Problem D: Dungeon Crawler

In the famous video game Wizards & Wyvern, the protagonist finds himself in some kind of maze structure and has to find a way out. Each level is constructed with several caves, platforms, rooms or other kind of interesting spots that are connected with paths of different types. In each spot, there is a single distinguishable and unique artefact that can be activated by the hero. To escape from one level, the player has to activate the correct artefacts in the correct order. Riddles hidden inside the maze help him to solve the level.

You hate riddles and therefore obtained a map of a single level and its solution from the internet. You only need to find out whether this map represents the current level you are in.



Overview of the first sample case.

While wandering around from spot to spot, the player can see the artefact in his current spot as well as all paths leading away from it. The game implements 26 different types of paths such as paved roads (`P`), bridges (`B`) or dirt roads (`D`). In each spot all paths leading away are of different types.

This is an interactive problem. You do not know your initial position but you can always decide which path to follow. For each spot you visit, you are provided with the name of the artefact there and the types of paths leading away. Your task is to find out whether this level matches the one on your map.

Interaction Protocol

Your submission will be interacting with a special program called the *grader*. This means that the output of your submission is sent to the grader and the output of the grader is sent to the standard input of your submission. This interaction must follow a specific protocol:

The grader starts with the description of your map:

- One integer n ($2 \leq n \leq 1\,000$), the number of spots on the map.
- n lines, the ith of which contains an integer k followed by k pairs $t_1\ m_1, \ldots, t_k\ m_k$, describing the paths leading away from spot i. Each path is given by its type t_j (t_j is an uppercase letter) and destination m_j ($1 \leq m_j \leq n, m_j \neq i$).

All paths are bidirectional and appear twice in the input, once for each side. You can safely assume that all spots are reachable from any position in the maze.

Then, the game proceeds in turns. In each turn, the grader sends you the description of your current spot, in the following format:

- One line with a lowercase string a ($1 \leq |a| \leq 20$) and an uppercase string s ($1 \leq |s| \leq 26$), the name of the artefact you see at this spot and the description of the paths leading away. No two spots will have the same artefact. All letters in s are distinct and uppercase with no particular order. The order can vary when entering the same spot again.

Your submission must respond with one of the following:

- `W` c – you want to walk along the path of type c, where c is a valid uppercase character.
- `R` i – you are sure the level corresponds to the map. The integer i ($1 \leq i \leq n$) should be the number of the spot on the initial map that you believe you are currently in.
- `R ambiguous` – the map matches the level, but the matching is not unique.
- `R no` – the level does not correspond to your map.

Continue on next page ...

After every valid response starting with `W`, the grader gives you the description of the new spot you are currently in. You are not allowed to do more than $250\,000$ walks. After a response starting with `R`, the game ends. When you send an invalid response, the game ends and your submission will be judged as "Wrong Answer". After each command you send, you should *flush* the standard output to ensure that it is sent to the game. For example, you can use `fflush(stdout)` in C++, `System.out.flush()` in Java, and `sys.stdout.flush()` in Python.

Sample Input 1

```
3
2 D 2 B 3
2 P 3 D 1
2 P 2 B 1
fountain DB

obelisk PD

crystals PB

fountain DB

```

Sample Output 1

```


W D

W P

W B

R 1
```

Sample Input 2

```
3
2 D 2 B 3
2 P 3 D 1
2 P 2 B 1
obelisk PD

fountain LD

```

Sample Output 2

```


W D

R no
```

Sample Input 3

```
2
2 P 2 D 2
2 D 1 P 1
fountain PD

obelisk PD

fountain PD

```

Sample Output 3

```


W P

W D

R ambiguous
```

In the example interactions above, the output of the grader is on the left and one possible correct output of a submission is on the right.
The empty lines on both sides only serve to emphasise the chronological order of requests and answers. The grader will never output any empty lines.