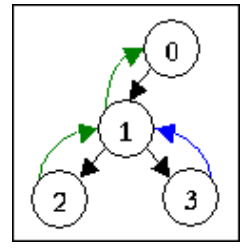
You are given a tree (a connected graph with no cycles), and the edges of the tree which are for some reason directed; your task is to add **minimum** number of special paths in the tree such that it's possible to go from any node to another. The rules for the special paths are noted below:

1. A special path consists of some continuous edges (from the tree) and nodes.
2. In a special path, the edges should be in opposite directions as they are in the tree.
3. A node or an edge can be visited at most once in a special path.
4. Multiple special paths may have common nodes or edges.

For example, in the picture below, a tree is drawn, the black arrows represent the edges and their directions, circles represent nodes. Then we need two special paths. One path is **2-1-0** (green arrow), another is **3-1** (blue arrow). Instead of the path **3-1** we can add **3-1-0**. You cannot add a path like **1-3** or **0-1-2** because of rule 2. You cannot add **0-2** or **2-3-0** because of rule 1.



Input

Input starts with an integer T (≤ 30), denoting the number of test cases.

Each case starts with a line containing an integer N ($2 \leq N \leq 20000$), where N denotes the number of nodes. The nodes are numbered from 0 to $N-1$. Each of the next $N-1$ lines contains two integers u v ($0 \leq u, v < N$, $u \neq v$) meaning that there is an edge from u to v.

Output

For each case, print the case number and the minimum number of special paths required such that it's possible to go from any node to another.

Sample Input

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

Sample Output

```
Case 1: 2
Case 2: 3
```