How Many Perfect Ways to Work?

Filename: `paths`

Nick enjoys the beauty and perfection in nature. Unfortunately, Nick lives in a huge city that is mostly devoid of the trees and serenity of nature. Instead, Nick's city has many, many roads set up in a perfect grid, so he must settle for finding perfect paths within this grid. In the grid, a path consists of a sequence of movements in one of four possible directions: north(N), south(S), east(E) or west(W). Assume that each movement in a direction is a single block. A perfect path does not contain any movements in directly opposite directions. Thus, the sequence of movements NNENE forms a perfect path, but the sequence NEESN does not, since it contains both a north and south movement. To complicate matters, there are some street intersections that have graffiti which Nick considers imperfect. Thus, any path that goes through these intersections is not a perfect path.

The Problem:

Given grid coordinates for Nick's starting location and destination, as well as coordinates of all the imperfect intersections Nick is to avoid, you are to determine the number of perfect paths Nick can take.

The Input:

There will be several sets of input. The first line will contain a single positive integer n $(n < 100)$ describing the number of test cases in the data set. The first line in each data set has a single integer m $(0 \leq m < 10)$, which represents the number of intersections in the town for that particular data set that Nick must avoid. The following m lines contain the coordinates (x, y) of the m intersections to avoid. All x and y coordinates will be non-negative integers less than 10, with the x coordinate appearing first on a line, followed by the y coordinate, separated by a single space. The next line in the data set will be a single positive integer p $(0 < p < 10)$ that represents the number of trips for which you will be calculating the number of perfect paths for that data set. The last p lines of the data set contain the p trips. Each line will contain two pairs of (x,y) coordinates. The first pair will be the coordinates for Nick's starting location and the second pair will be the coordinates for Nick's destination. Each coordinate on each line is separated by a single space from the previous and subsequent coordinates. You are guaranteed that none of Nick's starting locations or destinations will be an intersection he is supposed to avoid. All of these coordinates will also be non-negative integers less than 10 and be separated by spaces on each line.

The Output:

For each data set, you will output a single line header of the following format:

`Data Set k:`

where k is an integer in between 1 and n, inclusive.

Follow this with a blank line, and then p lines, each with one of the two following formats.

```
Test Case c: Nick can take P perfect paths.

Test Case c: Nick can take 1 perfect path.
```

where c is an integer in between 1 and p, inclusive and P will be a non-negative integer less than 2147483647. Use the second format only if P=1. Please indent each of these lines exactly 2 spaces from the left margin. Also, leave a blank line in between data sets.

Sample Input:

```
2
4
2 2
3 5
1 0
4 4
5
0 0 1 9
0 1 2 3
2 1 0 5
0 1 4 5
0 0 0 0
3
1 1
2 2
3 3
1
4 4 9 7
```

Sample Output:

```
Data Set 1:

  Test Case 1: Nick can take 9 perfect paths.
  Test Case 2: Nick can take 3 perfect paths.
  Test Case 3: Nick can take 5 perfect paths.
  Test Case 4: Nick can take 0 perfect paths.
  Test Case 5: Nick can take 1 perfect path.

Data Set 2:

  Test Case 1: Nick can take 56 perfect paths.
```