2026 ICPC Nanchang Invitational Contest

WHU ICPC 命题组　XDU ICPC 命题组

May 17th, 2026

比赛小结

- 本次比赛共收到 5092 份提交代码。
- 其中 1534 份代码正确。
- 382 个队伍有提交记录。
- 382 个队伍至少通过一题。

比赛小结

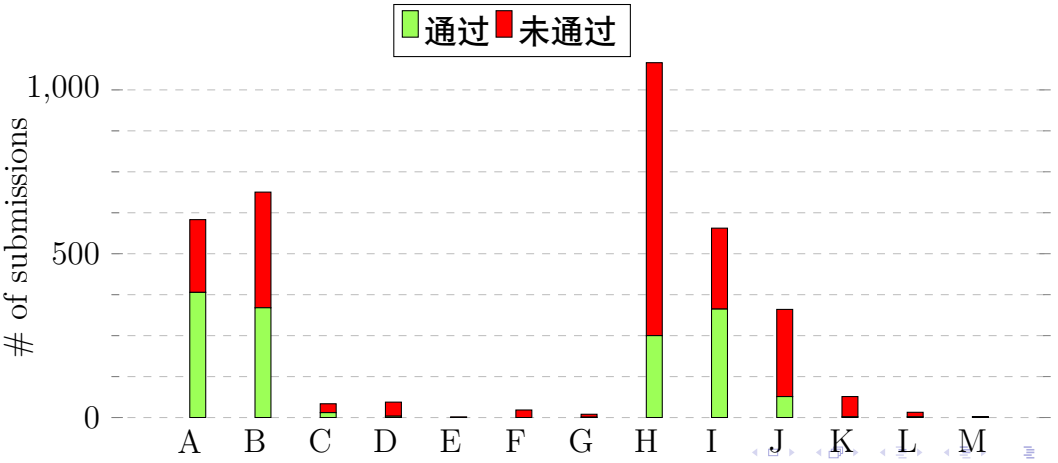
预估难度：

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

拟合难度：

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

各题通过情况



A. 马拉松的困难

- 读入 M, N,
- 判断 M 能否被 N 整除。

A. 马拉松的困难

- 读入 M, N,
- 判断 M 能否被 N 整除。
- 注意输入输出格式。

B. UNO

- 给定 $4n$ 张牌，每张牌为一个二元组 (c, x)，其中颜色 $c \in \{\mathrm{R}, \mathrm{Y}, \mathrm{G}, \mathrm{B}\}$，点数 $x \in \{1, \ldots, n\}$。每种颜色恰好出现 n 次（同色同点数可重复）。
- 若相邻两张牌有相同的颜色或点数，则可交换它们的位置。
- 判断从初始牌序列是否可通过若干次上述操作到达目标状态，即将所有牌按颜色排序为 $\mathrm{R} < \mathrm{Y} < \mathrm{G} < \mathrm{B}$，且同色内部按点数非递减。

B. UNO

- 如果两张牌不相关（颜色和点数都不同），那么它们的相对顺序不会改变。
- 无解当且仅当存在两张牌，它们无法交换且相对顺序错误。
- 枚举所有二元组进行判断即可。
- 时间复杂度 $O(n^2)$。

C. 旅行者的愿望

- 给定 n 个点 m 条边的有向无权图（边权均为 1），初始无重边无自环。
- 接下来依次加入 k 条有向边，第 i 天加入一条边（可能产生重边或自环）。
- 全局给定步数 w 和 q 个询问，每个询问指定起点 s 和终点 t。对每个询问，求最小的 d（$0 \le d \le k$，$d = 0$ 表示初始图），使得在第 d 天加边后，图中存在一条从 s 到 t 的长度恰好为 w 的路径（可重复经过点与边）。

C. 旅行者的愿望

- 一条路径出现的时间是这条路径上所有边被加入时间的最大值。
- 所以问题转化为，在所有长度为 w 的路径中，求所有边被加入时间最大值最小的那条。
- 设一条边的边权为它被加入的时间，那么我们需要求最小瓶颈路。
- 对于此类刚好走 w 步的问题，用邻接矩阵的幂的方式可以处理。
- 定义广义矩阵乘法 $c_{i,j} = \min_{k=1}^{n}\{\max\{a_{i,k}, b_{k,j}\}\}$，可以证明该定义满足结合律。设该图的邻接矩阵为 A，$A^{w}_{s_i,t_i}$ 即为答案。
- 时间复杂度 $O(n^3 \log w + q)$。

D. 吃樱桃

- 给定一棵 n 个结点的树，每个点有至多 m 种类型的樱桃，一次修改可以修改任意一个叶子结点上的所有樱桃（种类和种类数都可以修改）。
- 求以 $1, 2, ..., n$ 号点分别为根时，使所有叶子结点到根节点的路径上的樱桃集合均相等的最少修改次数。

