

Задача А. 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 ($1 \leq N \leq 10^5$) — 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 ($1 \leq u_i, v_i \leq N$).

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

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

<code>lenpath.in</code>	<code>lenpath.out</code>
3	0
2 1	1
3 2	
2	
2 2	
2 1	

Задача В. 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.

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

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

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

Формат выходных данных

Выведите m ответов на запросы.

Примеры

<code>minonpath.in</code>	<code>minonpath.out</code>
5	2
1 2	2
1 3	
2 5	
3 2	
2	
2 3	
4 5	

Задача С. 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 ($1 \leq n \leq 100\,000$) 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 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 D. Дерево

Имя входного файла: `tree.in`
Имя выходного файла: `tree.out`
Ограничение по времени: 4 секунды
Ограничение по памяти: 256 мегабайт

Задано подвешенное дерево, содержащее n ($1 \leq n \leq 1\,000\,000$) вершин. Каждая вершина покрашена в один из n цветов. Требуется для каждой вершины v вычислить количество различных цветов, встречающихся в поддереве с корнем v .

Формат входных данных

В первой строке входного файла задано число n . Последующие n строк описывают вершины, по одной в строке. Описание очередной вершины i имеет вид $p_i c_i$, где p_i — номер родителя вершины i , а c_i — цвет вершины i ($1 \leq c_i \leq n$). Для корня дерева $p_i = 0$.

Формат выходных данных

Выведите n чисел, обозначающих количества различных цветов в поддеревьях с корнями в вершинах $1, \dots, n$.

Примеры

tree.in	tree.out
5	1 2 3 1 1
2 1	
3 2	
0 3	
3 3	
2 1	

Задача E. Учиться!

Имя входного файла: `moscow.in`
Имя выходного файла: `moscow.out`
Ограничение по времени: 3 секунды
Ограничение по памяти: 64 мегабайта

Каждый год огненное количество выпускников, сдавшие ЕГЭ, выбирают, куда же они пойдут учиться. Не удивительно, что многие из них предпочитают перебраться поближе к столице. Транспортная инфраструктура страны переживает не лучшие времена, и в приемлемом качестве поддерживается минимально возможное число городов, необходимое для того, чтобы от любого города можно было добраться до любого другого.

Каждый выпускник оценивает свои результаты сдачи экзаменов, и решает, насколько далеко от своего родного города в сторону столицы он сможет уехать.

Выпускников настолько много, что вам не требуется выводить для каждого из них, до какого города он сможет доехать. Достаточно вывести сумму ответов для каждого выпускника.

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 $\langle (a_{2i-1} + v) \bmod n, a_{2i} \rangle$. В i -м запросе первое число соответствует городу, в котором окончил школу i -й выпускник, а второе — насколько далеко от родного города он может уехать. Все выпускники стараются перебраться как можно ближе к столице.

Формат входных данных

Первая строка содержит два числа: n ($1 \leq n \leq 100\,000$) и m ($1 \leq m \leq 10\,000\,000$). Столица имеет номер 0. Вторая строка содержит $n - 1$ целых чисел, i -е из этих чисел равно номеру следующего за городом i на пути к столице. 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 городов — ответов на все запросы.

Примеры

<code>moscow.in</code>	<code>moscow.out</code>
3 2 0 1 2 1 1 1 0	1
1 2 0 0 1 1 1	0