2025 江苏省大学生程序设计大赛 & 2025 广东省大学生程序设计竞赛题解

2025 年 5 月 29 日

概况

- Easy: D, F
- Easy-Medium: G, H, J
- Medium: A, I, K
- Medium-Hard: B, C, E
- Hard: L

D. 生成魔咒

题目大意

- 现在有一个计数器和一个按钮，每按按钮 2^x 秒就会得到 10^x 的权值，问什么时候把计数器调到一个给定的值

直接模拟即可。

F. 排名预测

题目大意

- 你知道比赛结束时你们队的过题数和罚时。你通过看榜知道了某个队伍封榜前的过题数和罚时，和封榜后什么时候提交了哪些题，你想知道这个队伍有没有可能排在你们队前面。
- 如果可能，输出至少过多少题才能排在你们队前面，如果不可能输出 -1

- 你所关注的队伍获得最好成绩的方式为：对于封榜前所有未通过的题，封榜后对此题的第一次提交通过。根据这一点，可以计算出对于所有封榜前未通过的题，封榜后第一次提交通过所带来的罚时贡献，之后按罚时贡献从小到大的顺序，贪心地选择通过其中的哪些题目会使得此队伍的排名高于自己。由于数据范围很小，也可以通过子集枚举的方式求出答案。
- 在实现上有一些细节，如通过某题目后，后续再对这道题进行的提交不会对通过题目数和罚时产生贡献，应忽略；排名比较采用严格比较方式，若通过题目数相同且罚时相同，你所关注的队伍排名并不严格高于自己。
- 时间复杂度：$\mathcal{O}(n \log n)$ 或 $\mathcal{O}(n2^n)$，空间复杂度：$\mathcal{O}(n)$。

J. 解谜比赛

题目大意

- 给你一个有向图，一个点被激活当且仅当其入边被激活的数目大于等于给定的参数 a_i，一个点被激活后会立即同时激活其所有出边，激活一条边的需要的时间为 w_j 秒
- 有一些特殊的装置，可以在开始后 x 秒将给定点集的 a_i 变成 0.
- 问你选中的某个点能否被激活，以及激活它需要的最小时间
- $n \le 10^5, m \le 10^6, k \le 10^5, a_i \le 10^5, w_i, x_i \le 10^9$

- 我们考虑类似拓扑排序的思想，每次有一条边被激活后，其会对到达的点做出 1 的贡献。
- 而特殊装置可以将指定点集的 a_i 变成 0，可以看作一个超级源点向其连了一条长度为 x ，贡献为 inf 的一条边。
- 利用优先队列来实现上述过程即可。

G. 货币系统

题目大意

- 给一个货币系统，$f(x)$ 表示在尽量多换大面额纸币的情况下用多少张纸币才能凑数 x 元，给定 m 问满足 $f(i) = m$ 的正整数 i 有多少个
- 多组询问
- $n \leq 10^5, q \leq 10^6, A_i \leq 10^6, m_i \leq 10^9$

- 对于给定的数组 $\{A_k\}_{k=1}^{n}$，显然有 $f(x + A_n, n) = f(x, n) + 1$，因此只需要先算出对于 $x \in [1, A_n]$ 的情况下 $f(x, n)$ 的值，对于 $x > A_n$ 情况，直接可得 $f(x, n) = f(x \bmod A_n, n) + \lfloor x/A_n \rfloor$。
- 设 $\{B_k\}_{k=1}^{\infty}$ 数组中的元素 B_k 表示对于 $x \in [1, A_n]$ 的情况下满足 $f(x, n) = k$ 的 x 的个数，显然这个数组的所有元素之和为 A_n，并且可以发现对于 $k > A_n$ 有 $B_k = 0$，因此只需要求这个数组的前 A_n 项。