D. 吃樱桃

- 由于 m 比较小，对 m 进行二进制压缩可以将每个节点上的樱桃转化为一个 $[0, 2^m - 1]$ 的权值。
- 根节点确定时，每一个非叶节点的值都会被或到路径中，求出这个总或值 x。对于每个叶节点，如果根到它的路径或值 y 不是 x 的超集，那么该叶子节点的值必定改变，否则可以不变。
- 因此贪心地从这些不变的叶节点中留下 y 的众数，改变剩下的所有叶子节点的值即可。
- 用一个 $n2^m$ 的数组存下每个点到其子树内所有叶子节点的路径或值，再进行换根 dp 即可。注意某个叶节点变成根时，这个点就不是叶节点了。
- 总时间复杂度为 $O(n2^m)$。

E. 完美无缺

- 给定序列，你可以进行若干次任意区间 $+1$ 操作，问至少经过多少次操作可以满足序列前缀和 s_i 在 $[l_i, r_i]$ 内。

E. 完美无缺

- 设 x_i 为第 i 个音符被 $+1$ 的次数，则增加后，前缀 $1, 2, \cdots, i$ 的总分是 $\sum_{k=1}^{i}(a_k + x_k)$。
- 记 $S_i = \sum_{k=1}^{i} a_k, \quad T_i = \sum_{k=1}^{i} x_k$，要求即为：对任意 $1 \le i \le n$，都有 $L_i - S_i \le T_i \le R_i - S_i$。
- 由于 x_i 是非负整数，所以 T_i 单调不减，所以条件可以强化为 $\max\{0, \max_{1\le j\le i}(L_j - S_j)\} \le T_i \le \min_{i\le j\le n}\{R_j - S_j\}$。
- 让我们考察下这个条件。记 $B_i = \max\left\{0, \max_{1\le j\le i}(L_j - S_j)\right\}, \quad C_i = \min_{i\le j\le n}\{R_j - S_j\}$

E. 完美无缺

- 现在问题转化成了下面的问题：

题意

给定整数序列 B 和 C 满足

$$0 \le B_1 \le B_2 \le \cdots \le B_n, C_1 \le C_2 \le \cdots \le C_n$$

需要找到整数序列 T 满足

$$T_1 \le T_2 \le \cdots \le T_n$$

且对任意 $1 \le i \le n$，都有

$$B_i \le T_i \le C_i$$

E. 完美无缺

题意

可以发现，这样的序列 T 存在，当且仅当对任意 $1 \le i \le n$，都有 $B_i \le C_i$。

在这样的序列 T 存在时，要求其对应的操作次数（下式记 $x_0 = 0$）

$$\sum_{i=1}^{n} \max\{0, x_i - x_{i-1}\}$$

的最小值.

- 这个操作次数，可以当成 $x_0, x_1, x_2, \cdots, x_n$ 上升部分的高度差之和.

E. 完美无缺

- 为了方便表达，记 $B_0 = C_0 = 0$。
- 考察区间 $[l, r]$，其中 $1 \le l \le r \le n$，我们有

$$B_r - C_{l-1} \le T_r - T_{l-1} \le C_r - B_{l-1}$$

- 即

$$B_r - C_{l-1} \le \sum_{i=l}^{r} x_i \le C_r - B_{l-1}$$

- 可以得出：
- 存在整数 $i \in [l, r]$，使得 $x_i \ge \lceil \frac{B_r - C_{l-1}}{r - l + 1} \rceil$；
- 存在整数 $i \in [l, r]$，使得 $x_i \le \lfloor \frac{C_r - B_{l-1}}{r - l + 1} \rfloor$。

E. 完美无缺

- 选取若干个区间 $[l_1, r_1], [l_2, r_2], \cdots, [l_{2m}, r_{2m}], [l_{2m+1}, r_{2m+1}]$，其中

$$1 \le l_1 \le r_1 < l_2 \le r_2 < \cdots < l_{2m} \le r_{2m} < l_{2m+1} \le r_{2m+1} \le n$$

- 则下面的命题都是成立的：
- 存在 $i_1 \in [l_1, r_1]$，使得 $x_{i_1} \ge \lceil \frac{B_{r_1} - C_{l_1-1}}{r_1 - l_1 + 1} \rceil$；
- 存在 $i_2 \in [l_2, r_2]$，使得 $x_{i_2} \le \lfloor \frac{C_{r_2} - B_{l_2-1}}{r_2 - l_2 + 1} \rfloor$；
- ……
- 存在 $i_{2m} \in [l_{2m}, r_{2m}]$，使得 $x_{i_{2m}} \le \lfloor \frac{C_{r_{2m}} - B_{l_{2m}-1}}{r_{2m} - l_{2m} + 1} \rfloor$；
- 存在 $i_{2m+1} \in [l_{2m+1}, r_{2m+1}]$，使得 $x_{i_{2m+1}} \ge \lceil \frac{B_{r_{2m+1}} - C_{l_{2m+1}-1}}{r_{2m+1} - l_{2m+1} + 1} \rceil$。

