思维型题目选讲

北京大学 郭羽冲

2024.1.19

- OI 中有许多题目仅考察基础知识点，但思维难度较高。近年来国内外各大 OI 赛事赛题均有重点考察思维型题目的趋势。

- OI 中有许多题目仅考察基础知识点，但思维难度较高。近年来国内外各大 OI 赛事赛题均有重点考察思维型题目的趋势。
- 本次交流我将分享若干道我认为比较偏向思维型的例题，以及一些常用的思维方式。

AGC068E Sort and Match

题目描述

对于一个长度为 m 的序列 x，令 y 为将 x 排序后的结果。定义 $f(x) = \prod_{i=1}^{m} A_{x_i, y_i}$。

给定 n, m, A。对于每个 $k = 1 \dots n$ 求所有长度为 m，值域为 $1 \sim n$，且 $x_1 = k$ 的整数序列 x 的权值之和。答案对 998244353 取模。

$1 \le n, m \le 50$。

AGC068E Sort and Match

解法

- 积和式？

AGC068E Sort and Match

解法

- 积和式？
- 暂不考虑 $x_1 = k$ 的限制。对于每个 i，将二元组 (x_i, y_i) 看作有向边 $y_i \to x_i$。

AGC068E Sort and Match

解法

- 积和式？
- 暂不考虑 $x_1 = k$ 的限制。对于每个 i，将二元组 (x_i, y_i) 看作有向边 $y_i \to x_i$。
- 显然为欧拉图，且每个点出边有顺序，与 BEST 定理有相似之处，因此考虑转化为有向图游走。

AGC068E Sort and Match

解法

- 积和式？
- 暂不考虑 $x_1 = k$ 的限制。对于每个 i，将二元组 (x_i, y_i) 看作有向边 $y_i \to x_i$。
- 显然为欧拉图，且每个点出边有顺序，与 BEST 定理有相似之处，因此考虑转化为有向图游走。
- 令当前所在点为 u，初始 $u = 1$。考虑如下过程：

AGC068E Sort and Match

解法

- 积和式？
- 暂不考虑 $x_1 = k$ 的限制。对于每个 i，将二元组 (x_i, y_i) 看作有向边 $y_i \to x_i$。
- 显然为欧拉图，且每个点出边有顺序，与 BEST 定理有相似之处，因此考虑转化为有向图游走。
- 令当前所在点为 u，初始 $u = 1$。考虑如下过程：
- 若 u 的所有出边均已被访问：若 $u = n$ 则终止过程，否则移动到 $u + 1$。
- 否则找到 u 的编号最小的未被访问的出边 $u \to v$，移动到 v。

AGC068E Sort and Match

解法

- 积和式？
- 暂不考虑 $x_1 = k$ 的限制。对于每个 i，将二元组 (x_i, y_i) 看作有向边 $y_i \to x_i$。
- 显然为欧拉图，且每个点出边有顺序，与 BEST 定理有相似之处，因此考虑转化为有向图游走。
- 令当前所在点为 u，初始 $u = 1$。考虑如下过程：
- 若 u 的所有出边均已被访问：若 $u = n$ 则终止过程，否则移动到 $u + 1$。
- 否则找到 u 的编号最小的未被访问的出边 $u \to v$，移动到 v。
- 因为每个点入度出度相同，所以最终所有边都会被访问。而访问方案与序列 x 一一对应。

AGC068E Sort and Match

解法

- 对于游走的方案设计 dp 即可。具体地，令 $dp_{u,x,c}$ 表示当前所在点为 u，当前回路的起点为 x，共经过 c 条边时的方案总权值和。

AGC068E Sort and Match

解法

- 对于游走的方案设计 dp 即可。具体地，令 $dp_{u,x,c}$ 表示当前所在点为 u，当前回路的起点为 x，共经过 c 条边时的方案总权值和。
- 转移时选择一条出边 $u \to v$ 并移动到 v。若 $u = x$ 可能出现 u 无出度的情况，此时有第二种转移，移动到 $u+1$。时间复杂度 $O(n^4)$。

