A Pay Check

Your boss has decided that rather than being paid by the hour, all programmers will receive exactly one penny per word of code written. Fortunately the application used to evaluate the number of words does not exclude symbols, so padding the numbers is fairly easy.

More precisely, one penny will be paid for each sequence of non-whitespace characters. These sequences must be separated by one or more whitespace characters, which may be either spaces (' '), tabs ('\t'), carriage returns ('\r'), or newlines ('\n').

Knowing how you will be paid, you have obviously become interested in how much different code formatting styles affect your paycheck. For this problem you will write a program to process code and determine how much it is worth.

Input

The input will consist of one or more programs to be evaluated. The end of each program is indicated by a line containing only "`---MORE---`" (without the quotation marks). The end of input is indicated by a line containing only "`---END---`" (without the quotation marks).

As shown in the sample input, the last program will not have a "`---MORE---`" line. No single line will be longer than 100 characters, and there will never be more than 1000 lines in the total input.

Output

For each program, print the number of cents paid for it on a line.

Sample Input

```
printf ( "Hello World!\n" ) ;
---MORE---
System . out . println ( "Hello World!" ) ;
---END---
```

Sample Output

```
6
10
```

B Line Testing

You are designing the input system for the next version of Black & White. The spells in this game are cast by making mouse movements. This system is almost completed, but it still needs one more component, the ability to test if a point is on the left hand or right hand side of a line.

This is useful for classifying what type of symbol is being drawn, but you have found that it was trickier than anticipated. First of all, what is the meaning of left and right for a line? The line has two points, p_0 and p_1. The direction of the line is defined by the vector $D = (p_1 - p_0)$. The normal to the vector is defined as $(-D_y, D_x)$. Now we can talk about left and right. Left is the direction in which the normal points, starting at the line, and right is the opposite of this direction.

There will no points that are co-linear (lie on the infinite line formed by (p_0, p_1)) with any of the lines.

Input

You will be given two integers, N and M. N is the number of points, and M is the number of lines you will be given. 0 < N <= 20 and 0 < M <= 10.

Then follows N lines where you will be given 2 doubles per line, separated by one space, with the first double being the x co-ordinate of the point and the second double being the y co-ordinate.

Then follows M lines where you will be given 4 doubles per line, each separated by one space. The doubles will be the co-ordinates of the line (x1, y1) and (x2, y2) in order.

Output

There will be M lines of output, each of which will contain N characters. The j-th character of the i-th line of output should be 'L' (without the quotes) if the point j is on the left of line i, and 'R' (without the quotes) if point j is to the right of line i. The lines and points should appear in the same order as they were given in the input.

Sample Input

```
5 2
5.0 5.0
1.0 5.0
5.0 1.0
1.0 1.0
2.0 3.0
0.0 4.0 8.0 4.5
1.5 8.0 1.2 0.0
```

Sample Output

```
LLRRR
LRLRL
```