FJCPC2025 题解

寒鸽儿

拼好题出题组

June 23, 2025

一些小彩蛋

- A 题是出题组在xhs上刷到的一个作弊哥，然后主播去钓了一手鱼
- B $\sim$ L 副标题名称致敬了 FJCPC2024 校排前 11 的队伍
- M 题致谢所有参赛选手，以及历届z赛事staff

难度排序

eazy：M 致谢
easy-medium：G 炒股高手、J 构造大师CFJ、K VERTeX
medium：C 中位数、E 卡牌游戏、H 难以控制的火箭滑板、I 割点
medium-hard：D 二叉树、L 众数、B XCPC、F 帕累托前沿
hard：A we are watching you!

M. 致谢

签到题，输入数字n，输出第n届FJCPC承办高校的英文缩写。
希望明年能有福建省内院校承办FJCPC2026比赛。

G. 炒股高手 I

基于贪心的思想，我们始终考虑低买高卖。对于任一一份鸡债对应的交易日区间，我们可以将其看作若干段连续上升序列和连续下降序列交替拼接而成，因此我们只会在上升段的开头买入股票，末尾卖入股票。

由于可以随意进行买卖操作，我们可以考虑一种等价的策略：对于第 $i(i>1)$ 天交易日，如果第 i 天的股票大于第 $i-1$ 天的股票，那么我们在第 $i-1$ 天买入股票，第 i 天卖出，这样我们就可以在第 i天使资产增长为原来的$e^{a_i-a_{i-1}}$倍。否则如果第 i 天的股票小于等于第 $i-1$ 天的股票，则不执行操作。

我们记 c_i表示第 i 天如果执行该策略可使得资产增长为原来的多少倍，那么对于任一鸡债，我们只需将询问区间内的所有c_i乘以初始鸡债数额即可得到最终资产。

G. 炒股高手 II

由于最终答案对资产取对数，此时乘以c_i对答案的影响变为加上其以 e 为底的指数。因此我们只需记录c_i的指数，且对于每个询问将区间内的指数相加即可得到答案。

J. 构造大师CFJ

一种比较简单的做法是考虑二进制构造。
不难发现 2^{2i} 一定是完全平方数。
我们回顾一下树状数组中的一个概念， $\text{lowbit}(n)$ 为可以整除 n 的最大的二的幂次方 2^i。
那么我们每次加 $x = \text{lowbit}(n)$ 即可，显然 $\text{lowbit}(n)$ 必然是递增的，当前的 x 必然和历史 x 不同。

题解

不妨设结点 1 的权值为 $b_1 = x$。
由于树上每个结点都与结点 1 连通，因此每个结点的权值可以表为 $w - x$ 或 $w + x$ 的形式，其中 w 可由边权计算得到。
对于每个结点有约束 $b_i > 0$，带入可得 $w - x > 0 \implies x < w$ 或 $w + x > 0 \implies x > -w$。
若有解则合法解构成一段区间，维护解区间端点即可。
时间复杂度 $O(n)$。

C. 中位数 I

给定n个数字，n为奇数，每次合并连续三个数变成他们的中位数，使得最后剩下的一个数尽可能大。
考虑二分答案mid，把 $a[i] \geq mid$ 标记成 1 ,把 $a[i] < mid$ 标记成 0 。
显然贪心地考虑要把 0，尽可能地消掉。
对于连续存在的超过2个的0，可以任选其中三个0变成一个0。这样最后剩下的一定是连续的2个0或者1个0。
注意由于 001 可以变成 0，因此如果对于00100这样的情况，可以先选择前三个数变成000，再变成1个0。因此只要能向后合并，尽可能向后合并连续存在的两个0，并合并掉三个0的情况。
以上过程可以使用一个栈来简单地辅助编码。最后检查栈内剩余1的个数是不是超过0的个数。
check为真说明此时的mid符合二分答案；否则不符合。

