Problem A. Снеговики

Time limit: 2 секунды
Memory limit: 64 мегабайта

Зима. 2012 год. На фоне грядущего Апокалипсиса и конца света незамеченной прошла новость об очередном прорыве в областях клонирования и снеговиков: клонирования снеговиков. Вы конечно знаете, но мы вам напомним, что снеговик состоит из нуля или более вертикально поставленных друг на друга шаров, а клонирование — это процесс создания идентичной копии (клона).

В местечке Местячково учитель Андрей Сергеевич Учитель купил через интернет-магазин «Интернет-магазин аппаратов клонирования» аппарат для клонирования снеговиков. Теперь дети могут играть и даже играют во дворе в следующую игру. Время от времени один из них выбирает понравившегося снеговика, клонирует его и:

- либо добавляет ему сверху один шар;
- либо удаляет из него верхний шар (если снеговик не пустой).

Учитель Андрей Сергеевич Учитель записал последовательность действий и теперь хочет узнать суммарную массу всех построенных снеговиков.

Input

Первая строка содержит количество действий n ($1 \le n \le 200\,000$). В строке номер $i+1$ содержится описание действия i:

- `t m` — клонировать снеговика номер t ($0 \le t < i$) и добавить сверху шар массой m ($0 < m \le 1000$);
- `t ∅` — клонировать снеговика номер t ($0 \le t < i$) и удалить верхний шар. Гарантируется, что снеговик t не пустой.

В результате действия i, описанного в строке $i+1$ создается снеговик номер i. Изначально имеется пустой снеговик с номером ноль.

Все числа во входном файле целые.

Output

Выведите суммарную массу построенных снеговиков.

Examples

stdin	stdout
8 0 1 1 5 2 4 3 2 4 3 5 0 6 6 1 0	74

Problem B. Персистентная очередь

Time limit:	2 секунды
Memory limit:	256 мегабайта

Реализуйте персистентную очередь.

Input

Первая строка содержит количество действий n ($1 \le n \le 200\,000$).

В строке номер $i + 1$ содержится описание действия i:

- `1 t m` — добавить в конец очереди номер t ($0 \le t < i$) число m;
- `-1 t` — удалить из очереди номер t ($0 \le t < i$) первый элемент.

В результате действия i, описанного в строке $i + 1$ создается очередь номер i.

Изначально имеется пустая очередь с номером ноль.

Все числа во входном файле целые, и помещаются в знаковый 32-битный тип.

Output

Для каждой операции удаления выведите удаленный элемент на отдельной строке.

Examples

stdin	stdout
10 1 0 1 1 1 2 1 2 3 1 2 4 -1 3 -1 5 -1 6 -1 4 -1 8 -1 9	1 2 3 1 2 4

Problem C. Персистентный массив

Time limit: 1 second
Memory limit: 256 MiB

Вам задан массив. Вы должны выполнять с ним два типа операций.

- `create` i j x ($a_{new} = a_i$; $a_{new}[j] = x$) — создать новую версию массива из i-й, присвоить j-й элемент x, остальные элементы остаются такими же как и в i-й версии.
- `get` i j (print $a_i[j]$) — вывести значение j-го элемента i-й версии.

Input

Первая строка содержит целое число n ($1 \le n \le 10^5$), а затем следует n целых чисел — исходные значения элементов массива. Исходный массив соответствует версии номер 1. Затем следует количество запросов m ($1 \le m \le 10^5$) и затем m запросов по одному на строке. Новая версия массива, созданная когда уже есть k версий, получает номер $k + 1$. Все присваиваемые значения лежат в диапазоне от 0 до 10^9, включительно. Массив проиндексирован от 1 до n, включительно.

Output

Для каждого запроса `get` выведите соответствующее значение.

Example

standard input	standard output
6 1 2 3 4 5 6 11 create 1 6 10 create 2 5 8 create 1 5 30 get 1 6 get 1 5 get 2 6 get 2 5 get 3 6 get 3 5 get 4 6 get 4 5	6 5 10 5 10 8 6 30

Problem D. Персистентное мультимножество

Time limit: 1 секунда
Memory limit: 256 мегабайт

Ваша задача реализовать персистентное мультимножество, с изменением только последней версии. Мультимножество состоит только из целых положительных чисел не превосходящих m. Мультимножество должно поддерживать следующие запросы:

- add x — добавить x в мультимножество;
- remove x — удалить один из элементов, который равен x из мультимножества, если хотя бы один такой элемент присутствует. Если такого элемента нет, то мультимножество не меняется;
- different v — сколько существует различных x, таких что x присутствует в v-й версии мультимножества;
- unique v — сколько существует различных x, таких что x присутствует в единственном экземпляре в v-й версии мультимножества;
- count x v — сколько экземляров x присутствует в v-й версии мультимножества.

