Задача A. Лексикографически минимальная подсеть

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

> Ты думаешь по-японски! Если взялся думать — думай по-немецки!
>
> Neon Genesis Evangelion

Рей Аянами изучает сети из генетически модифицированных деревьев. Сетью называется набор деревьев, соединённых двусторонними биокордами. Биокорды пронумерованы последовательными целыми числами, начиная с единицы. Каждый биокорд соединяет ровно два различных дерева.

Подсеть — это подмножество биокордов сети. Подсеть S устойчива, если:

- между любыми двумя различными деревьями есть путь из биокордов S,
- никакое собственное подмножество S не обладает этим свойством.

Подсети можно сравнивать лексикографически. Чтобы это сделать, следует записать номера биокордов подсети в возрастающем порядке, после чего сравнить лексикографически полученные последовательности чисел. Напомним, что последовательность a_1, a_2, …, a_p *лексикографически меньше* последовательности b_1, b_2, …, b_p, если для минимального i, для которого $a_i \neq b_i$, верно $a_i < b_i$.

Рей нужно найти одну конкретную подсеть заданной сети. Во-первых, эта подсеть должна быть устойчивой. Во-вторых, если устойчивых подсетей несколько, нужна та из них, у которой суммарная длина биокордов минимальна. В-третьих, если и таких подсетей несколько, из них нужна лексикографически минимальная.

Рей считает, что справится с этой задачей за несколько минут. А сможете ли вы это сделать?

Формат входных данных

Первая строка входных данных содержит два целых числа n и m — количество генетически модифицированных деревьев и количество двусторонних биокордов между ними ($2 \leqslant n \leqslant 100\,000$, $m \leqslant 100\,000$). Каждая из следующих m строк содержит три целых числа: номера деревьев, соединённых очередным биокордом, и длину этого биокорда. Длины биокордов неотрицательны и не превосходят $100\,000$. Гарантируется, что входные данные таковы, что ответ существует.

Формат выходных данных

Выведите $n-1$ число: номера биокордов в искомой подсети. Биокорды нумеруются целыми числами от 1 до m в том порядке, в котором они заданы во входных данных.

Пример

biocords.in	biocords.out
4 4 1 2 1 2 3 1 3 4 1 4 1 1	1 2 3

Задача B. Разрезание графа

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

Дан неориентированный граф. Над ним в заданном порядке производят операции следующих двух типов:

- `cut` — разрезать граф, то есть удалить из него ребро;
- `ask` — проверить, лежат ли две вершины графа в одной компоненте связности.

Известно, что после выполнения всех операций типа `cut` рёбер в графе не осталось. Найдите результат выполнения каждой из операций типа `ask`.

Формат входных данных

Первая строка ввода содержит три целых числа, разделённых пробелами — количество вершин графа n, количество рёбер m и количество операций k ($1 \leqslant n \leqslant 50\,000$, $0 \leqslant m \leqslant 100\,000$, $m \leqslant k \leqslant 150\,000$).

Следующие m строк задают рёбра графа; i-ая из этих строк содержит два числа u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$), разделённые пробелами — номера концов i-го ребра. Вершины нумеруются с единицы; граф не содержит петель и кратных рёбер.

Далее следуют k строк, описывающих операции. Операция типа `cut` задаётся строкой «`cut` u v» ($1 \leqslant u, v \leqslant n$), которая означает, что из графа удаляют ребро между вершинами u и v. Операция типа `ask` задаётся строкой «`ask` u v» ($1 \leqslant u, v \leqslant n$), которая означает, что необходимо узнать, лежат ли в данный момент вершины u и v в одной компоненте связности. Гарантируется, что каждое ребро графа встретится в операциях типа `cut` ровно один раз.

Формат выходных данных

Для каждой операции `ask` во вводе выведите на отдельной строке слово «`YES`», если две указанные вершины лежат в одной компоненте связности, и «`NO`» в противном случае. Порядок ответов должен соответствовать порядку операций `ask` во вводе.

Пример

cutting.in	cutting.out
3 3 7 1 2 2 3 3 1 ask 3 3 cut 1 2 ask 1 2 cut 1 3 ask 2 1 cut 2 3 ask 3 1	YES YES NO NO

Задача C. Система непересекающихся множеств

Имя входного файла:	`dsu.in`
Имя выходного файла:	`dsu.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мебибайт

Система непересекающихся множеств — структура данных, хранящая для некоторого множества его разбиение на подмножества в каждый момент времени. Она поддерживает две операции: слияние двух подмножеств и проверку принадлежности двух элементов одному подмножеству.

