Digger

This is an interactive problem

Task

During the excavation at the ruins of the ancient city-kingdom of Salamis archaeologist found a hidden royal tomb, buried at an unknown depth of n meters below the surface. Your task is to find the number n by running experiments with your digging bot.

In each experiment, you give the bot a list of integer numbers $a_1, a_2, ..., a_k$, and ask it to dig from the surface (depth 0) to depth n, using only steps of length a_i at a time. That means, if the bot's current depth is x, it can move to depths $x + a_1$, $x + a_2$, ..., or $x + a_k$. When the bot reaches the depth n, it tells you the number of steps it made. The bot is very intelligent and always makes the minimal possible number of steps to reach the depth n.

For example, let the hidden number be $n = 23$. If you give the bot a list $a = [1, 3, 8]$, it will reach depth n in 5 steps (for example $0 \to 3 \to 4 \to 7 \to 15 \to 23$), and if you give the bot a list $a = [9, 5]$, it will reach depth n in 3 steps ($0 \to 9 \to 18 \to 23$).

Your task is to guess the number n after making several experiments with the following constraints:

- The hidden number n is in range from 1 to 30000 (inclusive).
- The total number of experiments should be at most 20.
- The total length of lists a in all experiments should be at most 250.
- In each experiment, it should be possible for the bot to reach the depth n.

Interaction

To make an experiment, your program should print one line of the form: `? k a_1 a_2 ... a_k`, where k ($1 \le k \le 250$) is the length of the list and a_i $(1 \le a_i \le 30000)$ are the elements of the list. After that, your program should read a single integer m, the minimum number of steps the bot needed to do to reach the depth n.

When your program is ready to guess the number n, it should print one line of the form: `! n`. After that, your program should exit.

After printing each experiment do not forget to output the end of line and flush the output.

If you get -1 as the result of any experiment instead of valid data, your solution must exit immediately. This means that your solution will be considered incorrect because of an invalid experiment or any other mistake. Failing to exit can result in an arbitrary verdict because your solution will continue to read from a closed stream.

To flush, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `sys.stdout.flush()` in Python;

Example

Input	Output
	`? 3 1 3 8`
`5`	
	`? 2 9 5`
`3`	
	`! 23`

Scoring

Your solution will be tested on 100 tests. If your solution fails to guess the number n, or exceeds the given limits in at least one test, it will get 0 points.

If your solution successfully guesses the number n, and does not exceed the limits in all tests, the score will be calculated based on the total number of steps the bot made in all experiments.

Let s_i be the total number of steps the bot made in all experiments in test i, and let S be maximum of s_i for all 100 tests. Then your score will be calculated based on the following table:

S	**Score**
≤ 100	100
$101 - 150$	95
$151 - 200$	90
$201 - 250$	80
$251 - 300$	70
$301 - 500$	60
$501 - 1000$	50
$1001 - 5000$	20
$5001 - 20000$	10
$20001 - 30000$	5
≥ 30001	0