```
/**
 * Definition for an interval.
 * public class Interval {
 *     int start;
 *     int end;
 *     Interval() { start = 0; end = 0; }
 *     Interval(int s, int e) { start = s; end = e; }
 * }
 */
class Solution {
    public boolean canAttendMeetings(Interval[] intervals) {
        /*
        Arrays.sort(intervals, (a, b) -> a.start - b.start);

        for (int i = 0; i < intervals.length; i++) {
            if (i < intervals.length-1 && intervals[i].end > intervals[i+1].start) {
                return false;
            }
        }
        return true;
        */

        /*
        Queue<Interval> q = new PriorityQueue<>(new Comparator<Interval>(){
            @Override
            public int compare(Interval m1, Interval m2) {
                return m1.start == m2.start ? m1.end - m2.end : m1.start - m2.start;
            }
        });

        for (Interval i : intervals) {
            q.offer(i);
        }

        Interval i = null;
        while (!q.isEmpty()) {
            if (i != null) {
                if (i.end > q.peek().start)
                    return false;
            }
            i = q.poll();
        }
        return true;
        */
        int len = intervals.length;
        int[] first = new int[len];
        int[] second = new int[len];
        for (int i = 0; i < len; i++){
            first[i] = intervals[i].start;
            second[i] = intervals[i].end;
        }

        Arrays.sort(first);
        Arrays.sort(second);
        for (int i = 0; i < len-1; i++){
            if (second[i] > first[i+1]){
                return false;
            }
        }
        return true;
    }
}
```