Задача 1-1 (30 баллов). Пусть заданы n ключей $a_1, \ldots, a_n$ и m запросов на поиск k-й порядковой статистики $k_1, \ldots, k_m$, $m \geqslant 2$. Предложите алгоритм, который выполняет все эти запросы за время $O(n \log m + m)$. Указание: вообразите процесс одновременного (sic!) поиска ответов для всех k_i с помощью алгоритма `Quick-Select`, в котором на каждом вызове используется линейный детерминированный алгоритм поиска медианы. Очевидно, что одновременное знание всех k_i позволяет переиспользовать значительную часть результатов для разных k_i.

Решение. Предложим алгоритм, основанный на модификации детерминированного алгоритма выбора порядковой статистики (`Select`), который одновременно обрабатывает все запросы из множества K. Алгоритм работает рекурсивно и на каждом шаге делит как массив элементов, так и множество искомых порядковых статистик.

Шаг 1: Инициализация

Сортируем множество K по возрастанию:

$$k_1 \leqslant k_2 \leqslant \ldots \leqslant k_m.$$

Вызываем рекурсивную функцию $\texttt{Select}(A, K, l = 1)$, где l — смещение индексов относительно исходного массива (изначально равно 1).

Шаг 2: Рекурсивная функция `Select`

Функция $\texttt{Select}(A, K, l)$ выполняет следующие действия:

1. **Базовый случай**: Если $|K| = 0$ или $|A| = 0$, то возвращаемся.
2. **Поиск медианы**: Находим медиану массива A с помощью детерминированного линейного алгоритма поиска медианы. Обозначим найденную медиану как x.
3. **Разбиение массива**: Разбиваем массив A на три подмассива:

   $$A_L = \{a \in A \mid a < x\}, \quad A_M = \{a \in A \mid a = x\}, \quad A_R = \{a \in A \mid a > x\}$$

   Обозначим размеры подмассивов:

   $$n_L = |A_L|, \quad n_M = |A_M|, \quad n_R = |A_R|$$

4. **Разбиение множества K**: Для каждого $k \in K$ определяем, в какой подмассив попадает искомая порядковая статистика:

   - Если $k - l + 1 \leqslant n_L$, то k относится к подмножеству K_L.
   - Если $n_L < k - l + 1 \leqslant n_L + n_M$, то k относится к подмножеству K_M.
   - Если $k - l + 1 > n_L + n_M$, то k относится к подмножеству K_R.

   Обозначим соответствующие подмножества как K_L, K_M, K_R.

5. **Рекурсивные вызовы**:
   - Для левого подмассива: Вызываем `Select`(A_L, K_L, l).
   - Для элементов, равных медиане: Для каждого $k \in K_M$ присваиваем ответ x.
   - Для правого подмассива: Вызываем `Select`$(A_R, K_R, l' = l + n_L + n_M)$, где l' — новое смещение индексов.

Анализ сложности

Рассмотрим временную сложность алгоритма:

- **Поиск медианы**: На каждом уровне рекурсии поиск медианы выполняется за $O(|A|)$.
- **Разбиение массива**: Разбиение массива относительно медианы также занимает $O(|A|)$.
- **Разбиение множества K**: Так как множество K отсортировано, разбиение его на подмножества K_L, K_M, K_R выполняется за $O(|K|)$ с использованием техники двух указателей.

Общая временная сложность на каждом уровне рекурсии составляет $O(|A| + |K|)$.

Общее время работы

Так как на каждом уровне рекурсии размеры массивов уменьшаются как минимум вдвое благодаря выбору медианы в качестве опорного элемента, глубина рекурсии составляет $O(\log m)$, где m — число запросов.

Суммарное время работы алгоритма:

$$T(n, m) = \sum_{i=0}^{\log m} O\left(\frac{n}{2^i} + \frac{m}{2^i}\right) = O(n \log m + m).$$

Таким образом, общее время работы алгоритма составляет $O(n \log m + m)$.