

Problem L. The Five Bishops

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

This is an interactive problem.

There are five White Bishops and one Black King on an infinite chessboard. You are playing White. Your task is to **checkmate or stalemate** the King or to determine whether the King could avoid the checkmate or stalemate. White moves first.

You will be given 50 moves to do that. If after 50-th move of White side, the King is still not checkmated/stalemated, the answer will be considered incorrect.

The initial coordinates of pieces will be not greater than 10^6 by an absolute value. During the game, all coordinates must not exceed 10^9 by an absolute value.

Interaction Protocol

The first line of the input will contain only one single integer t , the number of problem instances to solve ($1 \leq t \leq 1000$).

Each instance starts with six pairs of integers x_i and y_i on a single line: the first five are coordinates of Bishops, the last pair is the coordinates of the King. All these integers do not exceed 10^6 by an absolute value.

Then the game starts. Your move consists of four integers $x_1 \ y_1 \ x_2 \ y_2$ on a single line: the initial and final coordinates of a Bishop. The interactor responds with four integers $x_1 \ y_1 \ x_2 \ y_2$ on a single line: the initial and final coordinates of the King. When the King is checkmated/stalemated, the interactor responds with four zeros, followed by the next problem instance if there is one. If your program decides that there is no way to checkmate/stalemate the King, it must output four zeros instead of the first move, in this case the interactor immediately responds with the next problem instance if there is one. The coordinates of moving pieces must not exceed 10^9 by an absolute value.

See sample interaction for more details.

If the checkmate/stalemate is possible, your program must do it in no more than 50 moves, otherwise the answer will be considered invalid (remember fifty-move rule). You don't need to find the optimal solution, but it is guaranteed that in every case where the solution exists it is possible to achieve the goal in no more than 50 moves.

If the checkmate/stalemate is not possible, your program should immediately respond with the line of four zeros (no moves allowed), otherwise the answer will be considered incorrect.

It is always guaranteed that every problem instance is correct, that is, no two pieces occupy the same cell and the King is not in check.

After each printed line, flush the output buffer: this can be done by calling, for example, `fflush (stdout)` in C or C++, `System.out.flush ()` in Java, or `sys.stdout.flush ()` in Python.

Example

standard input	standard output
4	
1 1 2 2 3 3 4 4 5 5 7 5	0 0 0 0
1 1 1 2 1 4 1 5 2 2 3 5	2 2 1 3
0 0 0 0	
1 2 2 3 4 5 5 6 6 7 5 3	0 0 0 0
3 2 4 2 3 6 4 6 6 4 3 4	6 4 5 5
0 0 0 0	

Explanation

Empty lines are added only to show the sequence of events.

In the first and third test cases there are five same-color bishops. In the second test case the checkmate could be achieved in one move. In the fourth case the stalemate could be achieved in one move.