

# XJTUPC 2025 题解

西安交通大学学生算法竞赛协会

2025.5.18

# 预期题目难度顺序

**Easy:** A, B, C, D, E

**Easy Medium:** F, O

**Medium:** N, L, H, I

**Medium Hard:** J, G, M

**Hard:** K

# A. 纸牌大师

**Idea:** Tom22l

**Data:** Tom22l

**Verify:** hwpvector, Meguhine

## A. 纸牌大师

语法题，考察条件语句的使用。

依照题意模拟即可。

## B. 全都登不上

**Idea:** Shirost

**Data:** Shirost

**Verify:** HCCH, PandaGhost, Meguhine

## B. 全都登不上

语法题，考察字符串读入、循环语句与条件语句的使用。

依照题意模拟即可。

## C. 量子力学

**Idea:** MCPlayer542

**Data:** MCPlayer542

**Verify:** cutewhite

## C. 量子力学

模拟。

对每个  $i \in [0, 2^n)$  求出量子系统转化成二进制后为  $i$  的概率  $p_i$ , 注意使用浮点数对来模拟复数的乘法运算:

$$(a + bi) \cdot (c + di) = (ac - bd) + (ad + bc)i$$

之后枚举二进制, 通过位运算对于每一位求和。

## D. 对称隔离：黑白之战

**Idea:** TAIYANGFENG

**Data:** TAIYANGFENG

**Verify:** MCPlayer542

## D. 对称隔离：黑白之战

模拟。

先假设平面有水平或者竖直对称轴。

若原先在某一位置有黑色地毯，则将其对称的位置染成黑色。这一操作是必要的，否则平面内将不存在对称轴。同时，这一操作也保证了平面内一定存在一个对称轴。

若我们继续进行染色，则一定不优（在满足对称轴条件的情况下，染色操作更多，会导致黑色地毯更有可能共享公共边）。

于是，再检验是否存在两个黑色地毯共享公共边即可。

## E. 公道杯

**Idea:** hwpvector

**Data:** PandaGhost

**Verify:** MCPlayer542, hwpvector

## E. 公道杯

博弈论。

首先，初始情形下  $E$  与  $F$  无异，因此将所有的  $F$  都更改为  $E$ 。现在序列中所有的元素均为  $E$  或者  $H$ 。

构造长度为  $n$  的序列  $\{b_i \in \{0, 1\}\}$ ，满足：

- $b_i = 0$  当且仅当第  $i$  为是  $E$ ；
- $b_i = 1$  当且仅当第  $i$  位是  $H$ 。

### 结论

- Alice 与 Bob 的操作决策与胜负无关；
- Alice 会输当且仅当  $\sum_{i=1}^n i \cdot b_i \equiv (n+1) \sum_{i=1}^n b_i \pmod{2}$ 。

## E. 公道杯 - 证明

令  $P = \{x \mid b_x = 1\}$ , 那么一次挪杯子操作可以概括如下:

选择  $x \in P$  且  $x \neq n+1$ , 若  $x+1 \notin P$ , 则新的位置集合为  $P' = (P \cup \{x+1\}) \setminus \{x\}$ , 否则  $P' = P \setminus \{x+1, x\}$ ,

注意到这个过程中恒有:

$$(\sum_{x \in P} x) + 1 \equiv \sum_{x \in P'} x \pmod{2}$$

这意味着无论怎么操作, 每个人对应的位置集合  $P$  的元素和的奇偶性是不变的, 而一个会输的格局是所有元素都被挪到了  $n+1$  的位置, 此时的奇偶性为  $(n+1) \sum_{i=1}^n b_i \pmod{2}$ .

## E. 公道杯 - Bonus

我们对题意进行修改：我们开始时额外给定正整数  $k$  ( $1 \leq k \leq |S|$ )。

修改 Alice 和 Bob 在博弈中的一次操作的定义：每次选择一个非空的公道杯，将其倒入右边的  $k$  个公道杯之一。若你选择了最右侧的  $k$  个公道杯之一，也可以直接选择将其倒掉。

假设 Alice 和 Bob 聪明绝顶，每次都执行最优操作，那么谁必胜？

## F. Beat Verdict: Precision Strike

**Idea:** HCCH

**Data:** HCCH

**Verify:** hwpvector

二分。思维。

关注到如下性质：

如果  $x \in [1, 4]$ ，那么我们回答 2；

如果  $x \in [5, 20]$ ，那么我们回答 10；

如果  $x \in [21, 84]$ ，那么我们回答 42 ...

