

Задача А. Разрезание графа

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

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 ($1 \leq n \leq 50\,000$, $0 \leq m \leq 100\,000$, $m \leq k \leq 150\,000$).

Следующие m строк задают рёбра графа; i -ая из этих строк содержит два числа u_i и v_i ($1 \leq u_i, v_i \leq n$), 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 ” ($1 \leq u, v \leq n$), которая означает, что из графа удаляют ребро между вершинами u и v . Операция типа **ask** задаётся строкой “**ask** u v ” ($1 \leq u, v \leq n$), 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	YES
1 2	YES
2 3	NO
3 1	NO
ask 3 3	
cut 1 2	
ask 1 2	
cut 1 3	
ask 2 1	
cut 2 3	
ask 3 1	

Задача В. Цветные волшебники

Имя входного файла: magic.in
Имя выходного файла: magic.out
Ограничение по времени: 10 секунд
Ограничение по памяти: 64 мегабайта

Сказочная страна представляет собой множество городов, соединённых дорогами с двухсторонним движением. Причем из любого города страны можно добраться в любой другой город либо непосредственно, либо через другие города. Известно, что в сказочной стране не существует дорог, соединяющих город сам с собой и между любыми двумя разными городами, существует не более одной дороги.

В сказочной стране живут желтый и синий волшебники. Желтый волшебник, пройдя по дороге, перекрашивает ее в желтый цвет, синий — в синий. Как известно, при наложении желтой краски на синюю, либо синей краски на желтую, краски смешиваются и превращаются в краску зеленого цвета, который является самым нелюбимым цветом обоих волшебников.

В этом году в столице страны (городе f) проводится конференция волшебников. Поэтому желтый и синий волшебники хотят узнать, какое минимальное количество дорог им придется перекрасить в зеленый цвет, чтобы добраться в столицу. Изначально все дороги не покрашены.

Начальное положение желтого и синего волшебников заранее не известно. Поэтому необходимо решить данную задачу для k возможных случаев их начальных расположений.

Формат входного файла

Первая строка входного файла содержит целые числа: n ($1 \leq n \leq 100\,000$) и m ($1 \leq m \leq 500\,000$) — количество городов и дорог в волшебной стране соответственно. Третья строка содержит одно целое число f ($1 \leq f \leq n$) — номер города, являющегося столицей сказочной страны. В следующих m строках, находится описание дорог страны. В этих m строк записано по два целых числа a_i и b_i , означающих, что существует дорога, соединяющая города a_i и b_i . Следующая строка содержит целое число k ($1 \leq k \leq 100\,000$) — количество возможных начальных расположений волшебников. Далее следуют k строк, каждая из которых содержит два целых числа — номера городов, в которых изначально находится желтый и синий волшебники соответственно.

Формат выходного файла

Для каждого из k случаев, ваша программа должна вывести в выходной минимальное количество дорог, которое придется покрасить в зеленый цвет волшебникам для того, чтобы добраться в столицу.

Пример

magic.in	magic.out
6 6	1
1	2
1 2	
2 3	
3 4	
4 2	
4 5	
3 6	
2	
5 6	
6 6	

Задача С. Цветные волшебники-easy

Задача «Цветные волшебники» с ограничениями $1 \leq n, k \leq 100$, $1 \leq m \leq 500$.

Задача D. LCA offline

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

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 .

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	1
ADD 1 2	1
ADD 1 3	1
ADD 2 4	2
GET 1 3	5
GET 2 3	
GET 3 4	
ADD 2 5	
GET 4 5	
GET 5 5	

Задача E. LCA offline-Easy

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

Задача F. LCA Problem Revisited

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

Задано подвешенное дерево, содержащее n ($1 \leq n \leq 100\,000$) вершин, пронумерованных от 0 до $n - 1$. Требуется ответить на m ($1 \leq m \leq 10\,000\,000$) 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, \dots, 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	2
0 1	
2 1	
1 1 0	

Задача G. LCA Problem Revisited-easy

Предыдущая задача, только $n, m \leq 100$.