最少数量线段覆盖

知识点排序 贪心

时间限制：1s 空间限制：256MB 限定语言：不限

题目描述：

给定坐标轴上的一组线段，线段的起点和终点均为整数并且长度不小于1，请你从中找到最少数量的线段，这些线段可以覆盖住所有线段。

输入描述：

第一行输入为所有线段的数量，不超过10000，后面每行表示一条线段，格式为"x,y"，x和y分别表示起点和终点，取值范围是[-105,105]。

输出描述：

最少线段数量，为正整数

示例1

输入：

3
1,4
2,5
3,6

输出：

2

说明：

选取2条线段[1,4]和[3,6]即可，这两条线段可以覆盖[2,5]

解题思路：

如例一：
1、先找出所有线段的最左端=1，和最右端=6
2、对所有线段进行排序。x小的排在前面，x相等的，y小的排在前面
【1，4】，【2，5】【3，6】
3、对所有最左侧线段进行遍历（x=1的线段即为最左侧线段）【1,4】

```
public class Main{

    public static List<Line> lineList = new ArrayList<>();    //线段集合
    public static int minLeft = Integer.MAX_VALUE; //左边界
    public static int maxRight = 0;          //右边界
    public static int res = Integer.MAX_VALUE;

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        sc.nextLine();

        for(int i=0; i<n; i++){
            String[] strings = sc.nextLine().split(",");
            int l = Integer.valueOf(strings[0]);
            minLeft = Math.min( minLeft, l);       //左边界为左坐标最小值
            int r = Integer.valueOf(strings[1]);
            maxRight = Math.max( maxRight, r);    //右边界为有坐标最大值
            Line line = new Line( l, r);
            lineList.add(line);
        }

        Collections.sort(lineList);
        for(int i=0; i<lineList.size(); i++){
            Line line = lineList.get(i);
            if(line.left == minLeft){     //第一根线段（左坐标等于左边界）
                handle( i+1, line.left, line.right, 1);
            }
        }

        System.out.println(res);
    }

    /**
     *
     * @param index        紧接着的一个线段的索引（因为排过序，前面的线段无需考虑）
     * @param left         前一线段的左坐标
     * @param right        前一线段的右坐标
     * @param count        线段的个数
     */
    public static void handle(int index, int left, int right, int count){
```

```
            if(right == maxRight){    //到了右边界，说明覆盖了所有线段
                res = Math.min( res, count);
            }else {
                for(int i=index; i<lineList.size(); i++){
                    Line line = lineList.get(i);
                    if(line.left > left && line.left <= right && line.right > right){
                        handle( i+1, line.left, line.right, count+1);
                    }
                }
            }

        }
    }

    class Line implements Comparable<Line>{

        int left;
        int right;

        public Line(int left, int right) {
            this.left = left;
            this.right = right;
        }

        @Override
        public int compareTo(Line o) {
            if(o.left == this.left){
                return this.right - o.right;
            }
            return this.left - o.left;
        }
    }
```