Задача A. Знакочередование

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

Реализуйте структуру данных из n элементов $a_1, a_2, \ldots, a_n$, поддерживающую следующие операции:

- присвоить элементу a_i значение j;
- найти знакочередующуюся сумму на отрезке от l до r включительно, т. е. $(a_l - a_{l+1} + a_{l+2} - \ldots a_r)$.

Формат входных данных

В первой строке входного файла содержится натуральное число n ($1 \leqslant n \leqslant 10^5$) — длина массива. Во второй строке записаны начальные значения элементов — неотрицательные целые числа, не превосходящие 10^4.

В третьей строке находится натуральное число m ($1 \leqslant m \leqslant 10^5$) — количество операций. В последующих m строках записаны операции:

- операция первого типа задаётся тремя числами `0` i j ($1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant 10^4$).
- операция второго типа задаётся тремя числами `1` l r ($1 \leqslant l \leqslant r \leqslant n$).

Формат выходных данных

Для каждой операции второго типа выведите на отдельной строке соответствующую знакочередующуюся сумму.

Пример

signchange.in	signchange.out
3 1 2 3 5 1 1 2 1 1 3 1 2 3 0 2 1 1 1 3	-1 2 -1 3

Задача B. 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 ($1 \leqslant N \leqslant 200\,000$) — 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 ($1 \leqslant M \leqslant 200\,000$) — 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 ($1 \leqslant k \leqslant N$). 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.

Примеры

kthzero.in	kthzero.out
5 0 0 3 0 2 3 u 1 5 u 1 0 s 1 5 3	4

Замечание

TL для Python 8 секунд

Задача C. Прибавление и максимум

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

Реализуйте эффективную структуру данных для хранения массива и выполнения следующих операций: увеличение всех элементов данного интервала на одно и то же число; поиск максимума на интервале.

Формат входных данных

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

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

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

Каждая из следующих M строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса (m – найти максимум на отрезке (оба конца включительно), a – увеличить все элементы на отрезке, то есть все элементы с l-го по r-й).

Следом за m вводятся два числа – левая и правая граница отрезка.

Следом за a вводятся три числа – левый и правый концы отрезка и число add, на которое нужно увеличить все элементы данного отрезка массива $(0 \leqslant add \leqslant 100000)$.

Формат выходных данных

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

Примеры

addandmax.in	addandmax.out
5 2 4 3 1 5 5 m 1 3 a 2 4 100 m 1 3 a 5 5 10 m 1 5	4 104 104

Задача D. 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

Замечание

TL для Python 10 секунд

Задача E. Число возрастающих подпоследовательностей

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

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

Необходимо найти число возрастающих подпоследовательностей наибольшей длины заданной последовательности a_1, ..., 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 ($1 \leqslant n \leqslant 100000$). Вторая строка входного файла содержит n целых чисел: a_1, a_2, ..., a_n. Все a_i не превосходят 10^9 по абсолютной величине.

Формат выходных данных

В выходной файл выведите ответ на задачу.

Примеры

subseq.in	subseq.out
5 1 2 3 4 5	1
6 1 1 2 2 3 3	8

Задача F. Диаграмма Юнга и Серёжа

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

Посмотрим на диаграмму Юнга, состоящую из n столбцов. Немногие знают, что можно делать не только положительные, но и отрицательные слагаемые. Тогда столбцы в диаграмме просто растут не вверх, а вниз.

Серёжа изучает диаграммы Юнга. Сегодня он рассматривает некоторые подотрезки столбцов и ищет на этих отрезках подотрезки с максимальной площадью.

Помогите Серёже, а то он хочет спать и ему не думается.

Формат входных данных

Входные данные содержат один или несколько тестовых примеров. Описание каждого из них начинается с двух чисел n и m — число стобцов диаграммы и количество интересующих Серёжу подотрезков.

В следующей строке содержится n чисел — высоты столбцов. Каждое из этих чисел по абсолютной величине не превосходит 10^4.