Изначально мультимножество имеет версию 0, а после i запросов — версию i.

Input

Первая строка содержит два целых числа n, m — количество операций над мультимножеством и максимальный возможный элемент мультимножества ($1 \le n, m \le 200\,000$). Следующие n строк содержат описание запросов. Запросы описываются названием и параметром, как в условии задачи. Для того, чтобы вы отвечали на запросы в online, вместо номеров версий v в запросах вам заданы числа y ($0 \le y \le n$). Пусть число s — сумма ответов на все предыдущие запросы different, unique и count. Число v для запроса different, unique или count с номером i вычисляется так: $v = (y+s) \bmod i$. Все числа x — целые, положительные и не превосходят m.

Output

Выведите ответы на запросы different, unique и count, по одному целому числу в строке.

Examples

stdin	stdout	Notes
9 3 add 2 add 1 add 2 different 3 unique 1 remove 2 unique 3 count 3 1 count 2 1	2 1 2 0 1	Неисправленные запросы: add 2 add 1 add 2 different 3 unique 3 remove 2 unique 6 count 3 6 count 2 6

Problem E. Персистентная приоритетная очередь

Time limit:	2 секунды
Memory limit:	256 мегабайт

Требуется реализовать структуру данных, которая хранит мультимножество и умеет изменять любую свою предыдущую версию, выполняя одну из этих операций:

1. Заданы v и x, требуется добавить в множество v элемент со значением x, после чего вывести минимальный элемент в получившемся множестве.
2. Заданы v и u, требуется объединить множества с номерами v и u, после чего вывести минимальный элемент в получившемся множестве.
3. Задано v, требуется вывести минимальный элемент в множестве v, после чего удалить минимальный элемент из множества v. Если множество пустое, то вывести, что множество пустое, и создать новое пустое множество.

Изначально есть одно пустое множество с номером 0. После операции с номером i множество, получаемое во время этой операции, получает номер i.

Input

Первая строка содержит число n — количество операций для выполнения.

От вас потребуется отвечать на запросы в онлайне, при этом поддерживая переменную s, которая пересчитывается следующим образом через предыдущее значение: если ответ на запрос равен x, то $s = (s_{old} + x) \bmod 239017$. Если же ответом на запрос является слово `empty`, то s не изменяется.

В следующих n строках заданы запросы.

Запросы первого типа описываются строкой `1 a b`, где a и b — неотрицательные целые числа, которые описывают v и x для соответствующего запроса, как $v = (a+s) \bmod i$ и $x = (b+17s) \bmod (10^9+1)$, где i — номер соответствующего запроса.

Запросы второго типа описываются строкой `2 a b`, где a и b — неотрицательные целые числа, которые описывают v и для соответствующего запроса, как $v = (a + s) \bmod i$ и $u = (b + 13s) \bmod i$, где i — номер соответствующего запроса.

Запросы третьего типа описываются строкой `3 a`, где a — неотрицательное целое число, которые описывает v для соответствующего запроса, как $v = (a + s) \bmod i$, где i — номер соответствующего запроса.

Число запросов не превышает 200 000. Гарантируется, что мощность любого созданного мультимножества не превышает 2^{63}.

Output

Требуется вывести ровно n строк, в каждой строке должно находиться неотрицательное целое число либо слово `empty`.

Для запросов первого и второго типа требуется вывести значение минимального элемента в только что созданном множестве, либо слово `empty`, если множество пустое.

Для запросов третьего типа требуется вывести минимальный элемент в множестве, либо слово `empty`, если множество пустое.

