Alternating List

Objective
The objective of this problem is to test the students' understanding on **Singly Linked List**.

Problem Description
A list is *alternating* if its elements alternate between positive and negative. For example:

Alternating lists	***Non-alternating lists***
[-1, 10, -2, 3, -5]	[-4, 2, -3, -7, 6]
[1, -2, 3, -4]	[4, 3]
[3]	[8, -8, 3, 4, -1]
[ ] (empty list)	[-2, -4, -6]

You are given a linked list (original) and **Q** updates. For each update, check if the updated linked list is *alternating*. **It is guaranteed that the elements in the linked list before and after update will not be 0.**

There are 3 valid update operations, as explained below. Note that the indices of a list begin with one, not zero.

1. **M <index> <size>**: Move a block of elements of length <size> starting at index <index> to the back of the list. For example, let the current linked list be [1, 3, 5, 4, 2, 6]. The operation "**M 2 3**" moves **[3, 5, 4]** to the end of the linked list. The updated linked list is [1, 2, 6, 3, 5, 4].
2. **R <index> <size>**: Remove a block of elements of length <size> starting at index <index> from the linked list. For example, let the current linked list be [1, 3, 5, 4, 6, 7]. The operation "**R 2 4**" removes **[3, 5, 4, 6]** from the list. After performing the operation, we will have [1, 7].
3. **A <index> <size> <value>**: Add the elements between index **<index>** and **<index + size – 1>** (inclusive) with **<value>**. For example, let the current linked list be [1, -3, -5, 6, 10]. The operation "**A 1 3 4**" adds **[-3, -5, 6]** with value **4**. The updated linked list is [1, **1, -1, 10**, 10].

*** It is guaranteed that <size> will not cause the index to go beyond the size of the Linked List (***i.e. size + index* – 1 ≤ *LinkedList.size***). For every operation, <size> is positive.**

Input
The first line of the input contains 2 integers **N (1 <= N <= 100)** and **Q (1 <= Q <= 100)**, where **N** is the size of the original linked list and **Q** the number of updates. The next line contains **N** integers, denoting the elements in the original linked list. The next **Q** lines are the update operations.

Output
Print "**YES**" if the updated linked list is alternating, otherwise print "**NO**".

Sample Input

```
4 4
1 -2 3 -4
M 1 3
A 1 1 14
R 2 2
A 2 1 -11
```

Sample Output

```
YES
NO
NO
YES
```

Explanation

N = 4 and Q = 4.
Update 1: [1, -2, 3, -4] →[-4, 1, -2, 3] and [-4, 1, -2, 3] is alternating.
Update 2: [-4, 1, -2, 3] →[10, 1, -2, 3] and [10, 1, -2, 3] is **NOT** alternating.
Update 3: [10, 1, -2, 3] →[10, 3] and [10, 3] is **NOT** alternating.
Update 4: [10, 3] →[10, -8] is alternating.