one

60分做法：三重循环，$O(n^3)$。然而我自己测暴力的时候没开O2，忽略了OJ上可以开O2这件事，所以不少同学这样直接过了，很抱歉。

正解考虑上课讲的思路，先枚举 j，那么 i, k 独立了，可以把不同 i 对应的 $a_i + j - i$ 排序，把 k 对应的那个值也排序。这样相当于有两个有序序列（分别对应 i, k 的取值），要统计从两个序列各取一个数的 $\max(x, y, C)$ 之和。枚举 x, y 中的最大值是哪个，用双指针求出有多少 (x, y) 的最大值是枚举的那个就好。

复杂度 $O(n^2 \log n)$，当然这题值域只有 n，所以可以用桶排序做到 $O(n^2)$。

two

30分做法：根据题意三重循环。

正解考虑枚举这个 a，那么可以算出有多少 (b, c) 满足 $a > b, a > c$。要求的答案其实就是这个减去 $a > b > c$ 的三元组个数，因为固定一个 a 之后，b, c 无关系相当于所有情况剪掉 b, c 有大于关系的。

这样就可以分成两部分计算。我们发现只要求出 f_i 表示有多少其它点比 i 小，g_i 表示有多少其它点比 i 大，则第一部分和第二部分都直接就求出来了。（第二部分枚举 b 即可计算。）这个 f, g 用树状数组即可求出。

复杂度 $O(n \log n)$。

50分做法：假设你不会树状数组，暴力求 f, g 就可以。

three

30分做法：搜索所有可能的删数顺序，根据期望的定义算答案就好（概率和对应的值都在DFS的过程中可以直接算）。

60分做法：用状压DP优化上述搜索过程。

这个题正解其实做法很简单的，但可能不太容易想到。我准备题目的时候可能有点低估它的难度了（？）

根据期望的线性性，i 的期望答案就是 $\sum_{j \neq i} P(j\text{ 删掉的时刻先于 }i) + 1$，其中 $+1$ 是因为它自己被删对应一次操作。

注意到 $P(j\text{ 删掉的时刻先于 }i) = a_j/(a_i + a_j)$，因为考虑模拟这个过程，则 i, j 之外的点都可以忽略，所以算这个就相当于 $n = 2$ 的情况下只考虑 i, j。

所以直接根据上述公式计算即可。

复杂度 $O(n^2)$。

four

这个题得先考虑怎么搞一个能求出正确答案的算法是吧。

可以枚举起点和终点 $s,t\ (s\le t)$，发现此时最多走的边数是好算的：首先往左可以一直走 $a_i\ge 2$ 的边（不然走过去就回不来了），走的次数是不超过它的最大偶数。然后从 s 走到 t，每条边走的次数为不超过这条边的最大奇数，之后再从 t 出发往右再续续命，和 s 往左的情况一模一样。这样就有一个 $O(n^3)$ 的暴力算法了。

怎么优化呢？首先可以预处理 $L[i],R[i]$ 表示 i 出发向左/右走最终回到自己，最多能走几条边，这个扫一遍就可以线性求出来。

然后答案其实就是 $\max_{s\le t} L[s]+R[t]+Val(s,t)$，这个 $Val(s,t)$ 就是从 s 到 t 的路径所有边不超过这条边的最大奇数之和。可以前缀和优化预处理。这样就 $O(n^2)$ 了。

其实可以把 Val 拆贡献到 s,t 上，这样枚举 t 之后对于 s 就是个前缀最大值，就 $O(n)$ 了。