---

令正整数序列  $\{a_n\}$  满足：  $a_1 = 1, a_{i+1} = 4a_i + 1$ ，

若  $x \in [a_i, a_{i+1})$  那么我们回答  $2a_i$ 。

通过打表可知  $a_{17} = 5.7 \times 10^9$ 。于是，我们可以将从 1 到  $5 \times 10^9$  之间的整数恰好分成 16 个连续的段。

## F. Beat Verdict: Precision Strike

之后像猜数游戏，二分询问中间段的边界。这样每次能够筛掉一半的段。

对于 16 个段，经过 4 次询问可以恰好知道  $x$  在哪一段中，之后根据预设的情形来回答即可。

注意二分实现的细节问题。

# O. 离散对数

**Idea:** Rhodoks

**Data:** Rhodoks

**Verify:** Yeyin\_0, PandaGhost

## O. 离散对数

数论。

首先对等式应用欧拉定理有：

$$a^b \equiv a^{b \bmod (p-1)}$$

$$b^c \equiv (b \bmod p)^c$$

设  $x = b \bmod (p-1)$ ,  $y = b \bmod p$ ,

由群论或数论知  $(\mathbb{N}_{p-1} \times \mathbb{N}_p, +)$  是以  $(1, 1)$  为生成元的循环群；

换一句话说,  $\forall x \in [0, p-1), y \in [0, p), \exists b \in [1, p(p-1)]$  满足：

$$b \bmod (p-1) = x \wedge b \bmod p = y$$

## O. 离散对数

现在我们需要找到  $x = b \bmod (p-1)$  和  $y = b \pmod{p}$ , 使得:

$$a^x \equiv y^c \pmod{p}$$

一旦找到一组  $x, y$ , 我们可以通过 CRT (中国剩余定理), 立刻构造出  $b$ 。

我们可以有如下构造方式:

- 令  $x = c, y = a$ , 根据 CRT 有
$$b = ((p-1)^2 \cdot a + p^2 \cdot c) \bmod (p \cdot (p-1));$$
- 令  $x = 0, y = 1$ , 根据 CRT 有  $b = (p-1)^2$ 。

## N. 结束乐队

**Idea:** Yeyin\_0

**Data:** Yeyin\_0

**Verify:** uuku

## N. 结束乐队 - 二分解法

二分。

对于一个排列  $\{p_n\}$ ，我们记子段  $a[l \dots r]$  的 mex 值为区间  $[l, r]$  上未出现的最小正整数， $pos_i$  为元素  $i$  在排列中出现的位置。

考虑一个结论：排列的一个子段  $a[l \dots r]$  的 mex 值大于  $x$  的一个充要条件是：

$$\forall i \in [1, x], l \leq pos_i \leq r$$

对于一次询问，如果我们已知  $pos_i$ ，那么我们可以处理出其前缀最大值  $max_i$  和最小值  $min_i$ ，之后我们只需二分出最大的  $x$  满足  $l \leq min_x$  且  $max_x \leq r$  即可，此时的  $x$  即为这个区间的 mex 值。

同时可以发现，后续的修改操作实际上为交换  $pos_x$  与  $pos_{x+1}$ ，这个操作只会修改  $max_x$  与  $min_x$ ，可通过  $max_{x-1}$  与  $min_{x-1}$  递推修改。此方法时间复杂度为  $\mathcal{O}(N + K \log N)$ 。

## N. 结束乐队 - 线段树解法

线段树。

由排列的特殊性质我们可以发现，对于一个排列  $\{p_n\}$  而言，这个排列的一个子段  $p[l \dots r]$  的 mex 其实就是  $\min\{p_1, \dots, p_{l-1}, p_{r+1}, \dots, p_n\}$  ( $l$  取 1 或  $r$  取  $n$  时需要特判)。

这样的话，对于一次询问，我们需要查询  $\min\{p_1, \dots, p_{l-1}\}$  与  $\min\{p_{r+1}, \dots, p_n\}$  的最小值，可以使用线段树维护区间最小值来完成。

修改操作记录  $pos_i$  为元素  $i$  在排列中出现的位置，每次修改交换  $pos_i$  和  $pos_{i+1}$ ，并对线段树上这两个位置进行单点修改。

此方法时间复杂度为  $\mathcal{O}(N + K \log N)$ 。

# L. 罗斯飞鸽

**Idea:** ShwStone

**Data:** ShwStone

**Verify:** hwpvector, PandaGhost

# L. 罗斯飞鸽

DP，最长不下降子序列。

考虑将题目的限制形式化：

钦定我们要得分的得分点的序列为  $\{a_i\}$ ，那么如果我们把这个序列按照判定点的时间先后顺序来排序后 ( $t_{a_i} < t_{a_j}$ )，那么应当满足：

$$\begin{aligned}\forall i < j, |x_{a_i} - x_{a_j}| &\leq v(t_{a_j} - t_{a_i}) \\ \Leftrightarrow -v(t_{a_j} - t_{a_i}) &\leq x_{a_i} - x_{a_j} \leq v(t_{a_j} - t_{a_i}) \\ \Leftrightarrow x_{a_i} + v \cdot t_{a_i} &\leq x_{a_j} + v \cdot t_{a_j} \wedge v \cdot t_{a_i} - x_{a_i} \leq v \cdot t_{a_j} - x_{a_j}\end{aligned}$$

于是我们得到了一个三维偏序的做法，可以使用树套树或者 CDQ 分治解决该问题，时间复杂度为  $\mathcal{O}(N \log^2 N)$ 。注意常数可能可以通过此题。

