Task 1: `AskOneGetOneFree`

You are playing a guessing game with Bob. First, Bob announces a positive integer N. Next, Bob chooses two secret numbers, say x and y, where both are integers within the range $0 \leq x < N$ and $0 \leq y < N$. It is possible that x is the same as y. You can ask Bob a series of questions regarding the value of x and y. Your goal is to correctly determine the values of x and y using as little number of questions as possible. You can't ask a question like "*What is the value of x and y?*". Your questions must be of a certain format. In each question, you choose a number r, and the question is of the form:

Are the numbers greater or equal to r?

Bob must honestly tells you whether $x \geq r$, and whether $y \geq r$. After you have received the answer, you can choose to ask the next question, or make a guess of the value of x and y. Below is an example of the interactions with Bob.

Bob:	Let's choose N to be 4.
You:	Are the numbers greater or equal to 3?
Bob:	Yes for x; no for y.
You:	Are the numbers greater or equal to 1?
Bob:	Yes for x; yes for y.
You:	Are the numbers greater or equal to 2?
Bob:	Yes for x; yes for y.
You:	$x = 3$ and $y = 2$.
Bob:	You are correct. You have asked 3 questions.

Bob always answers the questions honestly. However, at any point during the game, he may modify the secrets x, y as long as the modified secrets do not contradict his previous answers. For example, in the above game, Bob initially could had chosen $x = 3$ and $y = 0$, but after seen the second question, modified the secrets to $x = 3$ and $y = 2$. Note that the modified secrets do not contradict the first answer (i.e. "Yes for x; no for y,") and thus is allowed. With this flexibility, Bob can adopt different strategies. For instance, he can be very simple-minded: chooses two secrets randomly initially, and does not modify them. Bob can also be very cunning: to respond to a question, he modifies the secrets so as to force the opponent to ask many questions.

Interactions

Bob has written a program that uses the cunning strategy. You have to submit a program that interacts with Bob's program. The last few sections give details on how to submit your program. In this contest, we have invited a well-known programmer Ada, and she will also submit a program.

Subtasks

1. (30 marks) Bob always chooses $N = 12$. Let Q be the number of questions your program has asked. If your program has made a wrong guess, no mark will be awarded. Otherwise, the number of marks awarded is calculated as follow:

$8 < Q < 12$,	10 marks.
$6 < Q \leq 8$,	20 marks.
$Q \leq 6$,	30 marks.

   (Hints: Note that a simple strategy that asks many questions can already obtain some marks.)

2. (30 marks) Bob chooses a $N \leq 2000$. Bob also chooses the same value of N when playing with Ada. Let Q be the number of questions your program has asked, and X be the number of questions Ada's program has asked. If your program has made a wrong guess, no mark will be awarded. Otherwise,

$25 < (Q - X) < N$	10 marks
$2 < (Q - X) \leq 25$	15 marks
$(Q - X) = 2$	20 marks
$(Q - X) = 1$	25 marks
$(Q - X) \leq 0$	30 marks

3. (40 marks) Bob chooses a $N \leq 10000$. Similarly, let Q and X as defined in the previous subtask.

$(Q - X) = 2$	10 marks
$(Q - X) = 1$	20 marks
$(Q - X) \leq 0$	40 marks

Each subtask will be evaluated multiple times. The final number of marks awarded for that subtask is the minimum among the different evaluations. For example, suppose during the first evaluation of subtask 2, Bob chose N=1000 and your program was awarded 30 marks. However, during the second evaluation, Bob chose N=2000 and your program was awarded 25 marks. Without further evaluation, the final number of marks awarded would be $\min(25, 30) = 25$.

How to submit your solution

You can obtain a set of programs from the contest web page.

In this task, you only have to submit a subroutine (either c, c++ or Pascal). The following files are examples that you have to submit:

- `askonegetonefree.c`
- `askonegetonefree.cpp`
- `askonegetonefree.pas`

The program you submitted must implement the subroutine `guess(N)` where the parameter `N` gives the positive integer that Bob has chosen. Within the subroutine, you can call a function `query(r)` (which is written by Bob) multiple times. Calling the function `query(r)` corresponds to asking Bob the question: *are the numbers greater or equal to r?* The function `query(r)` returns an object `Answer`, containing two integers p and q (see the given programs on the datatype of `Answer`). If Bob's answer for the secret x is yes (i.e. $x \geq r$), then the returned value p is 1; otherwise it is 0. Likewise, If Bob's answer for the secret y is yes (i.e. $y \geq r$), then the returned value q is 1; otherwise it is 0.

When you (or rather your subroutine) believe that you have found the secrets, exit the routine and return the guess as an `Answer` object. For example, if you believe that the secrets are 3 and 2, store them in the `Answer` object and return it.

For debugging and testing, you may want to write a program for Bob. The programs `bob.pas`, `bob.c`, `bob.cpp` are examples of Bob's programs (of course, the programs by the cunning Bob are not provided). You can see that these programs call your subroutine `guess(N)`. The given Bob's programs use a silly strategy. It always chooses $N = 4$, $x = 3$ and $y = 2$. We stress that during evaluation, another program that uses the cunning strategy is used. To compile your program together with Bob's program, use the following command:

- For C: `gcc -O2 bob.c askonegetonefree.c -o game`
- For C++: `g++ -O2 bob.cpp askonegetonefree.cpp -o game`
- For Pascal: `fpc -O2 bob.pas -ogame`

Now, by running the compiled executable `./game`, a game between Bob and your subroutine/function will be played.

Important notes

Your routine `query(r)` must not perform input/output operations, like reading from the standard input (the keyboard), or writing to the standard output (the display).

Additional note for Pascal

Your program must include the library `askonegetonefreelib`, similar to the sample pascal program. In addition, you must not declare a variable/function with the following name:

`askonegetonefreecheckanswer`