Problem 1274: Number of Ships in a Rectangle

Problem Information

Difficulty: Hard
Acceptance Rate: 68.81%
Paid Only: Yes
Tags: Array, Divide and Conquer, Interactive

Problem Description

(This problem is an**interactive problem**.)

Each ship is located at an integer point on the sea represented by a cartesian plane, and each integer point may contain at most 1 ship.

You have a function `Sea.hasShips(topRight, bottomLeft)` which takes two points as arguments and returns `true` If there is at least one ship in the rectangle represented by the two points, including on the boundary.

Given two points: the top right and bottom left corners of a rectangle, return the number of ships present in that rectangle. It is guaranteed that there are **at most 10 ships** in that rectangle.

Submissions making **more than 400 calls** to `hasShips` will be judged _Wrong Answer_. Also, any solutions that attempt to circumvent the judge will result in disqualification.

Example :

![](https://assets.leetcode.com/uploads/2019/07/26/1445_example_1.PNG)

Input: ships = [[1,1],[2,2],[3,3],[5,5]], topRight = [4,4], bottomLeft = [0,0] **Output:** 3 **Explanation:** From [0,0] to [4,4] we can count 3 ships within the range.

Example 2:

Input: ans = [[1,1],[2,2],[3,3]], topRight = [1000,1000], bottomLeft = [0,0] **Output:** 3

Constraints:

* On the input `ships` is only given to initialize the map internally. You must solve this problem "blindfolded". In other words, you must find the answer using the given `hasShips` API, without knowing the `ships` position. * `0 <= bottomLeft[0] <= topRight[0] <= 1000` * `0 <= bottomLeft[1] <= topRight[1] <= 1000` * `topRight != bottomLeft`

Code Snippets

C++:

```
/**
* // This is Sea's API interface.
* // You should not implement it, or speculate about its implementation
* class Sea {
* public:
* bool hasShips(vector<int> topRight, vector<int> bottomLeft);
* };
*/

class Solution {
public:
int countShips(Sea sea, vector<int> topRight, vector<int> bottomLeft) {

}
};
```

Java:

```
/**
* // This is Sea's API interface.
* // You should not implement it, or speculate about its implementation
* class Sea {
* public boolean hasShips(int[] topRight, int[] bottomLeft);
* }
*/

class Solution {
public int countShips(Sea sea, int[] topRight, int[] bottomLeft) {

}
}
```

Python3:

```
# """
# This is Sea's API interface.
# You should not implement it, or speculate about its implementation
# """
#class Sea:
# def hasShips(self, topRight: 'Point', bottomLeft: 'Point') -> bool:
#
#class Point:
# def __init__(self, x: int, y: int):
# self.x = x
# self.y = y

class Solution:
def countShips(self, sea: 'Sea', topRight: 'Point', bottomLeft: 'Point') -> int:
```