Задача A. Дерево отрезков

Имя входного файла:	`build-tree.in`
Имя выходного файла:	`build-tree.out`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Дан массив a длины n (нумерация массива a начинается с нуля). Требуется построить дерево отрезков (с операцией минимума) на этом массиве.

Дерево отрезков на массиве из n элементов — это бинарное подвешенное дерево, отвечающее следующим свойствам:

- Каждой вершине соответствует некий полуинтервал $[L_i; R_i)$.
- Вершина является листом тогда и только тогда, когда $R_i - L_i = 1$
- Если вершина не является листом, то ее левому и правому сыновьям соответствуют полуинтервалы $[L_i; \lfloor \frac{L_i+R_i}{2} \rfloor)$ и $[\lfloor \frac{L_i+R_i}{2} \rfloor; R_i)$ соответственно.
- Корню дерева соответствует полуинтервал $[0; N)$.

Значением в вершине дерева, которой соответствует полуинтервал $[L_i; R_i)$, является минимальное среди чисел $a_{L_i}, \ldots, a_{R_i-1}$.

Вершинам дерева присваиваются номера по следующему принципу:

- Корень имеет номер 1.
- Левый и правый сын вершины с номером i имеют номера $2i$ и $2i + 1$ соответственно.

Формат входных данных

В первой строке вводится натуральное число $n \leqslant 30000$ — длина массива. Во второй строке вводится n чисел, не превышающих по модулю 10^5.

Формат выходных данных

В первой строке выведите число m - количество вершин в дереве.

В следующих m строках выведите описание вершин дерева. В $i + 1$-ой строке выведите шесть чисел, описывающих i-ую вершину дерева, через пробелы: номер i-ой вершины в дереве, границы полуинтервала L_i и R_i, значение в i вершине, а также номера вершин — левого и правого сына i-ой вершины (если у вершины нет сыновей, выведите два числа -1).

Левый и правый сыновья (если они есть) вершины с номером i, должны иметь номера $2i$ и $2i+1$ соответственно. Корень должен иметь номер 1.

Требуется выводить вершины в порядке возрастания номеров.

Примеры

build-tree.in	build-tree.out
6 -1 3 4 3 -1 2	11 1 0 6 -1 2 3 2 0 3 -1 4 5 3 3 6 -1 6 7 4 0 1 -1 -1 -1 5 1 3 3 10 11 6 3 4 3 -1 -1 7 4 6 -1 14 15 10 1 2 3 -1 -1 11 2 3 4 -1 -1 14 4 5 -1 -1 -1 15 5 6 2 -1 -1
3 6 1 5	5 1 0 3 1 2 3 2 0 1 6 -1 -1 3 1 3 1 6 7 6 1 2 1 -1 -1 7 2 3 5 -1 -1

Задача B. Поиск максимума

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

Реализуйте структуру данных для эффективного вычисления номера максимального из нескольких подряд идущих элементов массива.

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 1 до 100 000 — элементы массива.

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

В следующих K строках вводится по два числа — номера левого и правого элементов отрезка массива (считается, что элементы массива нумеруются с единицы).

Формат выходных данных

Для каждого запроса выведите индекс максимального элемента на указанном отрезке массива. Если максимальных элементов несколько, выведите любой их них.

Числа выводите в одну строку через пробел.

Примеры

index-max.in	index-max.out
5 2 2 2 1 5 2 2 3 2 5	3 5

Задача C. 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 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

Задача E. 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