Задача A. Разрезание графа

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

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 входного файла содержит три целых числа, разделённые 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-я 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 входном файле 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 входном файле.

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 B. LCA - 2

Имя входного файла:	`lca2.in`
Имя выходного файла:	`lca2.out`
Ограничение по времени:	5 секунды
Ограничение по памяти:	256 мегабайт

Задано подвешенное дерево, содержащее n ($1 \leqslant n \leqslant 100\,000$) вершин, пронумерованных от 0 до $n-1$. Требуется ответить на m ($1 \leqslant m \leqslant 10\,000\,000$) 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 входных данных

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 выходных данных

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 C. Самое дешевое ребро

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

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 входных данных

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.

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

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

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

Формат выходных данных

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 D. Расстояние в дереве

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

Вам дано неориентированное дерево (связный граф без циклов) на N вершинах. Требуется отвечать на запросы вида «найти длину кратчайшего пути между данной парой вершин».

Формат входных данных

В первой строке находится одно целое число N ($1 \leqslant N \leqslant 10^5$) — количество вершин в дереве.

В каждой из следующих $N - 1$ строк задано по два числа, разделенных пробелом, u_i, v_i — вершины, соединенные i-ым ребром дерева ($1 \leqslant u_i, v_i \leqslant N$).

В следующей строке задано число M — количество запросов ($0 \leqslant M \leqslant 10^5$).

В каждой из следующих M строк задано по два числа — номера вершин, соответствующих очередному запросу.

Формат выходных данных

Выведите M строк, в каждой из которых содержится одно целое число — ответ на очередной запрос. Ответы необходимо вывести в порядке, соответствующем порядку запросов на входе программы.

Примеры

lenpath.in	lenpath.out
3 2 1 3 2 2 2 2 2 1	0 1

Задача E. Генеалогия

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

Во время обсуждений в Парламенте лорды, с похожими взглядами на решение проблемы, обычно объединяются в группы. Как правило, результат обсуждения зависит от решения наиболее влиятельной группы лордов. Именно поэтому подсчёт влиятельности группы является наиболее важной задачей.

Естественно, каждый лорд дорожит древностью своего рода, поэтому влиятельность лорда равна древности его рода. Древность рода лорда — количество предков лорда: его отец, его дед, его прадед, и т.д. Чтобы посчитать влиятельность группы лордов, требуется посчитать количество лордов в группе вместе с их предками. Отметим, что если лорд является предком двух или более лордов в группе, то этот лорд должен быть посчитан только один раз.

Вам дано фамильное дерево лордов (удивительно, но все лорды произошли от одного пра-лорда) и список групп. Для каждой группу найдите её влиятельность.

Формат входных данных

Первая строка входного файла содержит число n — количество лордов ($1 \leqslant n \leqslant 100\,000$). Лорды нумеруются целыми числами от 1 до n. Следующая строка содержит n целых чисел $p_1, p_2, \ldots, p_n$, где p_i — отец лорда с номером i. Если лорд является основателем рода, то p_i равно -1. Гарантируется, что исходные данные формируют дерево. Третья строка входного файла содержит одно число g — количество групп ($1 \leqslant g \leqslant 3\,000\,000$). Следующие g строк содержат описания групп. j-ая строка содержит число k_j — размер j-ой группы, после которого следуют k_j различных чисел — номера лордов, состоящих в j-ой группе. Гарантируется, что сумма всех k_j во входном файле не превосходит $3\,000\,000$.

Формат выходных данных

В выходной файл выведите g строк. В j-ой строке выведите единственное число: влиятельность j-ой группы. Гарантируется, что размер выходного файла не превосходит шести мегабайт.

Примеры

genealogy.in	genealogy.out
4 -1 1 2 3 4 1 4 2 3 4 3 2 3 4 4 1 2 3 4	4 4 4 4
5 2 -1 1 2 3 10 3 3 4 1 3 2 4 3 4 1 3 5 4 1 4 2 2 3 3 1 4 3 1 2 3 3 4 5 1 1 3 1 2 4	4 4 5 2 3 4 1 5 2 3