Problem A. Anxious About Airdrops

Time limit: 1 second

There is a new game called "Well-known problemgrounds" out there. In this game n people are standing on a 2-dimensional grid, hoping to solve some problems.

The game proceeds as follows. At some point the judges summon an airdrop somewhere on the board. The airdrop contains a statement of a well-known problem. The players rush to the place of the airdrop immediately with constant velocity of 1 cell per second, longing to solve the task (and vainly hoping for some fresh ideas). However, they only move horizontally and vertically.

More specifically, all of them follow one and the same algorithm: at any second, if their column is not the same as the column of the airdrop, they move horizontally in the direction of the airdrop's column. If the airdrop is in the same column, they move vertically in the direction of the airdrop. The first player (or players) to reach the airdrop stops and solves the task in 1 second (the other players will continue moving towards the airdrop during this second) and announces this to everyone else. At this point everyone stops and starts waiting for the next airdrop to be deployed.

As a jury, you came up with $n - 1$ problems and want to deploy some of them in such a way that in the end all players will gather in one and the same place. You know the starting position of each player. Decide where to deploy the airdrops to achieve your goal.

Input

The first line contains a single integer $1 \le n \le 5000$ — the number of players. Each of the next n lines contains two integers x_i, y_i ($-10^8 \le x_i, y_i \le 10^8$) — the initial coordinates of the i-th player.

Output

On the first line of the output print a single integer $0 \le k \le n - 1$ — the number of airdrops deployed. Then output k lines, each containing two integers x and y ($-10^8 \le x, y \le 10^8$): the i-th of those lines should contain the coordinates of the point where the i-th airdrop is deployed. If there are several possible answers, output any one of them. In particular, you don't have to minimize k.

Examples

standard input	*standard output*
2 0 0 4 5	1 4 0
8 3 4 4 3 -3 4 4 -3 -3 -4 -4 -3 3 -4 -4 3	1 0 0

Note

In the first example the distances between the players and the location of the airdrop are 4 and 5 respectively. The first player will be the first to reach the location, but since he needs 1 second to solve the task, the second player will also reach the location.

In the second example all players are at the same distance $3 + 4 = 7$ from the airdrop, so everyone will get there simultaneously.