Annotation:

1. This a 2013 Bronze problem, very difficult, need strong java programing basis

2. Do not use a inner/nested class unless you want it be more capsulated, because its initiation is complex: OuterClass.InnerClass innerObject = outerObject.new InnerClass();

3. wormhole problem has 5 main difficulties:

(1). how to create a convenient wormhole class?
My design is to contain axis: int x, y; its partner: wormhole wh; and boolean accessed;
The answer only includes int x, y, and its partner: int
(mention: it is very important to use int, cause int could be used in array as index)()

(2). how to store wormhole's partner?
My plan is use a wormhole partner to represent its pair, null means not pair yet. But actually using int partner; and an point[0] as an extra space to represent no pair (wormholes[i] = new point[N + 1]; next_on_right = new int[N + 1];) is more convenient.

(3). how to find next_on_right wormhole?
For me, no way. The answer use a global static int[] next_on_right; and them use two nested for loop to set next_on_right wormhole, and a next_on_right[0] means no right side wormhole, this is very hard to think out!

(4). how to pairing the worms(all possibility) , and how can count the "trap pairings"?
For me, I use recursion to return count, to solve the base case, I use a checked list to see if all the pairing has been down. The answer take really good use of for loop, like:
for (i = 1; i <= N && wormholes[i].partner != 0; i++) {}
If the for loop is quit for some wormhole's partner not found, then pair it with closest next wormhole, then use: total += pairing(); to recursively solve the problem and includes all the possibilities.

If the for loop is quit for i > N, means all partner found, now just need to check using hasCycle() to decide return 1 or 0;

(5). how to checkHasCycle?
For me, I use a very original way to check by polling each wormhole to see if they have been accessed before. The answer, however, by taking advantage of "int partner", use:
pos = next_on_right[wormholes[pos].partner];(wrapped in a for loop) to check. If after quit for loop, pos != 0, means there is a cycle.

4. Finally, Arrays.fill(next_on_right, 0) is a good a way to initiate an array.