1: 区间问题:

例题:

给定 N 个闭区间 $[a_i,b_i]$,请你在数轴上选择尽量少的点,使得每个区间内至少包含一个选出的点。 输出选择的点的最小数量。

位于区间端点上的点也算作区间内。

输入格式

第一行包含整数 N,表示区间数。

接下来 N 行,每行包含两个整数 a_i, b_i ,表示一个区间的两个端点。

输出格式

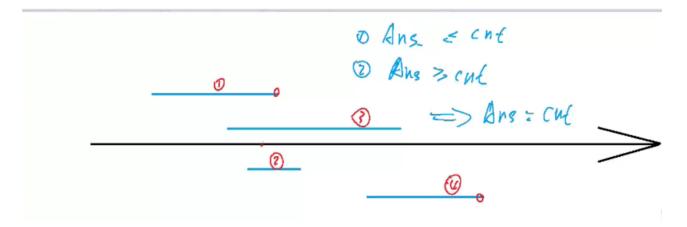
输出一个整数,表示所需的点的最小数量。

输入样例:

```
3
-1 1
2 4
3 5
```

输出样例:

2



- 1 思路:
- 2 1: 将每个区间按右端点从小到大排序
- 3 2: 从前往后依次枚举每个区间
- 4 如果当前区间中已经包含该点,则直接pass,否则,选择当前区间的右端点

```
import java.util.*;
public class Main{
public static int N = 100010;
public static int n;
```

```
public static int INF = 0x3f3f3f3f;
       public static Range[] range = new Range[N];
6
       public static void main(String[] args){
           Scanner sc = new Scanner(System.in);
8
           n = sc.nextInt();
9
           for(int i = 0; i < n; i++){</pre>
10
                int 1 = sc.nextInt();
11
                int r = sc.nextInt();
12
                range[i] = new Range(1, r);
13
           }
14
           Arrays.sort(range, 0, n);
15
           int res = 0;
16
           int ed = -INF;
17
           for(int i = 0; i < n; i++){
18
                if(range[i].l > ed){
19
20
                    res ++;
                    ed = range[i].r;
                }
22
           }
23
24
           System.out.println(res);
       }
25
26
   class Range implements Comparable<Range>{
       int 1, r;
28
       public Range(int 1, int r){
29
           this.l = 1;
           this.r = r;
31
32
       public int compareTo(Range o){
           return Integer.compare(r, o.r);
34
35
36
```

例题:

```
    1 思路:
    2 1:将所有区间按左端点从小到大排序
    3 2: 从前往后处理每个区间:
```

- 4 判断能否将其放到某个现有的组中 1[i] > Max_r
- 5 **3**: 如果存在这样的组,将其放进去,并更新当前组的Max_r,如果不存在这样的组,则开新组,然后再将其放进去

```
import java.util.*;
   public class Main{
       public static int N = 100010;
       public static int n;
4
       public static Range[] range = new Range[N];
5
       public static void main(String[] args){
6
           Scanner sc = new Scanner(System.in);
           n = sc.nextInt();
8
           for(int i = 0; i < n; i++){
                int 1 = sc.nextInt();
                int r = sc.nextInt();
11
                range[i] = new Range(1, r);
12
13
           Arrays.sort(range, 0, n);
14
           Queue<Integer> minheap = new PriorityQueue<>();
15
           for(int i = 0; i < n; i++){</pre>
16
                if(minheap.isEmpty() || minheap.peek() >= range[i].1){
                    minheap.add(range[i].r);
18
                }else{
19
                    minheap.poll();
20
                    minheap.add(range[i].r);
21
                }
22
23
           System.out.println(minheap.size());
24
       }
26
   class Range implements Comparable<Range>{
       int 1, r;
28
       public Range(int 1, int r){
29
           this.l = 1;
30
           this.r = r;
32
       }
       public int compareTo(Range o){
33
           return Integer.compare(1, o.1);
34
```

2: Huffman 树:

例题:

```
import java.util.*;
   public class Main{
       public static void main(String[] args){
           Scanner sc = new Scanner(System.in);
4
           int N = 100010;
           int n = sc.nextInt();
6
           Queue<Integer> minheap = new PriorityQueue<>();
           for(int i = 0; i < n; i++){
               int x = sc.nextInt();
               minheap.add(x);
10
11
           int res = 0;
12
           for(int i = 0; i < n; i++){
               if(minheap.size() > 1){
14
                   int a = minheap.poll();
                   int b = minheap.poll();
16
                   res += a + b;
17
                   minheap.add(a + b);
18
19
           }
20
           System.out.println(res);
       }
22
23 }
```

3: 排序不等式:

例题:

```
import java.util.*;
public class Main{
   public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
}
```

```
int N = 100010;
           int[] w = new int[N];
6
           int n = sc.nextInt();
           for(int i = 0; i < n; i++){
               w[i] = sc.nextInt();
10
           Arrays.sort(w, 0, n);
11
           long res = 0;
12
           for(int i = 0; i < n; i++){
13
               res += w[i] * (n - 1 - i);
14
15
           System.out.println(res);
16
       }
17
18 }
```

4: 绝对值不等式:

$$|x - a| + |x - b| > = |a - b|$$

例题:

```
import java.util.*;
public class Main{
       public static void main(String[] args){
           Scanner sc = new Scanner(System.in);
4
           int N = 100010;
           int[] a = new int[N];
           int n = sc.nextInt();
7
           for(int i = 0; i < n; i++){</pre>
               a[i] = sc.nextInt();
9
10
           Arrays.sort(a, 0, n);
11
           int sum = 0;
12
           for(int i = 0; i < n; i++){
13
               sum += Math.abs(a[i] - a[n / 2]);
14
           }
15
           System.out.println(sum);
       }
17
18 }
```