E. 卡牌游戏 I

最大化最坏情况下所得到的卡牌数字之和，等价于最小化最坏情况下双方卡牌数字之和的差的绝对值。

给定操作等价于选择一段长度为偶数的区间（可以为空），并交换其中元素编号的奇偶性。不妨设选择的区间为 $[l, r]$，显然 $l-1, r$ 同奇偶。不失一般性地，设 $l-1, r$ 均为奇数，设前缀和

$$O_k = \sum_{1 \le 2i+1 \le k} a_{2i+1}, \quad E_k = \sum_{1 \le 2i \le k} a_{2i}$$

则其中一位玩家得到的卡牌数字之和为 $O_n - (O_r - O_{l-1}) + (E_r - E_{l-1})$，另一位玩家得到的卡牌数字之和为 $E_n - (E_r - E_{l-1}) + (O_r - O_{l-1})$。

两玩家得到卡牌的差值为：

E. 卡牌游戏 II

$$\begin{aligned}\Delta &= (O_n - (O_r - O_{l-1}) + (E_r - E_{l-1})) \\ &\quad - (E_n - (E_r - E_{l-1}) + (O_r - O_{l-1})) \\ &= O_n - E_n + 2(E_r - O_r) - 2(E_{l-1} - O_{l-1})\end{aligned}$$

考虑枚举 r，对于每个 r 需要选择合适的 l 使得 Δ 最接近 0，也即使得 $2(E_{l-1} - O_{l-1})$ 最接近 $O_n - E_n + 2(E_r - O_r)$，这可以使用 set 维护 $l < r$ 的 $2(E_{l-1} - O_{l-1})$ 完成。对于 $l-1, r$ 为偶数的情形类似地使用 set 维护即可。

时间复杂度 $O(n \log n)$。

H. 难以控制的滑板火箭

当 $l \neq r$ 的时候在开始的时候全部用 r 步跑满，最后用 r 或者 $r-1$ 一定可以组成出与最短路奇偶性相同的（可以在最后两步反复横跳来浪费步数），所以答案就是最短路长度除以 r 向上取整。

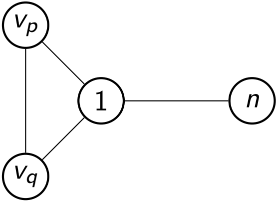
当 $l=r$ 且 l 为偶数的时候，只有步数为偶数的路径才可能是答案，在 BFS 的时候加上距离的奇偶性的维度，得到长度为偶数的最短路，答案就是这个最短路除以 r 向上取整的结果。

当 $l=r$ 且 l 为奇数的时候，类似上面的，先计算出步数为奇数的最短路和偶数的最短路，通过向上取整可以计算出最少需要的时间，如果需要的时间的奇偶性与最短路的奇偶性不同，那么需要多一秒钟来浪费。

I. 割点 I

假设所有节点为 $v_1 \sim v_n$。分三种情况讨论。

- 若 $v_2 \sim v_{n-1}$ 中有至少两个点 v_p, v_q 不是割点，可以先构造出以下基本图形：



然后对于所有其它节点，如果它为割点，则把它加到右边的链中间。否则加到左边的环中间。这样，所有环上的点（除 v_1）以及 v_n 都不是割点，其余点为割点。且 $deg_1 = 3, deg_n = 1$，其余点度数都为 2。

I. 割点 II

- 若 $v_2 \sim v_{n-1}$ 中仅有一个点不是割点，则该点若是 v_{n-1} 则有解，否则无解。
  证明可以考虑图的圆方树。由圆方树的定义可知，所有叶子节点均不是割点。而当 $n \geq 4$ 时，树至少有两个叶子节点（因为节点度数和为 $2n-2$，由鸽巢原理可得至少两个点度数为 1）。因此图至少有两个非割点，且只在一条链的情况取得。
- 若 $v_2 \sim v_{n-1}$ 中没有割点，由上面的分析可知无解。

D. 二叉树 I

本题有许多做法，包括树哈希、以及一些剪枝乱搞。这里给出一个思路比较流畅的做法：
根据所给树的节点数，可以先得到分割后两棵树的大小 n_1, n_2 ，不妨设 $n_1 < n_2$ 。先特判掉 $n_1 = 1$ 的情况，那么这条额外的边有以下几种情况：

