Task: DIV
Divisor game

Nowy Sącz, POLAND 2005
The 17-th International
Olympiad in Informatics

Source file `div.*`

Available memory: 32 MB. Maximum running time: 1 s.

Alice and Bob invented a two-player game. At first, Alice chooses a positive integer k in the range from 1 to some fixed integer n. Then Bob asks questions of the form 'Is k divisible by m?', where m is a positive integer. Alice answers each such question with 'yes' or 'no'. Bob wants to know what number Alice bears in mind by asking as few questions as possible. Your task is to write a program, which plays the game as Bob.

Let us denote by $d(n)$ the minimal number of questions, which have to be asked to find k, regardless of what k Alice chooses (for given n). Your program's answer for a test case will be considered correct, if k is correctly determined using no more than $d(n)$ questions.

Library

Your program must use a special library to play the game. The library consists of the files: `alice.pas` (for Pascal), `alice.h` and `alice.c` (for C/C++). The library provides the following functionality:

- `function get_n: longint` / `int get_n()` — Your program should call this function to initialize a game, before it calls any other function/procedure provided by the library. Function `get_n` returns n, the upper bound on the number that Alice has in mind. Number n satisfies the limitations $1 \le n \le 1\,000\,000$. Moreover, in 50 % of test cases n satisfies $1 \le n \le 10\,000$.

- `function is_divisible_by(m: longint): boolean` / `int is_divisible_by(int m)` — Your program may ask questions by calling this function. Function `is_divisible_by` returns `True`/`1` if the number k Alice has in mind is divisible by m. Otherwise it returns `False`/`0`. The parameter m must satisfy $1 \le m \le n$. Your program should ask as few questions as possible.

- `procedure guess(k: longint)` / `void guess(int k)` — To end the game your program should report the number k Alice has in mind, by calling the procedure `guess(k)`. The parameter k should satisfy $1 \le k \le n$. After calling this procedure your program will be terminated automatically.

If your program makes an illegal call, it will be terminated.

Your program should communicate only by means of the above functions and procedures. Your program must not read or write any files, it must not use standard input/output and it must not try to access any memory outside your program.

Compilation

If your program is written in Pascal, then you must include '`uses alice;`' statement in your source code. To compile your program, use the following command:
`ppc386 -O2 -XS div.pas`

If your program is written in C or C++, then you must include '`#include "alice.h"`' statement in your source code. To compile your program, use one of the following commands:

```
gcc -O2 -static div.c alice.c -lm
g++ -O2 -static div.cpp alice.c -lm
```

Experimentation

To let you experiment with your solution, you are given an example library playing as Alice: its sources are in `alice.pas`, `alice.h` and `alice.c` files. When you run your program, it will be playing against this simple library. You can modify this library, but please do not change the interface part of it. Please remember, that during the evaluation your program will be playing against a different opponent.

When you submit your program using the TEST interface[1], it will be compiled with the unmodified example opponent library. The submitted input file will be given to your program's standard input. The input file should consist of two lines, each containing one integer. The first line should contain number n, and the second line should contain number k.

You are also provided with two simple programs illustrating usage of the library: `div.c` and `div.pas`. (Please remember, that these programs are not correct solutions.)

Sample interaction

Below there is a sample interaction between a program and the library.

Your program calls	**What happens**
`get_n()`	returns 1 000
`is_divisible_by(10)`	returns `True`/`1`
`is_divisible_by(100)`	returns `True`/`1`
`is_divisible_by(1000)`	returns `False`/`0`
`is_divisible_by(200)`	returns `False`/`0`
`is_divisible_by(500)`	returns `False`/`0`
`is_divisible_by(300)`	returns `False`/`0`
`is_divisible_by(700)`	returns `False`/`0`
`guess(100)`	Alice's secret number is 100. Your program wins and is terminated automatically.

[1]TEST interface will be available during the trial session.