Задача A. Range Variation Query

Имя входного файла:	`rvq.in`
Имя выходного файла:	`rvq.out`
Ограничение по времени:	0.5 секунда
Ограничение по памяти:	64 мегабайта

В начальный момент времени последовательность a_n задана следующей формулой: $a_n = n^2 \bmod 12345 + n^3 \bmod 23456$.

Требуется много раз отвечать на запросы следующего вида:

- найти разность между максимальным и минимальным значениями среди элементов $a_i, a_{i+1}, \ldots, a_j$;
- присвоить элементу a_i значение j.

Формат входных данных

Первая строка входного файла содержит натуральное число k — количество запросов ($1 \leqslant k \leqslant 100\,000$). Следующие k строк содержат запросы, по одному на строке. Запрос номер i описывается двумя целыми числами x_i, y_i.

Если $x_i > 0$, то требуется найти разность между максимальным и минимальным значениями среди элементов $a_{x_i}, \ldots, a_{y_i}$. При этом $1 \leqslant x_i \leqslant y_i \leqslant 100\,000$.

Если $x_i < 0$, то требуется присвоить элементу $a_{|x_i|}$ значение y_i. В этом случае $-100\,000 \leqslant x_i \leqslant -1$ и $|y_i| \leqslant 100\,000$.

Формат выходных данных

Для каждого запроса первого типа в выходной файл требуется вывести одну строку, содержащую разность между максимальным и минимальным значениями на соответствующем отрезке.

Примеры

rvq.in	rvq.out
7 1 3 2 4 -2 -100 1 5 8 9 -3 -101 2 3	34 68 250 234 1

Задача B. Дерево отрезков с операцией на отрезке

Имя входного файла:	`segment-tree.in`
Имя выходного файла:	`segment-tree.out`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	64 мегабайта

Реализуйте эффективную структуру данных для хранения элементов и увеличения нескольких подряд идущих элементов на одно и то же число.

Формат входных данных

В первой строке вводится одно натуральное число N ($1 \leqslant N \leqslant 100\,000$) — количество чисел в массиве.

Во второй строке вводятся N чисел от 0 до 100 000 — элементы массива.

В третьей строке вводится одно натуральное число M ($1 \leqslant M \leqslant 30\,000$) — количество запросов.

Каждая из следующих M строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса (g — получить текущее значение элемента по его номеру, a — увеличить все элементы на отрезке).

Следом за g вводится одно число — номер элемента.

Следом за a вводится три числа — левый и правый концы отрезка и число add, на которое нужно увеличить все элементы данного отрезка массива ($1 \leqslant add \leqslant 100\,000$).

Формат выходных данных

Выведите в одну строку через пробел ответы на каждый запрос g.

Примеры

segment-tree.in	segment-tree.out
5 2 4 3 5 2 5 g 2 g 5 a 1 3 10 g 2 g 4	4 2 14 5

Задача C. Ближайшее большее число справа

Имя входного файла:	`nearandmore.in`
Имя выходного файла:	`nearandmore.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан массив a из n чисел. Нужно обрабатывать запросы:

0. `set(i, x)` – присвоить новое значение элементу массива `a[i] = x;`
1. `get(i, x)` – найти $\min k$: $k \geqslant i$ и $a_k \geqslant x$.

Формат входных данных

Первая строка входных данных содержит два числа: длину массива n и количество запросов m ($1 \leqslant n, m \leqslant 200\,000$).

Во второй строке записаны n целых чисел – элементы массива a ($0 \leqslant a_i \leqslant 200\,000$).

Следующие m строк содержат запросы, каждый запрос содержит три числа t, i, x. Первое число t равно 0 или 1 – тип запроса. $t = 0$ означает запрос типа `set`, $t = 1$ соответствует запросу типа `get`, $1 \leqslant i \leqslant n$, $0 \leqslant x \leqslant 200\,000$. Элементы массива нумеруются с единицы.

Формат выходных данных

На каждой запрос типа `get` на отдельной строке выведите соответствующее значение k. Если такого k не существует, выведите -1.

Примеры

nearandmore.in	nearandmore.out
4 5 1 2 3 4 1 1 1 1 1 3 1 1 5 0 2 3 1 1 3	1 3 -1 2

Задача D. Мега-инверсии

Имя входного файла:	`mega.in`
Имя выходного файла:	`mega.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Инверсией в перестановке $p_1, p_2, \ldots, p_N$ называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовём мега-инверсией в перестановке $p_1, p_2, \ldots, p_N$ тройку (i, j, k) такую, что $i < j < k$ и $p_i > p_j > p_k$. Напишите алгоритм для быстрого подсчёта количества мега-инверсий в перестановке.

Формат входных данных

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 100\,000$). Следующие N чисел описывают перестановку: $p_1, p_2, \ldots, p_N$ ($1 \leqslant p_i \leqslant N$), все p_i попарно различны. Числа разделяются переводами строк.

Формат выходных данных

Единственная строка выходного файла должна содержать одно число, равное количеству мега-инверсий в перестановке $p_1, p_2, \ldots, p_N$.

Примеры

mega.in	mega.out
4 4 3 2 1	4