Задача A. Сумма на отрезке

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

Дан массив из N элементов, нужно научиться находить сумму чисел на отрезке.

Формат входных данных

Первая строка входного файла содержит два целых числа N и K — количество чисел в массиве и количество запросов ($1 \leqslant N \leqslant 100\,000$, $0 \leqslant K \leqslant 100\,000$). Следующие K строк содержат следующие запросы:

1. `A i x` — присвоить i-му элементу массива значение x ($1 \leqslant i \leqslant n$, $0 \leqslant x \leqslant 10^9$);
2. `Q l r` — найти сумму чисел в массиве на позициях от l до r ($1 \leqslant l \leqslant r \leqslant n$).

Изначально в массиве живут нули.

Формат выходных данных

На каждый запрос вида `Q l r` нужно вывести единственное число — сумму на отрезке.

Примеры

sum.in	sum.out
5 9 A 2 2 A 3 1 A 4 2 Q 1 1 Q 2 2 Q 3 3 Q 4 4 Q 5 5 Q 1 5	0 2 1 2 0 5

Замечание

TL для Python 4 секунды

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:	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 ($1 \leqslant N \leqslant 100\,000$) — 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

Замечание

TL для Python 2 секунды

Задача C. Катый ноль

Имя входного файла:	`kthzero.in`
Имя выходного файла:	`kthzero.out`
Ограничение по времени:	1 секунда
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

Задача D. 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

Замечание

TL для Python 4 секунды

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:	256 мегабайт

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 ($0 \leqslant add \leqslant 100\,000$).

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 F. Дерево отрезков с операцией НОД

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

Реализуйте структуру данных для эффективного вычисления НОД 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 НОД.

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 НОД всех чисел соответствующего участка массива. Числа выводите в одну строку через пробел.

Примеры

segment-gcd.in	segment-gcd.out
5 2 2 2 1 5 2 2 3 2 5	2 1

Замечание

TL для Python 3 секунды

Задача G. 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 H. Четвертый этаж

Имя входного файла:	`floor4.in`
Имя выходного файла:	`floor4.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Знаете ли вы, почему четвертый этаж заперт и там не останавливается лифт? Потому что на самом деле четвертый, запертый, этаж, где не останавливается лифт, содержит бесконечное количество комнат, пронумерованных натуральными числами. На этот этаж регулярно приезжают дети, каждый из которых заранее выбрал, в какую комнату он хочет заселиться. Если выбранная комната оказывается свободна, то ребенок занимает ее, в противном случае он занимает первую свободную комнату с бóльшим номером.

Кроме того, некоторые дети уезжают в середине смены. Сразу после отъезда ребенка его комната становится доступна для заселения следующего.

Промоделируйте работу преподавателей, ответственных за четвертый этаж и научитесь быстро сообщать приезжающим детям, какую комнату им следует занимать.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество прибытий и отъездов, происходящих в течение смены ($1 \leqslant n \leqslant 100\,000$).

Следующие n строк содержат информацию об ЛКШатах. Число $a > 0$ обозначает, что приехал школьник, желающий занять комнату номер a ($1 \leqslant a \leqslant 100\,000$). Число $a < 0$ обозначает, что из комнаты номер $|a|$ уехал школьник. (Гарантируется, что эта комната не была пуста).

Формат выходных данных

Для каждого приезжающего школьника выведите одно натуральное число — номер комнаты, в которую он поселится.

Примеры

floor4.in	floor4.out
6 5 5 5 -6 5 5	5 6 7 6 8

Задача I. Количество инверсий

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайта

Напишите программу, которая для заданного массива $A = \langle a_1, a_2, \ldots, a_n \rangle$ находит количество пар (i, j) таких, что $i < j$ и $a_i > a_j$. Обратите внимание на то, что ответ может не влезать в int.

Формат входных данных

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 100\,000$) — количество элементов массива. Вторая строка содержит n попарно различных элементов массива A — целых неотрицательных чисел, не превосходящих 10^9.

Формат выходных данных

В выходной файл выведите одно число — количество инверсий в массиве.

Примеры

stdin	stdout
5 6 11 18 28 31	0
8 999994 999989 999982 999972 999969 999961 999954 999950	28