- 由此可以发现，满足 $f(x,n)=1$ 的 x 只能有 B_1 个，满足 $f(x,n)=2$ 的 x 有 B_1+B_2 个，因为对于 B_1 中的每个 x，$f(x+A_n,n)=2$，同理满足 $f(x,n)=3$ 的 x 有 $B_1+B_2+B_3$ 个，因为对于 B_1 中的每个 x，$f(x+2A_n,n)=3$，而对于 B_2 中的每个 x，$f(x+A_n,n)=3$。
- 以此类推可得，满足 $f(x,n)=m$ 的 x 恰好有 $\sum_{k=1}^{m}B_k$ 个，因此要解出本题，只需要先算出对于 $x\in[1,A_n]$ 的情况下 $f(x,n)$ 的值，再利用这些值算出 $\{B_k\}_{k=1}^{A_n}$ 数组，再对其做前缀和，根据 $\{B_k\}_{k=1}^{\infty}$ 的性质，当询问一个小于等于 A_n 的 m 时答案即为 $\sum_{k=1}^{m}B_k$，而当询问一个大于 A_n 的 m 时答案即为 $\sum_{k=1}^{m}B_k=\sum_{k=1}^{A_n}B_k$。
- 预处理复杂度为 $\mathcal{O}(A_n)$，处理询问复杂度为 $\mathcal{O}(1)$，因此总的复杂度为 $\mathcal{O}(\mathsf{max}(A_n,q))$。

H. 松散子序列

题目大意

- 给定一个长为 n 的小写字母串 S 和非负整数 k
- 对于 S 的一个子序列 T，定义 pos_i 为 T 中第 i 个字符在 S 中的下标，称 T 是好的当且仅当 $pos_i - pos_{i-1} > k$
- 问 S 有多少本质不同的子序列是好的。
- 多组数据
- $n \leq 10^6, k \leq n, \sum n \leq 10^6$

- 当 $k = 0$ 时，是一个经典的本质不同子序列问题。
- 令 $dp[i]$ 表示考虑前 i 个字符且以第 i 个字符结尾的本质不同子序列数目。
- 对于一个 i，当 $a[i]$ 未曾出现过时，$dp[i]$ 可以从之前的所有状态转移过来，即 $dp[i] = \sum_{j=0}^{i} dp[j]$。
- 令 $occ[i]$ 表示 $a[i]$ 上一次出现位置的下标。那么当 $a[i]$ 出现过时 $0 \sim occ[i] - 1$ 这一段会被重复计算，需要减去，则 $dp[i] = \sum_{j=occ[i]}^{i-1} dp[j]$。
- 当 $k \neq 0$ 时，不同之处在于考虑 $a[i]$ 时 $dp[i-k] \sim dp[i-1]$ 都没有贡献，于是：

$$dp[i+k] = \sum_{j=occ[i]}^{i-1} dp[j]$$

- 利用前缀和优化可以做到 $\mathcal{O}(n)$。

A. 矩阵游戏

题目大意

- 给定一个 $n \times m$ 的 01 矩阵，以及整数 A, B，你每次可以将一行/一列的所有元素取反
- 问 $\sum_{i=1}^{n} \sum_{j=1}^{m} (A \times i + B \times j)[a_{i,j} = 1]$ 的最大值。
- $n \leq 10^6, m \leq 10, |A|, |B| \leq 10^6, a_{i,j} \in \{0, 1\}$

- 先 $\mathcal{O}(2^m)$ 枚举每一列是否翻转，再 $\mathcal{O}(2^m)$ 枚举所有行状态。
- 对于一个行状态 s，令 $W = \sum_{j=1}^{m} j\,[s_j = 1]$，那么对于第 i 行，

  如果第 i 行等于 s：
  - 不翻转这行，贡献为 $S_1 = A \cdot i \cdot cnt_1 + B \cdot W$；
  - 翻转这行，贡献为 $S_2 = A \cdot i \cdot (m - cnt_1) + B \cdot (\frac{m(m+1)}{2} - W)$。

