Linked List

1992 is a different time. It's before the time that the entire Windows division at Microsoft could only afford to get one computer with a staggering 96 MB of RAM to test Windows 95 on. The average computer back then had less than 4 MB of RAM. It's less than what modern Java requires to start up!

To secure your release back to the modern age, you came across me. I would gladly let you back to the future if only you do me one little favour. I need to implement a linked list that can store N elements, up to $1\,000\,000$, each storing integers ranging from -8×10^{12} to 8×10^{12}. But wait, I am not very rich! I can only afford 10 MB of RAM (remember it's 1992) to run the program.

Your linked list must support these operations:

- `<` select the previous element (rewinding).
- `>` select the next element (advancing).
- `=` changes the current element to a value (updating).
- `+` creates a new element with value, shifting the current element back (inserting). The pointer will point at the inserted element.
- `-` removes the current element, shifts pointer to the next element (deleting).
- `!` prints the value of the current element (printing).

Note that your linked list starts empty, and it is possible to delete the last element in the list. In both cases, the pointer goes beyond the end of the list. In no case will the past-the-end "element" be updated, deleted, or queried, nor will there be attempts to advance past the past-the-end "element", or rewind past the first element.

Input Specification

The first line contains the integer M, the number of operations to perform, up to $10\,000\,000$.

The next M lines will contain one of the operation labels, `<`, `>`, `=`, `+`, `-`, or `!`. If the operation takes a value, in case of `=` or `+`, then the label is followed by a space followed by an integer.

Output Specification

For every `!` operation, print the value at the linked list pointer.

Subtasks

There are four subtasks:

1. $N \leq 1\,000$, $M \leq 10\,000$, 10% of points
2. $N \leq 10\,000$, $M \leq 100\,000$, 20% of points
3. $N \leq 100\,000$, $M \leq 1\,000\,000$, 30% of points

4. $N \leq 1\,000\,000$, $M \leq 10\,000\,000$, 40% of points

Sample Input

```
10
+ 100
+ 200
>
!
<
!
-
!
= 300
!
```

Sample Output

```
100
200
100
300
```

Explanation

1. Insert 100 to list: $[100]$, pointer at `0`.
2. Insert 200 to list: $[200, 100]$, pointer at `0`.
3. Advance pointer: pointer at `1`.
4. Print: prints 100 which is at index `1` of list $[200, 100]$.
5. Rewind pointer: pointer at `0`.
6. Print: prints 200 which is at index `0` of list $[200, 100]$.
7. Delete element: $[100]$, pointer still at `0`.
8. Print: prints 100 which is at index `0` of list $[100]$.
9. Update to 300: $[300]$, pointer at `0`.
10. Print: prints 300 which is at index `0` of list $[300]$.

Warning

It might be unwise to use languages that use a lot of memory, and even more unwise to use languages that won't start in the memory limit. Your mileage may vary. Problem is guaranteed to be solvable in any language that allows direct memory access, such as C, or C++.