Circular Buffer

Time Limit: 1 second
Memory limit: 13 MB

Your task is to implement a circular buffer of **size N**. A circular buffer is basically an array of fixed **size N**, but when more elements than **N** are added to it over time, they overwrite the oldest values.

Initially it is empty.

Your program will be given a sequence of the following **4 commands** to update/print its contents:

A n: Append the following n lines to the buffer. If the buffer is full then replace the older entries. Each of the following n lines will contain strings (without any spaces / tabs) of maximum length 100. They will be single words with alphabets and numbers.

R n: Remove first n elements of the buffer. These n elements are the ones that were added earliest among the current elements.

L : List the elements of buffer in order of their inserting time.

Q Quit : Stop scanning for input and exit your program. Your task is to execute these commands on circular buffer.

Input format:
First line of input contains N, the size of the buffer, followed by a sequence of these commands:
A n append the following n lines to the buffer
R n remove first n elements of the buffer
L list the elements of buffer in order of their inserting time.
Q quit

Output format:
Whenever L command appears in the input, print the elements of buffer in order of their inserting time. Element that was added first should appear first.

Sample Input:

```
10
A 3
str1
str2
```

```
str3
L
R 1
L
A 1
str4
R 2
L
Q
```

Sample Output:

```
str1
str2
str3
str2
str3
str4
```

Constraints:
The following conditions will be satisfied by the test cases, so you need not bother:
1. 0 <= N <= 10000
2. For append command, n can be any non-negative integer.
3. The length of the strings to be added will be at least 1 and at most 100.
4. For remove command, n <= Number of elements currently presents in circular buffer.
5. 1 <= Total number of commands <= 5000.