AGC068E Sort and Match

解法

- 对于游走的方案设计 dp 即可。具体地，令 $dp_{u,x,c}$ 表示当前所在点为 u，当前回路的起点为 x，共经过 c 条边时的方案总权值和。
- 转移时选择一条出边 $u \to v$ 并移动到 v。若 $u = x$ 可能出现 u 无出度的情况，此时有第二种转移，移动到 $u+1$。时间复杂度 $O(n^4)$。
- $x_1 = k$ 的限制相当于要求编号最小的出边到达点为 k。将过程倒置，最后一步特殊转移即可。时间复杂度依然为 $O(n^4)$。

AGC055F Creative Splitting

题目描述

定义一个长度为 nm 的序列为好的当且仅当可以将它分为 n 个长度为 m 的子序列，使得每个子序列中第 i 个元素 $\leq i$。
给定 n, m，对于每一对 x, y 求 $a_x = y$ 的好序列个数。答案对质数 P 取模。
$1 \leq n, m \leq 20$。

AGC055F Creative Splitting

解法

- 可以贪心判断，过程可被抽象为：维护 $b_1 \dots b_n$，初始 $b_i = m$。从后往前依次考虑每个 a_i，每次找到最小的 $b_j \geq a_i$，并将 b_j 减少 1。

AGC055F Creative Splitting

解法

- 可以贪心判断，过程可被抽象为：维护 $b_1 \ldots b_n$，初始 $b_i = m$。从后往前依次考虑每个 a_i，每次找到最小的 $b_j \geq a_i$，并将 b_j 减少 1。
- 若时刻保持 $b_1 \ldots b_n$ 有序，可以看作：每次选择一个 i，将 b_i 减少 1，方案数为 $b_i - b_{i-1}$（操作前）。

解法

- 可以贪心判断，过程可被抽象为：维护 $b_1 \dots b_n$，初始 $b_i = m$。从后往前依次考虑每个 a_i，每次找到最小的 $b_j \geq a_i$，并将 b_j 减少 1。
- 若时刻保持 $b_1 \dots b_n$ 有序，可以看作：每次选择一个 i，将 b_i 减少 1，方案数为 $b_i - b_{i-1}$（操作前）。
- 令 $c_i = b_i - b_{i-1}$，则过程变为：每次选择一个 i，将 c_i 减少 1，将 c_{i+1} 增加 1，方案数为 c_i（操作前）。

AGC055F Creative Splitting

解法

- 可以贪心判断，过程可被抽象为：维护 $b_1 \dots b_n$，初始 $b_i = m$。从后往前依次考虑每个 a_i，每次找到最小的 $b_j \ge a_i$，并将 b_j 减少 1。
- 若时刻保持 $b_1 \dots b_n$ 有序，可以看作：每次选择一个 i，将 b_i 减少 1，方案数为 $b_i - b_{i-1}$（操作前）。
- 令 $c_i = b_i - b_{i-1}$，则过程变为：每次选择一个 i，将 c_i 减少 1，将 c_{i+1} 增加 1，方案数为 c_i（操作前）。
- 初始 $c_1 = m$，最终 $c_{n+1} = m$。相当于有 m 个不同的球，初始均在 1，每次选择一个球往右移动一格，最终将所有球全部移动到 $n + 1$。

AGC055F Creative Splitting

解法

- 枚举第 $nm-x+1$ 步所操作的位置 t，则我们要求 $\sum_{i=1}^{t-1} c_i < y \le \sum_{i=1}^{t} c_i$。

AGC055F Creative Splitting

解法

- 枚举第 $nm-x+1$ 步所操作的位置 t，则我们要求 $\sum_{i=1}^{t-1} c_i < y \le \sum_{i=1}^{t} c_i$。
- 可以将它容斥为 $[y \le \sum_{i=1}^{t} c_i] - [y \le \sum_{i=1}^{t-1} c_i]$ 分别计算。

