

## Car Trouble

CPU TIME LIMIT

1 second

MEMORY LIMIT

1024 MB

DIFFICULTY

medium (3.9)

The city center of an unnamed Nordic university town consists of what was once a medieval city with narrow winding streets completely surrounded by a high wall protecting the city against Swedish invaders and other unwanted elements. The wall has since been removed and replaced by a system of interconnecting roads completely circumscribing the old part of the town. The roads inside still remains more or less the same as it was in the middle ages, which of course comes in conflict with modern requirements for accessibility by car, resulting in a maze of twisty little one-way streets, all alike, mixed with slightly wider two-way streets.



Making changes to the traffic routes in such a city can easily cause unexpected side-effects if you do not plan carefully ahead. The story goes that a prominent member of the city council once submitted a proposal to the council regarding extensive changes to how the traffic should be organized in the city center. The proposal did have the merit that it would be very easy to drive in to the central square, but it would unfortunately also be impossible to drive out again. The council member in question later went on to become minister of justice in the country under the parole that society should be harder on criminals – “it should be easy to go to jail, but difficult to get out again”.

To avoid mistakes as the one above, the city planners need you to develop a tool that can help them discover any traffic problems in the planning stage. The planners need to be alerted of two different situations. The first situation is that a street exists in the city center from which you cannot reach the surrounding, circular, system of roads, i.e., you are *trapped* inside the city. The second situation is that a street exists in the city that cannot be reached from the surrounding system of roads, i.e., it is *unreachable*.

## Input

The input consists of a description of how streets connect to each other and the surrounding circular road system. Each street (or a segment of a street) within the city center is represented by an arbitrary integer id number ( $0 < \text{id} < 1000$ ). The surrounding circular road system is represented by the special id number 0.

First line: An integer giving the number of streets (including the surrounding road system,  $0 < \text{streets} \leq 1000$ ).

The following lines: One line for each street (no particular order required and the surrounding road system is included) consisting of a number of integers. First an integer giving the id number of the street. Second, the number of (other) streets that can be reached from this street. Third, a sequence of distinct street id numbers indicating which streets can be reached from this street.

## Ouptut

One line for each street on which you would be trapped within the city consisting of the text "TRAPPED  $X$ " where " $X$ " is replaced by street id number in question.

Then, one line for each street within the city that is unreachable from the surrounding system of roads consisting of the text "UNREACHABLE  $X$ " where  $X$  should be replaced by the street id in question.

If no problems are found, i.e., you are not trapped in any street and every street is reachable, you should print a single line containing the text "NO PROBLEMS".

If multiple streets cause you to get trapped – or are unreachable – you should list them in the same order they were entered in the input (within respective category).



### Sample Input 1

```
6
0 1 1
1 1 2
2 3 1 3 0
3 0
4 2 5 0
5 1 4
```

### Sample Output 1

```
TRAPPED 3
UNREACHABLE 4
UNREACHABLE 5
```

### Sample Input 2

```
2
1 1 0
0 1 1
```

### Sample Output 2

```
NO PROBLEMS
```