line

算法 1

暴力枚举把人分成哪些段，然后计算答案即可。

时间复杂度$O(2^n * poly(n))$，期望得分9分。

算法 2

我们换一种计算答案的方式。设每一段人编号最大的那个是r调程序的时间为t，那么它对答案的贡献是$\sum_{i=r+1}^{n} w_i t$，即不耐烦指数的后缀和。

那么我们可以考虑dp，设f_i表示考虑到第i个人，第i个人在末尾，贡献的最小值。容易写出转移方程。

复杂度$O(n^2)$。

算法 3

一种处理转移方程式中后缀max一项的常见技巧是运用单调栈。它本质上是动态维护了以i结尾的后缀max有哪些段。然后相同的段取f数组维护最小的那个值即可。

而子任务3，4的特征是单调栈中元素个数是期望或严格$O(\log n)/O(1)$的。所以我们可以用RMQ（ST表，线段树等）维护f数组最小的值，并且暴力枚举每个单调栈上的元素，更新答案。

注意到有l_i的限制，所以我们还剩一段不能用单调栈上的元素表示。二分找到那一段，然后特殊处理即可。

当然，对于子任务四，你可以直接使用线段树等数据结构优化dp。

算法 4

注意到我们维护单调栈的入栈，出栈序列可以建成一棵树（类比dfs的过程）。

那么我们每次询问相当于询问树上的一条链组成的所有一次函数在某一点的最小值。

（Q：什么，你说动态凸壳？ A：你来写啊！）

这是一个典型的斜率优化dp的问题。

我们可以先把树建出来，然后做树链剖分。在每条链上维护线段树，线段树上每个节点存储这一段的一次函数组成的凸壳。

注意到斜率的单调性，所以每个线段树上的凸壳可以用单调队列维护。每次对于链上的每个询问，分成的线段树的每个节点得到的值取min即可，这样子避免了凸包合并，动态凸壳等很难实现的操作）。

时间复杂度$O(n \log^2 n)$，期望得分100分。

算法 5

其实我们可以做的更好。我们所需要维护的东西是单调栈的尾端删除，加入，询问一段凸壳上某一点的值。使用二进制分组 + 延迟重构的思想，可以做到$O(n \log n)$

总结

本题作为一道送分题，思维难度不高，考查点是选手熟知的数据结构，动态规划的常见技巧。并且为了增加选手信心，本题也没有刻意地去区分$O(n \log n)$,$O(n \log^2 n)$的算法。这道题目能够给选手更多的时间去攻克后面的难题，给他们一个有力的援助！