AGC055F Creative Splitting

解法

- 枚举第 $nm-x+1$ 步所操作的位置 t，则我们要求 $\sum_{i=1}^{t-1} c_i < y \le \sum_{i=1}^{t} c_i$。
- 可以将它容斥为 $[y \le \sum_{i=1}^{t} c_i] - [y \le \sum_{i=1}^{t-1} c_i]$ 分别计算。
- 将操作过程分为两个阶段：第一阶段为前 $nm-x$ 步，第二阶段为后 x 步。
- 令 $f_{t,i,j,k}$ 表示考虑前 i 个球，共进行 im 步操作，当前有 j 个球在第一阶段结束后位置 $\le t$，共有 k 步操作处于第一阶段。

AGC055F Creative Splitting

解法

- 枚举第 $nm-x+1$ 步所操作的位置 t，则我们要求 $\sum_{i=1}^{t-1} c_i < y \le \sum_{i=1}^{t} c_i$。
- 可以将它容斥为 $[y \le \sum_{i=1}^{t} c_i] - [y \le \sum_{i=1}^{t-1} c_i]$ 分别计算。
- 将操作过程分为两个阶段：第一阶段为前 $nm-x$ 步，第二阶段为后 x 步。
- 令 $f_{t,i,j,k}$ 表示考虑前 i 个球，共进行 im 步操作，当前有 j 个球在第一阶段结束后位置 $\le t$，共有 k 步操作处于第一阶段。
- 转移时只需要枚举当前球在第一阶段中的操作步数即可。根据上述分析，利用 f 计算出最终答案是容易的。

AGC055F Creative Splitting

解法

- 枚举第 $nm-x+1$ 步所操作的位置 t，则我们要求 $\sum_{i=1}^{t-1} c_i < y \le \sum_{i=1}^{t} c_i$。
- 可以将它容斥为 $[y \le \sum_{i=1}^{t} c_i] - [y \le \sum_{i=1}^{t-1} c_i]$ 分别计算。
- 将操作过程分为两个阶段：第一阶段为前 $nm-x$ 步，第二阶段为后 x 步。
- 令 $f_{t,i,j,k}$ 表示考虑前 i 个球，共进行 im 步操作，当前有 j 个球在第一阶段结束后位置 $\le t$，共有 k 步操作处于第一阶段。
- 转移时只需要枚举当前球在第一阶段中的操作步数即可。根据上述分析，利用 f 计算出最终答案是容易的。
- 时间复杂度 $O(n^3m^3)$。

CF1806F2 GCD Master

题目描述

给定一个长度为 n 的序列，进行 m 次操作，每次选择两个数合并为它们的 gcd，最大化剩余数之和。

$1 \le n \le 10^6, 1 \le a_i \le 9 \times 10^{18}$。

CF1806F2 GCD Master

解法

- 相当于将所有数划分为 $n-m$ 个集合，最大化每个集合 $\gcd$ 之和。

CF1806F2 GCD Master

解法

- 相当于将所有数划分为 $n-m$ 个集合，最大化每个集合 gcd 之和。
- 对于可重集 S，若 $\min(S) < \max(S)$，则有 $\max(S) \geq 2\gcd(S)$。

CF1806F2 GCD Master

解法

- 相当于将所有数划分为 $n-m$ 个集合，最大化每个集合 gcd 之和。
- 对于可重集 S，若 $\min(S) < \max(S)$，则有 $\max(S) \geq 2\gcd(S)$。
- 若方案中有两个这样的集合 S_1, S_2，则可调整为 $\max(\max(S_1), \max(S_2))$ 单独一组，剩余部分一组，得到不劣的新方案。因此方案中至多有一个可重集 S 满足 $\min(S) < \max(S)$。

