There is a rooted tree, some edges are undirected, while others are directed. We want to change maximum number of undirected edges to directed edges, but we don't want the length of the *longest* directed chain to be increased. Note that undirected edges are not allowed in a directed chain.

For example, if we have a "linear graph" $1 \to 2 \to 3 - 4$, we cannot change $3 - 4$ into $3 \to 4$ because the previous longest directed chain $(1 \to 2 \to 3)$ would be extended to $1 \to 2 \to 3 \to 4$. However, we can change $3 - 4$ into $3 \leftarrow 4$ without extending the longest chain.

Input

There will be at most 1200 test cases. Each test case contains several lines. In each line, the first integer u is the node that is described, followed by its sons, terminated by a zero. The direction of an edge can be from father to son, and can also be from son to father. If the edge is from father to son, then we put a letter '`d`' after that son (meaning that it is a downward edge). If the edge is from son to father, then we put a letter '`u`' after that son (meaning that it is an upward edge). If the edge is undirected then we do not put any letter after the son. Nodes are numbered 1 to n $(2 \leq n \leq 300)$ from top to down, left to right (so the first line is always root). Leaves are not given in the input. The test case ends with $u = 0$. **Most test cases have very few nodes.**

Output

For each test case, print the case number, the number of changed edges, followed by the changed edges with directions. Each directed edge is formatted as (i, c), which means the i-th undirected edge is changed to direction c. The undirected edges are numbered from 1, in the same order they appear in the input.

If there are several optimal solutions, print the lexicographically smallest one. When comparing two changes (i_1, c_1) and (i_2, c_2) lexicographically, we first compare i_1 and i_2, if i_1 and i_2 are equal, we compare c_1 and c_2. For example $(2, u) < (11, d)$, and $(3, d) < (3, u)$.

Sample Input

```
1 2d 3 0
3 4 5 0
0
1 2d 0
2 3d 0
3 4 0
0
1 2d 0
2 3 0
3 4u 0
0
1 2u 3 0
3 4u 5 0
0
```

Sample Output

```
Case 1: 3 (1,d) (2,u) (3,u)
Case 2: 1 (1,u)
Case 3: 0
Case 4: 1 (2,u)
```