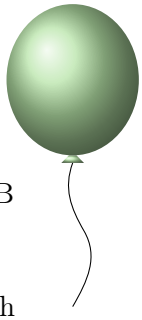
G November Rain



Time limit: 1.0s
Memory limit: 1024MB

You are conducting a grand, evolving symphony. The ensemble consists of musicians, where each musician plays a specific harmonic frequency (represented by a non-negative integer). Initially, the stage is completely empty. Over n sequential steps, exactly one action is taken to change the arrangement. For each step $i = 1, 2, \ldots, n$, an operation is performed:

- If the operation is `+` (Enter): A new musician joins the ensemble. You must decide the exact frequency b_i they will play.
- If the operation is `-` (Exit): A musician leaves the stage. You must choose the frequency b_i of a musician currently performing and have exactly one of them stop playing.

At every step, the performance is anchored by the "Phantom Note." Because of the unique acoustics of the symphony, the Phantom Note is never actually played by anyone on stage. Instead, its pitch is always determined by the *lowest frequency that is currently missing from the performance.*[1]

After the i-th action, it is required that the Phantom Note must resonate at exactly a_i.

Your task is to determine whether a valid sequence of chosen frequencies $b_1, b_2, \ldots, b_n$ exists that perfectly orchestrates the required Phantom Note progression at every step, and to construct one such sequence if it does.

Input

This problem has multiple test cases. The first line of input contains a single integer t ($1 \le t \le 3 \cdot 10^5$), representing the number of test cases.

For each test case, each performance is described over three lines:

- The first line contains a single integer n ($1 \le n \le 5000$), representing the total number of sequential steps in the performance.
- The second line contains a string op of length n, consisting exclusively of the characters `+` and `-`. The character op_i dictates the nature of the i-th action: `+` signifies a musician Entering, and `-` signifies a musician Exiting.
- The third line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le n$), representing the exact required pitch of the Phantom Note after the i-th action.

It is strictly guaranteed that the sum of n^2 over all performances does not exceed 5000^2.

Output

For each performance, output your result as follows:

[1] The pitch of the Phantom Note is mathematically defined as the `mex` (minimum excluded value). Let S be a multiset of non-negative integers representing the collection of frequencies currently being played by the ensemble. The minimum excluded value, denoted as $\text{mex}(S)$, is the smallest non-negative integer x such that $x \notin S$.

If it is impossible to orchestrate the required Phantom Note progression, print a single line containing the word `NO`.

Otherwise, print two lines:

- The first line must contain the word `YES`;
- The second line must contain n non-negative integers $b_1, b_2, \ldots, b_n$, representing the specific frequency played by the entering or exiting musician at each corresponding step.

Each frequency b_i must perfectly satisfy the acoustic constraints and operational rules described in the problem statement. You are permitted to output any valid non-negative integer, provided it is representable by a standard signed 64-bit integer.

If there are multiple valid sequences of frequencies that satisfy the performance, you may print any one of them.

Samples

Sample input 1	Sample output 1
4 2 ++ 0 2 3 +++ 0 1 3 3 +-+ 1 0 2 6 ++-+-+ 0 2 0 0 0 1	YES 1 0 YES 2 0 1 NO YES 1 0 0 7 1 0

Explanation of sample 1.
There are four test cases in sample 1.

- In the first test case, inserting 1 keeps the mex (Phantom Note) 0, then inserting 0 makes the mex become 2.
- In the second test case, inserting $2, 0, 1$ makes the mex sequence $0, 1, 3$.
- In the third test case, mex 1 after the first operation forces insertion of 0; after erasing it, one more insertion cannot make mex 2, so the answer is `NO`.
- In the fourth test case, the printed values make the multiset evolve with mex sequence $0, 2, 0, 0, 0, 1$.