Archaeology

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

This is an interactive problem.

In one of the N^2 integer points of the square $\mathbb{D} = [1; N] \times [1; N]$, a mysterious artifact is hidden. Your task is to find this artifact. You have a radar, and its single use is arranged as follows:

- First, you place the radar at any integer point belonging to $\mathbb{D}$.
- If the artifact is exactly at that point, the radar lights up green, the coordinates of the point where the artifact and the radar are located will be displayed on the screen, and the archaeological excavation ends there.
- Otherwise, suppose you placed the radar at point R, and the artifact is at point $A \neq R$. Then the radar will light up yellow and display the coordinates of an integer point $B \in \mathbb{D}$ such that $\angle ARB < 90°$. Among all such integer points, the radar will display one uniformly at random; in particular, if $A \neq R$, there is always a chance that the radar will show the coordinates of point A (since $\angle ARA = 0° < 90°$), but it will never show point R.

You can assume that, except for the first five tests (described in the Notes section), among all possible points in the square $\mathbb{D}$, the artifact is hidden at a uniformly selected random point, which is fixed in each test and does not depend on the responses of your program. All the random choices described above are made independently.

Your radar is not fully charged, so you can use it no more than 90 times. Write a program that will indicate where to place the radar so that it eventually lights up green.

Interaction Protocol

The first line contains a single integer N: the side of the square in which the artifact is hidden ($1 \leq N \leq 10^9$). Then your program should print requests to use the radar. Each request consists of two integers, R_{xi} and R_{yi}: the coordinates of the next point where you want to place the radar ($1 \leq R_{xi}, R_{yi} \leq N$). After each request, you must print a newline and flush the output buffer. After that, you can read two integers, B_{xi} and B_{yi}: the coordinates of the point displayed on the radar screen ($1 \leq B_{xi}, B_{yi} \leq N$). If $B_{xi} = R_{xi}$ and $B_{yi} = R_{yi}$, you have successfully passed the test, and the program should terminate immediately to receive the `OK` verdict on this test. Otherwise, if you still have not made 90 requests, you should continue making requests.

Your program should terminate immediately if, after making 90 requests, the radar has not yet lit up green, to receive a `Wrong Answer` verdict; otherwise, the verdict may be unpredictable.

Examples

standard input	*standard output*
1 1 1	1 1
2 2 2 1 2 1 2 2 2 1 1 1 2 1 1 1 2	1 1 1 1 2 1 2 1 2 1 2 2 2 2 1 2

Note

In this problem, there are exactly 50 tests, including examples.

Test 1 coincides with the first example.

In tests 2–5, $N = 2$. The artifact in the second test is hidden in the point $(1, 2)$ as shown in the second example; however, it is not guaranteed that in the second test, the interactor will provide the same responses to requests as in the example. Tests 2–5 are guaranteed to feature all four possible artifact places.

In the next 28 tests, N consecutively goes through numbers $4, 8, 15, 30, \ldots, \frac{10^9}{4}, \frac{10^9}{2}$. Formally, for each $i \in \{6, \ldots, 33\}$, in the i-th test $N = \lceil 10^9 / 2^{34-i} \rceil$. In each of these tests, the artifact is hidden at a uniformly selected random point.

In the next 17 tests, N consecutively goes through numbers $10^9 - 5, \ldots, 10^9 - 1, 10^9, 10^9, \ldots, 10^9$. Formally, for each $i \in \{34, \ldots, 50\}$, in the i-th test $N = 10^9 + \min\{0, i - 39\}$. In these 17 tests, the artifact is also each time hidden at a uniformly selected random point.

If several submissions are made and in some test the corresponding programs make the same queries, they will receive identical answers from the interacting jury program.