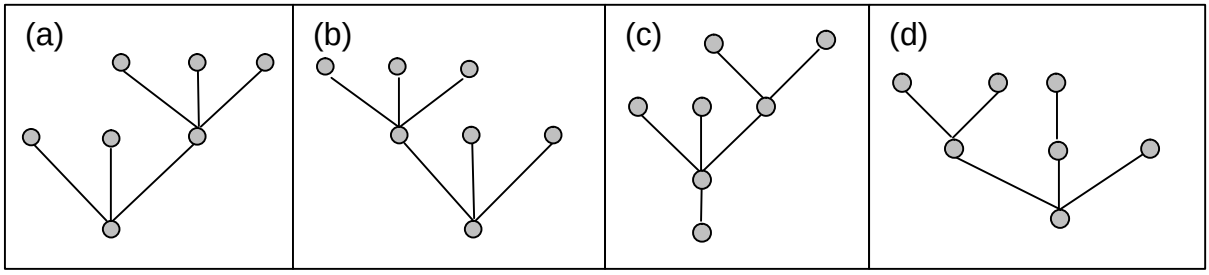
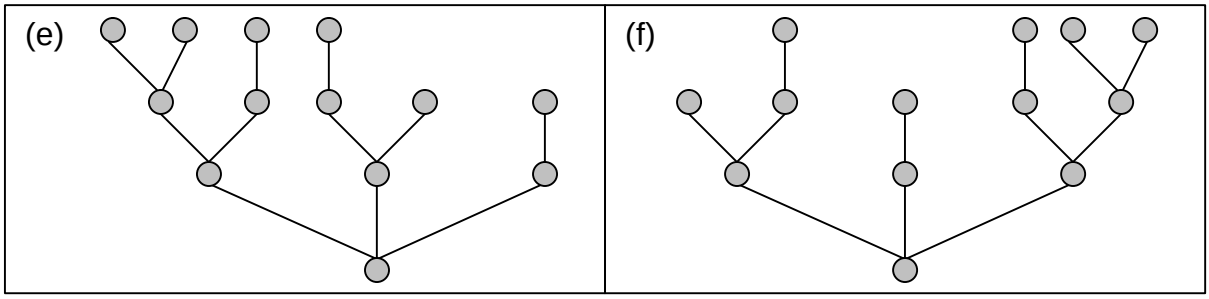
PROBLEM 9 - TRICKY TREES

The stream of tourists passing through the enchanted forest of Acmagorn is drying up because most people have seen it already and don't see the point of seeing it again. This is worrying the custodians of the forest because they rely on the money that tourists spend to manage the forest (and to pay their salaries). In an attempt to bring the tourists back again, they have devised a plan to make the forest look a bit different every day someone sees it. According to this plan, every night the trees randomly rearrange the order of their branches. Although the rearrangement doesn't change the structure of the trees, it can make their appearance quite different.

For example, images (a) and (b) below could show two possible rearrangements of the same tree. Of course, trees that are structurally different trees will always look different, no matter what rearrangement occurs. The trees in images (c) and (d) will never look the same as each other, because they are structurally different – nor will either (c) or (d) look the same as either (a) or (b).



As another example, the images (e) and (f) below could also show two possible rearrangements of the same tree.



This plan is a great success and the tourists are flocking back to see the constantly changing forest. Our friends Mary and Paul have been there several times and each time they've brought home many good pictures. They have now decided to put a bit of order in their collection of pictures, by identifying all matching tree images, i.e., all images that could represent the same tree. To achieve this result, they intend to associate a tag number with each image, such that images of trees are given the same tag if, and only if, they match. For examples, the images (a), (b), (c), (d) above could receive the tags 0, 0, 1, 2, in this order.

They have started by coding their collection of tree images. For each tree they labelled each node, in no particular order, with a distinct number in the range 0 to *k*-1, where *k* is the number of the nodes in that tree.

Your task is to write a program that will allow them to identify matching tree images.

Input Format [1]

Input consists of one or more scenarios, with each scenario consisting of a number of tree images. Each scenario starts with a line containing a single number, n ($1 \le n \le 100$) specifying the number of images in the scenario, followed by n lines containing image descriptions. Each image description line has at most 5,000 characters and consists of a number k ($1 \le k \le 1000$) specifying the number of nodes in the tree, followed by k numbers in the range -1 to k-1, specifying the parent of each node in turn, using -1 for the root node (which has no parent). The sequence of scenarios is terminated by an "empty scenario", i.e., a line consisting of single zero ('0').

Sample Input:

```
4
7 -1 0 0 6 6 6 0
7 -1 3 3 0 3 0 0
7 -1 0 1 1 6 6 1
7 -1 3 3 0 5 0 0
2
13 -1 0 0 0 1 1 2 2 3 4 4 5 6
13 -1 0 0 0 1 1 2 3 3 5 7 8 8
5
2 -1 0
3 -1 0 0
2 1 -1
3 -1 0 1
3 2 2 -1
0
```

Output Format

Output consists of one line for each scenario, beginning with a scenario sequence number (starting with 1), a colon ('`:`'), a space, and followed by a succession of tag numbers, one for each image, in input order, and separated by single spaces. Tag numbers are allocated successively in input image order, starting with 0, and increased by 1 at the first occurrence of an image not matching any of the previous images.

Sample Output:

```
1: 0 0 1 2
2: 0 0
3: 0 1 0 2 1
```

[1] Caveat: The input data for this program may contain lines up to 5000 characters long.