Jumbled Primes

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	256 megabytes

The great oracle has chosen a uniformly random permutation of the numbers from 1 to 100: $p_1, p_2, \ldots, p_{100}$.

The only way to get information from the oracle is by asking for the greatest common divisor (gcd) of a pair of numbers in the permutation, p_a and p_b, $(a \neq b)$.

Find all the positions of the prime numbers and the number one, and report them. The output should be a bitstring with 1 if position i is a prime or the number one, 0 otherwise.

Your solution is run once, and in this run, it will be tested on a 1000 testcases. Note that each time you submit, the tests are newly generated. Let S be the sum of the number of queries used on all tests. For getting accepted, it must hold that: $S \leq 1000 \times 600$

Interaction Protocol

In a testcase, your program can immediately start doing queries.

For finding out the gcd of p_a and p_b, output a single line in the format "`? a b`", where a and b are two distinct integers, such that $1 \leq a, b \leq 100$. Afterwards, you should read a single integer g. g will be equal to $\gcd(p_a, p_b)$.

When you have determined all the positions of the prime numbers and the number one, print a single line of the form "`! answer`", where `answer` is a bitstring of length 100, with a 1 at position j if p_j is a prime or the number one. When p_j is composite, the bitstring should contain a 0 at that position. This does not contribute to the total query count S.

Afterwards, this testcase is finished, and your program should immediately move on to the next testcase, or stop, if it completed all 1000 tests.

Do not forget to flush the output after each query.

Example

standard input	standard output
1 2 2 3	? 1 3 ? 2 4 ? 6 4 ? 6 3 ! 111010

Note

This example is not a valid testcase. For illustration, here p is a permutation of the numbers from 1 to 6. The random permutation just happened to be $p = [1\ 2\ 3\ 4\ 5\ 6]$.

Although the information obtained by the queries is not enough to find out where the primes are located, the output is correct: A bitstring of length 6, with a 1 at places $1, 2, 3$ and 5. p_1, p_2, p_3 and p_5 contain the prime numbers and one.