## L. 罗斯飞鸽

我们记  $\alpha_i = x_i + v \cdot t_i, \beta_i = v \cdot t_i - x_i,$

那么不难发现一个蕴涵:  $\alpha_i \leq \alpha_j \wedge \beta_i \leq \beta_j \Rightarrow t_i \leq t_j.$

这意味着在前面提到的三个维度的时间维度是无意义的, 所以问题退化成了一个二维偏序问题。

---

所以可以以  $\alpha_i$  来排序, 然后在  $\beta_i$  上跑最长不下降子序列, 时间复杂度为  $\mathcal{O}(N \log N)$ 。

# H. 9-Nine

**Idea:** wyhao

**Data:** wyhao

**Verify:** uuku, HCCH

## H. 9-Nine - 图论解法

图论，BFS 最短路。

对于每个可能的矩阵  $A$  和可能的矩阵  $B$  所构成的状态  $(A, B)$ ，我们建立一个节点，那么最多建立了  $2^9 \cdot 2^9 = 2^{18}$  个节点。

之后每个节点分别向其经过一次 BL, BR, AL, AR, C1, C2, C3 操作后得到的节点连一条边权为 1 的边，之后在这张图上跑一次单元最短路即可。由于边权均为 1，所以可以使用 BFS 算法。这样我们得到了从该点到全 0 和全 1 矩阵代表的结点的最少操作次数。

这种做法的时间复杂度为  $\mathcal{O}(4^{N^2} \cdot 7)$ ，其中  $N$  表示方格的边长。

## H. 9-Nine - 构造解法

构造。

实际上，观察可知，矩阵  $A$  和  $B$  可以代表两个三阶魔方的上下两个面。你的目标是复原这两个面。

容易证明（考虑每一次操作过程中，这些位置上的数构成的可重集合不变），若存在解，则：

- 中心块 ( $A_{2,2}, B_{2,2}$ ) 中有 1 个 1 和 1 个 0；
- 边块 ( $A_{1,2}, A_{2,1}, A_{2,3}, A_{3,2}, B_{1,2}, B_{2,1}, B_{2,3}, B_{3,2}$ ) 中有 4 个 1 和 4 个 0；
- 角块 ( $A_{1,1}, A_{1,3}, A_{3,1}, A_{3,3}, B_{1,1}, B_{1,3}, B_{3,1}, B_{3,3}$ ) 中有 4 个 1 和 4 个 0。

## H. 9-Nine - 构造解法

依次考虑将中心块、边块、角块移动到正确的位置上。

- 中心块复原：若中心块颜色不对，则可以操作 C2，使得中心块处于正确的位置。本步至多操作 1 次。
- 边块复原：若存在  $A$  的边块为 1，则此时一定存在一个  $B$  的边块为 0。将这两个边块，通过至多  $2 \times 2 = 4$  次的旋转操作移动到  $A_{2,1}$  和  $B_{2,1}$  位置，再进行 C1 操作，即可将这两个边块移动到正确的位置上。本步至多操作  $4 \times (4 + 1) = 20$  步。
- 角块复原：可以发现样例已经给出了一个角块  $A_{1,1}$  和  $B_{1,1}$  交换的构造方式（即 AL, C1, AL, C1, AR, C1）。同样的，若存在  $A$  的角块为 1，则此时一定存在一个  $B$  的角块为 0。将这两个角块，通过至多  $2 \times 2 = 4$  次的旋转操作移动到  $A_{1,1}$  和  $B_{1,1}$  位置，再进行之前所述操作。本步至多操作  $4 \times (4 + 6) = 40$  步。

总计  $1 + 20 + 40 = 61 \leq 81$  步，可以通过本题。

## H. 9-Nine - 加强版解法

考虑题意所述的加强版中  $9 \times 9$  的矩阵，保证有解，你需要在 999 次操作复原。

考虑如何在不改变其他块的方式下，交换  $A_{i,j}$  和  $B_{i,j}$ 。构造方式如下：

- 若  $i \neq j$ ，则可以进行 11 次操作交换：AL, Cj, Ci, AL, Ci, AR, Cj, AL, Cj, AR, Cj；
- 若  $i = j$ ，则可以进行 6 次操作交换：AL, Ci, AL, Ci, AR, Ci。

通过至多 2 次的旋转操作，可以将我们想要交换的块移动到对应的位置上。即每次交换最多需要的操作次数为  $11 + 2 = 13$ 。

一共需要最多的操作次数为  $13 \times (9 \times 9) = 1053$ 。若实现较为精细，注意特判中心块  $A_{5,5}, B_{5,5}$  的交换，并对两个斜对角中的元素  $A_{i,i}, A_{i,10-i}, B_{i,i}, B_{i,10-i}$  特判，使用 6 次交换的操作，则可以在 999 次操作内完成。

