Problem 5.1 **Stock-Revealing Backroom Dealing** (page 1 of 1)

Overview: Interactively guess a number based on knowledge that the desired number is less than, greater than, or equal to the guesses.

Description: Nagilent and PAMD, both well-established companies in the technology industry, are considering the possibility of merging. The CEO of Nagilent is bartering some Nagilent stock for PAMD stock with the CEO of PAMD, in order to artificially boost stock trading for both companies. It is possible for the two CEOs to trade anywhere from 1 to *n* shares of stock, but the CEO of PAMD only wants to trade an unknown number *m* shares of stock and will only tell the CEO of Nagilent whether his offer is too high, too low, or right on the dot. The CEO of Nagilent has requested your assistance in this latest endeavor of questionable business ethics.

After we give you *n*, you are allowed to query us about any number *x* within the range of 1 to *n* and we will tell you if *m* is less than, equal to, or greater than *x*. Your task is to use this information to successfully query *m*. Good luck!

Time Allocation: 1 second

Input/Output: This is an interactive problem. This means that your program will receive input from the grading environment based on the output your program produces. All input and output will be done through the console.

Rules of interaction:

1. Your program should begin by reading in a single integer *n*.
2. Then your program should output an integer *x*, representing a query about the hidden integer *m*.
3. You MUST output a new line character and flush the output stream after each output!
4. Each query will result in an integer response *k*, which will be either `-1`, `0`, or `1`, where `-1` indicates that *m* is less than *x*, `1` indicates that *m* is greater than *x*, and `0` indicates that you have successfully queried *m*.
5. You can make as many queries as you want, but your program should terminate after *m* has been successfully queried.

Assumptions and Expectations: *n* will be an integer from 1 to 1,000,000,000 inclusive.
m will be an integer from 1 to *n*, inclusive.
x must be an integer from 1 to *n*, inclusive.
If any output is invalid, your program will be deemed incorrect.

Sample Run:

```
Input: 4
Output: 2
Input: 1
Output: 3
Input: 0
```