Problem F. Ursa Minor

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes



Pixel Art: Ambition, Sulfox, and Charles

Sulfox the fennec fox celebrated his 20th birthday with great joy as he received a special gift — a pixel game named World Rebuilder. In this game, he plays the role of an interstellar explorer who journeys toward Polaris, the brightest star in the constellation Ursa Minor, and terraforms planets along the way.

A game level can be determined by a target sequence $a_1, a_2, \ldots, a_n$ and a tool sequence $b_1, b_2, \ldots, b_m$. At the beginning of this level, there are n (the length of the target sequence) continents with initial altitudes of 0, arranged circularly around the equator of the planet where Sulfox landed.

The "Batch Edit" tool allows Sulfox to modify the altitudes of the continents simultaneously. Whenever he uses it, he should firstly select a number k from the tool sequence and choose an arbitrary **real number** x. Then, he can select exactly k **consecutive** continents and add x to each of their altitudes. The level is cleared when the altitudes of the n continents, starting from some continent in some direction, are $a_1, a_2, \ldots, a_n$ in order.

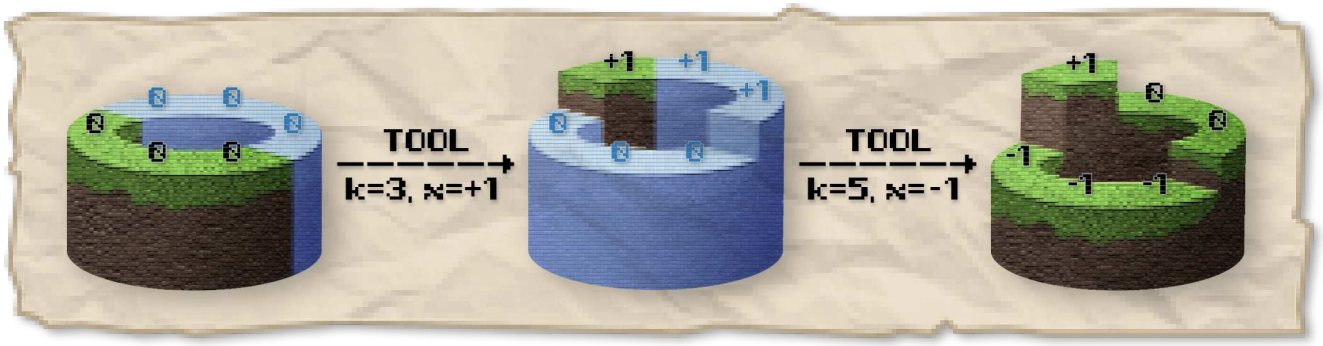


Figure: How to adjust the altitudes with the "Batch Edit" tool
(The selected continents are highlighted)

In fact, the level designer for this game is quite a lazy guy, for he generated each level by selecting a contiguous subsequence from a global target sequence $A_1, A_2, \ldots, A_N$ and a global tool sequence $B_1, B_2, \ldots, B_M$ respectively. What's worse, he didn't even bother to check whether each level is possible to clear, leaving Sulfox stuck at some levels marked as "easy"!

Luckily, as a video game, it certainly has multiple version updates to fix bugs (or maybe introduce new bugs, who knows?). Now given the global target sequence $A_1, A_2, \ldots, A_N$ and the global tool sequence $B_1, B_2, \ldots, B_M$ in the initial version, you need to handle Q events of two kinds:

- Game Update: Change the value of A_p to v in a new version.

- Level Query: Query if Sulfox can clear the level determined by target sequence $A_l, A_{l+1}, \ldots, A_r$ and tool sequence $B_s, B_{s+1}, \ldots, B_t$ in the latest version if he can use the tool any number of times.

Input

The first line contains three integers N, M, and Q ($1 \le N, M, Q \le 2 \times 10^5$), denoting the length of the global target sequence, the length of the global tool sequence, and the number of events, respectively.

The second line contains N integers $A_1, A_2, \ldots, A_N$ ($0 \le A_i \le 10^9$), denoting the global target sequence in the initial version.

The third line contains M integers $B_1, B_2, \ldots, B_M$ ($1 \le B_i \le N$), denoting the global tool sequence.

Then, each of the next Q lines contains an event in one of the following two formats:

- `U p v` ($1 \le p \le N$, $0 \le v \le 10^9$), denoting an update that changes the value of A_p to v in a new version.
- `Q l r s t` ($1 \le l \le r \le N$, $1 \le s \le t \le M$), denoting a query of if Sulfox can clear the level determined by target sequence $A_l, A_{l+1}, \ldots, A_r$ and tool sequence $B_s, B_{s+1}, \ldots, B_t$ in the latest version. It is guaranteed that each value of $B_s, B_{s+1}, \ldots, B_t$ does not exceed $r - l + 1$.

Output

For each query, output "`Yes`" in one line if Sulfox can clear the level determined by the given target sequence and the given tool sequence in the latest version, or otherwise, output "`No`" in one line.

Example

standard input	standard output
6 4 5 1 1 4 5 1 4 3 3 2 4 Q 1 5 1 2 Q 2 5 3 4 U 5 2 Q 1 6 1 2 Q 2 5 3 4	Yes No No Yes