Далее следуют описания подотрезков, каждое описание состоит из двух чисел l и r, обозначающих левый и правый конец подотрезка ($1 \leqslant l \leqslant r \leqslant n$).

Суммарная длина всех диаграмм, а также суммарное число подотрезков не превосходит 10^5.

Формат выходных данных

Для каждого из тестовых примеров выведите m чисел: искомую максимальную площадь для каждого из подотрезков.

Примеры

maxsum.in	maxsum.out
10 3 -100 1 2 3 4 -10 50 -100 -1 2 1 10 1 5 9 9 5 2 -1 2 -1 2 -1 1 5 2 4	50 10 -1 3 3

Задача G. Марио и трубы

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

Марио собирается проходить уровень, состоящий из N последовательно расположенных труб, высота i-й трубы — a_i. Он еще не знает, где он будет располагаться изначально, и куда ему надо добраться, поэтому хочет рассмотреть несколько вариантов.

Находясь на трубе, Марио может переместиться только на соседние трубы слева и справа (если они существуют). Спускаться он может с любой высоты, также он может перемещаться между одинаковыми трубами. Подниматься Марио может только на трубу, высота которой больше высоты текущей на 1. Более формально, Марио может переместиться с трубы i на трубу j, если $|i - j| = 1$ и $a_j - a_i \leqslant 1$.

Однако злой динозавр Боузер хочет помешать Марио пройти уровень, для чего иногда увеличивает высоту нескольких подряд идущих труб на одно число k. Теперь Марио не может понять, удастся ли ему пройти уровень и поэтому просит вас обрабатывать два типа запросов — Боузер изменяет высоту некоторых труб, и Марио пытается пройти от одной трубы до другой.

Формат входных данных

В первой строке заданы два целых числа N и M — число труб и число запросов соответственно ($2 \leqslant N \leqslant 3 \cdot 10^5, 1 \leqslant M \leqslant 10^6$).

Следующая строка содержит N целых чисел a_i — высоты труб на уровне ($1 \leqslant a_i \leqslant 10^9$).

Далее идут M строк, содержащие описание запросов. Каждая строка имеет вид:

- `1 x y` — может ли Марио пройти от трубы с номером x до трубы с номером y ($1 \leqslant x, y \leqslant N$). Гарантируется, что номера x и y не совпадают.
- `2 l r d` — Боузер увеличивает высоты труб с l-й до r-й на величину d ($1 \leqslant l \leqslant r \leqslant N$, $-10^9 \leqslant d \leqslant 10^9$).

Формат выходных данных

Для каждого запроса запроса первого типа нужно на отдельной строке вывести «`YES`», если Марио может дойти от одной трубы до другой и «`NO`» в противном случае (без кавычек).

Примеры

стандартный ввод	стандартный вывод
5 7 1 2 3 4 5 1 5 1 2 2 4 3 1 5 4 1 1 3 2 2 3 3 1 2 4 1 1 3	YES NO NO YES NO

Задача H. Блин, че-то я тупанул, конечно

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

> Письмо на Балабановскую спичечную фабрику: «Я 11 лет считаю спички у вас в коробках — их то 59, то 60, а иногда и 58. Вы там сумасшедшие что ли все???»

Гриша очень педантичный препод, который любит спорт. Поэтому он записал количество голов, которые он, *по его словам*, забил за каждый футбольный матч (может даже отрицательное, если он пропускал голы).

Еще Гриша считает, что главное в спорте — прогресс. Но прогресс должен быть плавным. Поэтому он хочет для разных последовательностей дней узнавать, насколько же прогресс был плавным. Мера плавности — длина максимального подотрезка вида $(x, x+1, ..., x+l)$, где l — длина такого отрезка.

Сокомандник Гриши, Женя, завидует результативности Гриши. Поэтому иногда он убавляет количество голов в блокнотике Гриши за некоторые матчи (иногда даже добавляет, чтобы ему не так обидно было).

А еще Гриша старший преподаватель, поэтому у него нет времени че-то там считать, поэтому вам дали данную задачу.

