Waterslide

Mouse Stofl likes to ride water slides. His favourite amusement park features a large network of slides connecting *N* small pools. At each pool, there is a ladder which can be climbed to reach some number of other slides. To prevent collisions, only one slide finishes per pool.

Mouse Stofl starts from one pool and would like to have as much fun as he can. He has most fun if he can ride as many distinct water slides as possible without ever walking between two pools. Can you help him determine such a maximum enjoyable sequence of slides? Note that Stofl is allowed to ride the same slide multiple times, although this wouldn’t increase his pleasure.

Subtask 1 (20 Points)

Help Stofl find the number of distinct slides on a maximum enjoyable sequence if he wants to start from a given pool.

Input

There are multiple test cases. The first line of the input contains the number *T*, which indicates the number of test cases. *T* test cases follow.

A single test case starts with two numbers *N* and *S* on separate lines (1 ≤ *S* ≤ *N*). The pools are numbered from 1 to *N*. *N* is the total number of pools and *S* indicates the starting pool. *N* lines follow, which describe the pools in order. Each of these lines contains the number of pools that can be reached from the described pool by going down a single slide, followed by the numbers of those pools. While each pool can have an arbitrary number of outgoing slides, it is guaranteed that there is exactly one slide leading to it.

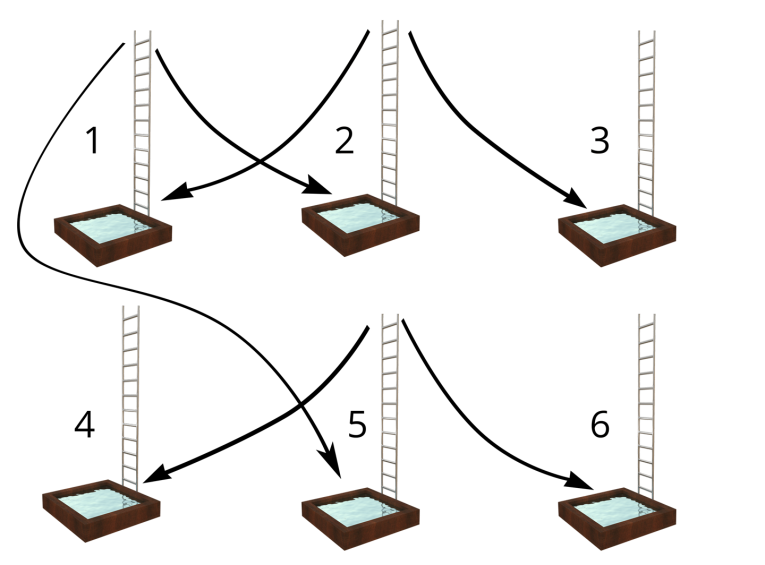
Output

For each test case you should output “Case #t:”, where *t* is the number of the case, starting from one. On the same line, one number should follow: The maximum number of distinct slides Stofl can visit without walking between two pools.

Constraints

* We have 1 ≤ *N* ≤ 1 000.

Example



*First test case of sample input visualized.*

Input:

3

6

1

2 2 5

2 1 3

0

0

2 4 6

0

6

3

1 2

1 3

1 4

1 5

1 6

1 1

6

2

1 1

1 2

1 4

2 3 5

1 6

0

Output:

Case #1: 4

Case #2: 6

Case #3: 1

Comment:

Case #1: The best sequence of platforms is either 1 → 2 → 1 → 5 → 6 or 1 → 2 → 1 → 5 → 4. Note that in this example, Stofl has to visit a platform twice to have the most fun.

Case #2: The best sequence of platforms is 3 → 4 → 5 → 6 → 1 → 2 → 3. Stofl can visit every platform because they are connected in a loop.

Case #3: The best sequence of platforms is 2 → 2. In this example, Stofl is trapped on a platform which is only connected to itself. Thus he can’t reach any of the other platforms.

Subtask 2 (20 Points)

The starting pool can now be chosen by Stofl himself. Your task is to find the number of distinct slides on a maximum enjoyable sequence starting from any of the *N* pools. In the input format, *S* is therefore omitted now.

Example

Input:

2

6

2 2 5

2 1 3

0

0

2 4 6

0

6

1 1

1 2

1 4

2 3 5

1 6

0

Output:

Case #1: 4

Case #2: 4

Comment:

Case #1: The best sequence of platforms is either 1 → 2 → 1 → 5 → 6 or 1 → 2 → 1 → 5 → 4.

Case #2: The best sequence of platforms is 4 → 3 → 4 → 5 → 6. Stofl can avoid the loops at platform one and two and instead chooses the interesting ride from platform three to six.

Constraints

* We have 1 ≤ *N* ≤ 1 000.

Subtask 3 (20 Points)

In anticipation of even more fun, Stofl decided to visit an even larger amusement park, featuring nearly one million unique pools. Stofl is still allowed to choose the starting pool.

Constraints

* We have 1 ≤ *N* ≤ 500 000.

Subtask 4 (40 Points)

This is a theoretical subtask, you do not have to provide any source code. Submit a proof that your solution to subtask 2 and/or 3 is correct (see [guide](http://2015.soi.ch/media/files/howto_en.pdf)) including an asymptotic analysis of the algorithm’s runtime and memory usage.