E. 完美无缺

- 我们可以发现以下的上升：
- 从 x_0 到 x_{i_1}，至少提供 $\lceil \frac{B_{r_1} - C_{l_1-1}}{r_1 - l_1 + 1} \rceil$ 的上升高度差；
- 从 x_{i_2} 到 x_{i_3}，至少提供 $\lceil \frac{B_{r_3} - C_{l_3-1}}{r_3 - l_3 + 1} \rceil - \lfloor \frac{C_{r_2} - B_{l_2-1}}{r_2 - l_2 + 1} \rfloor$ 的上升高度差；
- $\cdots\cdots$
- 从 $x_{i_{2m}}$ 到 $x_{i_{2m+1}}$，至少提供 $\lceil \frac{B_{r_{2m+1}} - C_{l_{2m+1}-1}}{r_{2m+1} - l_{2m+1} + 1} \rceil - \lfloor \frac{C_{r_{2m}} - B_{l_{2m}-1}}{r_{2m} - l_{2m} + 1} \rfloor$ 的上升高度差。
- 因此，操作次数的下界为：

$$\lceil \frac{B_{r_1} - C_{l_1-1}}{r_1 - l_1 + 1} \rceil - \lfloor \frac{C_{r_2} - B_{l_2-1}}{r_2 - l_2 + 1} \rfloor + \lceil \frac{B_{r_3} - C_{l_3-1}}{r_3 - l_3 + 1} \rceil - \cdots - \lfloor \frac{C_{r_{2m}} - B_{l_{2m}-1}}{r_{2m} - l_{2m} + 1} \rfloor + \lceil \frac{B_{r_{2m+1}} - C_{l_{2m+1}-1}}{r_{2m+1} - l_{2m+1} + 1} \rceil$$

E. 完美无缺

- 因此：最少操作次数大于或等于所有选法的对应式子中的最大值，且这个最大值就是最少操作次数。
- 考虑 DP 解决。

E. 完美无缺

- 设 f_r 是只看位置 $1, 2, \cdots, r$，选出奇数个（设是 $2m+1$ 个）区间，并且 $r_{2m+1} = r$ 时，下面式子的最大值。

$$\lceil \frac{B_{r_1} - C_{l_1-1}}{r_1 - l_1 + 1} \rceil - \lfloor \frac{C_{r_2} - B_{l_2-1}}{r_2 - l_2 + 1} \rfloor + \lceil \frac{B_{r_3} - C_{l_3-1}}{r_3 - l_3 + 1} \rceil - \cdots - \lfloor \frac{C_{r_{2m}} - B_{l_{2m}-1}}{r_{2m} - l_{2m} + 1} \rfloor + \lceil \frac{B_{r_{2m+1}} - C_{l_{2m+1}-1}}{r_{2m+1} - l_{2m+1} + 1} \rceil$$

- 并规定 $f_0 = -\infty$.
- g_r 是只看位置 $1, 2, \cdots, r$，选出偶数个（设是 $2m$ 个）区间，并且 $r_{2m} = r$ 时，下面式子的最大值。

$$\lceil \frac{B_{r_1} - C_{l_1-1}}{r_1 - l_1 + 1} \rceil - \lfloor \frac{C_{r_2} - B_{l_2-1}}{r_2 - l_2 + 1} \rfloor + \lceil \frac{B_{r_3} - C_{l_3-1}}{r_3 - l_3 + 1} \rceil - \cdots - \lfloor \frac{C_{r_{2m}} - B_{l_{2m}-1}}{r_{2m} - l_{2m} + 1} \rfloor$$

- 并规定 $g_0 = 0$。

E. 完美无缺

- 转移如下：

$$\forall 1 \le r \le n, f_r = \max_{1 \le l \le r} \left\{ \left(\max_{0 \le u \le l-1} g_u \right) + \lceil \frac{B_r - C_{l-1}}{r-l+1} \rceil \right\}$$

$$\forall 1 \le r \le n, g_r = \max_{1 \le l \le r} \left\{ \left(\max_{0 \le u \le l-1} f_u \right) - \lfloor \frac{C_r - B_{l-1}}{r-l+1} \rfloor \right\}$$