- 根连根。此时，原树中没有度数为 2 的节点，没有度数为 4 的节点。额外边是 $3-3$ （度数）。
- 根连叶子。此时原树中两个度数为 2 的节点，没有度数为 4 的节点。额外边是 $3-2$ 。
- 根连其他。此时原树中一个度数为 2 的节点，一个度数为 4 的节点。额外边是 $3-4$ 。

D. 二叉树 II

- 叶子连其他。此时原树中三个度数为 2 的节点，一个度数为 4 的节点。额外边是 $2-4$ 。
- 其他连其他。此时原树中两个度数为 2 的节点，两个度数为 4 的节点。额外边是 $4-4$ 。
- 叶子连叶子。此时原树中四个度数为 2 的节点，没有度数为 4 的节点。额外边是 $2-2$ 。

于是，根据树节点的度数分布，可以唯一的判断出这条额外边属于哪种情况。
注意到后五种情况中，满足度数情况的边都至多只有 $O(1)$ 条，直接枚举判断即可。再考虑第一种情况怎么做，发现只要跑一个类似换根dp的东西就可以了。

题解 I

最大值加次小值（次大值加最小值），可以构造出类似 $1\ 1\ 1\ \ldots\ 1\ 2$，那么能出现 $2^{n-1}-1$ 次，或者 $1\ 2\ 2\ 2\ \ldots\ 2\ 3$，那么有 $2^{n-1}-2$ 次。也可能成为众数。

对 a_i 排序后，假设众数为 k，那么一定存在一个最小的 l，此时存在最大的 r，$a_l+a_r=k$，如果 $r-l+1<n-1$，那么至多只有 $2^{r-l+1}-1$ 种情况为 k，也就是小于 2^{n-2} 种，所以只有可能是上述三种情况。

对于后面这两种情况，如果说最小值和最大值都只出现恰好一次，那么方案数最多也就是 $2^{n-2}-1$ 没有达到阈值，所以最大值或者最小值至少出现多次。下面考虑最大值出现多次的情况。此时最小值必然只出现一次，否则根据上面的结论一定是最大值加最小值是答案。假设最大值出现了 p 次，那么最大值加最小值的方案数为 $(2^p-1)2^{n-p-1}$ 种，如果除了存在第三种数，那

题解 II

么最大值加次小值出现的次数的上限为 $(2^p - 1)(2^{n-p-1} - 1)$ 种，小于最大值加最小值的方案数。如果只有两种数，那么方案数均为 $2^{n-1} - 1$ 种。也就是说只有类似 1 2 2 ... 2 2 的情况才会出现次小值加最大值是众数。所以只有在最小值出现一次，且只有两种数出现时答案为最大值乘 2，否则就是最小值加最大值。

观察 I

考虑合成一个奖牌对总价值的影响：

- 合成一个金牌需要耗费 8 个铁牌，价值从 8 变为 4，总价值 -4。
- 合成一个银牌需要耗费 4 个铁牌，价值从 4 变为 3，总价值 -1。
- 而合成一个铜牌则不会影响总价值。

朴素的枚举做法 I

考虑一个 n^2 的做法，枚举最终四元组中的金牌和银牌数量，分别称为 g 和 s，那么此时总价值为 $n-4g-s$。
若此时总价值小于 p，则该方案不合法。否则，剩下的铁牌可以任意转换为铜牌，计算得到剩余 $n-8g-4s$ 个铁牌。奖牌最多的情况是不合成任意一个铜牌，最终会有 $g+s+(n-8g-4s)$ 个奖牌；奖牌最少的情况是尽可能把铁牌合成铜牌，最终会有 $g+s+(n-8g-4s+1)/2$ 个奖牌。因此，对于 $i \in [g+s+(n-8g-4s+1)/2, g+s+(n-8g-4s)]$ 的问题，答案加一。
此时，做法变为：