## H. 9-Nine - 加强版解法

优化：注意到，在最开始的时候，我们可以通过至多 9 次列交换，把每一列  $A$  和  $B$  矩阵上至少 9 个块移动到正确的位置上。

这样操作后，最多只有 81 个块未被移动到正确的位置上，每次交换使 2 个块移动到正确的位置上（实际上，由于奇偶性，最多只有 80 个块）。

对所有未被归位的块，使用之前所述的构造方式，至多  $9 + 13 \times \frac{80}{2} = 529$  次操作。

# I. 奥日

**Idea:** hwpvector

**Data:** hwpvector

**Verify:** Meguhine, Yeyin\_0

# I. 奥日

图论，树的直径。思维。构造。

- 一条简单路径对重复度  $s$  的贡献为 0；
- 树上的一个大小为  $m$  的环对重复度  $s$  的贡献为  $m$ 。

根据离散数学知识可知，所有的路径都可以被拆分成若干个环和一条简单路径的复合。

又根据欧拉环游序的知识可知，树上的任意一个联通块，假设其大小为  $x$ ，对于联通块内的任意起点，都可以通过一个大小为  $2x-1$  的环来完全遍历。

所以最终的答案一定是树的直径与直径上的若干个除了根节点以外不包含直径的点的子树所复合而成，这些子树的大小之和为  $k$ ，起点与终点则是直径的两个端点。

# I. 奥日 - 证明

关于一些记号：

- 记有根树上的一个非根节点  $u$  的父节点为  $fa_u$ ;
- 记树上的路径  $l$  是简单的，当且仅当路径上的重复结点个数  $s = 0$ ;
- 记有根树上一个结点  $u$  的深度为  $p$ ，表示  $u$  到根节点的简单路径的结点个数（根节点深度为 1）;
- 记无根树的直径长度为  $p$ ，表示树上任意两结点之间最长的简单路径的结点个数。

# I. 奥日 - 证明

## 引理 1:

对于一个  $n$  个结点的有根树  $T = (V, E)$ , 记根结点为  $v$ 。对于一条路径  $l = (u_1, u_2, \dots, u_m)$ , 若满足  $u_1 = u_m = v \wedge |V| = n$ , 则  $m \geq 2n - 1$ 。

## 证明:

考虑有根树上的每一条边  $(w, fa_w)$  s.t.  $w \neq v$ 。对于任意一条满足题意的路径  $l$ , 必须满足:

存在一个  $i (1 \leq i \leq m - 1)$ , 满足  $u_i = fa_w \wedge u_{i+1} = w$ , 否则无法访问结点  $w$  (由于  $u_1 = v$ ), 即  $|V| < n$

存在一个  $i (1 \leq i \leq m - 1)$ , 满足  $u_i = w \wedge u_{i+1} = fa_w$ , 否则在存在  $u'_i = fa_w \wedge u'_{i+1} = w$  后,  $u_m \neq v$

综上, 每条边至少在  $l$  中被访问两次。树  $T$  的边数为  $n - 1$ , 故  $l$  的边数至少为  $2(n - 1)$ , 点数  $m$  比边数多 1, 即  $m \geq 2(n - 1) + 1 = 2n - 1$ 。

# I. 奥日 - 证明

## 引理 2:

对于一个  $n$  个结点的有根树  $T = (V, E)$ , 记根结点为  $v$ , 记树上深度最大的一个结点的深度为  $p$ 。对于一条路径  $l = (u_1, u_2, \dots, u_m)$ , 若满足  $u_1 = v \wedge |V| = n$ , 则  $m \geq 2n - p$ 。

## 证明:

考虑反证法。若存在一条路径  $l$ , 有  $m < 2n - p$ , 则:

记  $l' = (u_m, fa_{u_m}, \dots, v)$  表示从  $u_m$  到  $v$  的一条简单路径。则  $l'$  的点数  $x$  有  $x \leq p$ 。

构造路径  $L = (u_1, u_2, \dots, u_m, fa_{u_m}, \dots, v)$ , 则  $L$  的长度  $y$  有  $y = m + x - 1 < 2n - 1$ , 同时路径  $L$  满足引理 1 中的条件, 应有  $y \geq 2n - 1$ 。矛盾, 得证。

# I. 奥日 - 证明

## 引理 3:

对于一个  $n$  个结点的无根树  $T = (V, E)$ , 记直径长度为  $p$ 。对于一条路径  $l = (u_1, u_2, \dots, u_m)$ , 若满足  $|V| = n$ , 则  $m \geq 2n - p$ , 即重点数量  $s = m - |V| \geq n - p$ 。

## 证明:

枚举  $u_1$ , 枚举集合为  $V$ 。若  $u_1 = v$ , 则考虑将  $T$  以  $v$  为根, 构造一个有根树。应用引理 2, 记该有根树最深结点深度为  $p'$ , 有  $m \geq 2n - p'$ , 同时  $p' \leq p$ 。故  $m \geq 2n - p$ 。

# I. 奥日 - 证明

## 引理 4:

对于一个  $n$  个结点的无根树  $T = (V, E)$ , 记树的直径长度为  $p$ 。任意路径  $l = (u_1, u_2, \dots, u_m)$ , 要求重点数量  $s \leq k$ , 则  $|V| \leq \min\{p + k, n\}$ 。

