BRCKTS - Brackets

We will call a **bracket word** any word constructed out of two sorts of characters: the opening bracket "(" and the closing bracket ")". Among these words we will distinguish **correct bracket expressions**. These are such bracket words in which the brackets can be matched into pairs such that

- every pair consists of an opening bracket and a closing bracket appearing further in the bracket word
- for every pair the part of the word between the brackets of this pair has equal number of opening and closing brackets

On a bracket word one can do the following operations:
- **replacement** -- changes the i-th bracket into the opposite one
- **check** -- if the word is a correct bracket expression

Task

Write a program which

- reads (from standard input) the bracket word and the sequence of operations performed,
- for every check operation determines if the current bracket word is a correct bracket expression,
- writes out the outcome (to standard output).

Input

Ten test cases (given one under another, you have to process all!). Each of the test cases is a series of lines. The first line of a test consists of a single number n $(1<=n<=30000)$ denoting the length of the bracket word. The second line consists of n brackets, not separated by any spaces. The third line consists of a single number m -- the number of operations. Each of the following m lines carries a number k denoting the operation performed. $k=0$ denotes the check operation, $k>0$ denotes replacement of k-th bracket by the opposite.

Output

For every test case your program should print a line:
Test i:
where i is replaced by the number of the test and in the following lines, for every check operation in the i-th test your program should print a line with the word YES, if the current bracket word is a correct bracket expression, and a line with a word NO otherwise. (There should be as many lines as check operations in the test.)

Example

```
Input:
4
()((
4
4
0
2
0
[and 9 test cases more]
Output:
Test 1:
YES
NO
[and 9 test cases more]
```

Warning: large Input/Output data, be careful with certain languages