ACM-ICPC Live Archive

5895 Locksmith

Mooks A, C, and M have been charged with creating locks for the Riddler's lair. They need to design locks comprised of flat interlocking pieces that can be unlocked only by separating these pieces while sliding them around on the surface of the doors. Unfortunately, while the mooks have come up with quite a few designs, they are not very good at determining which designs can actually be unlocked.

The pieces of the lock are axis-aligned polygons (that is, the sides are all either horizontal or vertical) in the plane. The lock can be manipulated by sliding any one polygon at a time in any direction, so long as doing so does not cause the polygon to overlap with any other polygon. Rotating a polygon is not allowed. A polygon is considered separable from the rest of the lock if there exists a sequence of moves (possibly involving several polygons) leading to a configuration such that it is possible to draw a straight line between the polygon and the remainder of the lock. Given a proposed lock configuration, your job is to determine the number of separable polygons.

Input

The input file will contain multiple test cases. Each test case begins with a line with a single integer N denoting the number of pieces in the lock. This line is followed by N lines of space-separated integers each with the following format:

$c\ x_1\ y_1\ \ldots\ x_c\ y_c$

Here, c denotes the number of vertices in the piece, and x_i and y_i give the locations of the vertices in clockwise order. The sides of each piece are all either horizontal or vertical, and they alternate between the two. The pieces are guaranteed to have no overlapping area. Each test case has either 2 or 3 pieces, and for each test case, these pieces have a total of at most 30 vertices. All coordinates given are integers between 0 and 1000 inclusive. Input will be terminated by a case with 0 pieces, which should not be processed.

Output

For each input test case, print the number of separable pieces in the lock.

Sample Input

```
2
12 0 0 0 6 9 6 9 0 6 0 6 1 8 1 8 5 1 5 1 1 3 1 3 0
4 2 2 2 4 7 4 7 2
2
12 0 0 0 6 9 6 9 0 6 0 6 1 8 1 8 5 1 5 1 1 3 1 3 0
4 4 2 4 4 7 4 7 2
0
```

Sample Output

```
0
2
```