- 答案是 $\max_{1 \le r \le n} f_r$。
- 时间复杂度 $O(n^2)$。

F. 一笔画

- 给定一张无向图，可能含重边和自环，不保证连通。图上有 k 对互不相同的节点构成“跳跃点对”。
- 若某次沿边移动到达一个传送点，则立即强制传送到配对的另一节点。
- 求是否存在一条路径恰好经过每条边一次，若有则输出方案。

F. 一笔画

- 题目要求在一张特殊图上的欧拉路径。
- 考虑如何转化跳跃点对才能不破坏普通无向图上的欧拉路性质。
- 发现我们可以将一个跳跃点对看作是同一个点的两端。
- 这样我们只需要确定起点跑一遍带强制传送的欧拉路径即可。

F. 一笔画

- 对于跳跃点对 (x, y)，设 d_i 为度数，做如下分类讨论：
- 若 $d_x = d_y + 1$，那么存在欧拉路径，且 y 为起点或终点。$d_y = d_x + 1$ 同理。
- 若 $d_x = d_y + 2$，那么存在欧拉回路，且 x 为起点。
- 若 $d_x = d_y$，那么起点或终点可能是任意一个有连边的点。
- 其他情况必然无解。
- 注意判断图不连通或者有边没有走到的情况。
- 时间复杂度 $O(n + m)$。

G. 世界的意义

- 给定括号串，维护单点修改、区间查询最长合法子串。

G. 世界的意义

- 首先考虑暴力匹配怎么做：用一个栈来维护当前没有被匹配的左括号以及起始位置。遍历整个区间，如果遇到左括号就入栈，如果为右括号就出栈并尝试更新答案。如果出栈后栈内为空，就将当前位置重新作为起始位置入栈。
- 注意到：

结论

对于任意括号串，删除所有的合法括号子串，必然得到形如))))(((的结果。

- 基于这个性质可以解决本题。
- 不难发现，在具体的匹配过程中，匹配的栈序列是固定的。

G. 世界的意义

- 从匹配的左端点出发。对于散块直接暴力，对于整块如下操作：
- 需要知道当前全来了几个左括号。设为 m_0。
- 对于整块分为三部分：一个右括号前缀、一个合法括号串、一个左括号后缀。注意这里我们是对于原始的字符串进行操作，即这三部分内也会包含若干合法串。方便起见设这三部分长度为 a_1, a_2, a_3，两侧的括号数量为 b_1, b_2。
- 考虑匹配的过程：我们拿着 m 个左括号去和这个块进行匹配。然后首先面对的是这个块的右括号：
- 如果 $b_1 \leq m$，则可以接着匹配下去。
- 如果 $b_1 > m$，则匹配不下去了，这时就需要截断重新开始。
- 然后面对中间的完美匹配序列，发现没有影响。
- 然后面对这个块的左括号，给 m 加上 b_2。

G. 世界的意义

- 记 $l_{k,i}$ 表示第 k 个块，如果 a_3 部分用 i 个右括号匹配上，最靠左可以到达的位置；$r_{k,i}$ 表示第 k 个块，如果 a_1 部分用 i 个左括号匹配上，最靠右可以到达的位置。
- 维护一个栈 s，用于存放当前的所有未匹配左括号。发现这样做是 $O(n)$ 的，所以我们只去维护每一个块内有多少个左括号未被匹配，以及这些括号的来源是哪个块。当然散块只能暴力。特别地，栈底维护当前匹配的括号序列的左端点。
- 对于整块做上述的匹配操作即可。栈内元素是 $O(\sqrt{n})$ 级别的。每一个元素最多只会被入栈一次，出栈一次。
- 然后注意到，这个过程只考虑了横跨两个块的答案，没有考虑最优解在块内的情况。因此我们需要额外处理块内容案，计算的时候顺便取一下 $\max$ 即可。
- 时间复杂度 $O(n\sqrt{n})$

G. 世界的意义

考虑用线段树维护区间信息。每次合并仅影响中间新产生的合法括号子串，且答案单调不减。具体地：

- 每个区间记录：右括号前缀长度、左括号后缀长度、前后缀中左/右括号数量，以及区间总长。
- 合并时：
- 若左区间左括号数 $=$ 右区间右括号数，新区间前后缀直接继承，中间部分 $=$ 总长 $-$ 前后缀长。
- 否则，必有一侧完全用尽，另一侧需部分选取。以左区间左括号后缀数 $>$ 右区间右括号前缀数（记为 len）为例：递归地在左区间中取出长度为 len 的左括号后缀（若左子区间左括号后缀不足 len，则全取；否则跨左右子区间递归取）。对称情况同理。
- 维护前后缀及括号数量，并用合并产生的合法串大小更新答案。

