Problem G. Galilei

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Galileo Galilei was a famous astronomer. He was the first to use the telescope to look at the sky. He discovered many celestial bodies, and it appeared that tracking all of them in mind was quite hard... There is a rumor that he invented a tracking system for his inventions and discoveries called *Galilei's inventions tracker*, or simply *git*. It allowed to save many versions of various information. For example, if you would like to look at some parameters of the star you observed a few months ago, you could just query this system. Your task is to reproduce the behavior of a simplified version of this tracker. A brief description follows.

A *commit* is a representation of the entire system state. Commits are named with sequential integers starting from 0. Initially, there is only one commit number 0. Commits are ordered as a rooted tree with root at 0: every commit except the root has exactly one parent. Commit a is called an *ancestor* of b if it is possible to come from b to a going up to parent several times. One commit is always selected as the *working copy*. Several operations are supported:

- `co` x, "checkout": select commit x as the working copy.
- `ci`, "commit": create a new commit whose parent is the current working copy and select it as the working copy. The new commit gets the lowest possible id.
- `re` x, "rebase": let your working copy be y. Find a commit t which is the lowest common ancestor of x and y. Copy the path from t to y (not including t) on top of x. New commits are numbered as follows: at first, the closest one to x gets the lowest possible number, then the second commit gets the next lowest possible number, and so on. The last copied commit becomes your working copy.

  If x is an ancestor of y, nothing happens and the path is not copied. If y is an ancestor of x, then the path is not copied and x becomes the working copy.

Your task is to implement all these operations.

Input

The first line of input contains one integer n ($0 \leqslant n \leqslant 10^5$). Each of the next n lines contains one of the following commands:

- `co x`: checkout at commit x
- `ci`: commit
- `re x`: rebase over commit x

It is guaranteed that the commit number is always valid. Additionally, it is guaranteed than after all operations, there will be no more than 10^{18} commits.

Output

After every rebase operation, print the number of the working copy commit after this operation.

Example

standard input	standard output
7 ci ci co 0 ci ci re 1 re 4	6 9