1. 枚举 g, s；

朴素的枚举做法 II

2. 计算出相应区间，并在答案数组上做区间加操作，这个操作可以转换为答案数组的差分数组上的单点修改操作。从而得到了 $O(n^2)$ 时间复杂度的做法。

线性的枚举做法 I

以下简称"答案数组的差分数组"为差分数组。
在朴素的枚举做法的基础上，只枚举金牌的数量 g，此时银牌数量 s 的取值范围受到两个约束：

1. 初始时仅有 n 个铁牌，即 $8g+4s \leq n$；
2. 总价值需要大于等于 p，即 $4g+3s+(n-8g-4s) \geq p$。

每一个 s 的取值都对应着对一个答案数组上的区间加一操作，展示其中的规律如下：

- $[4, 4]$
- $[5, 7]$
- $[6, 10]$
- $[7, 13]$

线性的枚举做法 II

会发现这些区间的左端点是公差为 1 的等差数列，右端点是公差为 3 的等差数列。因此这相当于：在差分数组上做一个区间加一操作；和一个区间每隔2减一操作。二者都相当于在差分数组的差分数组（即，二阶差分数组）上做单点修改操作。
在代码实现上，参考程序中分别使用了两个二阶差分数组来维护不同公差的区间修改操作。

1. 彩蛋
2. 题目难度定位
3. M. 致谢
4. G. 炒股高手
5. J. 构造大师CFJ
6. K. VERTeX
7. C. 中位数
8. E. 卡牌游戏
9. H. 难以控制的滑板火箭
10. I. 割点
11. D. 二叉树
12. L. 众数
13. B. XCPC
14. F. 帕累托前沿
15. A. We are watching you!

F. 帕累托前沿 I

先对每个位置 i 求出 L_i, R_i。其中 L_i 的值为最大的 $j < i$，使得 $x_j \geq x_i$ 且 $y_j \geq y_i$，类似地，R_i 的值为最小的 $j > i$，使得 $x_j \geq x_i$ 且 $y_j \geq y_i$。为了方便，令 $x_0 = y_0 = x_{n+1} = y_{n+1} = +\infty$。

求解方法为：以三元组 $(x_i, y_i, -i)$ 为关键字，对所有下标从大到小排序，依次求解每个下标的答案。由于对称性，这里仅说明求解 L_i 的方法。

我们维护一棵线段树，线段树下标为原序列下标，值为区间已插入下标 y 的最大值。对于每个下标，由于前面已插入线段树的下标的 x 都大于等于自己的 x，所以仅需要找到左边最大的下标，使得其 y 大于等于自己的 y。通过维护区间 y 最大值，这一步可以通过线段树上二分在 $\mathcal{O}(n \log n)$ 复杂度内完成。询问完成后，再将当前下标插入线段树中。

F. 帕累托前沿 II

接着考虑每个下标对询问的贡献。对于节点 i，其显然对 $l < L_i \leq i$ 且 $i \leq R_i < r$ 的询问区间 $[l, r]$ 有 1 的共享。将询问看作是一个二维平面上的点，那么就是一个矩形加，单点查询的问题。可以通过扫描线 + 树状数组在 $\mathcal{O}(n \log n)$ 的复杂度内求解。

两个部分时间复杂度都是 $\mathcal{O}(n \log n)$，但也可以用 cdq 分治类似的方法求解，实现常数不太大的 $\mathcal{O}(n \log^2 n)$ 也可以通过。

A. We are watching you!

给一个字符串 S，对于 S 的后缀自动机的每个节点有权值 c_i。
定义一个字符串的价值为，该字符串在 S 的后缀自动机上经过的节点的最大值。
求 S 的所有子串的价值的总和。

A. We are watching you!

后缀自动机是一个 DAG，对其进行 DAG 剖分。
后缀自动机的每个节点维护一个集合，表示可以走到该节点的子串的价值的集合。
每个节点向后序节点转移。对于轻边的转移，暴力集合合并即可。对于重边，将集合继承过去即可。
时间复杂度是每个后缀经过的轻边的总和，$O(n\log n)$。