G. 世界的意义

- 维护一个合并的开销是 $\log n$ 级别的，线段树深度也为 $\log n$ ，所以单次修改的开销就是 $\log^2 n$
- 那我们再看看求取答案的时候，其实可以视作至多 $\log n$ 个区间顺次合并并维护答案，所以我们可以按合并顺序建一个临时的树结构并逐层合并，单次查询的开销也是 $\log^2 n$
- 至此我们可以在 $\mathrm{O}(n \log n + q \log^2 n)$ 的时间复杂度内完成这道题。

题意

Nim 游戏，但是第一次拿的不能比 k 多，之后每一次都不能比上一次拿的多，拿走最后一个石头的人赢。

$k = 1$ 的情况是平凡的，判断石头总数奇偶性即可。

考虑 $k >= 2$ 的情况，我们发现如果当前是奇数那么取走 1 个直接胜利了，如果是偶数，先手如果选择取走奇数个的话就会留给后手一个奇数的局面，直接输掉了，于是相当于先手只会取走偶数个石头。

于是导致接下来每一步都会是总数是偶数的情况，直到每一堆都只有一个或者零个石头，此时这个玩家不管是没石头拿了还是被迫只拿走一个石头都会输掉。

于是我们发现若 $k \geq 2$，且总和为偶数，这个游戏和一个新的游戏等价：

$$a_i \leftarrow \lfloor a_i/2 \rfloor, k \leftarrow \lfloor k/2 \rfloor$$

继续把 2 推广到 2^t 。我们会发现如果存在 $2^t \le k$ 且 $\sum \lfloor \frac{a_i}{2^t} \rfloor$ 是奇数，并且对于所有 $t' < t$ 都不满足这个条件，那么先手就会选择取走 2^t 个石头直接获胜。否则，先手无论怎么取都会留给后手一个上述局面。

考虑这个 t ，实际上 2^t 就是 a_i 异或和的 lowbit。

于是相当于只需要比较异或和的 lowbit 和 k 的大小，如果 lowbit 小于等于 k 那么先手必胜，否则先手必败。

时间复杂度 $O(n)$。$O(n \log n)$ 的解法和正解本质一致，也能通过这题。

I. 搞快点搞快点

- 初始给定一个蛋仔柱，四个蛋仔自下而上依次为 A、B、C、D，堆叠成高度为 4 的柱体，底部位于坐标 0。赛道由 l 段长度为 1 的连续线段组成，第 i 段连接坐标 $i-1$ 与 i，其上地形由字符 $S_i \in \{0,1,2\}$ 给出。
- 当柱体整体从 x 前进至 $x+1$ 时，耗时仅由第 $x+1$ 段地形决定：若为 0（平地）耗时 t_0，若为 1（泥潭）耗时 t_1，若为 2（加速带）耗时 t_2。
- 此外，若当前柱体底部位于 x 且包含 $j \geq 2$ 个蛋仔，则可执行“超级起步”：底部的蛋仔瞬间将上方 $j-1$ 个蛋仔（包括 D）抛至坐标 $\min(l, x+d)$，该操作耗时为 0，随后原地底部蛋仔消失，新位置形成一个高度为 $j-1$ 的柱体。
- 目标是使得蛋仔 D（始终保持在所在柱体的顶端）到达坐标 l 的最短时间。

I. 搞快点搞快点

- 注意到初始时只有 4 个蛋仔。每使用一次超级起步，最底部的蛋仔都会被淘汰一个，因此超级起步最多只会使用 3 次。
- 这说明我们不需要记录当前还剩哪些蛋仔，也不需要记录蛋仔的具体顺序。只要知道已经使用了多少次超级起步，就能知道接下来是否还能继续使用超级起步。
- 设 $dp[k][x]$ 表示已经使用了恰好 k 次超级起步，并且当前包含蛋仔 D 的蛋仔柱位于坐标 x 时的最小耗时，其中 $0 \le k \le 3$，$0 \le x \le l$。
- 初始 $dp[0][0] = 0$

I. 搞快点搞快点

- 考虑一个已经可达的状态 $dp[k][x]$：
- 若 $x < L$，可以普通前进到 $x+1$。这会经过第 $x+1$ 段线段，设 S_{x+1} 对应的耗时为 $cost(x+1)$，则可以更新：

$$dp[k][x+1] = \min(dp[k][x+1], dp[k][x] + cost(x+1))$$

- 若 $k < 3$，还可以使用一次超级起步。它不花费时间，并且会到达 $\min(L, x+d)$，因此可以更新：

$$dp[k+1][\min(L, x+d)] = \min(dp[k+1][\min(L, x+d)], dp[k][x])$$

- 最终的答案为：

$$\min_{0 \le k \le 3} dp[k][L]$$

J. GCD 之旅

- 有一个 n 个点的完全图，其中连接点 u 和点 v 的无向边的边权为 $\gcd(a_u, a_v)$。
- 有 q 组询问，每次询问在所有从点 x 出发到达点 y 的路径中，边权最小值的最大值是多少。
- $1 \leq n, q, a_i \leq 10^6$。

