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";
  static String LEFT = "left";

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

  public static Map<String, Map<String, List<Point>>> getCoordsTable(Point[] coords) {
    Map<String, Map<String, List<Point>>> coordsTable =
        new HashMap<String, Map<String, List<Point>>>();
    for (Point coord1 : coords) {
      Map<String, List<Point>> coord1Directions = new HashMap<String, List<Point>>();
      coord1Directions.put(UP, new ArrayList<Point>());
      coord1Directions.put(RIGHT, new ArrayList<Point>());
      coord1Directions.put(DOWN, new ArrayList<Point>());
      coord1Directions.put(LEFT, new ArrayList<Point>());
      for (Point coord2 : coords) {
        String coord2Direction = getCoordDirection(coord1, coord2);
        if (coord1Directions.containsKey(coord2Direction))
          coord1Directions.get(coord2Direction).add(coord2);
      }
      String coord1String = coordToString(coord1);
      coordsTable.put(coord1String, coord1Directions);
    }
    return coordsTable;
  }

  public static String getCoordDirection(Point coord1, Point coord2) {
    if (coord2.y == coord1.y) {
      if (coord2.x > coord1.x) {
        return RIGHT;
      } else if (coord2.x < coord1.x) {
        return LEFT;
      }
    } else if (coord2.x == coord1.x) {
      if (coord2.y > coord1.y) {
        return UP;
      } else if (coord2.y < coord1.y) {
        return DOWN;
      }
    }
    return "";
  }

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

  public static int clockwiseCountRectangles(
      Point coord,
      Map<String, Map<String, List<Point>>> coordsTable,
      String direction,
      Point origin) {
    String coordString = coordToString(coord);
    if (direction == LEFT) {
      boolean rectangleFound = coordsTable.get(coordString).get(LEFT).contains(origin);
      return rectangleFound ? 1 : 0;
    } else {
      int rectangleCount = 0;
      String nextDirection = getNextClockwiseDirection(direction);
      for (Point nextCoord : coordsTable.get(coordString).get(direction)) {
        rectangleCount += clockwiseCountRectangles(nextCoord, coordsTable, nextDirection, origin);
      }
      return rectangleCount;
    }
  }

  public static String getNextClockwiseDirection(String direction) {
    if (direction == UP) return RIGHT;
    if (direction == RIGHT) return DOWN;
    if (direction == DOWN) return LEFT;
    return "";
  }

  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;
    }

    public boolean equals(Object a) {
      return this.x == ((Point) a).x && this.y == ((Point) a).y;
    }
  }
}
```