```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  // O(n^2) time | O(n) space - where n is the number of coordinates
  public static int rectangleMania(Point[] coords) {
    Set<String> coordsTable = getCoordsTable(coords);
    return getRectangleCount(coords, coordsTable);
  }

  public static Set<String> getCoordsTable(Point[] coords) {
    Set<String> coordsTable = new HashSet<String>();
    for (Point coord : coords) {
      String coordString = coordToString(coord);
      coordsTable.add(coordString);
    }
    return coordsTable;
  }

  public static int getRectangleCount(Point[] coords, Set<String> coordsTable) {
    int rectangleCount = 0;
    for (Point coord1 : coords) {
      for (Point coord2 : coords) {
        if (!isInUpperRight(coord1, coord2)) continue;
        String upperCoordString = coordToString(new Point(coord1.x, coord2.y));
        String rightCoordString = coordToString(new Point(coord2.x, coord1.y));
        if (coordsTable.contains(upperCoordString) && coordsTable.contains(rightCoordString))
          rectangleCount++;
      }
    }
    return rectangleCount;
  }

  public static boolean isInUpperRight(Point coord1, Point coord2) {
    return coord2.x > coord1.x && coord2.y > coord1.y;
  }

  public static String coordToString(Point coord) {
    return Integer.toString(coord.x) + "-" + Integer.toString(coord.y);
  }

  static class Point {
    public int x;
    public int y;

    public Point(int x, int y) {
      this.x = x;
      this.y = y;
    }
  }
}

```