  其中 $cnt_1 = \mathsf{popcount(s)}$。

- 那么如果第 i 行要翻转，则需满足 $S_2 > S_1$，即 $A \cdot i \cdot (m - 2cnt_1) > B(2W - \frac{m(m+1)}{2})$。
- 不难发现对于一个固定的行状态，除了 i 之外的所有项都固定了。
- 这意味着我们的行翻转策略肯定是：存在一个分界点 k，当 $i \le k$ 时，翻转第 i 行；当 $i > k$ 时，不翻转第 i 行。或者当 $i \le k$ 时，不翻转第 i 行；当 $i > k$ 时，翻转第 i 行。
- 对每个行状态，可以采取二分的方式求出这个分界点。
- 确定分界点后，分界点前后两段贡献可以预处理前缀和快速计算。
  时间复杂度 $\mathcal{O}(2^{2m} \log n)$。

I. 队伍取名

题目大意

- 一堆名字三个字的人，问有多少三人一队组队方式使得每个人名选一个不同位置的字组成的名字是队里的一个人名
- $n \leq 10^5$

- 考虑因为队伍名不同算不同方案，而且姓名都不同，所以可以考虑先枚举一个人 i 作为队伍名，然后算有多少个选其他两个人的方案，这样就可以不重不漏的枚举所有方案。
- 首先因为相同的四元组，也可能有多个排列顺序构成同一种方案（这个易错点在样例同样有体现），不过我们考虑已经枚举了一个人，这个人的名字显然随便选任何一个字都是可以的。那么我们考虑剩下的人，只需要满足和 i 的名字里任意两个字相同就可以了（剩下第三个字 i 自己选就行）。
- 为了避免算重，我们考虑计算 $f_{i,s}$ 表示和 i 的名字里每个字是否相同的状态恰好为二进制数 s 的人名个数，这个并不好直接统计，可以先统计 $g_{i,s}$ 表示和 i 的名字里每个字是否相同的状态为二进制数 s 的超集的人名个数，然后进行 $fwt/SOSdp$ 或者暴力容斥都可以（因为 s 的集合大小只有 2^3）计算出 f。g 可以直接 2^3 枚举每个人名字的子集统计。统计的方式可以直接用 $map/unordered\_map$ 存 $vector$ 或者名字的哈希。

- 然后考虑如何利用 f 统计答案，显然 $f_{i,0}$ 先忽略，然后考虑两种情况：两个人与 i 的相同状态是否相同，如果不相同，那么显然任何两个人选出来，因为与 i 的名字相同的地方不完全一样，所以一定都是可以组成 i 的名字的，暴力 (8^2) 枚举，个数相乘就可以了。如果相同，那么要求就是 $popcount(s) \geq 2$ 的状态 s 才行，那么这一部分的答案就是 $\binom{f_{i,s}}{2}$，然后累加起来就行。
- 根据容斥和统计的实现方式，复杂度为 $\mathcal{O}(2^3 n \log n)$ 或者 $\mathcal{O}(4^3 n \log n)$ 或者 $\mathcal{O}(2^3 n)$ 或者 $\mathcal{O}(4^3 n)$。

K. 打字机

题目大意

- 给一个字符串 S，求最短的 $T = aLb$，使得 S 是 $(aLbL^R)^\infty$ 的前缀
- 对 S 的所有前缀求出上述问题的答案
- 多组数据，$|S| \le 10^6, \sum |S| \le 10^6$

- 首先 $|T| = 1$ 的情况可以特判，后文我们默认 $|T| \geq 2$。不妨记 $T' = T\,T[2..(|T|-1)]^R = T[1]T[2]\ldots T[|T|-1]T[|T|]T[|T|-1]\ldots T[2]$。
- 观察到如果 S 可以由 T 作为模版从打字机中打出，则 S 一定是 $(T')^\infty$ 的前缀。此处我们可以分三种情况讨论：
  - 1. $|S| > |T'| = 2|T| - 2$，此时 $S[1..(2|T|-1)]$ 是一个回文串，且 $|T'|$ 是 S 的一个周期
  - 2. $|T| < |S| \leq |T'|$，则 $S[(2|T| - |S|)..|S|]$ 是一个回文串
  - 3. $|S| \leq |T|$，则 S 是 T 的一个前缀

