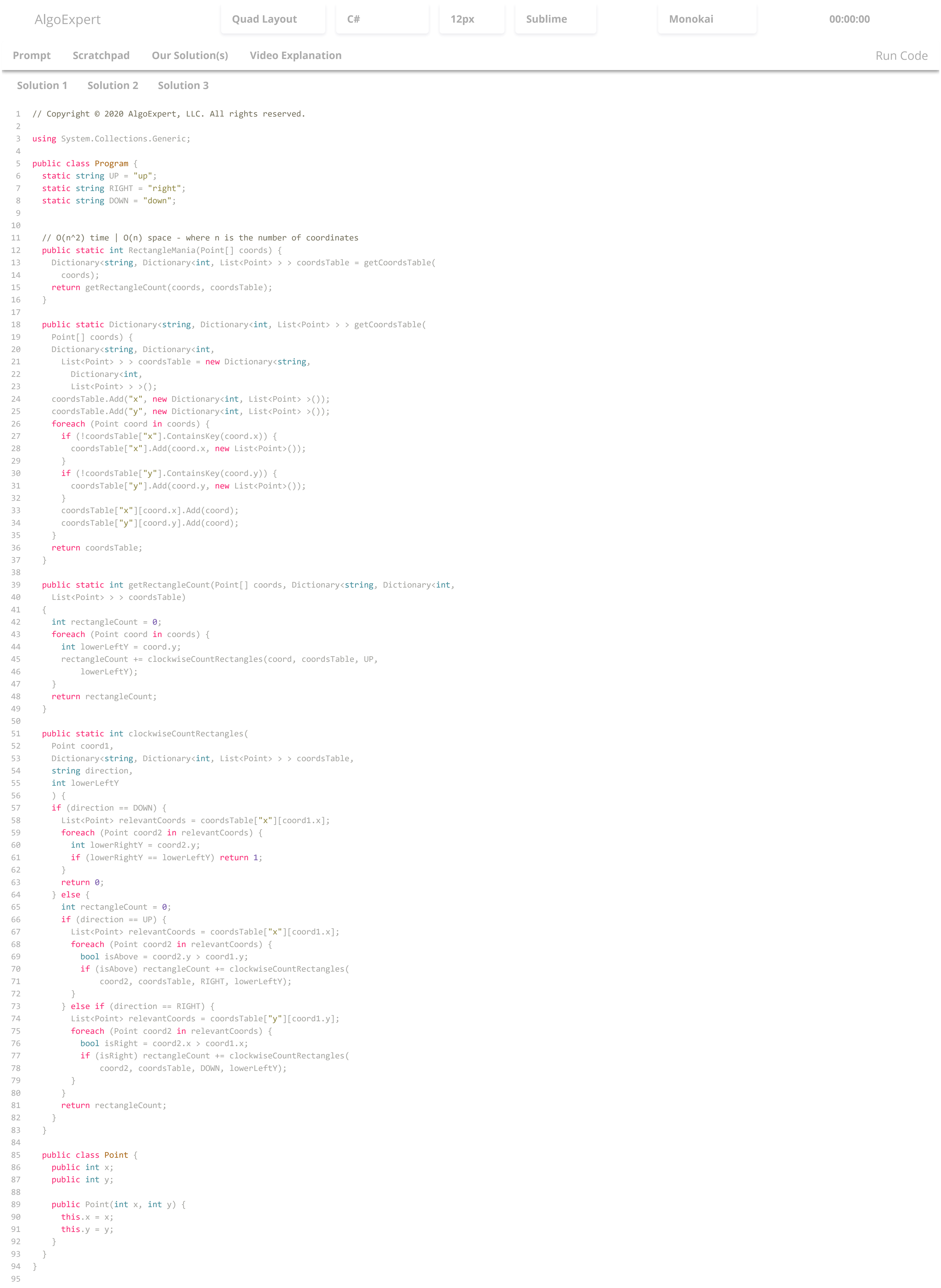

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

using System.Collections.Generic;

public 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) {
    Dictionary<string, Dictionary<int, List<Point> > > coordsTable = getCoordsTable(
      coords);
    return getRectangleCount(coords, coordsTable);
  }

  public static Dictionary<string, Dictionary<int, List<Point> > > getCoordsTable(
    Point[] coords) {
    Dictionary<string, Dictionary<int,
      List<Point> > > coordsTable = new Dictionary<string,
        Dictionary<int,
        List<Point> > >();
    coordsTable.Add("x", new Dictionary<int, List<Point> >());
    coordsTable.Add("y", new Dictionary<int, List<Point> >());
    foreach (Point coord in coords) {
      if (!coordsTable["x"].ContainsKey(coord.x)) {
        coordsTable["x"].Add(coord.x, new List<Point>());
      }
      if (!coordsTable["y"].ContainsKey(coord.y)) {
        coordsTable["y"].Add(coord.y, new List<Point>());
      }
      coordsTable["x"][coord.x].Add(coord);
      coordsTable["y"][coord.y].Add(coord);
    }
    return coordsTable;
  }

  public static int getRectangleCount(Point[] coords, Dictionary<string, Dictionary<int,
    List<Point> > > coordsTable)
  {
    int rectangleCount = 0;
    foreach (Point coord in coords) {
      int lowerLeftY = coord.y;
      rectangleCount += clockwiseCountRectangles(coord, coordsTable, UP,
          lowerLeftY);
    }
    return rectangleCount;
  }

  public static int clockwiseCountRectangles(
    Point coord1,
    Dictionary<string, Dictionary<int, List<Point> > > coordsTable,
    string direction,
    int lowerLeftY
    ) {
    if (direction == DOWN) {
      List<Point> relevantCoords = coordsTable["x"][coord1.x];
      foreach (Point coord2 in relevantCoords) {
        int lowerRightY = coord2.y;
        if (lowerRightY == lowerLeftY) return 1;
      }
      return 0;
    } else {
      int rectangleCount = 0;
      if (direction == UP) {
        List<Point> relevantCoords = coordsTable["x"][coord1.x];
        foreach (Point coord2 in relevantCoords) {
          bool isAbove = coord2.y > coord1.y;
          if (isAbove) rectangleCount += clockwiseCountRectangles(
              coord2, coordsTable, RIGHT, lowerLeftY);
        }
      } else if (direction == RIGHT) {
        List<Point> relevantCoords = coordsTable["y"][coord1.y];
        foreach (Point coord2 in relevantCoords) {
          bool isRight = coord2.x > coord1.x;
          if (isRight) rectangleCount += clockwiseCountRectangles(
              coord2, coordsTable, DOWN, lowerLeftY);
        }
      }
      return rectangleCount;
    }
  }

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

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