Problem B
Binary Assignment

Vuong is one of the greatest mathematicians of all time! His hobby is to find out mathematical properties of everything, and sometimes even of non-existing things! On his birthday, his programmer friend gave him a binary string S of length n. After a while, he has found out two very interesting properties of S:

- $X(S)$ – the length of the **shortest** string that is **not** a *subsequence* of S
- $Y(S)$ – the number of the strings that are **not** *subsequence* of S of length $X(S)$

Seeing Vuong had fun finding out these two properties, his programmer friend think that it would be great to also change the string S a little bit. The programmer will sequentially do q modifications to the string S. Each modification is one of the following types:

- `0` l r – set $S_l, S_{l+1}, \ldots, S_r$ to 0.
- `1` l r – set $S_l, S_{l+1}, \ldots, S_r$ to 1.
- `F` l r – flip $S_l, S_{l+1}, \ldots, S_r$. That is, for $l \le i \le r$, if S_i is 0, set it to 1, else set it to 0.

And of course, for each modified version of S, Vuong was also gladly to find $X(S)$ and $Y(S)$, because it was his birthday!

But a puzzle is not complete without an answer. Given the string S and the list of q modifications to the string S, help the programmer friend finding $X(S)$ and $Y(S)$ for each modification, so that he can check Vuong's result with the answer.

Because the answer can be very large, please output the answer modulo $10^9 + 7$.

A string a is a subsequence of a string b if a can be obtained from b by deletion of several (possibly, zero or all) characters. For example, "`bd`", "`acd`", "`b`" are subsequences of "`abcd`", while "`da`" is not.

Input

The first line contains two integers n and q ($1 \le n, q \le 100\,000$) – the length of string S, and the number of modifications.

The second line contains the binary string S of length n.

The i-th line on the next q lines contains the description of the i-th operation in one of the following forms:

- `0` l r ($1 \le l \le r \le n$) – set $S_l, S_{l+1}, \ldots, S_r$ to 0.
- `1` l r ($1 \le l \le r \le n$) – set $S_l, S_{l+1}, \ldots, S_r$ to 1.
- `F` l r ($1 \le l \le r \le n$) – flip $S_l, S_{l+1}, \ldots, S_r$.

Output

For each modification of S, output on a line two integers $X(S)$ and $Y(S)$ modulo $10^9 + 7$.

Explanation of the samples

In the example, the string S is `0110`, and there are $q = 3$ modifications to S.
The following table demonstrates the modifications of S.

Order	Modification	Value of S	$X(S)$	$Y(S)$
Initial		`0110`	3	5
1	0 2 3	`0000`	1	1
2	1 3 4	`0011`	2	1
3	F 2 3	`0101`	3	4

- For $S = $ `0000`, $X(S) = 1$ and $Y(S) = 1$, because there is one string of length 1 that is **not** a *subsequence* of S, which is the string `1`.
- For $S = $ `0011`, the string `10` is the shortest, and is the only string of length 2 that is **not** a *subsequence* of S.
- For $S = $ `0101`, the list of strings of shortest length that are **not** *subsequences* of S is {`000`, `100`, `110`, `111`}.
- For the *initial* string $S = $ `0110`, the list of strings of shortest length that are **not** *subsequences* of S is
{`000`, `001`, `100`, `101`, `111`}. So $X(S) = 3$ and $Y(S) = 5$, but you don't have to print these numbers.

Sample Input 1

```
4 3
0110
0 2 3
1 3 4
F 2 3
```

Sample Output 1

```
1 1
2 1
3 4
```