Task 1: `Palindromic FizzBuzz`

Gug finds the classic FizzBuzz problem to be too boring, and has decided to add a twist to it. Print a list of consecutive integers, one on each line, starting with S on the first line and ending with E on the last line. As Gug likes palindromes, print the string `Palindrome!` in place of an integer if it is palindromic, i.e. it can be read the same way forwards and backwards.

Input

Your program must read from standard input. The input is a line with 2 integers, S and E, in a single line.

Output

Your program must print to standard output. Output $E - S + 1$ lines, with each line containing either an integer or the string `Palindrome!` if the integer is palindromic.

Subtasks

The maximum execution time on each instance is 1.0s. For all testcases, the input will satisfy the following bounds:

- $E - S + 1 \leq 10^5$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	S, E
1	7	$1 \leq S = E \leq 9$
2	11	$1 \leq S \leq E \leq 9$
3	14	$1 \leq S \leq E \leq 100$
4	8	$1 \leq S \leq E \leq 10^5$
5	9	$1 \leq S = E \leq 10^9$
6	20	$1 \leq S \leq E \leq 10^9$
7	31	$1 \leq S \leq E \leq 10^{18}$

Sample Testcase 1

This testcase is valid for subtasks 3, 4, 6 and 7.

Input	Output
8 13	Palindrome! Palindrome! 10 Palindrome! 12 13

Sample Testcase 1 Explanation

8, 9 and 11 are palindromes.

Sample Testcase 2

This testcase is valid for all subtasks.

Input	Output
3 3	Palindrome!

Sample Testcase 2 Explanation

3 is a palindrome.

Sample Testcase 3

This testcase is valid for subtasks 6 and 7.

Input	Output
999999997 1000000000	999999997 999999998 Palindrome! 1000000000

Sample Testcase 3 Explanation

999999999 is a palindrome.

Task 2: `Lost Array`

Rar the Cat has an array X of N positive integers. He is a teacher and he wants to give his students a homework based on his array. The students in his class had learnt the `min` function, and Rar would like to test them on this. He have already set M homework questions, and all of them are of this form:

$$\min(X_i, X_j) \ = ?$$

Unfortunately, Rar has lost his array! Given the M homework questions, as well as the answer key, help Rar to reconstruct a possible array that matches all of his homework answers. Such an array is guaranteed to exist.

Input

Your program must read from standard input.

The first line of the input will contain 2 numbers, N and M.

The next M lines of input will contain 3 numbers, A_i, B_i, and C_i. For all $i = 1, 2, ..., M$, $\min(X_{A_i}, X_{B_i}) = C_i$.

Output

Output N numbers in a single line (separated by spaces), the array X. If multiple solutions exist, all of them will be accepted. All elements of X must be between 1 and 10^9 (inclusive).

Subtasks

The maximum execution time on each instance is 1.0s. For all testcases, the input will satisfy the following bounds:

- $1 \leq N, M \leq 10^5$
- $1 \leq A_i, B_i \leq N$
- $A_i \neq B_i$
- $1 \leq C_i \leq 10^9$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	5	$N = 2, M = 1$
2	6	$M \leq 3$
3	20	$N, M \leq 1000$
4	21	$C_i \leq 10, N \leq 5$
5	48	-

Sample Testcase 1

This testcase is valid for all subtasks.

Input	Output
2 1 2 1 7	9 7

Sample Testcase 1 Explanation

The reconstructed array satisfies the given constraints:

- $\min(X_2, X_1) = \min(7, 9) = 7$

Sample Testcase 2

This testcase is valid for subtasks 3, 4 and 5.

Input	Output
5 6 1 2 1 3 5 4 1 5 3 1 3 3 2 3 1 2 4 1	3 1 4 1 5

Sample Testcase 2 Explanation

The reconstructed array satisfies the given constraints:

- $\min(X_1, X_2) = \min(3, 1) = 1$

- $\min(X_3, X_5) = \min(4, 5) = 4$
- $\min(X_5, X_1) = \min(5, 3) = 3$
- $\min(X_1, X_3) = \min(3, 4) = 3$
- $\min(X_3, X_2) = \min(4, 1) = 1$
- $\min(X_4, X_2) = \min(1, 1) = 1$

Sample Testcase 3

This testcase is valid for subtasks 3 and 5.

Input	Output
5 1 1 2 123	123 1000000000 3 4 26311337

Sample Testcase 3 Explanation

The only condition for the array is that $\min(X_1, X_2) = 123$, the rest of the array can be any value between 1 and 10^9.

Task 3: `Experimental Charges`

There are N charged particles, each with either a positive or a negative charge. The charge on each particle is unknown, but it is known that if two particles with the same charge are brought close together, they will repel each other and if two particles with a different charge are brought together, they will attract each other.

In an experiment, Q events happen in chronological order. Each event is of one of the following 2 types:

1. Two particles are brought close together and you are told whether they repel or attract,
2. You are asked when two particles are brought together, whether they will repel, attract or either is possible, based on the experimental observations so far.

It is guaranteed that there is at least one configuration of particles that match all the experimental observations given.

Input

Your program must read from standard input.

The input starts with a line with 2 integers, N and Q. N denotes the number of charged particles, Q denotes the number of events.

Q lines will then follow with 1 character and 2 integers each, the i^{th} line will contain T_i, A_i and B_i. If T_i = '`A`', it denotes type 1 event where particles A_i and B_i attract. If T_i = '`R`', it denotes type 1 event where particles A_i and B_i repel. If T_i = '`Q`', it denotes type 2 event asking whether particles A_i and B_i attract or repel (or either is possible).

Output

Your program must print to standard output.

