Где корень?

Это интерактивная задача

Вам дано дерево из n вершин. Деревом называется такой граф, что между каждой парой его вершин существует ровно один простой путь. **Также гарантируется, что хотя бы одна вершина напрямую соединена ребром не менее чем с 3 вершинами.** Одна из вершин является корнем, и ваша задача найти его. Для этого вы можете делать запросы следующего вида:

- Для заданного множества вершин $a_1, a_2, \ldots, a_m$ проверить, лежит ли их наименьший общий предок в этом множестве .

Вершина v является общим предком множества вершин S, если пути от всех вершин в S к корню проходят через v. Наименьшим общим предком (LCA) множества вершин S является общий предок S, наиболее удаленный от корня.

Протокол взаимодействия

Взаимодействие начинается со считывания одного целого числа n ($4 \le n \le 500$) — количество вершин.

Затем считайте следующие $n-1$ строк. В i-й строке будут записаны два целых числа a_i, b_i ($1 \le a_i, b_i \le n$), указывающие на наличие ребра между вершинами a_i, b_i в дереве.

Гарантируется, что данные $n-1$ ребер образуют дерево и хотя бы одна вершина напрямую соединена ребром не менее чем с 3 вершинами..

Чтобы сделать запрос, сначала выведите "`?`", затем целое число m, а затем m различных целых чисел $a_1, a_2, \ldots, a_m$ ($1 \le m \le n$, $1 \le a_i \le n$, все a_i различны) - вершины, для которых вы хотите проверить, есть ли среди них их LCA.

В качестве ответа программа выдаст "`YES`", если их LCA является одним из $a_1, a_2, \ldots, a_m$, и "`NO`" в противном случае.

Вы можете задать не более 1000 запросов, но вы получите разное количество баллов в зависимости от того, сколько запросов вы задали. Вывод ответа не считается запросом. Подробности смотрите в разделе система оценки.

Когда вы определили корень, выведите символ "`!`" и затем одно целое число v ($1 \le v \le n$) - корень. Затем завершите свою программу.

После вывода запроса не забудьте вывести перевод строки и сбросить буфер вывода. Для этого используйте:

- `fflush(stdout)` или `cout.flush()` в C++;
- `stdout.flush()` в Python;

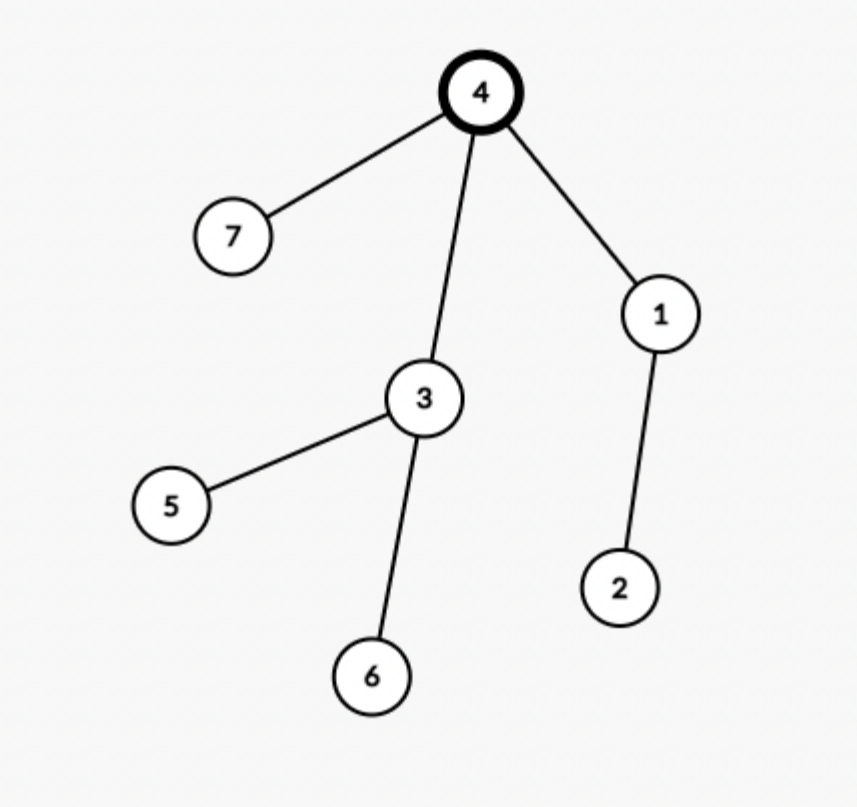
Гарантируется, что для каждого набора входных данных дерево и его корень фиксируются до начала взаимодействия. Другими словами, **интерактор не адаптивен**.

Не забывайте сбрасывать буфер вывода после каждого запроса.

Пример

```
Input:
7
4 1
1 2
4 3
3 5
3 6
4 7
Output:
? 2 5 6
Input:
NO
Output:
? 3 6 3 5
Input:
YES
Output:
? 2 1 7
Input:
NO
Output:
? 2 4 6
Input:
YES
Output:
! 4
```

Примечание



Скрытый корень — это вершина 4.

В первом запросе LCA вершин 5 и 6 — это вершина 3, которой нет среди вершин 5 и 6, поэтому ответ «NO».

Во втором запросе LCA вершин 3, 5 и 6 — это вершина 3, поэтому ответ «YES».

В третьем запросе LCA вершин 1 и 7 — это вершина 4, поэтому ответ «NO».

В четвертом запросе LCA вершин 4 и 6 — это вершина 4, поэтому ответ — «YES».

После этого мы можем предположить, что корнем является вершина 4, что является правильным ответом.

Система оценки

1. (7 баллов): $n \leq 9$
2. (10 баллов): $n \leq 30$
3. (до 83 баллов): $n \leq 500$

В первой и второй подзадачах можно делать не более 1000 запросов.

В третьей подзадаче обозначим k - максимальное количество выполненных запросов в каждом тесте. Если $k \leq 9$, вы получите 83 баллов. Иначе, вы получите $\lfloor \max(10, 83 \cdot (1 - \frac{\ln(k-6)}{7})) \rfloor$ баллов.

Код C++, который вычисляет количество баллов:

```
((k <= 9) ? 83: max(10, int(83 * (1 - log(k - 6.0) / 7))))
```