Example

стандартный ввод	стандартный вывод
9 1 0 2 1 0 999999970 2 2 0 3 0 2 4 4 3 0 3 0 3 0 3 8	2 3 2 2 2 2 2 3 empty

Problem F. Intercity Express

Time limit: 5 seconds
Memory limit: 256 MiB

Андрей — разработчик системы для продажи билетов на поезда Он решил проверить ее на маршруте, соединяющем два больших города, а также содержащем $n-2$ промежуточных пункт, так что всего на пути n станций, пронумерованные от 1 до n

Поезд имеет s посадочных мест, пронумерованных от 1 до s. Система имеет доступ к базе данных, которая содержит информацию о проданных на текущий момент билетах со станции 1 до станции n и нуждается в ответе на вопрос возможно ли продать билет со станции a до станции b, и если возможно, какое минимальное число мест доступно на всех отрезках между a и b. К сожалению, программа имеет доступ только в режиме чтения, поэтому если она видит свободное место, то она должна сообщить об этом, но не изменять базу данных.

Помогите Андрею протестировать его систему, напишите программу, которая выдает ответы на такие вопросы.

Input

Первая строка содержит число n — количество станций, s — количество сидений в поезде и m — количество уже проданных билетов ($2 \le n \le 10^9$, $1 \le s \le 100\,000$, $0 \le m \le 100\,000$). Далее следуют m линий, которые описывают уже проданные билеты с помощью чисел c_i, a_i, and b_i — место которое занято, начальная станция пути и конечная станция пути ($1 \le c_i \le s$, $1 \le a_i < b_i \le n$).

В следующей строке есть q — количество запросов ($1 \le q \le 100\,000$). Специальная переменная p должны быть заведена при чтении запросов. Изначально $p = 0$. Последующие $2q$ чисел описывают запросы. Каждый запрос характеризуется двумя числами: x_i and y_i ($x_i < y_i$). Получить номера городов a и b, существование билета между которыми нам нужно проверить, можно следующим образом: $a = x_i + p$, $b = y_i + p$. Ответ на запрос 0, если нет места на отрезке между a и b. Иначе выведите количество доступных мест.

После ответа на запрос прибавьте полученное число к p.

Output

Для каждого запроса выведите ответ

Example

stdin	stdout
5 3 5 1 2 5 2 1 2 2 4 5 3 2 3 3 3 4 10 1 2 1 2 1 2 2 3 -2 0 2 4 1 3 1 4 2 5 1 5	1 2 2 3 0 2 0 0 0 0

Запросы: $(1,2)$, $(2,3)$, $(3,4)$, $(4,5)$, $(1,3)$, $(2,4)$, $(3,5)$, $(1,4)$, $(2,5)$, $(1,5)$.

Problem G. Откат

Time limit: 2 секунды
Memory limit: 256 мегабайт

Сергей работает системным администратором в очень крупной компании. Естественно, в круг его обязанностей входит резервное копирование информации, хранящейся на различных серверах и «откат» к предыдущей версии в случае возникновения проблем.

В данный момент Сергей борется с проблемой недостатка места для хранения информации для восстановления. Он решил перенести часть информации на новые сервера. К сожалению, если что-то случится во время переноса, он не сможет произвести откат, поэтому процедура переноса должна быть тщательно спланирована.

На данный момент у Сергея хранятся n точек восстановления различных серверов, пронумерованных от 1 до n. Точка восстановления с номером i позволяет произвести откат для сервера a_i. Сергей решил разбить перенос на этапы, при этом на каждом этапе в случае возникновения проблем будут доступны точки восстановления с номерами $l, l+1, \ldots, r$ для некоторых l и r.

Для того, чтобы спланировать перенос данных оптимальным образом, Сергею необходимо научиться отвечать на запросы: для заданного l, при каком минимальном r в процессе переноса будут доступны точки восстановления не менее чем k различных серверов.

Помогите Сергею.

Input

Первая строка входного файла содержит два целых числа n и m, разделенные пробелами — количество точек восстановления и количество серверов ($1 \le n, m \le 100\,000$). Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ — номера серверов, которым соответствуют точки восстановления ($1 \le a_i \le m$).

Третья строка входного файла содержит q — количество запросов, которые необходимо обработать ($1 \le q \le 100\,000$). В процессе обработки запросов необходимо поддерживать число p, исходно оно равно 0. Каждый запрос задается парой чисел x_i и y_i, используйте их для получения данных запроса следующим образом: $l_i = ((x_i + p) \bmod n) + 1$, $k_i = ((y_i + p) \bmod m) + 1$ ($1 \le l_i, x_i \le n, 1 \le k_i, y_i \le m$). Пусть ответ на i-й запрос равен r. После выполнения этого запроса, следует присвоить p значение r.

