# 435. Non-overlapping Intervals

Given a collection of intervals, find the minimum number of intervals you need to remove to make the rest of the intervals non-overlapping.

**Example 1:**

```
Input: [[1,2],[2,3],[3,4],[1,3]]
Output: 1
Explanation: [1,3] can be removed and the rest of intervals are non-overlapping.
```

**Example 2:**

```
Input: [[1,2],[1,2],[1,2]]
Output: 2
Explanation: You need to remove two [1,2] to make the rest of intervals non-overlapping
```

**Example 3:**

```
Input: [[1,2],[2,3]]
Output: 0
Explanation: You don't need to remove any of the intervals since they're already non-ov
```

**Note:**

1. You may assume the interval's end point is always bigger than its start point.
2. Intervals like [1,2] and [2,3] have borders "touching" but they don't overlap each other.

# 移除重复区间的最少个数

## Greedy, Array, Interval

### acvtivity selection problem to compare the end and start

1. sort the 2D array by creating a comparator class that compares the second value of two arrays
2. select the first interval and initilize the end, therfore, iterate [1,size) to compare the start and end
3. if end is greater, count++, else update the end to current interval
4. return count;

```
class arrayComparator implements Comparator<int[]> {
     public int compare(int[] arr1, int[] arr2) {
         return arr1[1] - arr2[1];
     }
 }
```