問題 8 – 不能抄襲！(No Plagiarism!)

(20 分)

問題敘述

阿夸教授是「Brainfuck 導論」的任課教師。如果你沒聽過 Brainfuck 的話，Brainfuck 是一種極小化的程式語言，它是由 Urban Müller 在 1993 年創造的。這種語言由八種運算子構成，列表如下：

字元	含義
>	指標加一
<	指標減一
+	指標指向的位元組的值加一
-	指標指向的位元組的值減一
.	輸出指標指向的單元內容（ASCII 碼）
,	輸入內容到指標指向的單元（ASCII 碼）
[	如果指標指向的單元值為零，向後跳轉到對應的] 指令的次一指令處
]	如果指標指向的單元值不為零，向前跳轉到對應的 [指令的次一指令處

因為這個語言實在是太難了，每次要繳交作業時，大多數學生都會向其他同學「借」作業來「參考」。阿夸教授當然不樂見這種狀況，因此她拜託你來幫她抓抄襲。
幸運的是，你不需要學會 Brainfuck 也能達成這項任務。給你所有學生繳交的程式碼，然後阿夸教授會問你學生i和j的程式碼相似程度。為了簡化問題，你只需要找出他們的程式碼的最長共同前綴就好。

輸入格式

第一行有兩個整數 n和 m，以空白分隔。
接下來 n行，每行都是一個由 Brainfuck 指令組成的字串，代表學生 0, 1, …, $n-1$提交的程式碼。
接下來 m行，每行有兩個整數 i, j介於 0 到 $n-1$之間，代表阿夸教授詢問你學生 i和 j的程式碼相似程度。

輸出格式

對每個詢問輸出一個整數後換行，代表詢問的最長共同前綴。

資料範圍

$1 \leq n, m \leq 1000000$
所有程式碼的長度總和不超過 1000000

輸入範例 1

```
2 1
++--><[ ]
++-++<>[ ]
0 1
```

輸出範例 1

```
3
```

輸入範例 2

```
3 3
+++++
+++++--
+++++-
0 1
0 2
1 2
```

輸出範例 2

```
5
5
6
```

輸入範例 3

```
4 5
><><><><
++++++++
------++
++++++--
0 1
0 3
1 2
1 3
2 3
```

輸出範例 3

```
0
0
0
6
0
```

範例說明

第一個範例的“++--><[ ]” 和 “++-++<>[ ]” 的最長共同前綴是 “++-”，長度為 3，因此輸出 3。
第二個範例的程式碼 0, 1 和程式碼 0, 2 皆有前綴 “+++++”，所以前兩行輸出皆為 5。程式碼 1, 2 皆有前綴 “++++-”，所以第三行輸出是 6.
對於第三個範例，詢問 1, 2, 3, 5 的最長共同前綴都是空字串，所以對應到的輸出皆為 0。
第四個詢問的最長共同前綴是”++++++”，所以第四行輸出是 6。

Q8: No Plagiarism!

(20 points)

Description

Prof. Aqua is the Instructor of the course “Introduction To Brainfuck”. If you haven’t heard about Brainfuck, it is an esoteric programming language created in 1993 by Urban Müller, and it only consists of eight commands, listed below.

Character	Meaning
`>`	Increment the data pointer (to point to the next cell to the right).
`<`	Decrement the data pointer (to point to the next cell to the left).
`+`	Increment (increase by one) the byte at the data pointer.
`-`	Decrement (decrease by one) the byte at the data pointer.
`.`	Output the byte at the data pointer.
`,`	Accept one byte of input, storing its value in the byte at the data pointer.
`[`	If the byte at the data pointer is zero, then instead of moving the instruction pointer forward to the next command, jump it *forward* to the command after the *matching* `]` command.
`]`	If the byte at the data pointer is nonzero, then instead of moving the instruction pointer forward to the next command, jump it *back* to the command after the *matching* `[` command.

Since the language is very hard to read and write, every time when there is an assignment, most of the students “borrowed” other students’ code “for reference.” Of course, Prof. Aqua is not happy about this situation, so she asked you to help her detect plagiarism.
Fortunately, you don’t need to learn Brainfuck to do this. Now given all students’ code submissions, and then Prof. Aqua will ask you the similarity of student i’s code and student j’s code. To simplify the problem, you only need to find the length of the longest common prefix of their codes.

Input Format

In the first line consists of two positive integers $n$and m, separated by space.
In the next $n$lines, each line is a string only consisting of the 8 Brainfuck commands, which represents the code submission of student 0, 1, …, $n-1$, respectively.
The next $m$lines, each line consists of two integers i, j ranging from 0 to $n - 1$(inclusive), which represents that Prof. Aqua asks you the length of the longest common prefix of student i's and student j's code submissions.

Output Format

For each query (i, j) Prof. Aqua asked, output the length of the longest common prefix of student i and student j's code submissions.

Data Range

$1 \leq n, m \leq 1000000$
The sum of length of all code submissions is no more than 1000000.

Input Example 1

```
2 1
++--><[ ]
++-++<>[ ]
0 1
```

Output Example 1

```
3
```

Input Example 2

```
3 3
+++++
+++++--
+++++-
0 1
0 2
1 2
```

Output Example 2

```
5
5
6
```

Input Example 3

```
4 5
><><><><
++++++++
------++
++++++--
0 1
0 3
1 2
1 3
2 3
```

Output Example 3

```
0
0
0
6
0
```

Example Explanation:

For example 1, the longest common prefix of “++--><[]” and “++-++<>[]” is “++-”, so it outputs 3.
For example 2, code 0 and 1, code 0 and 2 share the prefix “+++++”, so the first two lines of output are both 5. Code 1 and 2 share the prefix “+++++-”, so the third line of output is 6.
For example 3, queries 1, 2, 3, 5 all only have the common prefix “” which is an empty string, so the corresponding lines of output are all 0. Query 4 has the longest common prefix “++++++”, so the fourth line of output is 6.