В этой задаче структура используется так. Дан граф из n вершин, изначально не имеющий рёбер. В граф последовательно предлагаются рёбра для добавления. Каждый раз, когда предлагается ребро между какими-то вершинами u и v веса c, следует проверить, лежат ли на этот момент u и v в одной компоненте связности. Если да, ребро игнорируется. Если нет, ребро добавляется в граф.

Найдите суммарный вес ребёр, добавленных в граф после всех указанных операций.

Формат входных данных

В первой строке заданы через пробел целые числа n и k — количество вершин в графе и количество строк, описывающих рёбра ($1 \leqslant n \leqslant 10^7$, $0 \leqslant k \leqslant 10^5$).

В следующих k строках заданы рёбра. Каждая из них имеет вид $u\ v\ c\ \Delta u\ \Delta v\ \Delta c\ m$ ($0 \leqslant u, v < n$, $0 \leqslant c < 10^9$, $|\Delta u|, |\Delta v|, |\Delta c| < 10^9$, $1 \leqslant m \leqslant 10^7$, все числа в строке целые). Такая строка означает, что последовательно поступают предложения о добавлении в граф m рёбер. Первое из них — ребро между вершинами u и v, имеющее вес c. Второе — ребро между $(u + \Delta u) \bmod n$ и $(v + \Delta v) \bmod n$, имеющее вес $(c + \Delta c) \bmod 10^9$, и так далее. Последнее из этих m рёбер соединяет вершины $(u + (m-1) \cdot \Delta u) \bmod n$ и $(v + (m-1) \cdot \Delta v) \bmod n$, а его вес равен $(c + (m-1) \cdot \Delta c) \bmod 10^9$. Напомним, что $x \bmod y$ — это наименьшее неотрицательное число z такое, что величина $z - x$ делится нацело на y.

Общее количество предлагаемых рёбер, равное сумме чисел m во всех строках, не превосходит 10^7. Среди предлагаемых рёбер могут быть кратные рёбра и петли. Обратите внимание на то, что вершины в графе нумеруются с нуля.

Формат выходных данных

В первой строке выведите одно число — суммарный вес рёбер, добавленных в граф после всех указанных операций.

Примеры

dsu.in	dsu.out
8 3 0 1 1 2 2 0 4 0 1 2 1 1 0 5 0 3 6 9 12 15 2	29
4 1 1 2 1 1 1 1 2	3

Задача D. ЛАЖУ

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

На мат-мехе, в аудитории 00, стоит бесконечное количество парт, расставленных в ряд и пронумерованных целыми положительными числами. Сегодня в этой аудитории проходит экзамен по Линейному Анализу Жутких Уравнений. Время от времени приходят студенты. Каждый студент знает, за какую парту он хочет сесть, но на экзамене двум студентам нельзя сидеть вместе, поэтому, если парта занята, он занимает первую свободную парту с бóльшим номером (подальше от преподавателя).

Кроме того, некоторые студенты не сдают экзамен и уходят. После этого за парту, которую занимал ушедший студент, может сесть вновь прибывший.

Преподаватель знает, когда студенты будут приходить и когда они будут уходить учить заново. Вам требуется по этой информации узнать, за какой партой будет сидеть каждый из студентов (чтобы заранее положить туда конспект).

Формат входных данных

Первая строка входных данных содержит натуральное число n — количество событий, происходящих в течение экзамена ($n \leqslant 100\,000$).

Следующие n строк содержат информацию об этих событиях. Число $a > 0$ обозначает, что пришёл студент, желающий занять парту номер a ($a \leqslant 100\,000$). Число $a < 0$ обозначает, что студент освободил парту с номером $-a$. (Гарантируется, что он за ней действительно сидел.)

Формат выходных данных

Для каждого студента выведите одно целое положительное число — номер парты, которую он займёт.

Пример

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

Задача E. Гитара

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

Спать днём нельзя!

А. С. Лопатин

Девочка Соня всегда берёт с собой в поход гитару. Однажды вечером Соня и её друзья собрались вокруг костра и стали петь песни. Все очень утомились, поэтому решили, что каждый сыграет не больше одной песни за этот вечер — это поможет раньше уйти спать.

После того, как очередной человек исполнил песню, он передаёт гитару человеку, которого заранее выбрал, а сам уходит спать. Если же оказывается, что тот выбранный человек уже не сидит у костра, спать уходит вся компания.

Кроме того, как только у человека сыграют оба его соседа, ему становится скучно и он тоже уходит спать.

Определите, в каком порядке люди будут играть на гитаре.

Формат входных данных

В первой строке входных данных записано целое число n — количество ребят у костра ($3 \leqslant n \leqslant 100\,000$). Во второй строке записаны n чисел, отражающих предпочтения: i-е число — это номер человека, которому i-й человек хотел бы передать гитару. Люди перечислены в порядке обхода против часовой стрелки.