Output

На каждый запрос выведите одно число — искомое минимальное r, либо 0, если такого r не существует.

Examples

stdin	stdout
7 3 1 2 1 3 1 2 1 4 7 3 7 1 7 1 2 2	1 4 0 6

Problem H. Шары и урны

Time limit: 2 секунды
Memory limit: 256 мебибайт

Рассмотрим n различных шаров и n различных урн, стоящих в ряд. Изначально каждая урна содержит ровно один из шаров.

Для перемещения шаров есть специальное устройство. Пользоваться им очень просто. Сначала необходимо выбрать непрерывный отрезок урн. После этого устройство поднимает шары изо всех урн этого отрезка. Наконец, необходимо выбрать другой непрерывный отрезок урн такой же длины, после чего устройство перемещает поднятые шары в этот отрезок. Каждая урна может вместить любое количество шаров.

По данной последовательности перемещений шаров при помощи устройства выясните для каждого шара, в какой урне он будет находиться после всех перемещений.

Input

Первая строка ввода содержит два целых числа n и m — количество урн и количество перемещений ($1 \le n \le 100\,000$, $1 \le m \le 50\,000$). Каждая из следующих m строк содержит по три числа count_i, from_i и to_i, которые означают, что устройство одновременно перемещает все шары из урны from_i в урну to_i, все шары из урны $\mathrm{from}_i + 1$ в урну $\mathrm{to}_i + 1$, ..., все шары из урны $\mathrm{from}_i + \mathrm{count}_i - 1$ в урну $\mathrm{to}_i + \mathrm{count}_i - 1$ ($1 \le \mathrm{count}_i, \mathrm{from}_i, \mathrm{to}_i \le n$, $\max(\mathrm{from}_i, \mathrm{to}_i) + \mathrm{count}_i \le n + 1$).

Output

Выведите ровно n чисел от 1 до n: конечные позиции всех шаров. Первое число должно задавать конечную позицию шара, который изначально был в первой урне, второе число — позицию шара из второй урны и так далее.

Examples

stdin	stdout
2 3 1 1 2 1 2 1 1 2 1	1 1
10 3 1 9 2 3 7 3 8 3 1	1 2 1 2 3 4 1 2 2 8

Problem I. Жадность

Time limit: 2 секунды
Memory limit: 256 мебибайт

В этой задаче вам предстоит решить хорошо известную задачу о рюкзаке. К сожалению, это будет не NP-трудная версия задачи, а более простая модификация.

Дано n предметов в фиксированном порядке, i-й имеет вес s_i и стоимость c_i. Также есть q различных рюкзаков, в i-й из которых помещаются предметы суммарным весом w_i. Вы заполняете рюкзак, жадным образом помещая в него предметы по одному (помните, что порядок предметов фиксирован и важен). Это значит, что вы никогда не вынимаете предметы и всегда помещаете их в рюкзак, если возможно, то есть если суммарный вес предметов в рюкзаке после этой операции не превысит его вместимости. Вы всегда пытаетесь поместить в рюкзак каждый из n предметов по порядку независимо от того, получилось ли поместить в него все предыдущие предметы.

Каждый из рюкзаков нужно заполнить по данному алгоритму и вывести суммарную стоимость предметов, которые в него попали. Все рюкзаки заполняются независимо, то есть каждый рюкзак заполняется всеми предметами независимо от того, были ли эти предметы использованы для других рюкзаков.

Input

В первой строке записано целое число N — количество предметов ($1 \le N \le 10^4$).

Во второй строке записано N целых чисел s_1, s_2, …, s_n — веса предметов ($1 \le s_i \le 10^{13}$).

В третьей строке записано N целых чисел c_1, c_2, …, c_n — стоимости предметов ($1 \le c_i \le 10^4$).

В четвёртой строке записано целое число q — количество рюкзаков, которые нужно попробовать заполнить ($1 \le q \le 10^6$).

В пятой строке записаны q целых чисел w_1, w_2, …, w_q — вместительности рюкзаков ($1 \le w_i \le 10^{18}$).

Output

Выведите q целых чисел — суммарную стоимость поместившихся вещей для каждого рюкзака.

Example

стандартный ввод	стандартный вывод
5 5 3 2 4 1 1 2 3 4 5 3 4 8 100	7 3 15