The Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

You are given an infinite binary tree. The tree has the root and infinitely many vertices, each vertex has left son and right son, each vertex except the root has a parent.

Every vertex can be painted in one of c colors or be colorless. Initially, all vertices are colorless.

You need to process two types of requests:

1. `color(`u`, `x`)` You are given a vertex u, paint vertex u with color x, and then call `color(`L`, `$(x+1) \bmod c$`)` for its left son L and `color(`R`, `$(x-1+c) \bmod c$`)` for its right son R. Note that this operation repaints the entire (infinite) set of vertices in the subtree of the vertex u. Here mod is the operation of taking a division remainder. If the vertex has already been painted, then its color changes to the new one.
2. Given a vertex, output its current color.

Input

The first line contains two integers q, c — the number of queries and colors, respectively ($1 \le q \le 5 \cdot 10^5$, $1 \le c \le 10^9$). This is followed by q queries, each starts with an integer t_i — the type of the i-th query.

If $t_i = 1$, then an integer x ($0 \le x \le c-1$) is given in the line, the color that the vertex of the query u should be colored. The next line describes the path to the vertex u in the form of a non-empty string s_i consisting of the characters '`L`' and '`R`'. This string specifies the path from the root of the tree to the vertex u, where '`L`' denotes going to the left son, and '`R`' — going to the right son.

If $t_i = 2$, then the next line specifies the path to the vertex whose color should be output, described similarly to the previous query.

It is guaranteed that the sum of the lengths of the paths to all the vertices of the queries does not exceed $5 \cdot 10^5$.

Output

For each request of the second type print the color of the corresponding vertex on a separate line. If the vertex is colorless, print -1.

Example

standard input	standard output
6 3 1 2 L 2 LL 1 0 LL 2 LLR 2 LR 2 R	0 2 1 -1