Farmer John has a boolean statement that is N keywords long ($1 \leq N < 2 \cdot 10^5$, N odd). Only `true` or `false` appear in odd positions, while only `and` and `or` appear in even positions.

A phrase of the form x OPERATOR y, where x and y are either `true` or `false`, and OPERATOR is `and` or `or`, evaluates as follows:

- x `and` y: This evaluates to true if both x and y are true, and false otherwise.
- x `or` y: This evaluates to true if either x or y is true, and false otherwise.

When evaluating the statement, FJ has to take the order of precedence in Moo Language into account. Similar to C++, `and` takes priority over `or`. More specifically, to evaluate the statement, repeat the following step until the statement consists of only one keyword.

1. If the statement contains an `and`, choose any of them and replace the phrase surrounding it with its evaluation.
2. Otherwise, the statement contains an `or`. Choose any of them and replace the phrase surrounding it with its evaluation.

It may be proven that if multiple phrases can be evaluated during a given step, it does not matter which one is chosen; the statement will always evaluate to the same value.

FJ has Q ($1 \leq Q \leq 2 \cdot 10^5$) queries. In each query, he gives you two integers l and r ($1 \leq l \leq r \leq N$, l and r are both odd), and deletes the segment from keyword l to keyword r inclusive. In turn, he wishes to replace the segment he just deleted with just one simple `true` or `false` so that the whole statement evaluates to a certain boolean value. Help FJ determine if it's possible!

INPUT FORMAT (input arrives from the terminal / stdin):

The first line contains N and Q.

The next line contains N strings, a valid boolean statement.

The following Q lines contain two integers l and r, and a string `true` or `false`, denoting whether he wants the whole statement to evaluate to true or false.

OUTPUT FORMAT (print output to the terminal / stdout):

Output a string of length Q, where the i'th character is Y if the i'th query is possible, otherwise N.

SAMPLE INPUT:

```
5 7
false and true or true
1 1 false
1 3 true
1 5 false
3 3 true
3 3 false
5 5 false
5 5 true
```

SAMPLE OUTPUT:

```
NYYYNYY
```

Let's analyze the first query:

If we were to replace delete the segment $[1, 1]$ and replace it with `true`, then the whole statement becomes:

```
true and true or true
```

We evaluate the `and` keyword from at position 2 and obtain

```
true or true
```

Since we have no `and` keywords left, we have to evaluate the `or` keyword. After evaluation, all that is left is

```
true
```

It can be shown that if we were to replace the segment with `false`, the statement will still evaluate to `true`, so we output N since the statement cannot possibly evaluate to `false`.

For the second query, we can replace the segment $[1, 3]$ with `true` and the whole statement will evaluate to `true`, so we output Y.

For the third query, since $[1, 5]$ is the whole statement, we can replace it with anything, so we output Y.

SAMPLE INPUT:

```
13 4
false or true and false and false and true or true and false
1 5 false
3 11 true
3 11 false
13 13 true
```

SAMPLE OUTPUT:

```
YNYY
```

SCORING:

- Inputs 3-5: $N, Q \le 10^2$
- Inputs 6-8: $N, Q \le 10^3$
- Inputs 9-26: No additional constraints.

Problem credits: Chongtian Ma