J. GCD 之旅

- 考虑按照边权从大到小建出 Kruskal 重构树。但是边数是 $O(n^2)$ 的，不能枚举所有边进行建图。
- 不妨先将点权 a 去重。
- 假设有 u, v, w 三个点，并且 $\gcd(a_u, a_v) = a_w$，则将路径 $u \to v$ 拆分成 $u \to w \to v$ 不会改变边权最小值。因此可以建 n 个中转点，其中中转点 k 向所有满足 $k \mid a_u$ 的点 u 连边，边权为 k。这样，原图的所有边都可以等价成利用中转点来转移。
- 这样建图的边数是 $\sum_{k=1}^{n} \frac{n}{k} = O(n \ln n)$。
- 建出 Kruskal 重构树后，题目等价于求出点 x 和点 y 的 lca。
- 时间复杂度为 $O(n\alpha(n) \ln n)$。

K. 祖传代码

- 有 m 行代码，每行代码都是 $a_x \leftarrow a_u \times a_v$ 的形式。
  $(x, u, v \in \{1, 2, \ldots, n\})$
- 给定变量的初始值，预测这些变量在代码循环 k 遍以后的最终值。
- 一共有 q 组询问，答案对 $MOD = 998\,244\,353$ 取模。
- $1 \leq n \leq 200$, $1 \leq m \leq 2 \times 10^5$, $1 \leq q \leq 200$, $1 \leq k \leq 10^6$。
- 时间限制为 5s。

K. 祖传代码

- 每一次循环相当于执行 $a_i \leftarrow \prod_{j=1}^{n} a_j^{c_{i,j}}$。
- 定义矩阵右乘列向量：$(C \otimes \alpha)_i = \left(\prod_{j=1}^{n} a_j^{c_{i,j}} \right) \mod MOD$。
- 因此答案等价于求出 $\underbrace{C \otimes \left(\cdots (C \otimes (C \otimes \alpha)) \cdots \right)}_{k\text{个}C}$。

K. 祖传代码

- 可以发现最终每个数都可以写成 $\left(\prod_{j=1}^{n} a_j^{d_{i,j}}\right) \bmod MOD$ 的形式。
- 考虑继续化简 $\underbrace{C \otimes \Big(\cdots\big(C \otimes (C \otimes \alpha)\big)\cdots\Big)}_{k\text{个}C}$。
- 定义矩阵乘法：$(A \oplus B)_{i,j} = \left(\sum_{k=1}^{n} a_{i,k} \times b_{k,j}\right) \bmod (MOD-1)$。
- 定义矩阵的幂：$A^k = \underbrace{A \oplus A \oplus \cdots \oplus A}_{k\text{个}A}$。
- 由费马小定理，上面的式子等价于 $C^k \otimes \alpha$。

K. 祖传代码

- 直接矩阵快速幂计算 $C^k \otimes \alpha$ 的总复杂度是 $O(qn^3 \log k)$，不能接受。
- 单次矩阵乘法是 $O(n^3)$ 的，单次矩阵乘向量是 $O(n^2 \log MOD)$ 的。
- 考虑预处理 C^{2^t}，来减少矩阵乘法的次数。
- 于是可以把 $C^k \otimes \alpha$ 拆成 $C^{2^{t_r}} \otimes \left(\cdots \left(C^{2^{t_2}} \otimes \left(C^{2^{t_1}} \otimes \alpha \right) \right) \cdots \right)$ 计算。
- 这样总复杂度是 $O(n^3 \log k + qn^2 \log k \log MOD)$。
- 但是时间限制只有 5s，这样还不足以通过。

K. 祖传代码

- 方法一：
  - 既然瓶颈在快速幂，可以想办法把快速幂省掉。
  - 考虑使用 BSGS 算法对所有数取离散对数，把乘法转化成加法。
  - 这样总复杂度就是 $O(\sqrt{nq \cdot MOD} + n^3 \log k + qn^2 \log k)$ 了。
  - 该做法 std 最大运行时间为 1.024s。
- 方法二：
  - 把 k 拆成二进制太慢了，考虑拆成 B 进制，并预处理 $C^{r \times B^t}$。
  - 总时间复杂度 $O(Bn^3 \log_B k + qn^2 \log_B k \log_2 MOD)$。
  - 取 $B = 20$ 足以通过。该做法 std 最大运行时间为 3.875s。
- 还有其他合理的优化方法（如分块快速幂），也足以通过本题。
- 注意特判 0 的传播。
- Fun Fact: GPT 的做法最大运行时间为 0.185s。

