Маленькому Пете на День рождения папа подарил массив из n целых чисел, в котором каждое из чисел было в пределах от 1 до m. Петя любит числовые массивы, так что он очень обрадовался и тут же начал задавать папе запросы на этом массиве. Поскольку Петя — активный мальчик, он очень любит разнообразие и краткость. Поэтому при каждом запросе он выбирал l-й элемент массива и спрашивал папу, при каком минимальном значении r среди элементов с l-го по r-й включительно будет не менее чем k различных чисел.

Сначала папа пытался честно отвечать на Петины запросы, но их было слишком много, поэтому он быстро устал и попросил вас помочь ему.

Первая строка входного файла содержит два целых числа n и m, разделенные пробелами — количество чисел в массиве и максимальное возможное значение числа в массиве (1 ≤ n, m ≤ 100 000).

Вторая строка содержит n целых чисел a1, a2, …, an — элементы массива (1 ≤ ai ≤ m).

Третья строка входного файла содержит q — количество запросов, которые необходимо обработать (1 ≤ q ≤ 100 000).

В процессе обработки запросов необходимо поддерживать число p, исходно оно равно 0. Каждый запрос задается парой чисел xi и yi, Петя генерирует их для получения данных очередного запроса следующим образом: https://contest.yandex.ru/testsys/tex/render/bF9pID0gXGxlZnQoKHhfaSArIHApIFxibW9kIG5ccmlnaHQpICsgMQ==.png, https://contest.yandex.ru/testsys/tex/render/a19pID0gXGxlZnQoKHlfaSArIHApIFxibW9kIG1ccmlnaHQpICsgMQ==.png(1 ≤ li,xi ≤ n, 1≤ ki, yi ≤ m). Пусть ответ на i-й запрос равен r. После выполнения этого запроса, следует присвоить p значение r.

**Решение.**

1. Решим задачу для L = 1:

Пусть мы имеем массив B, где B[i] равно 1, если число a[i] не встречалось ранее, и 0-лю, в противном случае. Тогда ответ на запрос будет такое **r**, что .

Что бы решить эту задачу, построим дерево отрезков на массиве B, в каждой вершине которого будет хранится сумма подотрезка, за который отвечает вершина.

Тогда отвечать на запрос можно будет за O(logn) путем спуска по дереву: число в вершине поддерева говорит о количестве разлиных чисел на этом подотрезке, поэтому мы точно будем знать в левом подотрезке, или в правом находится k-e различное число.

1. Теперь решим задачу для любого L.

Что изменится, когда мы из запросов с L = 1, перейдем в запросы с L = 2? Изменится лишь 2 элемента массива B. Элемент с индексом 1 станет равен нулю, а B[x], где a[x] = a[1] (x минимально) станет равным единице. В случае, когда не существует такого x>L, что a[x]=a[L], мы лишь изменяем элемент B[L]. Следовательно, изменится лишь O(logn) вершин дерева отрезков. Таким образом, мы можем перейти от запросов L, R к запросам L + 1, R, лишь добавив O(logn) вершин в персистентное дерево отрезков.

Таким образом перед тем, как обрабатывать запросы, нам понадобится выделить O(n \* logn) памяти.

1. Теперь, имея **n** указателей на корневые вершины персистентного дерева отрезков, мы можем за O(logn) отвечать на любой поступающий запрос.

Следовательно общее время работы O(m \* logn), где m – количество запросов.

А количество выделяемой памяти – O(n \* logn).