Задача A. Задание котенку

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

Маленький котенок заблудился в лабиринте. Дни и ночи он плутал, совсем обессилев, пока не наткнулся на Бабу-Ягу. Баба-Яга сначала хотела съесть котенка, но потом передумала, увидев, что от него остались кожа да кости. Но котенок все мяукал и мяукал, а топить его противно — воды Баба-Яга боится! Поэтому-то Баба-Яга и решила спровадить бедного котенка, дав ему подробную карту лабиринта, — все равно эта карта только место занимает!

Но не в правилах злодеев делать что-то особенно доброе просто так. Поэтому Баба-Яга дала задание котенку — вести учет вызванных чертей. Баба-Яга хочет построить чертей в шеренгу. Шеренга будет расположена на прямой, причем черти вызываются в целых точках этой прямой. Если в какой-то точке прямой черт уже призван, то повторный призыв черта не изменяет состояния этой точки, иначе эта клетка заполняется чертом.

Бабе-Яге очень приятно восхищаться черным делом рук своих, и она хочет периодически задавать вопросы: а сколько, собственно, чертей находится на заданном отрезке? И этой грязной работой должен заниматься котенок, пока Бабе-Яге не надоест. Но это его единственный шанс выбраться из лабиринта!

Помогите котенку написать программу, которая по заданной последовательности запросов Бабы-Яги и вызовов чертей ответит на каждый запрос.

Формат входного файла

Входной файл состоит из запросов Бабы-Яги и вызовов чертей. События происходят в том порядке, в котором они описаны во входном файле. Если соответствующая строка содержит одно число A_i, то оно соответствует вызову черта в позицию A_i. Иначе в строке содержатся два числа A_i и B_i, которые означают, что Баба-Яга пожелала узнать, сколько же чертей заключено между точками с координатами между A_i и B_i.

Запросов во входном файле не более 200 000, все числа A_i и B_i целые и по модулю не превосходят 1 000 000. Последняя строка входного файла обязательно завершается переводом строки.

Формат выходного файла

Для каждого из запросов Бабы-Яги в отдельной строке необходимо вывести число — ответ на запрос.

Примеры

kitten.in	kitten.out
2 3 5 2 6 5 6 7 8 9 3 9 5 3 9	0 1 1 0 2 2

Задача B. Хипуй!

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

В этой задаче вам необходимо организовать структуру данных Heap для хранения целых чисел, над которой определены следующие операции:

- `Insert(X)` — добавить в Heap число X;
- `Extract` — достать из Heap наибольшее число (удалив его при этом).

Формат входного файла

Во входном файле записано количество команд N ($1 \leqslant N \leqslant 100\,000$), потом последовательность из N команд, каждая в своей строке.

Каждая команда имеет такой формат: „`0 <число>`“ или „`1`“, что означает соответственно операции `Insert(<число>)` и `Extract`. Добавляемые числа находятся в интервале от 1 до 10^7 включительно.

Гарантируется, что при выполенении команды `Extract` в структуре находится по крайней мере один элемент.

Формат выходного файла

В выходной файл для каждой команды извлечения необходимо вывести число, полученное при выполнении команды `Extract`.

Примеры

heap.in	heap.out
7 0 100 0 10 1 0 5 0 30 0 50 1	100 50

Задача C. Range Variation Query

Имя входного файла:	`rvq.in`
Имя выходного файла:	`rvq.out`
Ограничение по времени:	2 секунды
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 входного файла

Первая строка входного файла содержит натуральное число k — количество запросов ($k \leqslant 100\,000$). 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 выходного файла

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. 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 входного файла

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

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

Формат выходного файла

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 E. Хорошие дни

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

Билл разрабатывает новую математическую теорию, описывающую человеческие эмоции. Его последние исследования посвящены изучению того, насколько хорошие и плохие дни влияют на воспоминания людей о различных периодах жизни.

Недавно Билл придумал методику, которая описывает, насколько хорошим или плохим был день человеческой жизни с помощью сопоставления дню некоторого неотрицательного целого числа. Билл называет это число *эмоциональной значимостью* этого дня. Чем больше это число, тем лучше этот день. Билл полагает, что значимость некоторого периода человеческой жизни равна сумме эмоциональных значимостей каждого из дней периода, помноженной на минимум эмоциональных значимостей дней этого периода. Эта методика отражает то, что период, который в среднем может быть весьма неплох, бывает испорчен одним плохим днем.

Теперь Билл хочет проанализировать свою собственную жизнь и найти в ней период максимальной значимости. Помогите ему это сделать.

Формат входного файла

Первая строка входного файла содержит число n — количество дней в жизни Билла, которые он хочет исследовать ($1 \leqslant n \leqslant 100\,000$). Оставшаяся часть файла содержит n целых чисел $a_1, a_2, \ldots, a_n$, все в пределах от 0 до 10^6 — эмоциональные значимости дней. Числа во входном файле разделяются пробелами и переводами строки.

Формат выходного файла

В первой строке выходного файла выведите максимальную значимость периода жизни Билла. Во второй строке выведите два числа l и r, означающие, что значимость периода с l-го по r-й день (включительно) в жизни Билла была максимально возможной.

Примеры

feelgood.in	feelgood.out
6 3 1 6 4 5 2	60 3 5

Задача F. Разреженные таблицы

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

Дан массив из n чисел. Требуется написать программу, которая будет отвечать на запросы следующего вида: найти минимум на отрезке между u и v включительно.

Формат входного файла

В первой строке входного файла даны три натуральных числа n, m ($1 \leqslant n \leqslant 10^5$, $1 \leqslant m \leqslant 10^7$) и a_1 ($0 \leqslant a_1 < 16\,714\,589$) — количество элементов в массиве, количество запросов и первый элемент массива соответственно. Вторая строка содержит два натуральных числа u_1 и v_1 ($1 \leqslant u_1, v_1 \leqslant n$) — первый запрос.

Элементы $a_2, a_3, \ldots, a_n$ задаются следующей формулой:

$$a_{i+1} = (23 \cdot a_i + 21563) \bmod 16714589.$$

Например, при $n = 10$, $a_1 = 12345$ получается следующий массив: a = (12345, 305498, 7048017, 11694653, 1565158, 2591019, 9471233, 570265, 13137658, 1325095).

Запросы генерируются следующим образом:

$$u_{i+1} = \big((17 \cdot u_i + 751 + ans_i + 2i) \bmod n\big) + 1,$$
$$v_{i+1} = \big((13 \cdot v_i + 593 + ans_i + 5i) \bmod n\big) + 1,$$

где ans_i — ответ на запрос номер i.

Формат выходного файла

В выходной файл выведите u_m, v_m и ans_m (последний запрос и ответ на него).

Примеры

sparse.in	sparse.out
10 8 12345 3 9	5 3 1565158