Изначально гитара у Сони, которая имеет номер 1. Гарантируется, что никто не планирует передавать гитару сам себе.

Формат выходных данных

В первой строке выведите целое число — количество песен, которые будут исполнены этим вечером. В следующей строке выведите номера людей, которые будут играть на гитаре, в том порядке, в котором они будут это делать.

Пример

guitar.in	guitar.out
5 3 3 5 1 2	3 1 3 5

Задача F. Гармонический ряд

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

Дано целое число n. Выведите число, которое получится в r после выполнения следующей программы на языке, аналогичном C++:

```
r = 0;
for (i = 1; i <= n; i++)
  for (j = i; j <= n; j += i)
    r += 1;
```

Все переменные — 64-битные целые числа со знаком.

Формат входных данных

В первой строке записано целое число n ($1 \leqslant n \leqslant 10^{14}$).

Формат выходных данных

В первой строке выведите одно целое число r — значение результата после выполнения программы.

Примеры

harmonic.in	harmonic.out
3	5
10	27

Задача G. Список степеней

Имя входного файла:	`list-powers.in`
Имя выходного файла:	`list-powers.out`
Ограничение по времени:	3.5 секунды
Ограничение по памяти:	256 мебибайт

Пусть задано простое число p и число a такое, что $0 < a < p$. Рассмотрим все числа от l до r включительно, представимые в виде $a^k \bmod p$ для какого-то целого неотрицательного числа k. Пусть известно, что таких чисел не более 100. Выведите все эти числа в порядке возрастания.

Формат входных данных

В единственной строке заданы через пробел четыре целых числа p, a, l и r ($0 < a < p \leqslant 10^9$, p простое, $0 \leqslant l \leqslant r < p$).

Формат выходных данных

Выведите все числа от l до r включительно, представимые в виде $a^k \bmod p$ для какого-то целого неотрицательного числа k, в порядке возрастания, разделяя соседние числа пробелом. Гарантируется, что входные данные таковы, что в правильном ответе не более 100 чисел.

Примеры

list-powers.in	list-powers.out
5 3 0 3	1 2 3
5 4 2 3	

Пояснения к примерам

В первом примере требуется найти все числа от $l = 0$ до $r = 3$ включительно, которые представимы в виде $3^k \bmod 5$ для некоторого целого $k \geqslant 0$. Это числа $3^0 \bmod 5 = 1$, $3^1 \bmod 5 = 3$ и $3^3 \bmod 5 = 27 \bmod 5 = 2$. Число 0 не представимо в таком виде, поскольку 3^k не делится на 5 ни для какого целого $k \geqslant 0$. Значит, следует вывести числа 1, 2 и 3 в порядке возрастания.

Во втором примере требуется найти все числа от $l = 2$ до $r = 3$ включительно, которые представимы в виде $4^k \bmod 5$ для некоторого целого $k \geqslant 0$. Выпишем первые несколько чисел такого вида:
$4^0 \bmod 5 = 1$,
$4^1 \bmod 5 = 4$,
$4^2 \bmod 5 = 16 \bmod 5 = 1$,
$4^3 \bmod 5 = 64 \bmod 5 = 4$,
$4^4 \bmod 5 = 256 \bmod 5 = 1$, ...
Можно доказать, что в этой последовательности встречаются лишь числа 1 и 4. Поэтому список в ответе будет пустым.

Задача H. Числа на отрезке

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

Вова нарисовал на доске горизонтальную прямую, отметил на ней N точек и пронумеровал их слева направо натуральными числами от 1 до N. После этого он стал обводить некоторые точки кружочками. Время от времени Паша, чтобы оторвать его от этого занятия, спрашивает его, сколько точек на отрезке от A до B, включительно, Вова уже обвёл кружочками. Ответьте Паше на все его вопросы, чтобы не отвлекать Вову.

Формат входных данных

В первой строке входных данных записаны два целых числа N и K — количество точек на отрезке и количество событий, соответственно ($1 \leqslant N \leqslant 1\,000\,000$, $1 \leqslant K \leqslant 100\,000$).

В следующих K строках заданы события в порядке, в котором они случались. Каждая из этих строк либо содержит целое число C от 1 до N, включительно, которое означает, что Вова обвёл кружочком точку с номером C, либо имеет вид 0 A B, где $1 \leqslant A \leqslant B \leqslant N$, что означает, что Паша спросил, сколько точек на отрезке от A до B, включительно, уже обведено кружочками.

Вова обводит каждую точку не более одного раза.

Формат выходных данных

Выведите ответ на каждый вопрос Паши на отдельной строке в том порядке, в котором эти вопросы даны во входных данных.