For every type 2 event, your program must output one line with one character. Output '`A`' if the charges will attract, '`R`' if the charges will repel, or '`?`' if either is possible.

Subtasks

The maximum execution time on each instance is 2.0s. For all testcases, the input will satisfy the following bounds:

- $1 \leq N, Q \leq 10^5$
- $1 \leq A_i \neq B_i \leq N$
- T_i = '`A`', '`R`', or '`Q`'

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	N, Q	T_i, A_i, B_i
1	7	$N = 2, Q \leq 10$	-
2	11	-	$A_i = 1$ or $B_i = 1$
3	14	-	T_i = '`R`', or '`Q`'
4	12	-	All updates occur before queries
5	25	$1 \leq N, Q \leq 10^3$	-
6	31	-	-

Sample Testcase 1

This testcase is valid for subtasks 1, 2, 3, 5 and 6.

Input	Output
2 3 Q 1 2 R 1 2 Q 1 2	? R

Sample Testcase 2

This testcase is valid for subtasks 4, 5 and 6.

Input	Output
4 5 R 1 2 A 2 3 A 1 4 Q 2 4 Q 1 3	A A

Task 4: `Square or Rectangle?`

Note that only C++ and Java may be used for this task.

I can be a square, or a rectangle. Am I a square, or a rectangle?

Consider a grid of N by N squares. Suppose there is either a square or a rectangle covering some grid squares, and each grid square is either completely inside or outside the shape, i.e. the boundary of the shape follows the grid lines. Two examples are shown below:

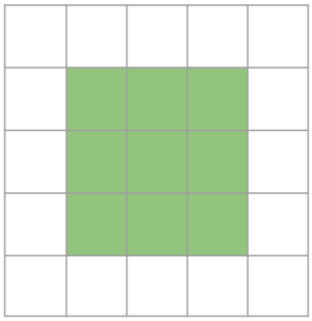

Figure 1: I am a **square**.

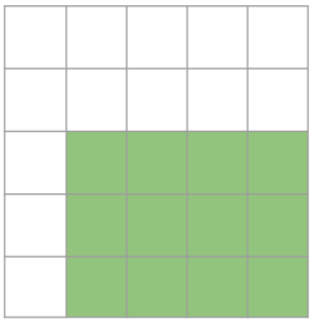

Figure 2: I am a **rectangle**.

It is also guaranteed that the shape covers **at least 4% of the total area** of the grid. Your task is to determine if the shape is a square, or a rectangle. To do that, you are allowed to ask at most Q queries, each query allows you to find out if a certain grid square is inside the shape. The grid squares are identified by coordinates (x, y) where $1 \leq x, y \leq N$.

Implementation Details

This is an interactive task. Do not read from standard input or write to standard output.

You are to implement the following function:

- C++: `bool am_i_square(int N, int Q)`
- Java: `public static boolean am_i_square(int N, int Q)`

The `am_i_square` function will be called at most T times. Each call to this function represents a separate instance of the problem, i.e. the shape may be different and Q queries are allowed in each instance of the problem.

Within the `am_i_square` function, you are allowed to call the following grader functions to complete the task:

- C++: `bool inside_shape(int X, int Y)`
- Java: `public static boolean inside_shape(int X, int Y)`

The `inside_shape` function will return `true` if the grid square (X, Y) is inside the shape, or `false` otherwise. If your program calls this function more than Q times or with invalid parameters, the program will terminate immediately and you will be given a *Wrong Answer* verdict.

Sample Interaction

Consider the grid found in Figure 1. Suppose $Q = 25$. Your function will be called with the following parameters:

```
am_i_square(5, 25)
```

A possible interaction could be as follows:

- `inside_shape(3, 3) = true`

  The `inside_shape` function is asked if (3, 3) is inside the shape. As the grid square is inside the shape (highlighted in green as shown in Figure 1), the function returns `true`.

- `inside_shape(5, 4) = false`

  The `inside_shape` function is asked if (5, 4) is inside the shape. As the grid square on the fifth row and fourth column is outside the shape, the function returns `false`.

- `inside_shape(1, 1) = false`

  The `inside_shape` function is asked if (1, 1) is inside the shape. As the grid square on the top-left corner is outside the shape, the function returns `false`.

- `inside_shape(2, 4) = true`

  The `inside_shape` function is asked if (2, 4) is inside the shape. As the grid square is within the shape, the function returns `true`.

At this point, it decides that it has enough information to conclude that the shape is a **square**. As such, it will return `true`. As the shape is indeed a square and the program has used less than 25 queries, it would be deemed as correct for this testcase.

Subtasks

The maximum execution time on each instance is 3.0s. For all testcases, the input will satisfy the following bounds:

- $N = 100$
- $1 \leq T \leq 1000$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	14	$Q = 10^4$
2	19	$Q = 100$
3	18	$Q = 40$, shape will cover at least 25 % of the grid area.
4	49	$Q = 50$. Please read "Scoring" for further details.

Scoring

Subtask 4 is a special scoring subtask. Your score depends on the maximum number of queries q you make in any testcase.

- If $q > 50$, you will score 0 points.
- If $34 \leq q \leq 50$, you will score $40 - 30 * \frac{q-34}{17}$ points.
- If $q \leq 33$, you will score 49 points.

Testing

You may download the grader file, the header file (for C++) and a solution template under *Attachments*. The sample testcases are also provided for your reference. Please use the compile and run scripts provided (the .sh files) for testing.

Grader Input Format for Testing

- one line with three integers, T, N and Q.
- T lines, containing four integers X_1, Y_1, X_2 and Y_2, where (X_1, Y_1) is the top-left corner of the shape and (X_2, Y_2) is the bottom-right corner of the shape.