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. 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 C. 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 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. Парковка

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

На кольцевой парковке есть n мест пронумерованых от 1 до n. Есть два вида событий прибытие машину на парковку и отъезд машины с парковки. Если машина приезжает на парковку, а её место занято, то она едет далее по кругу и встаёт на первое свободное место.

Формат входных данных

В первой строке входного файла находится два числа n и m — размер парковки и количество запросов($1 \leqslant n, m \leqslant 100000$). В следующих m строках находятся события. Каждая из этих строк имеет следующий вид:

- enter x — приехала машина, которая хочет встать на место x. Для каждой такой команды выведите какое место займёт эта машина.
- exit x — уехала машина занимавшая место x. Гарантируется, что на этом месте была машина.

Формат выходных данных

Выведите последовательно результаты выполнения всех операций `enter`.

Примеры

parking.in	parking.out
3 5 enter 1 enter 1 exit 1 enter 2 enter 2	1 2 3 1

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. 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 8 секунд

Задача H. 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 I. 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 J. 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

Задача K. Задача Иосифа

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

N мальчиков стоят по кругу. Они начинают считать себя по часовой стрелке, счет ведется с единицы. Как только количество посчитанных достигает p, последний посчитанный (p-й) мальчик покидает круг, а процесс счета начинается со следующего за ним мальчика и вновь ведется с единицы.

Последний оставшийся в кругу выигрывает.

Можете ли вы посчитать, номер выигрывшего мальчика в исходном кругу? (мальчики нумеруются числами от 1 до N по часовой стрелке, начиная с того самого мальчика, с которого начинался счет).

Формат входных данных

Во входном файле два целых числа — N и P ($1 \leqslant N, P \leqslant 10^5$).

Формат выходных данных

Выведите номер выигравшего мальчика.

Примеры

joseph.in	joseph.out
3 4	2

Задача L. Совет стаи

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

— Смотрите хорошенько, смотрите хорошенько, о волки!

Акела

Закон Джунглей очень ясно говорит, что каждый вновь женившийся волк может отделиться от своей стаи; однако едва его волчата вырастут настолько, чтобы хорошо держаться на ногах, он обязан привести их и представить Совету стаи, который обыкновенно собирается в полнолуние; это делается для того, чтобы остальные волки узнали их. После такого осмотра волчата имеют право бегать куда им угодно и пока они не поймают первого оленя.

На совете стаи волки представляют своих детенышей другим волкам, чтобы они могли принять решение об их вступлении в стаю. Совет проходит в полночь, и волчата располагаются вдоль полоски лунного света, чтобы их можно было лучше рассмотреть.

Процесс выбора вступающих в стаю волчат не совсем определен, но зависит от решения каждого из волков. Волк, оценивающий потомство, как правило, не изучает всех прибывших детенышей, но ограничивается лишь некоторыми, стоящими подряд. Также, иногда та или другая волчица выталкивает носом своего детёныша в полосу лунного света, желая, чтобы его непременно заметили. В таком случае какой-то другой волчонок уходит в тень, а новый встает на его место.

Показателем будущих способностей маленького волка является его рост, главным образом зависящий от длины его ног. Взрослые волки внимательно рассматривают волчат и выбирают из них несколько рядом стоящих, по их мнению, наиболее приспособленных для охоты. Приспособленность детеныша волки определяют, как число $h - e$, где h — рост волчонка, а e — некоторая эталонная величина, передающаяся волками из поколения в поколение. Таким образом, среди кандидатов, рассматриваемых им, каждый волк выбирает подряд стоящих, так, чтобы сумма их приспособленностей была максимальна. При этом, каждый волк понимает важность преемственности и выбирает хотя бы одного волчонка, даже если все кандидаты неважные.

Предскажите выбор каждого из волков на совете стаи.

Формат входных данных

В первой строке входного файла записано одно число N ($1 \leqslant N \leqslant 100\,000$) — количество волчат-кандидатов, выстроившихся в ряд в начале процесса отбора. Во второй строке записаны N целых чисел x_i ($|x_i| \leqslant 10^9$) — приспособленности детенышей.

В третьей строке записано одно число M ($0 \leqslant M \leqslant 100\,000$) — количество событий, произошеших на совете. В следующих M строках описываются события:

- `1` a_i x_i ($1 \leqslant a_i \leqslant N, |x_i| \leqslant 10^9$) означает, что мать-волчица поставила своего детеныша с приспособленностью x_i на позицию a_i;
- `2` l_i r_i ($1 \leqslant l_i \leqslant r_i \leqslant N$) означает, что волк производит выбор, просматривая детенышей на позициях $[l_i, r_i]$.

Формат выходных данных

Для каждого события типа `2` выведите ответ — максимальную возможную сумму приспособленностей расположенных подряд волчат на отрезке $[l_i, r_i]$.

Примеры

council.in	council.out
3 1 2 3 5 2 1 3 1 2 -1 2 1 3 2 2 3 2 1 2	6 3 3 1