Формат входных данных

В первой строке находится одно целое число n — количество матчей по футболу за смену, в которых Гриша принял участие. $1 \leqslant n \leqslant 10^5$ Во второй строке находятся n чисел q_i ($|q_i| \leqslant 10^9$) — количество голов, который Гриша забил в каждом матче. В третьей строке находится одно целое число m ($0 \leqslant m \leqslant 100\,000$) — количество действий с блокнотиком Гриши. В следующих m строках содержится описание эксперимента.

- `+` l_i r_i d_i — Женя добавил d_i голов в матчах с l_i-го по r_i-й. ($1 \leqslant l_i \leqslant r_i \leqslant n$, $|d_i| \leqslant 10^9$)
- `?` l_i r_i — Гриша попросил Вас узнать насколько у него был плавный прогресс в матчах с l_i-го по r_i-й. ($1 \leqslant l_i \leqslant r_i \leqslant n$)

Формат выходных данных

Для каждого действия второго типа выведите в новой строке одно число — меру плавности.

Примеры

atoms.in	atoms.out
6 2 3 4 3 4 4 5 ? 1 6 + 6 6 1 ? 2 6 + 4 6 2 ? 1 5	3 3 5

Задача I. Присваивание, прибавление и сумма

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Есть массив из n элементов, изначально заполненный нулями. Вам нужно написать структуру данных, которая обрабатывает три вида запросов:

- присвоить всем элементам на отрезке от l до $r-1$ значение v,
- прибавить ко всем элементам на отрезке от l до $r-1$ число v,
- узнать сумму на отрезке от l до $r-1$.

Формат входных данных

Первая строка содержит два числа n и m ($1 \leqslant n, m \leqslant 100000$) — размер массива и число операций. Далее следует описание операций. Описание каждой операции имеет следущий вид:

- 1 l r v — присвоить всем элементам на отрезке от l до $r-1$ значение v ($0 \leqslant l < r \leqslant n$, $0 \leqslant v \leqslant 10^5$).
- 2 l r v — прибавить ко всем элементам на отрезке от l до $r-1$ число v ($0 \leqslant l < r \leqslant n$, $0 \leqslant v \leqslant 10^5$).
- 3 l r — узнать сумму на отрезке от l до $r-1$ ($0 \leqslant l < r \leqslant n$).

Формат выходных данных

Для каждой операции третьего типа выведите соответствующее значение.

Примеры

стандартный ввод	стандартный вывод
5 7 1 0 3 3 2 2 4 2 3 1 3 2 1 5 1 1 0 2 2 3 0 3 3 3 5	8 10 4

Задача J. Перестановки

Имя входного файла:	`permutation.in`
Имя выходного файла:	`permutation.out`
Ограничение по времени:	1.5 секунды
Ограничение по памяти:	256 мегабайт

Вася выписал на доске в каком-то порядке все числа от 1 по N, каждое число ровно по одному разу. Количество чисел оказалось довольно большим, поэтому Вася не может окинуть взглядом все числа. Однако ему надо всё-таки представлять эту последовательность, поэтому он написал программу, которая отвечает на вопрос — сколько среди чисел, стоящих на позициях с x по y, по величине лежат в интервале от k до l. Сделайте то же самое.

Формат входных данных

В первой строке лежит два натуральных числа — $1 \leqslant N \leqslant 100\,000$ — количество чисел, которые выписал Вася и $1 \leqslant M \leqslant 100\,000$ — количество вопросов, которые Вася хочет задать программе. Во второй строке дано N чисел — последовательность чисел, выписанных Васей. Далее в M строках находятся описания вопросов. Каждая строка содержит четыре целых числа $1 \leqslant x \leqslant y \leqslant N$ и $1 \leqslant k \leqslant l \leqslant N$.

Формат выходных данных

Выведите M строк, каждая должна содержать единственное число — ответ на Васин вопрос.

Примеры

permutation.in	permutation.out
4 2 1 2 3 4 1 2 2 3 1 3 1 3	1 3