## 证明:

对于任意路径  $l$ , 考虑导出子图  $T' = T[V']$ 。  $T'$  仍然是树。

对  $T'$  和路径  $l$  应用引理 3, 记树  $T'$  的直径长度为  $p'$ , 有  $s \geq |V'| - p'$ , 同时  $p' \leq p$ 。故有  $s \geq |V'| - p$ 。结合  $s \leq k$ , 有  $|V'| \leq p + k$ 。同时,  $|V'|$  有上限  $n$ 。故  $|V'| \leq \min\{p + k, n\}$ 。

# I. 奥日 - 证明

对于任意树  $T = (V, E)$ , 下给出一种构造方式, 使得  $|V'|$  取到上界  $\min\{p + k, n\}$ 。

找到任一直径  $P = (u_1, u_2, \dots, u_p)$ ,  $p$  为直径长度。记直径点集  $V_p = \{u_1, u_2, \dots, u_p\}$ 。在  $V$  中选择  $r = \min\{k, n - p\}$  个结点, 构成结点集  $V_r = \{v_1, v_2, \dots, v_r\}$ , 满足:

- 对于任意  $1 \leq i \leq p, 1 \leq j \leq r$ , 有  $u_i \neq v_j$ ;
- 导出子图  $T' = T[V_p \cup V_r]$  连通;

可以通过对  $u_1, u_2, \dots, u_p$  进行 DFS (初始时标记直径结点已访问), 来选取结点集  $V_r$ 。

# I. 奥日 - 证明

构造初始路径  $l = (u_1, u_2, \dots, u_p)$ 。进行如下操作  $r$  次：

- 当前路径为  $l = (w_1, w_2, \dots, w_m)$ ；
- 根据  $T'$  的连通性，一定存在一个未被插入的  $v_i$ ，满足  $\exists j, (w_j, v_i) \in E$ ；
- 插入  $v_i$ ，此时  $l = (w_1, w_2, \dots, w_j, v_i, w_j, w_{j+1}, \dots, w_m)$ ，并将  $v_i$  标记为已插入；

每一次进行插入后，路径  $l$  的重复点数量  $s$  增加 1，本质不同结点数  $|V|$  增加 1。完成操作后， $s = \min\{k, n-p\}$  且  $|V| = \min\{p+k, n\}$ 。

很容易将上述操作优化到线性时间复杂度。

## J. 棱堡

**Idea:** Rhodoks

**Data:** ykliu

**Verify:** HCCH, Yeyin\_0

## J. 棱堡

计算几何，凸包。

**结论：**

一个简单非退化多边形是棱堡当且仅当该多边形的任一条边都不在多边形的点集的凸包上。

于是这题仅需求出多边形点集的凸包，然后查看是否存在一个边的两个端点都在凸包上，若是，则不是棱堡，否则是棱堡。

经典的求凸包的算法是  $\mathcal{O}(N \log N)$  的。

## J. 棱堡 - 证明

若多边形存在边在凸包上, 假设这个边是  $PQ$ , 那么多边形的所有点都在  $PQ$  的一侧, 这意味着在  $PQ$  的这一侧的邻域  $\bar{U}_{PQ}$  都在多边形内, 那么对于  $PQ$  上任意一个非端点的点, 它与多边形上任意点的连线要么经过  $PQ$ , 要么一定会经过  $\bar{U}_{PQ}$ , 所以这个边上的除了端点的点都是火力盲区, 因而这个多边形不是棱堡。

对于多边形的一个边, 它两侧的邻域  $\bar{U}_{PQ} - in, \bar{U}_{PQ} - out$  一定恰好有一侧在多边形内, 另一侧在多边形外, 规定多边形外的一侧为外侧。若多边形不是棱堡, 那么存在无限个火力盲区, 故存在多边形的一个边  $PQ$  满足边上存在非端点的火力盲区  $X$ , 则多边形在边  $PQ$  的外侧一定没有点。否则取之一点  $A$  那么线段  $AX$  与多边形的边交点集合 (不包含  $X$ ) 中一定存在距离  $X$  最近的点  $R$ , 则  $R$  能攻击到  $X$ , 那么  $X$  不是火力盲区。则多边形所有点都在  $PQ$  的一侧, 所以  $PQ$  在凸包上。

# G. Primal Core Optimization: Attribute Balance

**Idea:** TAIYANGFENG

**Data:** TAIYANGFENG, HCCH

**Verify:** Rhodoks, MCPlayer542

## G. Primal Core Optimization: Attribute Balance

分析，三分。

形式化的，定义两个点  $X_1 = (x_1, y_1, z_1), X_2 = (x_2, y_2, z_2)$  的距离：

$$\text{dist}(X_1, X_2) :=$$

$$\max(x_1 - x_2, y_1 - y_2, z_1 - z_2, 0) - \min(x_1 - x_2, y_1 - y_2, z_1 - z_2, 0)$$

