UVa Online Judge

12476 Fun with Binary Tree

Once upon a time there was a **Froogrammer** (a very special type of computer programmer), who fall in love with binary trees. A **binary tree** is a tree data structure in which each node has at most two child nodes, usually distinguished as "left" and "right". In this problem you are requested to solve one of his favorite games played in a binary tree. The game is played in the following way:

1. You will start the game from the root of a binary tree.
2. Then you will be given a set of instructions containing letter '`L`' or '`R`'.
3. After processing i-th instruction you will process $(i + 1)$-th instruction.
4. After receiving an instruction you can either accept the instruction or reject it.
5. If you accept an instruction '`L`' it will move you to the left child of the current node and if it is '`R`' than it will move you to the right child of the current node.
6. If there is no left child for an instruction '`L`' or right child for an instruction '`R`', that instruction will be considered as rejected. Thus you will remain on the same node you are in.
7. If you reject two consecutive instructions you will lose the game.
8. If you do not lose while processing all the instructions, you will be considered as the winner.

Given the description of the binary tree and the set of instructions S you need to process. You are to write a program that will output '`Yes`' if you can win the game and '`No`' if you cannot win the game.

Input

Input starts with an integer T ($T \leq 200$), denoting the number of test cases. Each case starts with a number N, total number of nodes in the tree ($1 \leq N \leq 20000$). Next N lines contains two numbers L, R ($2 \leq L, R \leq N$ or $L = -1$ or $R = -1$). (L, R) in i-th line represents the **1-based index** of node that is the left child and right child of i-th node. If the value of L or R is '`-1`' than there is no node connected in that corresponding end. Next line will contain a string S ($1 \leq length(S) \leq 40000$).

Node 1 is the root node of the tree.

Output

For each case, print the case number in the following format '`Case` x`:` y', where x is the case number and y is '`Yes`' or '`No`' indicating whether it is possible for you to go through all the instructions following the rules described in the problem statement. See the sample input/output for more details.

Sample Input

```
2
5
2 3
4 5
-1 -1
-1 -1
-1 -1
```

```
LRLR
3
2 3
-1 -1
-1 -1
LL
```

Sample Output

```
Case 1: Yes
Case 2: Yes
```