CF1806F2 GCD Master

解法

- 相当于将所有数划分为 $n-m$ 个集合，最大化每个集合 gcd 之和。
- 对于可重集 S，若 $\min(S) < \max(S)$，则有 $\max(S) \geq 2\gcd(S)$。
- 若方案中有两个这样的集合 S_1, S_2，则可调整为 $\max(\max(S_1), \max(S_2))$ 单独一组，剩余部分一组，得到不劣的新方案。因此方案中至多有一个可重集 S 满足 $\min(S) < \max(S)$。
- 将 “合并不等元素” 与 “合并相等元素” 看作两种操作。枚举 t 表示前者进行的次数。后者产生的代价即为每种元素去掉一个后的前 $m-t$ 小之和。

CF1806F2 GCD Master

解法

- 此时问题转化为，在元素互不相等的集合中选出 $t+1$ 个元素，最小化 $\text{sum}(S) - \gcd(S)$。

CF1806F2 GCD Master

解法

- 此时问题转化为，在元素互不相等的集合中选出 $t+1$ 个元素，最小化 $\mathsf{sum}(S) - \gcd(S)$。
- 将所有元素从小到大排序得到 $a_1 \dots a_n$，若 $i > j > k, a_i \in S, a_j \in S, a_k \notin S$，则 $\gcd(S) \le a_i - a_j$，因此将 a_i 换成 a_k 不劣。

CF1806F2 GCD Master

解法

- 此时问题转化为，在元素互不相等的集合中选出 $t+1$ 个元素，最小化 $\mathsf{sum}(S)-\gcd(S)$。
- 将所有元素从小到大排序得到 $a_1\dots a_n$，若 $i>j>k, a_i\in S, a_j\in S, a_k\notin S$，则 $\gcd(S)\le a_i-a_j$，因此将 a_i 换成 a_k 不劣。
- 因此 S 一定形如 $a_1\dots a_t, a_p$，其中 $t+1\le p\le n$。

CF1806F2 GCD Master

解法

- 此时问题转化为，在元素互不相等的集合中选出 $t+1$ 个元素，最小化 $\text{sum}(S) - \gcd(S)$。
- 将所有元素从小到大排序得到 $a_1 \dots a_n$，若 $i > j > k, a_i \in S, a_j \in S, a_k \notin S$，则 $\gcd(S) \le a_i - a_j$，因此将 a_i 换成 a_k 不劣。
- 因此 S 一定形如 $a_1 \dots a_t, a_p$，其中 $t+1 \le p \le n$。
- 枚举 t，若 $\gcd(a_1 \dots a_t) = \gcd(a_1 \dots a_{t+1})$ 则 $p = t+1$ 最优。仅出现 $O(\log V)$ 个不满足上述条件的 t，暴力计算对应 p 即可。

CF1806F2 GCD Master

解法

- 此时问题转化为，在元素互不相等的集合中选出 $t+1$ 个元素，最小化 $\mathsf{sum}(S) - \gcd(S)$。
- 将所有元素从小到大排序得到 $a_1 \dots a_n$，若 $i > j > k, a_i \in S, a_j \in S, a_k \notin S$，则 $\gcd(S) \le a_i - a_j$，因此将 a_i 换成 a_k 不劣。
- 因此 S 一定形如 $a_1 \dots a_t, a_p$，其中 $t+1 \le p \le n$。
- 枚举 t，若 $\gcd(a_1 \dots a_t) = \gcd(a_1 \dots a_{t+1})$ 则 $p = t+1$ 最优。仅出现 $O(\log V)$ 个不满足上述条件的 t，暴力计算对应 p 即可。
- 时间复杂度 $O(n \log V)$ 或 $O(n \log^2 V)$。

CEOI2014 The Wall

题目描述

有一个 $n \times m$ 的矩阵，其中有若干个格子是关键的。相邻的格点之间可以建造城墙，并产生对应代价。对城墙有如下要求：

- 城墙形成一条包含格点 $(0, 0)$ 的回路（若一条边被经过多次，需要算多次代价）。
- 从网格外部任意一点走到任意一个关键格子，都必须穿过城墙。

求建造城墙的最小代价。

$1 \le n, m \le 400$。

CEOI2014 The Wall

解法

- 乍一看像是最小割，但由于回路的限制不好处理。

CEOI2014 The Wall

解法

- 乍一看像是最小割，但由于回路的限制不好处理。
- 关键结论：考虑 $(0,0)$ 到每个关键格子左上角的最短路径，则它一定被回路包含在内。

CEOI2014 The Wall

解法

- 乍一看像是最小割，但由于回路的限制不好处理。
- 关键结论：考虑 $(0,0)$ 到每个关键格子左上角的最短路径，则它一定被回路包含在内。
- 证明：若否，则存在最短路径的一段区间在回路外部，而最短路径的子区间依然为最短路径，可调整。