那么题目所求即为在  $\mathbb{Z}^3$  找到一个点  $S$  使得对于给定点列  $\{X_n\}$  ,  $\sum_{i=1}^n \text{dist}(X_i, S)$  的值最小。

## G. Primal Core Optimization: Attribute Balance

首先思考一个简单的问题，如果已知了  $S$  点的  $x, y$  坐标，那么如何选择  $z$  坐标呢？

实际上对于每个点  $X_i$ ，将  $\text{dist}(X_i, S)$  看作一个关于  $S$  的  $z$  坐标的函数  $D_i(z)$ ，那么这个函数是连续的，下凸的，并且其一定可以被写为如下形式：

$$D_i(z) = \begin{cases} (l - z) + c, & z \leq l \\ c, & l \leq z \leq r \\ (z - r) + c, & r \leq z \end{cases}$$

因此每个  $D_i(z)$  对于  $z$  的二阶差分仅有两个位置有值，于是我们可以对于所有函数标记其二阶差分的位置，再对于每个位置来计算  $D_i(z)$  的总和，这么做的时间复杂度为  $\mathcal{O}(N)$ 。

## G. Primal Core Optimization: Attribute Balance

基于上一个问题，如果仅知道  $S$  点的  $x$  坐标，那么如何确定  $y, z$  坐标呢？

### 结论 1

令  $F_{x_0}(y) = \min_z \{ \sum_{i=1}^n \text{dist}(X_i, (x_0, y, z)) \}$ ，那么  $F_{x_0}(y)$  是关于  $y$  的下凸函数。

我们所求即为  $\min_y \{ F_{x_0}(y) \}$ ，下凸函数求最小值的问题可以用二分或三分算法来解决，时间复杂度  $\mathcal{O}(N \log N)$ 。

## G. Primal Core Optimization: Attribute Balance

回到原始的问题上，如何求  $S$  点的  $x, y, z$  坐标？

### 结论 2

令  $G(x) = \min_{y,z} \{ \sum_{i=1}^n \text{dist}(X_i, (x, y, z)) \}$ ，那么  $G(x) = \min_y \{ F_x(y) \}$ ，并且  $G(x)$  是关于  $x$  的下凸函数。

所以我们可以二分或三分  $x$ ，假设当前二分或三分的  $x$  值为  $x_0$ ，那么嵌套地二分或三分来计算  $\min_y \{ F_{x_0}(y) \}$ ，从而完成对  $G(x_0)$  的计算，进而继续二分或三分。

本题嵌套地进行了两轮二分，时间复杂度为  $\mathcal{O}(N \log^2 N)$ ，本题略微卡常，选手要注意自己的实现方式。

以下仅证明 结论 2，其余结论可由相似思路证明出来。

## 引理

若函数  $f: \mathbb{R}^3 \rightarrow \mathbb{R}$  满足：“对于任意的三维直线参数方程  $\vec{s}(t) = (x_0 + a \cdot t, y_0 + b \cdot t, z_0 + c \cdot t)$ ，令  $g(t) = f(\vec{s}(t))$ ，则  $g(t)$  是下凸函数”（下称条件 A），那么关于  $x$  的函数  $F(x) = \min_{y,z} \{f(x, y, z)\}$  是下凸的。

假如  $F(x)$  不是下凸的，那么存在  $x_1 < x_2 < x_3$  使得  $F(x_2) > \frac{F(x_1) - F(x_3)}{x_1 - x_3} \cdot (x_2 - x_1) + F(x_1)$ ，假设  $f(x_1, y_1, z_1) = F(x_1)$ ， $f(x_3, y_3, z_3) = F(x_3)$ ，那么取同时经过  $(x_1, y_1, z_1)$ ， $(x_3, y_3, z_3)$  两点的直线  $L_1$ ，假设  $L_1$  与平面  $x = x_2$  交于点  $(x_2, y_2, z_2)$ ，那么有  $F(x_2) \leq f(x_2, y_2, z_2)$ ，于是，由  $L_1$  的参数方程生成的函数  $g(t)$  必定不是下凸的。

## G. Primal Core Optimization: Attribute Balance - 证明

设  $D_i(x, y, z) = \text{dist}(X_i, (x, y, z))$ , 显然  $D_i(x, y, z)$  满足条件 A, 由于下凸函数之和仍然是下凸函数, 所以  $f(x, y, z) = \sum_{i=1}^n \text{dist}(X_i, (x, y, z)) = \sum_{i=1}^n D_i(x, y, z)$  满足条件 A, 故  $G(x) = \min_{y,z} \{f(x, y, z)\}$  是关于  $x$  的下凸函数。

# M. 泰拉构史

**Idea:** lrefrain

**Data:** xhnehc

**Verify:** hwpvector



她真好看

# M. 泰拉构史

DP。

由于每个元素都是唯一的, 所以规定  $pos_{\{a_n\}}(x)$  表示  $x$  在序列  $\{a_n\}$  中出现的位置, 之后令  $P_{\{a_n\}}(x) = \{a_i | i < pos_{\{a_n\}}(x) \wedge |a_i - x| > 1\}$

于是注意到两点性质:

