AlgoExpert | Quad Layout | Java | 12px | Sublime | Monokai | 00:00:00

Prompt | Scratchpad | Our Solution(s) | Video Explanation | Run Code

Solution 1 | Solution 2 | Solution 3

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

import java.util.*;

class Program {
  static String UP = "up";
  static String RIGHT = "right";
  static String DOWN = "down";

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

  public static Map<String, Map<Integer, List<Point>>> getCoordsTable(Point[] coords) {
    Map<String, Map<Integer, List<Point>>> coordsTable =
        new HashMap<String, Map<Integer, List<Point>>>();
    coordsTable.put("x", new HashMap<Integer, List<Point>>());
    coordsTable.put("y", new HashMap<Integer, List<Point>>());
    for (Point coord : coords) {
      if (!coordsTable.get("x").containsKey(coord.x)) {
        coordsTable.get("x").put(coord.x, new ArrayList<Point>());
      }
      if (!coordsTable.get("y").containsKey(coord.y)) {
        coordsTable.get("y").put(coord.y, new ArrayList<Point>());
      }
      coordsTable.get("x").get(coord.x).add(coord);
      coordsTable.get("y").get(coord.y).add(coord);
    }
    return coordsTable;
  }

  public static int getRectangleCount(
      Point[] coords, Map<String, Map<Integer, List<Point>>> coordsTable) {
    int rectangleCount = 0;
    for (Point coord : coords) {
      int lowerLeftY = coord.y;
      rectangleCount += clockwiseCountRectangles(coord, coordsTable, UP, lowerLeftY);
    }
    return rectangleCount;
  }

  public static int clockwiseCountRectangles(
      Point coord1,
      Map<String, Map<Integer, List<Point>>> coordsTable,
      String direction,
      int lowerLeftY) {
    if (direction == DOWN) {
      List<Point> relevantCoords = coordsTable.get("x").get(coord1.x);
      for (Point coord2 : relevantCoords) {
        int lowerRightY = coord2.y;
        if (lowerRightY == lowerLeftY) return 1;
      }
      return 0;
    } else {
      int rectangleCount = 0;
      if (direction == UP) {
        List<Point> relevantCoords = coordsTable.get("x").get(coord1.x);
        for (Point coord2 : relevantCoords) {
          boolean isAbove = coord2.y > coord1.y;
          if (isAbove)
            rectangleCount += clockwiseCountRectangles(coord2, coordsTable, RIGHT, lowerLeftY);
        }
      } else if (direction == RIGHT) {
        List<Point> relevantCoords = coordsTable.get("y").get(coord1.y);
        for (Point coord2 : relevantCoords) {
          boolean isRight = coord2.x > coord1.x;
          if (isRight)
            rectangleCount += clockwiseCountRectangles(coord2, coordsTable, DOWN, lowerLeftY);
        }
      }
      return rectangleCount;
    }
  }

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

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