- 不难看出，对于相同的 S，情况 1 对应的 $|T_1| < \frac{1+|S|}{2}$，情况 2 对应的 $\frac{1+|S|}{2} \le |T_2| < |S|$，情况 3 $|T_3| \ge |S|$。
- 故而对于前缀 $S[1..i]$，我们应该优先考虑情况 1 对应的 T_1 是否存在（通过求解 $S[1..i]$ 的周期并判断是否为回文串），若不存在再考虑情况 2。情况 3 说明答案的上界为 $|S[1..i]| = i$。
- 这里提出一种 $\mathcal{O}(|S|)$ 的确定性做法：首先我们可以用 Manacher 求出所有奇回文串，由此可以求出所有前缀的最长回文后缀（对应情况 2），然后再利用 Z 函数配合之前 Manacher 求出的信息求出所有前缀的最短“回文周期”（对应情况 1）。最后将三种情况的答案求 $\min$ 即可。
- 其它的如带个 $\log$ 或者哈希的做法也可能通过，但是需要注意常数以及哈希冲突的风险。

C. 切牌

题目大意

- 把 1 到 n 切成非空的 k 段后，再任意排列，然后每轮按顺序依次从每一段拿出第一个元素，直到所有取完所有元素形成一个新的 1 到 n 的排列，
- 现在给你一个排列，Q 个修改，每次修改交换两个位置的数，每次修改后问你有多少种切分方法和排列方法能得到当前排列
- $n, Q \leq 10^5$

- 首先注意到答案一定不会超过 n，因为如果目标序列前 k 个数是属于某种切分方式中每一堆的第一个数，那么切分和排列方案是唯一的。
- 假设我们已经确定了前 k 个数是某一堆的第一个数，那么对于后面的 $n-k$ 个数，每个数 x 一定与 $x-1$ 在同一堆，且 $x-1$ 在目标序列的位置 p_{x-1} 在 x 的位置 p_x 左边，我们把 (p_{x-1}, p_x) 看成区间，是否合法的充要条件是 $p_{x-1} < p_x$ 且这些区间没有包含关系，证明如下：
- 必要性：如果有包含关系，假设 $p_{x-1} < p_{y-1} < p_y < p_x$，那么因为 $p_{x-1} < p_{y-1}$，x 所在堆要在 y 前面，又由于 $p_y < p_x$,x 所在的堆要在 y 后面，矛盾。
- 充分性：因为知道前 k 个数，可以直接推出切分关系，因为没有区间包含，我们可以根据前 k 个数得到一个堆之间的顺序，直接按这个排列就好。

- 通过这个我们可以推出一个结论，如果前 k 个数对应的切分是合法的，那么前 $k+1$ 个数的切分也是合法的，因为我们只是减少了区间数量，不会带来新的区间产生矛盾，于是每次询问就变成了去找最大的不满足条件的 k。
- 我们去维护每个位置 x 是否满足 $p_{a_x-1} < x$ 以及对应的区间 (p_{a_x-1}, x) 是否被右边一个位置的区间 $(p_{a_{x+1}-1}, x+1)$ 包含即可，因为如果 (p_{a_x-1}, x) 被某个区间包含，要么是 $(p_{a_{x+1}-1}, x+1)$，要么 $(p_{a_{x+1}-1}, x+1)$ 也被这个区间包含，即会在比 x 更右边的某个位置 y 产生矛盾，而我们的询问是找最右边会产生矛盾的位置，一定会找到 y。
- 维护某个位置会否合法可以 $\mathcal{O}(1)$ 做到，每次交换会影响 $\mathcal{O}(1)$ 个位置，查询最右边矛盾的位置可以用线段树实现，总体复杂度 $\mathcal{O}(n \log n)$

B. 整数生成器

题目大意

- 给你一个集合，每次可以选择两个数，把它们的 xor, and, or 中的某一个结果，插入集合
- 给定一个 x, 要在 70 次内造出 x
- $n \leq 10^5, x < 2^{30}$