- 无论如何操作, 对于任意  $x$  而言  $P_{\{a_n\}}(x)$  不变
- 任意一个元素在初始序列中的位置与最终序列的位置的差的绝对值不超过 2

于是可以得到一个结论:

## 结论

序列  $\{b_n\}$  能通过序列  $\{a_n\}$  经过若干次操作得到当且仅当对于任意  $x$ ,  $P_{\{a_n\}}(x) = P_{\{b_n\}}(x)$ 。

## M. 泰拉构史

于是我们得到了一个可以用来判定一个序列  $\{b_n\}$  是否能从初始序列经过若干次操作后得到的条件，之后考虑按照以下策略来对满足条件的序列进行计数：

一开始的时候序列  $\{b_n\}$  是空的，我们从前往后向序列中填数，每填完一个数字  $x$  后检验  $P_{\{b_n\}}(x)$  是否等于  $P_{\{a_n\}}(x)$ 。

这意味着还需要记录已经填了哪些数，然而由于第二点性质，在填第  $i$  位的时候只有可能是原序列中的  $a_{i-2}, a_{i-1}, a_i, a_{i+1}, a_{i+2}$  中的一个。并且，反之，假如已经填完了前  $i-1$  位，并且这  $i-1$  个数均满足条件，那么对于即将填入第  $i$  位的  $x$ ，假如  $\text{pos}_{\{a_n\}}(y) < i-2$ ，那么  $y$  一定已经被填过了，这意味我们不需要记录下全部填好的数，而只记录原序列中  $a_{i-2}, a_{i-1}, a_i, a_{i+1}, a_{i+2}$  这 5 个数是否被填写过即可。

形式化的, DP 状态及转移如下:

$DP[i][s]$  ( $s$  为长度为 5 的 0/1 串) 表示从前往后在  $\{b_n\}$  中填数, 已经填完了前  $i$  个位置上的数, 并且这  $i$  个位置上的数均满足  $P_{\{b_n\}} = P_{\{a_n\}}$  的条件, 并且  $a_{i-2}, a_{i-1}, a_i, a_{i+1}, a_{i+2}$  这 5 个数有哪些已经填了, 有哪些没有填 (用  $s$  状压来表示) 的方案数。

转移只需要考虑当前的位置填哪个数, 并且检验是否符合条件, 然后再根据当前位置填的数对  $s$  位移操作即可。

时间复杂度为  $\mathcal{O}(32 \cdot 5 \times N) \sim \mathcal{O}(N)$ 。

当然, 这个状态还有优化的空间, 在此不一一列出。

## M. 泰拉构史 - 证明

充分性显然，因为无论怎么操作  $P_{\{a_n\}}(x)$  均不变。

必要性：

以下给出一种由  $\{b_n\}$  还原回  $\{a_n\}$  的策略，

设  $a_1$  在  $\{b_n\}$  出现的位置为  $x$ ，即  $b_x = a_1$ ，那么显然  $P_{\{b_n\}}(a_1) = P_{\{a_n\}}(a_1) = \emptyset$ ，即在  $\{b_n\}$  中所有  $b_1, b_2 \dots b_{x-1}$  均可以与  $b_x$  交换，因此可以经过  $x-1$  轮操作把  $b_x$  挪到  $b_1$ ，之后把  $\{b_n\}$  继续如法炮制的把  $a_2$  放到  $b_2$  的位置上，经过至多  $n-1$  轮后，得到  $b_i = a_i$  即  $\{b_n\} = \{a_n\}$ 。

## M. 泰拉构史 - 验题人做法

我们称位置  $i$  和  $i+1$  处于同一段，当且仅当  $\{a_n\}$  经过一系列操作得到  $\{b_n\}$  之后，可以交换  $b_i$  和  $b_{i+1}$ （即  $|b_i - b_{i+1}| = 1$ ）。

若位置  $l$  和  $l+1$ ， $l+1$  和  $l+2$ ， $\dots$ ， $r-1$  和  $r$  处在同一段，则我们称区间  $[l, r]$  处在同一段。

若区间  $[l, r]$  处在同一段，且  $l-1$  和  $l$  不处在同一段， $r$  和  $r+1$  不处在同一段，则我们称区间  $[l, r]$  为极大的同一段。

我们发现，对于一个  $\{a_n\}$ ，其所有极大的同一段  $[l_i, r_i]$  其交永远为空，其并为  $[1, n]$ 。即极大的同一段，将原序列位置  $[1, n]$ ，划分成了不重叠的若干段。

我们对所有极大的连续段计数，最终答案为这些连续段计数的积。

## M. 泰拉构史 - 验题人做法

考虑这样一个性质：对一个处于同一段  $[l, r]$ ，其对应的子序列  $c_l, c_{l+1}, \dots, c_r$ ，一定能通过交换操作，形成一个连续递增的序列  $x, x+1, x+2, \dots, x+r-l+1$  或是连续递减的序列  $x, x-1, x-2, \dots, x-(r-l+1)$ 。使用归纳法，容易证明。

