D2. Constrained Tree

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You need to find a binary tree of size n that satisfies a given set of c constraints. Suppose that the nodes of the unknown binary tree are labeled using a *pre-order* traversal starting with 1. For the i-th constraint you are given two labels, a_i and b_i and a direction, left or right. In case of left direction, b_i is an element of the subtree rooted at a_i's left child. Similarly in the case of right direction b_i is an element of the subtree rooted at a_i's right child.

Input

The first line of input contains two integers n and c. The next c lines contain 2 integers a_i, b_i ($1 \le a_i, b_i \le n$) and either "`LEFT`" or "`RIGHT`" denoting whether b is in the subtree rooted at a_i's left child or in the subtree rooted at a_i's right child.

The problem consists of multiple subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem D1 (9 points), the constraints $1 \le n \le 100$, $1 \le c \le 50$ will hold.
- In subproblem D2 (8 points), the constraints $1 \le n \le 1000000$, $1 \le c \le 100000$ will hold.

Output

Output will be on a single line.

Any binary tree that satisfies the constraints will be accepted. The tree's nodes should be printed out as n space separated labels representing an *in-order* traversal, using the *pre-order* numbers as labels of vertices.

If there are no trees that satisfy the constraints, print "`IMPOSSIBLE`" (without quotes).

Examples

input
3 2 1 2 LEFT 1 3 RIGHT
output
2 1 3

input
3 2 1 2 RIGHT 1 3 LEFT
output
IMPOSSIBLE

Note

Consider the first sample test. We need to find a tree with 3 nodes that satisfies the following two constraints. The node labeled 2 with pre-order traversal should be in the left subtree of the node labeled 1 with pre-order traversal; the node labeled 3 with pre-order traversal should be in the right subtree of the node labeled 1. There is only one tree with three nodes that satisfies these constraints and its in-order traversal is $(2, 1, 3)$.

Pre-order is the "root – left subtree – right subtree" order. *In-order* is the "left subtree – root – right subtree" order.

For other information regarding *in-order* and *pre-order*, see http://en.wikipedia.org/wiki/Tree_traversal .