Задача Cutting. Разрезание графа

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

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 входного файла

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-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 выходного файла

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 LCA. Offline

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

Изначально имеется дерево состоящее только из корня (вершина с номером 1). Требуется научиться отвечать на следующие запросы:

- ADD a b — подвесить вершину b за вершину a (гарантируется, что вершина a уже существует).
- GET a b — вернуть LCA вершин a и b.

Все номера вершин от 1 до N.

В каждый момент времени у нас есть одно дерево.

Формат входного файла

В первой строке входного файла содержится число k — количество запросов. Следующие k строк содержат сами запросы. Гарантируется, что число запросов каждого из типов не превосходит 500 000.

Формат выходного файла

Для каждого запроса типа GET выведите в отдельную строку одно целое число — ответ на соответствующий запрос.

Примеры

lca.in	lca.out
9 ADD 1 2 ADD 1 3 ADD 2 4 GET 1 3 GET 2 3 GET 3 4 ADD 2 5 GET 4 5 GET 5 5	1 1 1 2 5

Задача LCA. Offline-Easy

Задача «LCA Offline» с ограничением на количество запросов — 1000.

Задача LCA-RMQ. LCA Problem Revisited

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

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 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 число 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

lca_rmq.in	lca_rmq.out
3 2 0 1 2 1 1 1 0	2

Задача LCA-RMQ-Easy. LCA Problem Revisited (простая)

Задача «LCA Problem Revisited» с ограничением 100 на количество вершин в дереве.