考虑每一个长度为  $m$  的递增或是递减序列对答案的贡献，记为  $f(m)$ 。考虑是否将最后一个数与前一个数交换，容易得到递推式  $f(m) = f(m-1) + f(m-2)$  ( $f(1) = 1, f(2) = 2$ )。

## M. 泰拉构史 - 验题人做法

接下来，我们只需要找到所有极大的同一段。根据上面的性质，将其转换为连续递增序列或是连续递减序列，反过来考虑我们需要判断哪些情况。于是，只需考虑如下情况中的位置  $i$  和位置  $i+1$  处在同一段，即是充要条件：

- $|a_i - a_{i+1}| = 1$ ;
- $|a_{i-1} - a_i| = 1$  且  $|a_{i-1} - a_{i+1}| = 1$ ;
- $|a_{i+1} - a_{i+2}| = 1$  且  $|a_i - a_{i+2}| = 1$ ;
- $|a_{i-1} - a_i| = 1$  且  $|a_{i+1} - a_{i+2}| = 1$  且  $|a_{i-1} - a_{i+2}| = 1$ 。

时间复杂度为  $\mathcal{O}(4 \times N) \sim \mathcal{O}(N)$ 。其中，4 为上述分类的四种情况。

## K. 我永远喜欢希儿 · 芙乐艾

**Idea:** Luan-233

**Data:** Luan-233

**Verify:** minstdfx

## K. 我永远喜欢希儿 · 芙乐艾

分块，离线，LCA。

无良出题人卡空间，所以本题实现的空间复杂度均为  $\mathcal{O}(N)$ 。

首先分析换根操作，

不难发现树根的位置其并不影响路径，而只影响我们求值的点的范围。

如果钦定 1 号节点为真正的树根，那么对于询问  $(2 \ x)$ ，若当前树根是  $x$  的祖先，那么就相当于直接询问  $x$  的子树的和；若当前的树根是  $x$ ，那么就相当于求所有节点的和；若  $x$  是当前树根的祖先，那么必然存在  $x$  的一儿子  $y$  使得当前树根在  $y$  的子树内，那么这次询问的答案为全体节点的和减去  $y$  的子树和。

所以我们可以把换根操作去掉，只关心如何在有根树（根为一号节点）内维护子树和。

## K. 我永远喜欢希儿 · 芙乐艾

对于修改一条路径的操作, 可以考虑将其放在树上差分里处理, 这样修改路径  $(x, y)$  只会更改  $x, y, LCA(x, y), fa_{LCA(x, y)}$  四个节点的差分值。

对于子树和, 假如树的差分数组为  $d[]$ , 则节点  $x$  的子树和为:

$$\begin{aligned} & \sum_{u \in Tree[x]} \sum_{v \in Tree[u]} d[v] \\ &= \sum_{v \in Tree[x]} d[v] \cdot (\sum_{u \in Tree[x] \wedge v \in Tree[u]} 1) \\ &= \sum_{v \in Tree[x]} d[v] \cdot (dep[v] - dep[x] + 1) \\ &= (\sum_{v \in Tree[x]} dep[v] \cdot d[v]) - (dep[x] - 1) \cdot \sum_{v \in Tree[x]} d[v] \end{aligned}$$

所以只需要维护  $dep[] \cdot d[]$  和  $d[]$  这两个元素的子树和即可。

由于对于空间的限制, LCA 需要用树链剖分来实现。

## K. 我永远喜欢希儿·芙乐艾

现在将操作序列  $\{A_n\}$  分块，块长为  $\sqrt{N_1}$ ，那么每次操作  $(l \ r)$  就可以被切分成至多根号个整块操作和至多根号个散块操作，而操作  $(l \ r)$  对于某次询问的贡献即为这次操作的整块操作和散块操作的贡献之和。

对于散块操作，注意到可以通过树的  $DFN$  序列将树的节点拍成一个序列，而一个节点的子树即为序列中的一个连续的子段，那么散块操作就是单点修改区间求和，由于有总共  $\mathcal{O}(M\sqrt{N_1})$  次单点修改而仅有  $\mathcal{O}(M)$  次询问，

所以需要实现  $\mathcal{O}(1)$  修改， $\mathcal{O}(\sqrt{N_1})$  询问，这用分块是很容易解决的。

## K. 我永远喜欢希儿 · 芙乐艾

对于整块操作，由于空间限制，需要将其离线。

具体地，枚举每个块，之后通过树上子树和来统计这个块对于每个节点  $i$  的贡献是多少，记作  $c[i]$ 。

接着，从前往后遍历整个操作序列，并实时记录这个块被整块操作了几次，假设被操作了  $t$  次，那么对于一次询问  $x$  的子树的和，令  $ans += t \cdot c[x]$  即可。

由于计算一个整块的时间复杂度为  $\mathcal{O}(M + N_2)$ ，而总共有  $\sqrt{N_1}$  个整块，故这一部分是时间复杂度为  $\mathcal{O}((M + N_2)\sqrt{N_1})$ 。

- 感谢各位选手参加 XJTUPC 2025。