L. 小 Z 的函数

- q 次询问，每次对区间 $[l, r]$ $(1 \le l < r \le n)$ 求：
- $\max_{l \le i < j \le r} \varphi(p_i \times p_j)$
- 时间限制为 3s

L. 小 Z 的函数

- 利用公式，将 $\varphi(p_i \times p_j)$ 拆开
- $\varphi(p_i \times p_j) = \varphi(p_i)\varphi(p_j) \cdot \frac{\gcd(p_i,p_j)}{\varphi(\gcd(p_i,p_j))}$
- 不妨假设 $\gcd(p_i, p_j) = g$，注意到对于任意 g 的正因子 d，都有 $\frac{d}{\varphi(d)} \leq \frac{g}{\varphi(g)}$ 成立。
- 我们考虑枚举 d，去计算所有满足 $d \mid p_i, p_j$ 的贡献：$\varphi(p_i)\varphi(p_j) \cdot \frac{d}{\varphi(d)}$

L. 小 Z 的函数

- 固定 d，我们现在只关心查询区间内最大和次大的两个 $\varphi(p_i)$ 的乘积是多少。
- 对每一个 p_i，找到满足 $\varphi(p_j) \geq \varphi(p_i), j < i$ 的最大的 j_{pre} 以及同理 $\varphi(p_j) \geq \varphi(p_i), j > i$ 的最小的 j_{suf}。
- 此时，若只考虑所有 (j_{pre}, i) 和 (i, j_{suf}) 对，则每个查询区间的最大乘积对将会是次大的 $\varphi(p_i)$ 处的两对之一。
- 综上，我们可以将这 $O(n \ln n)$ 对取出并进行扫描线，即可做到 $O(n \ln n \log n)$ 的复杂度。
- 该做法 std 最大运行时间为 1.807s，足以通过本题。

M. Kingdam of Construction

- 给定一个 $n \times m$ 的矩阵 a，
- 你可以执行最多 9961 次操作，每次操作为翻转一行或一列，
- 最后使 a 变为目标矩阵 $b = \begin{bmatrix} 1 & 2 & \cdots & m \\ m+1 & m+2 & \cdots & 2m \\ \vdots & \vdots & \ddots & \vdots \\ (n-1)m+1 & (n-1)m+2 & \cdots & nm \end{bmatrix}$，或报告无解。
- $n, m \leq 134$ 且均为偶数。

M. Kingdam of Construction

- 给定一个 $n \times m$ 的矩阵 a，
- 你可以执行最多 9961 次操作，每次操作为翻转一行或一列，
- 最后使 a 变为目标矩阵 $b = \begin{bmatrix} 1 & 2 & \cdots & m \\ m+1 & m+2 & \cdots & 2m \\ \vdots & \vdots & \ddots & \vdots \\ (n-1)m+1 & (n-1)m+2 & \cdots & nm \end{bmatrix}$，或报告无解。
- $n, m \leq 134$ 且均为偶数。

- 首先发现一个元素只会在 $(i,j)/(i,m-j+1)/(n-i+1,j)/(n-i+1,m-j+1)$ 这四个位置（“轨道”）上移动。如果一个轨道上的四个数的集合和对应目标轨道上四个数的集合不同则一定无解。

M. Kingdam of Construction

- 这提示我们把这个 2×2 的轨道作为研究对象。(不妨四个数从小到大重编号为 $0, 1, 2, 3$。)
- 视其为长为 4 的排列（轨道上的四个数按左上、右上、左下、右下顺序构成排列），翻转操作即对一行或一列的所有排列进行变换：交换特定两个位置的元素，目标把所有排列都变为 $p_0 = [0, 1, 2, 3]$。
- 记 $r_{i,j}$ 为排列 $[a_{i,j}, a_{i,m-j+1}, a_{n-i+1,j}, a_{n-i+1,m-j+1}]$ 的奇偶性，则我们每次操作会选一行或一列的轨道将其 r 值全部取反。
- 经典问题，记 row_i 和 col_j 分别为翻转第 i 行和第 j 列的次数的奇偶性，那么有 $r_{i,j} = row_i \operatorname{xor} col_j$。令 $row_1 = 0$，发现剩下都能求，如果遇到矛盾则无解。
- 那么对每行每列翻转 row_i/col_j 次后所有排列均为偶排列。

