Problem H
Hash Collision
Time limit: 1 second

For security reasons, TU Delft is going to place locks with numeric keypads on the doors of a large number of rooms. Each room will have its own pass code. The task of setting up the server on which all codes will be stored is given to Harry and Sharon.



Silhouettes free to use from pxfuel.com

Having paid attention in cybersecurity class, they know that the codes should be passed through a *hash function*, preferably multiple times, before storing.

Sharon came up with the nifty idea of letting the room number be the number of times the code is passed through the hash function. That way, even if two rooms happen to have the same pass code, they will not (necessarily) end up with the same hash value. However, they find that for some combinations of room number and code, the hash value happens to be the same as the original code, presenting a security risk.

Not to be outdone by Sharon, Harry came up with an idea of his own: to switch the roles, that is, let the code be the number of times the hash function is applied to the room number. In other words, if c is the code and r is the room number, the hash value will be $f^c(r) = \underbrace{f(\cdots f(}_{c \text{ times}} r) \cdots)$.

After some thought, Sharon claimed that, regardless of what the function f is, it would always still be the case for some room numbers and codes that the hash value is the same as the code; that is, that $f^c(r) = c$. In fact, Sharon thinks it would not be too difficult to find two such numbers, even without knowing the full details of f.

This dismissive statement made Harry angry, who believed that Sharon was just jealous of his idea. After a big argument that led nowhere, Harry decided to make Sharon prove her claim: he has written a program that, upon sending it a query, will return the hash value $f^c(r)$ for the c and r given in the query, using a secret hash function f he has chosen. The hash function accepts any r in $\{1, \ldots, n\}$, where n is given, and returns a value in the same range. The value of c should also be in the same range. The challenge for Sharon is to find c and r such that $f^c(r) = c$, using a limited number of queries.

You know that Sharon is right about her claim and decide to help her.

In the first sample case, the value of n is 6, and the hidden function is given by $f(1) = 4$, $f(2) = 3$, $f(3) = 5$, $f(4) = 2$, $f(5) = 4$, and $f(6) = 6$. In the second sample, $n = 4$, and $f(1) = 2$, $f(2) = 4$, $f(3) = 2$, and $f(4) = 2$.

Interaction

This is an interactive problem. Your submission will be run against an interactor, which reads the standard output of your submission and writes to the standard input of your submission. This interaction needs to follow a specific protocol:

The interactor first sends one line with an integer n ($1 \leq n \leq 2 \cdot 10^5$), indicating that the domain of the hidden function f is $\{1, \ldots, n\}$.

Then, your program should make at most 1000 queries to find the answer. Each query is made by printing one line of the form "`? c r`" ($1 \le c, r \le n$). The interactor will respond with an integer h ($1 \le h \le n$), the value of $f^c(r)$.

When you have determined some values of c and r such that $f^c(r) = c$, print one line of the form "`! c r`" ($1 \le c, r \le n$), after which the interaction will stop. Printing the answer does not count as a query.

If there are multiple valid solutions, you may output any one of them.

The interactor is not adaptive: the hidden function f is fixed up front, and does not depend on your queries.

Make sure you flush the buffer after each write.

A testing tool is provided to help you develop your solution.

Using more than 1000 queries will result in a wrong answer.

Sample Interaction 1

Read	Write
6	
	? 2 4
3	
	? 4 1
5	
	? 5 5
4	
	? 1 6
6	
	? 2 1
2	
	! 2 1

Sample Interaction 2

Read	Write
4	
	? 1 3
2	
	? 2 3
4	
	? 3 3
2	
	! 4 3