходного алгоритма

Почему такая сложность?

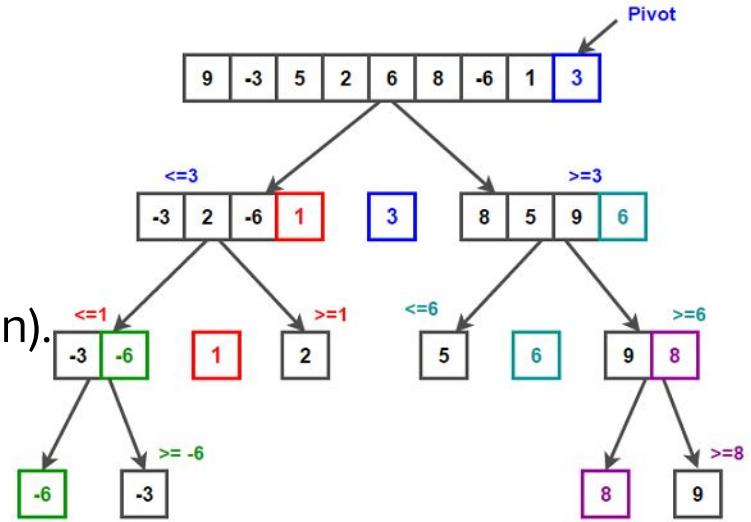
Время:

В худшем случаем опорный элемент k приведет к сдвигу (k-1) элементов. O(k*(k-1)) == O(k^2) или же O(n^2).

В случае выбора (или почти хорошего) опорного элемента рекурсия даст логарифмическую сложность O(n*log(n))

Память:

Необходимо сохранить каждый элемент (n).



n - объем входных данных

Полезные / используемые материалы

- Визуализация алгоритмов:
  - https://coddy.tech/visualize/ru/sorting
- Рассмотренные сортировки:
  - https://thecode.media/counting-sort/
  - https://thecode.media/merge-sort/
  - https://thecode.media/qsort/

Вопросы по курсу можно задавать:

Почта дисциплины - alg@moevm.pro
Иванов Дмитрий Владимирович - dmitry.ivanov@moevm.pro