Задача A. LCA

Имя входного файла:	`lca.in`
Имя выходного файла:	`lca.out`
Ограничение по времени:	5 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный граф. Подсчитайте, сколько пар вершин (i, j) имеют общего предка. Общим предком вершин i и j называется такая вершину k, что и i, и j достижимы из k.

Формат входного файла

В первой строке входного файла содержатся целые числа $1 \le N \le 10^4, 0 \le M \le 10^4$ — количество вершин и рёбер в графе. Далее следуют M строк по два числа от 1 до N. Пара чисел (a, b) означает, что из вершины a есть ребро в вершину b.

Формат выходного файла

Выведите одно целое число — количество пар.

Примеры

lca.in	lca.out
2 1 1 2	4
3 2 2 1 3 1	7

Задача B. Самое дешевое ребро

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

Дано подвешенное дерево с корнем в первой вершине. Все ребра имеют веса (стоимости). Вам нужно ответить на M запросов вида "найти у двух вершин минимум среди стоимостей ребер пути между ними".

Формат входного файла

В первой строке файла записано одно числ — n (количество вершин).

В следующих $n-1$ строках записаны два числа — x и y. Число x на строке i означает, что x — предок вершины i, y означает стоимость ребра.

$x < i,\ |y| \le 10^6$.

Далее m запросов вида (x, y) — найти минимум на пути из x в y $(x \ne y)$.

Ограничения: $2 \le n \le 5 \cdot 10^4$, $0 \le m \le 5 \cdot 10^4$.

Формат выходного файла

Выведите m ответов на запросы.

Примеры

minonpath.in	minonpath.out
5 1 2 1 3 2 5 3 2 2 2 3 4 5	2 2

Задача C. LCA - 2

Имя входного файла:	`lca2.in`
Имя выходного файла:	`lca2.out`
Ограничение по времени:	8 секунды
Ограничение по памяти:	256 мегабайт

Задано подвешенное дерево, содержащее n $(1 \le n \le 100\,000)$ вершин, пронумерованных от 0 до $n-1$. Требуется ответить на m $(1 \le m \le 10\,000\,000)$ запросов о наименьшем общем предке для пары вершин.

Запросы генерируются следующим образом. Заданы числа a_1, a_2 и числа x, y и z. Числа $a_3, \ldots, a_{2m}$ генерируются следующим образом: $a_i = (x \cdot a_{i-2} + y \cdot a_{i-1} + z) \bmod n$. Первый запрос имеет вид $\langle a_1, a_2 \rangle$. Если ответ на $i-1$-й запрос равен v, то i-й запрос имеет вид $\langle (a_{2i-1} + v) \bmod n, a_{2i} \rangle$.

Формат входного файла

Первая строка содержит два числа: n и m. Корень дерева имеет номер 0. Вторая строка содержит $n-1$ целых чисел, i-е из этих чисел равно номеру родителя вершины i. Третья строка содержит два целых числа в диапазоне от 0 до $n-1$: a_1 и a_2. Четвертая строка содержит три целых числа: x, y и z, эти числа неотрицательны и не превосходят 10^9.

Формат выходного файла

Выведите в выходной файл сумму номеров вершин — ответов на все запросы.

Примеры

lca2.in	lca2.out
3 2 0 1 2 1 1 1 0	2
1 2 0 0 1 1 1	0

Задача D. LCA-3

Имя входного файла:	`lca3.in`
Имя выходного файла:	`lca3.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Подвешенное дерево —это ориентированный граф без циклов, в котором в каждую вершину, кроме одной, называемой *корнем* ориентированного дерева, входит одно ребро. В корень ориентированного дерева не входит ни одного ребра. *Отцом* вершины называется вершина, ребро из которой входит в данную.

(по материалам Wikipedia)

Дан набор подвешенных деревьев. Требуется выполнять следующие операции:

1. 0 u v Для двух заданных вершин u и v выяснить, лежат ли они в одном дереве. Если это так, вывести вершину, являющуюся их наименьшим общим предком, иначе вывести 0.
2. 1 u v Для корня u одного из деревьев и произвольной вершины v другого дерева добавить ребро (v, u). В результате эти два дерева соединятся в одно.

Вам необходимо выполнять все операции online, т.е. вы сможете узнать следующий запрос только выполнив предыдущий.

Формат входного файла

На первой строке входного файла находится число n — суммарное количество вершин в рассматриваемых деревьях, $1 \le n \le 50000$. На следующей строке расположено n чисел — предок каждой вершины в начальной конфигурации, или 0, если соответстующая вершина является корнем. Затем следует число k — количество запросов к вашей программе, $1 \le k \le 100000$. Каждая из следующих строк содержит по три целых числа: вид запроса (0 — для поиска LCA или 1 — для добавления ребра) и два числа x, y. Вершины, участвующие в запросе можно вычислить по формуле: $u = (x - 1 + ans) \bmod n + 1$, $v = (y - 1 + ans) \bmod n + 1$, где ans - ответ на последний запрос типа 0 ($ans = 0$ для первого запроса).

Формат выходного файла

Для каждого запроса типа 0, выведите в выходной файл одно число на отдельной строке — ответ за этот запрос.

Примеры

lca3.in	lca3.out
5 0 0 0 0 0 12 1 5 3 0 2 5 1 4 2 1 1 5 0 1 5 1 3 4 0 1 5 0 3 1 0 4 2 0 1 4 0 5 2 0 4 1	0 5 5 3 2 3 3 2