Problem H. Hackerman

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

You, Mr. Hackerman, have gained access to the servers of a secure messaging app. They use a new ciphering method based on the difficulty of factoring the numbers that are products of three big primes.

You are reading the source code and found out how they actually generate these numbers. They define three recurrences:

- $x_n = (11 \cdot x_{n-1} + 7) \bmod 611\,953 \quad \forall n > 0$
- $y_n = (13 \cdot y_{n-1} + 5) \bmod 746\,773 \quad \forall n > 0$
- $z_n = (53 \cdot z_{n-1} + 3) \bmod 882\,389 \quad \forall n > 0$

The seeds x_0, y_0, z_0 seem to be stored in a secure file.

Now they get the three primes p_k, q_k, r_k for the k-th user:

- p_k is x_k-th number in a secure file `X.axx` (all its numbers are different primes with exactly 31 decimal digits).
- q_k is y_k-th number in a secure file `Y.axx` (all its numbers are different primes with exactly 32 decimal digits).
- r_k is z_k-th number in a secure file `Z.axx` (all its numbers are different primes with exactly 33 decimal digits).

Then, they compute the public key $n_k = p_k \cdot q_k \cdot r_k$.

You have temporary access to the public key database, and you can query up to 5 public keys for any user.

You are given two integers u and v. You have to intercept communications between the u-th user and the v-th user. For this task, you will need to compute $p_u, q_u, r_u, p_v, q_v, r_v$; having these values will allow you to completely manipulate communication.

For the sake of input/output simplicity, we ask you to output $p_u + q_u + r_u + p_v + q_v + r_v$ as your answer.

Interaction Protocol

First, you must read a line containing two integers u and v ($0 \le u, v < 7 \cdot 10^{12}$).

To ask for the public key (n_k) of the k-th user, print "`? k`", where k is the user index. Remember that you can do at most 5 queries of this type.

The first user has index 0, and there are exactly $7 \cdot 10^{12}$ users, so make sure $0 \le k < 7 \cdot 10^{12}$ in all your questions.

When you have computed the solution, print "`! a`", where $a = p_u + q_u + r_u + p_v + q_v + r_v$. Remember it's safer to stop your code after this query.

Don't forget to flush the output after printing each line!

The files `X.axx`, `Y.axx`, and `Z.axx` are the same for the whole problem.

Example

standard input	standard output
10 20 19227930940946299264548209033040475836840046972249992507604326690346496179418709407710724396749 1 27484854406533716638162995259016486377602039494141055337350245326304213427822762176892360055761 7 61991716112162091571854380103197141133071202062437636592434396525428433284775254050695414158203	? 10 ? 20 ? 3 ! 11886707251230740987903684471226 96

Note

Solution to the example ($u = 10$ and $v = 20$):

$p_{10} = 674571972811348479492069676788 1$

$q_{10} = 54398126832702965410665141463513$

$r_{10} = 523986762172023700466774225430947$

$p_{20} = 689903708532390014938395717956 9$

$q_{20} = 766079724656701501898022114673 09$

$r_{20} = 520033106839239897778822214813477$

$p_{10} + q_{10} + r_{10} + p_{20} + q_{20} + r_{20} = 11886707251230740987903684471226 96$