# ACM International Collegiate Programming Contest, Asia-Amritapuri Site, 2012

## Problem K: The Loyalty of the Orcs

*'I think that the enemy brought his own enemy with him,' answered Aragorn. 'These are Northern Orcs from far away. Among the slain are none of the great Orcs with the strange badges. There was a quarrel, I guess: it is no uncommon thing with these foul folk. Maybe there was some dispute about the road.'* - Aragorn describing the nature of Orcs.

Indeed, everyone knows that the Orcs are treacherous creatures who look for their own satisfaction and more often than not disregard the rules. The only way to keep them in line, is by maintaining the chain of command over a strict hierarchy among the ranks, wherein each Orc is responsible to his immediate superior all the way up to the army's head.

Further, the powers that be, have decided to have regular checks of their army's loyalties, just in case some Orc has been killed and all his juniors end up turning rogue!

There are N orcs, numbered 1 to N, wherein the lead orc is numbered 1.
Step 1. Randomly choose a fixed order in which to test Orcs' loyalties.
Step 2. Going in this order, you make a "roll-call" to check if the current Orc is alive or not.
Step 3. If the current Orc is dead, then he is marked as "deleted".

With this information, it is possible to tell which all Orcs will be loyal, and which won't be.
However, cunning Master Wormtongue suggests the following optimization:
In step 2, if any of the considered Orc's superiors (not necessarily immediate superior) is marked as deleted, then the roll-call is not made.

Now, given this algorithm and the hierarchy of the army, along with which Orcs are dead, what is the expected number of roll-calls (taken over all possible orderings in "Step 1") that you save by performing this optimization?

### Input (STDIN):

The first line contains T, the number of test cases.
The first line of each test case contains N, the number of orcs in the army. The next N-1 lines contain two space-separated integers u v, denoting that u is the immediate superior of v or vice-versa. The head of the army is the orc labelled 1. The next line contains m, the number of dead orcs. The next line contains m space separated integers, which are the labels of the dead orcs.

### Output (STDOUT):

Output one real number for each test case containing the expected number of roll-calls that you save. The results should be accurate within an error range of 10^-6.

### Constraints:

1 <= T <= 5
1 <= N <= 100,000
1 <= u, v <= N

u != v
The given set of u-v pairs form a valid chain of command. That means every Orc, except the Orc labeeled 1, has exactly one immediate superior.

Time Limit: 2s
Memory Limit: 64MB

## Sample Input:

```
2
2
1 2
1
1
2
1 2
1
2
```

## Sample Output:

```
0.5
0
```

## Notes/Explanation of Sample Input:

For the first test case, the Orc labelled 1 is dead. The two possible orderings are [1, 2] and [2, 1]. With the optimization, for the order [1, 2], we save the roll-call to 2. So, the total number of roll-calls without the optimization is 4, and with the optimization is 3. Expected number of roll-calls is therefore, (4 - 3) / 2 = 0.5 .
For the second test case, the Orc labeled 2 is dead. Since he does not have any sub-ordinates, the optimization does not have any effect.