CS 4820, Spring 2018	**Homework 5, Problem 3**
Name: Ronguang Wang	
NetID: rw564	
Collaborators: Siyuao Liu (sl2928); Yihao Chen (yc2288)	

(3) *(10 points)* A closed axis-parallel rectangle in the plane is a subset $R \subset \mathbb{R}^2$ that is the Cartesian product of two closed intervals,

$$R = [a, b] \times [c, d] = \{(x, y) \mid a \leq x \leq b \text{ and } c \leq y \leq d\}.$$

We call the 4-tuple (a, b, c, d) the *description* of R. Given the descriptions of n rectangles, design an algorithm to decide whether there exists a point in the plane that belongs to two or more of the rectangles. Your algorithm's running time should be $O(n \log n)$.

Solution:

The problem is aiming to develop an algorithm which can find whether there exists a point where two or more rectangles are overlapped on the plane. Given that a, b, c, d are the four corners of each rectangle and each rectangles edges are parallel to the x and y axises. The upper bound of the time complexity should be $O(n \log n)$.

The edges (a_i, b_i) of the rectangles which are parallel to the x-axis can be treated as intervals (s_i, f_i) which are the starting time and the finishing time of each interval. Similarly, the edges (c_i, d_i) of the rectangles which are parallel to the y-axis can be treated as intervals (s_j, f_j) which are the starting time and the finishing time of each interval. First, the intervals along the x-axis should be checked to see whether there is a pair of intervals are conflict which means that two rectangles may overlap. Then, the intervals along the y-axis should be checked to see whether there is a pair of intervals are conflict which means that two rectangles may overlap. After the sets X and Y of conflicting interval pairs in both x-axis and y-axis directions are collected, the intersection of the sets X and Y are the pairs of rectangles that overlapped on the plane.

A self-balancing data structure Red-Black Tree will be used to store the x-coordinates a and b, and y-coordinates c and d separately. Then, we can search the Red-Black Tree to check whether two rectangles are overlapped for x-axis direction and y-axis direction separately. The rectangle pairs who are overlapped can be stored in hash tables. Finally, we can insert the element from x-axis direction table to y-axis direction table to check the intersect set.

Algorithm:

```
pre-processing:

  sort x-coordinates (a and b) of the rectangles in increasing order
  sort y-coordinates (d and c) of the rectangles in decreasing order

insert the x-coordinates (a_i, b_i) of the rectangles into the Red-Black Tree X
insert the y-coordinates (c_i, d_i) of the rectangles into the Red-Black Tree Y
```

```
find the conflict intervals in x-axis direction by searching the Red-Black Tree X
and record the conflict intervals pair x(i, j) in hash table A
find the conflict intervals in y-axis direction by searching the Red-Black Tree Y
and record the conflict intervals pair y(i, j) in hash table B

find the intersection of x(i, j) and y(i, j) by inserting elements in hash table A
to hash table B
```

Algorithm's Correctness:

Lemma 1. *Two rectangles A and B are overlap if both the edges parallel to x-axis and the edges parallel to y-axis of them are conflict correspondingly.*

Proof. There must be at least one point lies in both rectangle A and rectangle B if they have common points in both x-axis direction and y-axis direction. □

Running Time Analysis:

The pre-processing which is sorting $2n$ nodes for both x-coordinates and y-coordinates of the rectangles costs $O(n \log n)$. The insertion to the self-balancing tree also costs $O(n \log n)$. Additionally, when searching for conflict intervals, $O(n \log n)$ is needed to search on the tree. Finally, the insertion for the hash table costs $O(n)$. Therefore, the total time complexity for the algorithm is $O(n \log n)$.