Bracket Query

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Bracket sequences are perfect examples of graceful balancing. Here we talk about bracket sequence consisting of '`(`' and '`)`'. For example, "`(())`", "`()(())`" are valid balanced bracket sequence.

More formally, a bracket sequence is valid if and only if it can be constructed by the following rules:

- "`()`" is a valid bracket sequence.
- If A is a valid bracket sequence, then (A), i.e., adding a left bracket before and a right bracket after, forms a valid bracket sequence.
- If A, B are valid bracket sequences, then AB, i.e., concatenating them, forms a valid bracket sequence.

Today Sayu is playing a game with you. She has a valid bracket sequence S with length n, and you can ask her questions in the form "What is the difference between the numbers of left and right brackets in the substring of S from the l-th character to the r-th character?", and then Sayu will tell you the answer.

You want to determine the bracket sequence using the above method. After playing for a while, however, Sayu escaped and went to grab some sleep. "Mission accomplished! Can I go back and sleep now?"

You want to retrieve a possible valid bracket sequence from already known queries. You may also find contradictions in the queries since Sayu was sleepy, in which case you should report no solution.

Input

The first line contains two integers n, q $\left(2 \le n \le 3000, 0 \le q \le \min(\binom{n+1}{2}, 5 \times 10^5)\right)$, denoting the length of the bracket sequence and the number of known queries.

In the following q lines, the i-th line contains three integers $l_i, r_i, c_i \, (1 \le l_i \le r_i \le n,\ -n \le c_i \le n)$, denoting the i-th query ranges $q_i = [l_i, r_i]$, and the answer to the query is c_i, which is the number of the left brackets subtracts the number of right brackets in the interval.

It's guaranteed that n is even, and the queried intervals are pairwise different.

Output

If there's no valid solution, print "?".

Otherwise, print "`!` S" in a single line where S is a valid bracket sequence consistent with input.

Examples

standard input	standard output

`4 1`
`1 2 0`
	`! ()()`

`4 1`
`1 2 2`
	`! (())`

`2 2`
`1 1 1`
`2 2 -1`
	`! ()`

`2 1`
`1 1 2`
	`?`

`4 0`	`! (())`

Note

For the first and the second test case, the only possible valid bracket sequences with length 4 are "`(())`" and "`()()`". By asking the substring containing the first two characters, one can distinguish two different substrings since the answer will be 2 and 0, respectively.