M. Kingdam of Construction

- 接下来考虑如何对单个轨道处理，即进行操作将当前偶排列变为 p_0，而不影响其他排列。容易发现如果公式内每种操作都出现了偶数次，执行前后就不会对其他排列产生影响。
- （记操作 U 为翻转轨道上方的行，D 为翻转下方的行，R 为翻转右列，L 为翻转左列。）样例已给出一种三轮换方法是 LULU，除右下角的元素外，其余三个元素顺时针轮换。类似地能得到其他 7 种三轮换公式。
- 偶排列一共有 12 种，对于三种非三轮换的情况也能手玩出公式：
  1. $p_1 = [1,0,3,2]$，$\begin{bmatrix}1,0\\3,2\end{bmatrix}$，即左右相邻换：URLURL 或 RDRD + URUR；
  2. $p_2 = [2,3,0,1]$，$\begin{bmatrix}2,3\\0,1\end{bmatrix}$，即上下相邻换：RUDRUD 或 URUR + LULU；
  3. $p_3 = [3,2,1,0]$，$\begin{bmatrix}3,2\\1,0\end{bmatrix}$，即对角换：URUDRD 或 URUR + RDRD。

M. Kingdam of Construction

- 此时我们已经有了一个 $cnt = \frac{n}{2} + \frac{m}{2} + 8\left(\frac{n}{2}\right)\left(\frac{m}{2}\right) \leq 36,046$ 的解。
  - 如果操作数要求为 $36,046$，所有偶排列都至多用两次顺时针三轮换变为 p_0。
- 但该题要求为 9961 次，也就是说均摊下来每个轨道只能用 $2+\epsilon$ 次！
- 事实：正常做法行不通，因为如果不影响其他排列，两次操作无法进行任何有意义的变换。
- 不可能完成的任务？

M. Kingdam of Construction

- 此时我们已经有了一个 $cnt = \frac{n}{2} + \frac{m}{2} + 8\left(\frac{n}{2}\right)\left(\frac{m}{2}\right) \le 36,046$ 的解。
  - 如果操作数要求为 $36,046$，所有偶排列都至多用两次顺时针三轮换变为 p_0。
- 但该题要求为 9961 次，也就是说均摊下来每个轨道只能用 $2+\epsilon$ 次！
- 事实：正常做法行不通，因为如果不影响其他排列，两次操作无法进行任何有意义的变换。
- 不可能完成的任务？
- 介绍…公共骨架复用！
- 我们发现对于一行的所有排列，行翻转会影响所有的排列，但是列翻转互不影响。所以考虑一行的轨道的所有操作可以化为常数级别次行翻转（“公共骨架”）中插入每个轨道的列翻转，
- 比如我们选取公共骨架为 UUDDUDUD，那么 $\mathrm{R}_1\mathrm{UR}_1\mathrm{U} + \mathrm{UR}_2\mathrm{UDR}_2\mathrm{D}$ 可以合并为 $\mathrm{R}_1\mathrm{UR}_1\mathrm{R}_2\mathrm{UDR}_2\mathrm{DUDUD}$。

M. Kingdam of Construction

- 注意，合并时必须保证骨架在每个公式执行前的部分不能对该轨道产生影响，比如如果公共骨架为 UUDDUDUD，公式为 DRUDRU，合并为 UUDDRUDRUD 是有问题的，因为相当于在公式前后各加了一个 D 导致起始状态不对。
- 那么我们搞出这个东西后（公共骨架为 UUDDUDUD）发现除了左右相邻换 p_1（URLURL），其他情况都只会在框架里插入不超过两次 L/R 操作，这样就有希望均摊每个轨道两次了！
- 左右相邻换 p_1 怎么办？注意到一次 UD 可以把这一行的所有 p_1 解决掉（变为目标态），但是会影响到该行其他的轨道的状态。
- 再注意到，UD 后只有 $p_0 = [0, 1, 2, 3]$ 会变成 $p_1 = [1, 0, 3, 2]$，所以如果 p_1 只要比 p_0 多的话就 UD，否则不管，之后 p_1 一定不多于 p_0，而 p_0 是 0 次，所以这些轨道均摊起来是不超过 2 次的！

M. Kingdam of Construction

- 做完了，分析次数：
  1. 最开始的奇偶调整（对每行每列翻转 row_i/col_j 次），为 $\frac{n}{2}+\frac{m}{2} \le 134$ 次；
  2. 对于每行的轨道，一次 UD 加上长度为 8 的框架加上每个轨道均摊插入 2 次，为 $2+8+2\left(\frac{m}{2}\right) \le 144$ 次每行；
  3. 所以方案不超过 $134+67\times 144=9782$ 次，符合要求。
- Fun Fact: 有大神验题人卡到了 9380 次。
- 构造的时空复杂度均为 $O(n^2)$（将 n 和 m 视为同阶）。