Алгоритмы и структуры данных.
Сортировки и порядковые статистики, решаем вместе.

1. Обсудите рандомизированный алгоритм поиска порядковой статистики. Используя схему доказательства асимптотики алгоритма по индукции, покажите, что рандомизированный поиск порядковой статистики работает за ожидаемое линейное время.

2. Покажите, как реализовать алгоритм сортировки вставками, чтобы время его работы составляло $O(n + inv)$, где inv — количество инверсий в массиве.

3. Пусть у вас есть массив с элементами некоторого типа, для любой пары элементов вам доступна операция сравнения. Некоторые элементы равны, но при этом всё равно отличимы по вспомогательной информации. Разрешается изменять вспомогательную информацию элементов (то есть никак не используемую при сравнении). Также имеется функция *MagicSort* которая за линейное время сортирует массив элементов данного типа. Обращение к данной функции возможно только как к чёрному ящику. Предложите, как на основе функции *MagicSort* построить устойчивый алгоритм сортировки за линейное время.

4. Оцените время работы алгоритма быстрой сортировки Хоара, в случае если в качестве *pivot* элемента выбирается:

   (a) Элемент стоящий посередине, то есть $a[(lb+rb)/2]$. Оцените время работы в худшем случае, в среднем (ожидаемое) и на случайных данных.

   (b) Медиана случайных $\sqrt{rb - lb + 1}$ элементов. Оцените время работы в худшем, в среднем (ожидаемое) и на случайных данных.

5. Даны n строк суммарной длины L. Чему будет равно математическое ожидание времени работы алгоритма быстрой сортировки, при условии что лексикографическое сравнение двух строк выполняется наивно?

6. Инверсией называется пара (i, j), такая что $i < j$ и $a_i > a_j$. Обсудите как модифицировать алгоритм сортировки слиянием, чтобы параллельно вычислять число инверсий.

7. Супер-инверсией называется тройка (i, j, k), такая что $i < j < k$ и $a_i > a_j > a_k$. Предложите алгоритм подсчёта количества супер-инверсий за время $O(n \log n)$, основанный на сортировке слиянием. Как обобщить алгоритм на поиск k-инверсий, то есть наборов $i_1 < i_2 < \ldots < l_k$, что $a_{i_1} > a_{i_2} > \ldots > a_{i_k}$.

8. Пусть некоторый алгоритм выбирает случайную пару элементов (i, j) и меняет их местами, если они нарушают условие упорядоченности. Покажите, что математическое ожидание количества действий данного алгоритма:

   (a) $O(n^4)$;

   (b) $O(n^2 \log n)$;

9. Докажите, что поиск элемента в отсортированном массиве, использующий лишь операции сравнения, в худшем случае работает не быстрее, чем за $O(\log\ n)$.