C. Replacement

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Daniel has a string s, consisting of lowercase English letters and period signs (characters '.'). Let's define the operation of *replacement* as the following sequence of steps: find a substring ".." (two consecutive periods) in string s, of all occurrences of the substring let's choose the first one, and replace this substring with string ".". In other words, during the replacement operation, the first two consecutive periods are replaced by one. If string s contains no two consecutive periods, then nothing happens.

Let's define $f(s)$ as the minimum number of operations of *replacement* to perform, so that the string does not have any two consecutive periods left.

You need to process m queries, the i-th results in that the character at position x_i $(1 \le x_i \le n)$ of string s is assigned value c_i. After each operation you have to calculate and output the value of $f(s)$.

Help Daniel to process all queries.

Input

The first line contains two integers n and m $(1 \le n, m \le 300\,000)$ the length of the string and the number of queries.

The second line contains string s, consisting of n lowercase English letters and period signs.

The following m lines contain the descriptions of queries. The i-th line contains integer x_i and c_i $(1 \le x_i \le n$, c_i — a lowercas English letter or a period sign), describing the query of assigning symbol c_i to position x_i.

Output

Print m numbers, one per line, the i-th of these numbers must be equal to the value of $f(s)$ after performing the i-th assignment.

Examples

input
10 3 .b..bz.... 1 h 3 c 9 f

output
4 3 1

input
4 4 .cc. 2 . 3 . 2 a 1 a

output
1 3 1 1

Note

Note to the first sample test (replaced periods are enclosed in square brackets).

The original string is ".b..bz....".

- after the first query $f($`hb..bz....`$)$ = 4 ("hb[..]bz...." → "hb.bz[..].." → "hb.bz[..]." → "hb.bz[..]" → "hb.bz.")
- after the second query $f($`hbc.bz....`$)$ = 3 ("hbc.bz[..].." → "hbc.bz[..]." → "hbc.bz[..]" → "hbc.bz.")
- after the third query $f($`hbc.bz..f.`$)$ = 1 ("hbc.bz[..]f." → "hbc.bz.f.")

Note to the second sample test.

The original string is ".cc.".

- after the first query: $f($`..c.`$)$ = 1 ("[..]c." → ".c.")
- after the second query: $f($`....`$)$ = 3 ("[..].." → "[..]." → "[..]" → ".")
- after the third query: $f($`.a..`$)$ = 1 (".a[..]" → ".a.")
- after the fourth query: $f($`aa..`$)$ = 1 ("aa[..]" → "aa.")