Problem K
Optimized Cheating

Time Limit: 1 second

Bob's favorite game just released a limited-time item for sale that will boost his game character's power significantly. However, there is not enough time for Bob to acquire sufficient in-game currency to purchase the item. Bob thus decides to resort to a cheat tool he found online to modify his in-game currency value.

The cheat tool will take two values x and y specified by Bob and overwrites all memory slots that contain the value x with the value y. Bob does not want to use the cheat tool unsafely. He does not want to modify any memory slots other than the one that stores his currency value and cause the game to crash. Therefore, Bob needs to make sure that his currency value does not have any duplicate in the game's memory space before running the cheat tool. Bob can use a set of operations provided by the game to modify his currency value, such as doing missions, purchasing items, and so on. Those operations can add, subtract, multiply or divide his currency value by a constant. All those operations will only change Bob's currency value, and they will not affect any other memory slots. Bob cannot use an operation that would cause his currency value to become negative (e.g. buying an item that costs more than his available currency).

Bob knows that the memory space of the game can be represented as an array consisting of n integers. He also knows the location of the memory slot within this array that stores his currency value. However, Bob does not know how to modify his currency value as quickly as possible in order to use the cheat tool safely. Can you help him?

Input

The first line of input contains three integers n, m, and k ($1 \leq n \leq 10^4, 1 \leq m \leq 1\,000, 1 \leq k \leq n$), where n is the number of memory slots in the game's memory space, m is the number of operations that Bob can use, and k is the 1-based index of the memory slot that stores Bob's in-game currency value in the game's memory space.

The next n lines each contain a single integer between 1 and 10^9, giving the values stored in the game's memory space in order.

The next m lines each contain a single character p (+, −, $\star$, or /) and an integer v ($1 \leq v \leq 10^9$) that describe one operation that Bob can use to change his currency value. If Bob's current currency value is u, then after applying the operation his currency value will become $u\ \mathrm{p}\ v$. For instance, applying an operation + 3 will increase Bob's currency value by 3. Divisions are integer divisions (e.g., 7 / 3 = 2). An operation cannot be applied if it would result in a negative currency value. Each operation can be applied multiple times (including zero).

Output

If it is possible for Bob to make to his in-game currency value unique in the game's memory space, output a single integer t on the first line, the minimum number of operations that Bob must apply. Then output t lines that each have an integer denoting the 1-based index of the operation that Bob should apply in order. If there are multiple ways to apply the operations, you may output any of them.

If it is impossible for Bob to make his currency value unique in the game's memory space, output -1.

Sample Input 1	Sample Output 1
5 3 4 2 1 3 2 3 - 1 * 2 + 3	1 2