

## Задача А. Поиск максимума

Имя входного файла: `index-max.in`  
Имя выходного файла: `index-max.out`  
Ограничение по времени: 3 секунды  
Ограничение по памяти: 256 мегабайт

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

### Формат входных данных

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

Во второй строке вводятся  $N$  чисел от 1 до 100 000 — элементы массива.

В третьей строке вводится одно натуральное число  $K$  ( $1 \leq K \leq 3\,000\,000$ ) — 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.

### Пример

<code>index-max.in</code>	<code>index-max.out</code>
5	3
2 2 2 1 5	5
2	
2 3	
2 5	

## Задача В. НОД на подотрезках с изменением элемента

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Реализуйте эффективную структуру данных, позволяющую изменять элементы массивы и вычислять НОД нескольких подряд идущих элементов.

### Формат входных данных

В первой строке вводится одно натуральное число  $N$  ( $1 \leq N \leq 100000$ ) – количество чисел в массиве.

Во второй строке вводятся  $N$  чисел от 0 до 100000 – элементы массива.

В третьей строке вводится одно натуральное число  $M$  ( $1 \leq M \leq 30000$ ) – количество запросов.

Каждая из следующих  $M$  строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса ( $s$  – вычислить НОД,  $u$  – обновить значение элемента).

Следом за  $s$  вводятся два числа – номера левой и правой границы отрезка.

Следом за  $u$  вводятся два числа – номер элемента и его новое значение.

### Формат выходных данных

Для каждого запроса  $s$  выведите результат. Все числа выводите в одну строку через пробел.

### Пример

стандартный ввод	стандартный вывод
5	2 4 4 32
2 8 4 16 12	
5	
s 1 5	
s 4 5	
u 3 32	
s 2 5	
s 3 3	

## Задача С. Катый ноль

Имя входного файла: `kthzero.in`  
Имя выходного файла: `kthzero.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Реализуйте эффективную структуру данных, позволяющую изменять элементы массива и вычислять индекс  $k$ -го слева нуля на данном отрезке в массиве.

### Формат входных данных

В первой строке вводится одно натуральное число  $N$  ( $1 \leq N \leq 200\,000$ ) — количество чисел в массиве. Во второй строке вводятся  $N$  чисел от 0 до 100 000 — элементы массива. В третьей строке вводится одно натуральное число  $M$  ( $1 \leq M \leq 200\,000$ ) — количество запросов. Каждая из следующих  $M$  строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса (`s` — вычислить индекс  $k$ -го нуля, `u` — обновить значение элемента). Следом за `s` вводится три числа — левый и правый концы отрезка и число  $k$  ( $1 \leq k \leq N$ ). Следом за `u` вводятся два числа — номер элемента и его новое значение.

### Формат выходных данных

Для каждого запроса  $s$  выведите результат. Все числа выводите в одну строку через пробел. Если нужного числа нулей на запрашиваемом отрезке нет, выводите  $-1$  для данного запроса.

### Пример

<code>kthzero.in</code>	<code>kthzero.out</code>
5	4
0 0 3 0 2	
3	
u 1 5	
u 1 0	
s 1 5 3	

### Замечание

TL для Python 8 секунд

## Задача D. Противник слаб

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 5 секунд  
Ограничение по памяти: 256 мегабайт

Римляне снова наступают. На этот раз их гораздо больше чем персов, но Шапур готов победить их. Он говорит: «Лев никогда не испугается сотни овец».

Не смотря на это, Шапур должен найти слабость римской армии чтобы победить ее. Как вы помните, Шапур — математик, поэтому он определяет насколько слаба армии как число — степень слабости.

Шапур считает, что степень слабости армии равна количеству таких троек  $i, j, k$ , что  $i < j < k$  и  $a_i > a_j > a_k$ , где  $a_x$  — сила человека, стоящего в строю на месте с номером  $x$ .

Помогите Шапуру узнать, насколько слаба армия римлян.

### Формат входных данных

В первой строке записано одно целое число  $n$  ( $3 \leq n \leq 10^6$ ) — количество солдат в римской армии. Следующая строка содержит  $n$  целых чисел  $a_i$  ( $1 \leq i \leq n, 1 \leq a_i \leq 10^9$ ) — силы людей в римской армии.

### Формат выходных данных

Выведите одно число — степень слабости римской армии.

### Примеры

стандартный ввод	стандартный вывод
3 3 2 1	1
3 2 3 1	0
4 10 8 3 1	4
4 1 5 4 3	1

## Задача E. 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  $\min k: k \geq i$  и  $a_k \geq x$ .

### Формат входных данных

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

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

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`,  $1 \leq i \leq n$ ,  $0 \leq x \leq 200\,000$ . Элементы массива нумеруются с единицы.

### Формат выходных данных

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

### Пример

<code>nearandmore.in</code>	<code>nearandmore.out</code>
4 5	1
1 2 3 4	3
1 1 1	-1
1 1 3	2
1 1 5	
0 2 3	
1 1 3	

## Задача F. Число возрастающих подпоследовательностей

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 256 мегабайт

Задана последовательность из  $n$  чисел  $a_1, a_2, \dots, a_n$ . Подпоследовательностью длины  $k$  этой последовательности называется набор индексов  $i_1, i_2, \dots, i_k$ , удовлетворяющий неравенствам  $1 \leq i_1 < i_2 < \dots < i_k \leq n$ . Подпоследовательность называется возрастающей, если выполняются неравенства  $a_{i_1} < a_{i_2} < \dots < a_{i_k}$ .

Необходимо найти число возрастающих подпоследовательностей наибольшей длины заданной последовательности  $a_1, \dots, a_n$ . Так как это число может быть достаточно большим, необходимо найти остаток от его деления на  $10^9 + 7$ .

### Формат входных данных

Первая строка входного файла содержит целое число  $n$  ( $1 \leq n \leq 10^5$ ). Вторая строка входного файла содержит  $n$  целых чисел:  $a_1, a_2, \dots, a_n$ . Все  $a_i$  не превосходят  $10^9$  по абсолютной величине.

### Формат выходных данных

В выходной файл выведите ответ на задачу.

### Примеры

стандартный ввод	стандартный вывод
5 1 2 3 4 5	1
6 1 1 2 2 3 3	8