

1. Поймите, как обрабатывается последовательность запросов `insert 1, insert 2, ..., insert n` в splay-дереве.
2. Покажите, что splay-дерево эффективно (линейно) работает, если использовать его как стек, то есть в качестве ключа при добавлении использовать только текущий размер дерева  $n$ , а удалять только ключ  $n - 1$ .
3. Приведите последовательность из  $n$  операций `insert`, после обработки которых splay-дерево вырождается в бамбук. Предположим, что процедура `erase` не вызывает `splay`. Приведите пример последовательности операций `erase`, которая в таком случае обрабатывается за  $\Omega(n^2)$ . Поймите, что происходит при корректной обработке этих запросов.
4. Как сделать `merge` и `split` в splay-дереве? Обратите внимание, что время обработки таких операций не должно превосходить время работы соответствующих операций `splay`.
5. (Неявное дерево поиска) Задан массив  $a_1, \dots, a_m$ . К нему поступают запросы вида: а) вставить  $x$  в позицию  $pos$  (то есть между двумя уже существующими элементами, нумерация смещается); б) удалить число на позиции  $pos$  (нумерация смещается); в) сообщить сумму на подотрезке массива. Обработайте каждый запрос за  $O(\log n)$ , где  $n$  – текущий размер массива.
6. Задан массив  $a_1, \dots, a_m$ . К нему поступают запросы вида: а) вставить  $x$  в позицию  $pos$  (то есть между двумя уже существующими элементами); б) удалить число на позиции  $pos$  (нумерация смещается); в) сообщить сумму на подотрезке массива; г) прибавить ко всем числам подотрезка одно и то же число; д) развернуть подотрезок. Обработайте каждый запрос за  $O(\log n)$ , где  $n$  — текущий размер массива.
7. (Своппер) Дан массив  $a_1, \dots, a_n$ . К нему поступает  $q$  запросов одного из двух видов: а) найти сумму на отрезке; б) по данным  $l$  и  $r$  поменять местами числа  $a_l, a_{l+1}$ , поменять местами  $a_{l+2}, a_{l+3}$ , и так далее, вплоть до пары  $a_{r-1}, a_r$  (считайте, что  $r - l + 1$  чётно). Ответьте на все запросы за  $O((n + q) \log n)$ .