F
Jumping monkey

You are a hunter chasing a monkey in the forest, trying to shoot it down with your all-powerful automatic machine gun. The monkey is hiding somewhere behind the branches of one of the trees, out of your sight. You can aim at one of the trees and shoot; your bullets are capable of going through the branches and killing the monkey instantly if it happens to be in that tree. If it isn't, the monkey takes advantage of the time it takes you to reload and takes a leap into a neighbouring tree without you noticing. It never stays in the same place after a shot. You would like to find out whether there is an strategy that allows you to capture the monkey for sure, irrespective of its initial location and subsequent jumps. If so, you need to determine the shortest sequence of shots guaranteeing this.

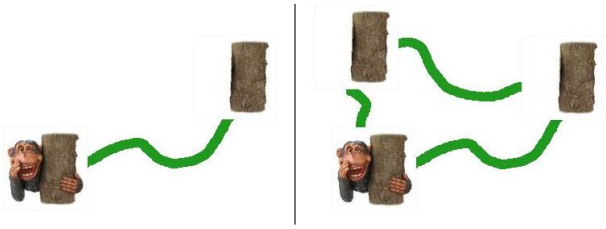


Figure 2

As an example, consider the situation in which there are only two neighboring trees in the forest (left hand side of Figure 2). It is then possible to make sure you capture the monkey by shooting twice at the same tree. Your first shot succeeds if the monkey happened to be there in the first place. Otherwise, the monkey was behind the other tree and it will necessarily have moved when you shoot for the second time.

However, depending on the shape of the forest it may not possible for you to ensure victory. One example of this is if there are three trees, all connected to one another (right hand side of Figure 2). No matter where you aim at, there are always two possible locations for the monkey at any given moment. (Note that here we are concerned with the worst-case scenario where the monkey may consistently guess your next target tree).

Input

The input consists of several test cases, separated by single blank lines. Each test case begins with a line containing two integers n and m ($1 \leq n \leq 21$); n is the number of trees in the forest, and m is the number of adjacency relations between trees. Each of the following m lines contains two distinct integers between 0 and $n-1$ (inclusive), the identifiers of the trees in an adjacent pair. The order of both trees within a pair carries no meaning, and no pair appears more than once. You may further assume that no tree is adjacent to itself, and there is always a path between any two trees in the forest.

The test cases will finish with a line containing only two zeros (also preceded with a blank line).

Output

Print a line for each test case. The line should contain the single word `Impossible` if the task is impossible. Otherwise, it must contain the shortest sequence of shots with the required property, in the format L: $V_1 V_2 \ldots V_L$, where L is the length of the sequence, and $V_1, V_2, \ldots, V_L$ are space-separated integers containing the identifiers of the trees to shoot at in the right order. If several shortest sequences exist, print the lexicographically smallest one. (A sequence is smaller than another in lexicographic order if the first element on which they differ is smaller in the first one).

Sample Input

```
2 1
0 1

3 3
0 1
1 2
2 0

4 3
0 1
2 3
1 3

0 0
```

Sample Output

```
2: 0 0
Impossible
4: 1 3 3 1
```