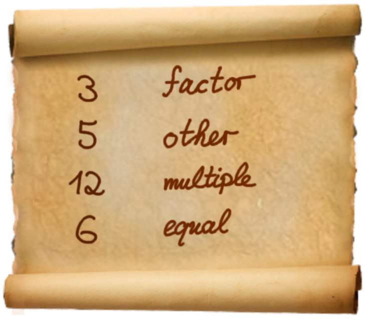
Problem E: Enchanted Exam

Time limit: 1 second



Parchment from Wikimedia Commons, cc by-sa 3.0

In the Wizarding World, Arithmancy is the study of numbers and their magical properties, and it is also Hermione's favourite class at Hogwarts. She is currently studying for the final exams, the *Nastily Exhausting Wizarding Test (N.E.W.T.)*, using last year's exam questions.

To this day, students at Hogwarts still use parchments and quills as writing material. While these have some disadvantages compared to muggles' pen and paper, an advantage is that using magically enchanted parchment allows for more quirky exam questions such as the following:

> You need to find a hidden integer number x between 1 and 100, inclusive. Up to 50 times you may write an integer number y on the provided piece of parchment. For each number you write, one of the following four words will appear next to it:
>
> - `equal`, if you found the correct number;
> - `factor`, if y is a divisor of x;
> - `multiple`, if y is a multiple of x;
> - `other`, if none of the above is true.

We decided to reuse this N.E.W.T. question as a problem for Winter Contest 2022. Unfortunately we were not able to obtain enough magical interactive parchment in time for the contest, so that we had to convert the original question into an interactive programming problem instead.

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:

Repeatedly output query numbers y ($1 \leq y \leq 100$). For each such number, the interactor replies with one of the strings `equal`, `factor`, `multiple` or `other`, as described above.

After every query you should *flush* the standard output to ensure that the query is sent to the interactor. For example, you can use `fflush(stdout)` in C++, `System.out.flush()` in Java, and `sys.stdout.flush()` in Python.

You may use at most 50 queries and your program must exit after you found the hidden number. The interaction will be run multiple times. The hidden number x is fixed in advance and will be different for each run.

A testing tool is provided to help you develop your solution. It can be downloaded from the DOMjudge problems overview page.

Sample Interaction 1

Read	Write
	`3`
`factor`	
	`5`
`other`	
	`12`
`multiple`	
	`6`
`equal`	

Sample Interaction 2

Read	Write
	`100`
`multiple`	
	`99`
`multiple`	
	`1`
`equal`	