**最大重叠区间个数**

1. 问题描述

有一个聚会，许多人来参加。一个记录仪记录下了每个人到达聚会的时间和离开聚会的时间。也就是说每个人在聚会的时间为[]，求问这个聚会至少能同时接纳多少人？（即这个聚会中最多能同时接纳多少人？）

1. 问题分析

这是最大重叠区间的一个具体样例，这种题的做法就是将所有的开始时间和结束时间（无论哪个事件的或人的达到结束时间）进行排序，如果是开始时间则计数加1，如果是结束时间则计数减1。在此过程中的计数的最大值便是最大重叠区间数目。

PS：在排序的时候，会有开始时间和结束时间重叠的情况，总共分为三种情况：

1. 两个开始时间重叠（即相等），这种情况不影响结果，因为无论哪个在前计数都要加1
2. 两个结束时间重叠（即相等），这种情况不影响结果，因为无论哪个在前计数都要减1
3. 开始时间和结束时间重叠（即相等），这种情况出现时，应该让结束时间排靠在前，开始时间排靠在后，否则若将开始时间排靠在前的话，会将两个区间连接但不重叠的情况考虑成两个区间连接但重叠
4. 代码实现
5. #include <vector>
6. #include <iostream>
7. #include <algorithm>
9. **using** **namespace** std;
11. **struct** node
12. {
13. **int** time;   //time为到达时间或者结束时间
14. **int** flag;  // flag为0则为到达时间，flag为1则为结束时间
15. node(**int** t, **int** f) : time(t), flag(f) {}
16. };
17. **int** main()
18. {
19. **int** n;   // 此聚会来过的总人数
20. cin >> n;
21. vector<node> v;
22. **while** (n--) {
23. **int** s, e;   // 输入每个人的到达时间s和结束时间e
24. cin >> s >> e;
25. v.push\_back(node(s, 0));
26. v.push\_back(node(e, 1));
27. }
29. //按时间排序（所有事件的开始时间和结束时间）
30. //注意排序时的特殊情况：当时间相等时，让结束时间在前
31. sort(v.begin(), v.end(), [](**const** node& a, **const** node& b) {
32. **if** (a.time == b.time) **return** a.flag > b.flag;
33. **return** a.time < b.time;
34. });
36. //遇到开始时间加1，遇到结束时间减1，找个过程中的最大值
37. **int** ans = 0, res = -1;
39. **for** (**int** i = 0; i < v.size(); i++) {
40. **if** (v[i].flag == 0)
41. ans++;
42. **else**
43. ans--;
44. res = max(res, ans);
45. }
47. cout << res << endl;
48. **return** 0;
49. }