Problem H: Trees

Source: `trees.{c,cpp,java}`

A graph consists of a set of vertices and edges between pairs of vertices. Two vertices are connected if there is a path (subset of edges) leading from one vertex to another, and a connected component is a maximal subset of vertices that are all connected to each other. A graph consists of one or more connected components.

A tree is a connected component without cycles, but it can also be characterized in other ways. For example, a tree consisting of n vertices has exactly n-1 edges. Also, there is a unique path connecting any pair of vertices in a tree.

Given a graph, report the number of connected components that are also trees.

Input

The input consists of a number of cases. Each case starts with two non-negative integers **`n`** and **`m`**, satisfying **`n ≤ 500`** and **`m ≤ n(n-1)/2`**. This is followed by m lines, each containing two integers specifying the two distinct vertices connected by an edge. No edge will be specified twice (or given again in a different order). The vertices are labelled **`1`** to **`n`**. The end of input is indicated by a line containing **`n = m = 0`**.

Output

For each case, print one of the following lines depending on how many different connected components are trees (T > 1 below):

```
Case x: A forest of T trees.
Case x: There is one tree.
Case x: No trees.
```

`x` is the case number (starting from 1).

Sample Input

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

Sample Output

```
Case 1: A forest of 3 trees.
Case 2: There is one tree.
Case 3: No trees.
```