Tree

假设有 k 个叶子，考虑先把叶子中的 1 都加入，然后删除再把所有 0 加入的过程。值域一定大于等于 k，所以答案至少为 $\lceil k/2 \rceil$。

接着考虑如何构造，首先可以整棵树可以由 $\lceil k/2 \rceil$ 条链覆盖。然后可以整棵树划分成 $\lceil k/2 \rceil$ 个集合，如果一个点属于多条链，把它加入其中任意某一个集合即可。每个集合都是链的子集。把每个集合按照链的顺序黑白染色即可。

注意到一个连通块和集合的交一定是连续一段，也就是差值不会超过 1，所以整体不会超过 $\lceil k/2 \rceil$。可以根据当前黑白点的个数决定下一条链的染色方案，保证整体黑白个数差不超过 1。

关于链划分，每次选择两个叶子，选取路径上未被选去的点，可以使用并查集。时间复杂度 $O(n \log n)$。或者可以使用选取叶子的重心，然后 DFS 的方法做到 $O(n)$。

#Number

首先可以把 A 中偶数贪心匹配。对于奇数 $2k+1$，可以把它当成二元组 $(k, k+1)$，表示可以变成这两个数中的一个，记这样的二元组为 X_k。

对于这些二元组，形成了一棵满二叉树。考虑最大流，二元组 $(k-1, k)$ 和 $(k, k+1)$ 向 Y_k 连边，Y_k 向汇点 T 连 b_k 的边。这些二元组有环状的结构，所以并不好贪心。

考虑最小割，如果有个 X_k 没有被割掉（在 S 集），那么它到根的路径上的所有二元组 X 都不会被割，而对应的 Y 都要被割掉。考虑树形 dp，每个子树对外部的影响之和它的左右链中最深的在 S 集中的点有关，即考虑 $dp_{u,l,r}$ 表示 u 这个子树，左右链在 S 中的最深点在什么地方，内部的 X 和 Y 对应的最小割。

子树合并的时候枚举左子树右子树左右链的深度，记录 Y 对应的贡献，合并即可。最后考虑 1 这个子树左右链的贡献，记入答案。

这个做法的复杂度为 $O(\sum_{i=1}^{m} 2^{m-i} i^4)$，~~注意到~~ $\sum_{i \ge 1} i^4 2^{-i} = 150$ ~~收敛，所以时间复杂度为线性。~~实际上上述转移可以通过枚举合并部分的较深深度，类似前缀最大值，优化到三方，更加合理的常数估计为 $\sum_{i \ge 2} (i-1)^3 2^{-i} = 13$。

Sequence

将 a 从大到小排序，令 $c_i = a_i - a_{i+1}$。

相当于如下问题：一张简单二分图，左边有 c_i 个度数为 i 的点，右边有 n 个点，求右边点度数序列方案数。

再次转化等价于如下问题：一个序列 d，初始 d 中有 c_i 个 i，每次操作选择一个 k，将 d 中前 k 大同时 -1，求 n 次操作将 d 变为全 0 的方案数。

考察每次操作对 a 的影响，相当于以下两种操作：

- 选择 i，将 a_i, a_{i+1} 合并为 x，其中 $a_{i+1} \le x < a_i$。
- 将 a_n 变为 x，其中 $0 \le x < a_n$。

求 n 次操作后 a 中仅剩一个 0 的方案数。

方便起见，在 a 中补一个 0，则只需要考虑第一种操作。

令 $f_{l,r,x}$ 表示用 $r-l-1$ 次操作将 $a_l \dots a_r$ 合并为 x 的方案数。

$f_{l,r,*}$ 可以表示成 $O(n)$ 段生成函数，每段生成函数形如 $\sum\limits_{i=0}^{n} \dfrac{w_i}{(1-x)^i}$。

转移时枚举最后一步合并之前的分段点即可。时间复杂度 $O(n^5)$。