Problem I. Points and Segments

Time limit:	1 second
Memory limit:	512 megabytes

This is interactive problem.

Alice and Bob are playing a new game "Points and Segments" at the boring Geometry lesson.

The game board is a sheet of paper, it has n distinct points, no three of those are on the same line. The players make moves in turn, Alice moves first. The current player's move is to choose two points and connect them by a segment. The new segment must not have common internal points with the previous segments (they can have a common endpoint). The player who cannot make a move loses.

In this problem your program will play "Points and Segments" with the judges program, and it must win. Your program will get a board with n points and must first choose whether it would like to play for Alice, or for Bob. After that your program must make moves for the chosen player until it wins.

Interaction Protocol

First your program must read a single integer n — the number of points on the board ($3 \le n \le 300$).

After that read n pairs of integers (x_i, y_i) — coordinates of the points ($-10^4 \le x_i, y_i \le 10^4$; all points are distinct, no three points are on the same line).

After analyzing the field, your program must decide whether it will play for Alice and move first, or play for Bob and move second. Print 1 if it wants to play for Alice, print 2 if it wants to play for Bob. Don't forget to end the line after this output.

After that the players make moves in turn, according to the choice your program has made. If it's your program's move, it should print two integers on a line: i and j — the indices of points that it would like to connect by a segment ($1 \le i, j \le n$; i and j must be distinct, the segment must not have common internal points with existing segments). If it is judges program's turn, it prints its move in the same format, your program must read it from its standard input.

If your program is in a situation that it cannot make a move, it will be forced to terminate by the judging system. The judging system will not wait for its reaction. If your program wins, and the judges program has no move, the judges program will print "`0 0`" instead of its move. Your program must read these two zeroes and terminate.

Examples

standard input	standard output
3	
0 0	
10 0	
0 10	
	1
	1 2
1 3	
	2 3
0 0	
4	
0 0	
10 0	
5 7	
5 3	
	2
1 2	
	1 4
1 3	
	2 4
2 3	
	3 4
0 0	

Note

In the example above the judges and participant's programs messages are formatted with empty lines to show which message is a response to which one. In real interaction there are no empty lines, do not print them. But you must end each printed message with a new line character.