Примеры

numbers.in	numbers.out
3 4 1 0 1 1 2 0 1 3	1 2
10 6 0 1 10 6 1 4 0 2 9 8	0 2

Задача I. Поиск минимума на отрезке

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

Задана последовательность целых чисел A длины N. Для каждой заданной пары чисел (k, l) требуется найти значение индекса минимального элемента в последовательности A, начиная с индекса k и заканчивая индексом l.

Формат входных данных

В первой строке входных данных содержатся два числа N и M — количество элементов последовательности $1 \leqslant N \leqslant 100\,000$ и количество запросов $1 \leqslant M \leqslant 100\,000$. В следующей строке через пробел перечислены элементы последовательности A. Все числа не превышают границ 32-битного числа со знаком. Последующие M строк содержат по два числа (k, l) — начало и конец отрезков, на которых требуется найти индекс минимального элемента.

Формат выходных данных

Выведите индексы минимальных элементов последовательности A на заданных отрезках для всех запросов.

Пример

rmq.in	rmq.out
6 3 1 8 4 5 3 7 1 6 2 2 3 5	1 2 5

Задача J. Поиск суммы на отрезке

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

Задана последовательность целых чисел A длины N. Для каждой заданной пары чисел (k, l) требуется найти значение суммы элементов в последовательности A, начиная с индекса k и заканчивая индексом l.

Формат входных данных

В первой строке входных данных содержатся два числа N и M — количество элементов последовательности $1 \leqslant N \leqslant 100\,000$ и количество запросов $1 \leqslant M \leqslant 100\,000$. В следующей строке через пробел перечислены элементы последовательности A. Все числа не превышают границ 32-битного числа со знаком. Последующие M строк содержат по два числа (k, l) — начало и конец отрезков, на которых требуется найти сумму элементов.

Формат выходных данных

Выведите суммы элементов последовательности A на заданных отрезках для всех запросов.

Пример

rsq.in	rsq.out
6 3 1 8 4 5 3 7 1 6 2 2 3 5	28 8 12

Задача K. Точки и отрезки

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

Дано n отрезков на числовой прямой и m точек на этой же прямой. Для каждой из данных точек определите, скольким отрезкам она принадлежит. Точка x считается принадлежащей отрезку с концами a и b, если выполняется двойное неравенство $\min(a, b) \leqslant x \leqslant \max(a, b)$.

Формат входных данных

Первая строка содержит два целых числа n ($1 \leqslant n \leqslant 10^5$) — число отрезков и m ($1 \leqslant m \leqslant 10^5$) — число точек. В следующих n строках записаны по два целых числа a_i и b_i — координаты концов соответствующего отрезка. В последней строке записаны m целых чисел — координаты точек. Все числа во входных данных не превосходят по модулю 10^9.

Формат выходных данных

Выведите m чисел — для каждой точки выведите количество отрезков, в которых она содержится.

Примеры

segments.in	segments.out
2 2 0 5 7 10 1 6	1 0
1 3 -10 10 -100 100 0	0 0 1

Задача L. CHM

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

Ваша задача — реализовать **Persistent Disjoint Set Union** (персистентную систему непересекающихся множеств). Что это значит?

Про **Disjoint Set Union**:

Изначально у вас есть n элементов, пронумерованных целыми числами от 1 до n и лежащих каждый в своём множестве. Нужно научиться отвечать на два типа запросов.

- `+ a b` — объединить множества, в которых лежат элементы a и b
- `? a b` — сказать, лежат ли элементы a и b сейчас в одном множестве

Про **Persistent**:

Теперь у нас будет несколько копий (версий) структуры данных **Disjoint Set Union**.

Запросы будут выглядеть так:

- `+ i a b` — запрос к i-й структуре: объединить множества, в которых лежат элементы a и b. При этом i-я структура остаётся неизменной, создаётся новая версия, ей присваивается новый номер (какой? читайте дальше)
- `? i a b` — запрос к i-й структуре: сказать, лежат ли элементы a и b сейчас в одном множестве

Формат входных данных

На первой строке два числа N ($1 \leqslant N \leqslant 10^5$) и K ($0 \leqslant K \leqslant 10^5$) — число элементов и число запросов. Эта изначальная копия (версия) структуры имеет номер 0.

Далее следуют K строк, на каждой описание очередного запроса. Формат запросов описан выше. Запросы нумеруются целыми числами от 1 до K.

При обработки j-го запроса, если он имеет вид «`+ i a b`», новая версия получит номер j.

Формат выходных данных

Для каждого запроса вида «`? i a b`» на отдельной строке нужно вывести «`YES`» или «`NO`».

Пример

snm.in	snm.out
4 7 + 0 1 2 ? 0 1 2 ? 1 1 2 + 1 2 3 ? 4 3 1 ? 0 4 4 ? 4 1 4	NO YES YES YES NO