- 设值域为 V。考虑 S 中的数构成的线性基，如果 x 在线性基里面，就可以在 $\log V = 30$ 次操作以内得到 x。
- 如果不在，我们要想办法让线性基变大。
- 一个猜测的想法是每次取两个数执行操作，看能不能插入线性基中。
- 这样操作次数一定不超过 $2 \log V = 60$ 次，但是暴力时间复杂度为 $\mathcal{O}(n^2 (\log V)^2)$。
- 但是实际上只需要每次考虑两个在线性基中的数，时间复杂度为 $\mathcal{O}(n \log V + (\log V)^4)$。

正确性证明：

- 设线性基为 $x_1, x_2, \ldots, x_m$，线性空间为 B，数为 $a_1, a_2, \ldots, a_n$。
- 我们证明分为两个部分：
  - 1. 若 $\forall i, j\ a_i \wedge a_j \in B$，则无论如何操作，都无法扩张 B。
  - 2. 若 $\forall i, j\ x_i \wedge x_j \in B$，则 $\forall i, j\ a_i \wedge a_j \in B$。

正确性证明：

- 对 1，由于 $a_i \vee a_j = (a_i \wedge a_j) \oplus a_i \oplus a_j$，所以只需要考虑与操作。
- 我们的操作结果一定是 S 的 $2^n - 1$ 种非空子集的与的线性组合，只需要证明每个非空子集的与都属于 B。
- 按子集大小 c 归纳，$c = 1$ 显然成立，$c = 2$，由假设成立，$c = 3$，只需证 $a_1 \wedge a_2 \wedge a_3 \in B$。
- $a_1 \wedge a_2 \in B$，因此存在 $\{q\}$，$a_1 \wedge a_2 = \oplus_i a_{q_i}$。
- 又有与对异或有分配律。
- 所以 $a_1 \wedge a_2 \wedge a_3 = (\oplus_i a_{q_i}) \wedge a_3 = \oplus_i (a_{q_i} \wedge a_3)$。
- 由归纳假设，右侧所有项属于 B，所以左式属于 B。
- 因此每个非空子集的与都属于 B，从而 1 成立。

正确性证明：

- 对 2，存在矩阵 $\{p\}$，满足 $a_i = \oplus_j x_{p_{i,j}}$。
- 而 $a_i \wedge a_j = (\oplus_l x_{p_{i,l}}) \wedge (\oplus_k x_{p_{j,k}}) = \oplus_{l,k} (x_{p_{i,l}} \wedge x_{p_{j,k}})$。
- 与上同理，2 成立。
- 所以我们只需要每次考虑线性基内的两个数即可。

E. 网格染色

题目大意

- 给你一个 $2 \times N$ 的格子，某些格子已经染色了，你要对剩余格子任意染色，使得最后同色的四连通块数量最少，给出方案
- $N \leq 2 \times 10^5$

- 我们从左往右一列一列看，我们维护之前的连通块是否能通过 0 与当前列连通，那么如果这一列有颜色且能与之前同色连通块连通的，可以直接贪心染色去连就好。
- 这里需要分这一列的两个格子是否都有颜色以及前一列是否有颜色来讨论，需要细致的代码实现，染色后更新每个颜色的连通性即可。
- 总复杂度 $\mathcal{O}(n)$。

L. 路线选择

题目大意

- 一个 $n \times m$ 网格，有一个人要从 $(1,1)$ 到 (n,m)，路上有 k 个关键点必须要经过，关键点在边上（x 坐标为实数，y 坐标为整数），每条边经过的速度不一样，问最短时间
- $n \leq 50, m \leq 4, k \leq 10^5$

- 考虑插头 dp。
- 我们要找的是固定起点终点，要经过所有关键点的路径。
- 我们对于轮廓线上的点维护两个值，度数的奇偶性和所属连通块的最小表示，只需要度数的奇偶性是因为只要起点终点度数是奇数，并且整个图连通，那么一定有欧拉通路。
- 转移的时候我们分别枚举行和列的走法，有 (1: 不走 2: 走一遍 3: 走两遍 4: 左/上走进来再回去 5: 右/下走进来再回去 6: 两端都走进来再回去)，分别去更新度数和连通性的状态即可。
- 总复杂度 $\mathcal{O}(nm7^m \times 36)$，其中 7^m 是度数奇偶性，连通性，36 是 6×6 的转移。

Thank you!