解法

- 乍一看像是最小割，但由于回路的限制不好处理。
- 关键结论：考虑 $(0, 0)$ 到每个关键格子左上角的最短路径，则它一定被回路包含在内。
- 证明：若否，则存在最短路径的一段区间在回路外部，而最短路径的子区间依然为最短路径，可调整。
- 任意一条不“穿过”这些路径的回路均满足条件。

解法

- 乍一看像是最小割，但由于回路的限制不好处理。
- 关键结论：考虑 $(0,0)$ 到每个关键格子左上角的最短路径，则它一定被回路包含在内。
- 证明：若否，则存在最短路径的一段区间在回路外部，而最短路径的子区间依然为最短路径，可调整。
- 任意一条不“穿过”这些路径的回路均满足条件。
- 格点拆成四个点，横向边拆成上下两条，纵向边拆成左右两条，可严格刻画“穿过”。找最小环即可。时间复杂度 $O(nm \log nm)$。

IOI2019 折线

题目描述

给定二维平面上 n 个点，求一条从 $(0,0)$ 出发，仅包含不超过 $n+3$ 条线段，且每条线段均平行于坐标轴的折线使得所有点都在折线上。

$1 \le n \le 10^5$。

注：原题为提交答案，但存在直接构造的方法。

IOI2019 折线

解法

- 直观想法是：一圈一圈往里绕。如果有点在角上则段数可能达到 $2n$。

解法

- 直观想法是：一圈一圈往里绕。如果有点在角上则段数可能达到 $2n$。
- 如果 $x_i = y_i = i$，则应当从左下开始依次经过每个点。

IOI2019 折线

解法

- 直观想法是：一圈一圈往里绕。如果有点在角上则段数可能达到 $2n$。
- 如果 $x_i = y_i = i$，则应当从左下开始依次经过每个点。
- 将两种想法结合。维护三个序列 a, b, c，依次对应绕圈，左上-右下，左下-右上。

IOI2019 折线

解法

- 直观想法是：一圈一圈往里绕。如果有点在角上则段数可能达到 $2n$。
- 如果 $x_i = y_i = i$，则应当从左下开始依次经过每个点。
- 将两种想法结合。维护三个序列 a, b, c，依次对应绕圈，左上-右下，左下-右上。
- 先尝试绕圈，四步为一轮，依次选择当前 x 最小，y 最小，x 最大，y 最大的点。

IOI2019 折线

解法

- 直观想法是：一圈一圈往里绕。如果有点在角上则段数可能达到 $2n$。
- 如果 $x_i = y_i = i$，则应当从左下开始依次经过每个点。
- 将两种想法结合。维护三个序列 a, b, c，依次对应绕圈，左上-右下，左下-右上。
- 先尝试绕圈，四步为一轮，依次选择当前 x 最小，y 最小，x 最大，y 最大的点。
- 若某次选择点之后发现无法与上一个点衔接，则说明上一个点在角上，可将其放入 b 或 c 中。

IOI2019 折线

解法

- 直观想法是：一圈一圈往里绕。如果有点在角上则段数可能达到 $2n$。
- 如果 $x_i = y_i = i$，则应当从左下开始依次经过每个点。
- 将两种想法结合。维护三个序列 a, b, c，依次对应绕圈，左上-右下，左下-右上。
- 先尝试绕圈，四步为一轮，依次选择当前 x 最小，y 最小，x 最大，y 最大的点。
- 若某次选择点之后发现无法与上一个点衔接，则说明上一个点在角上，可将其放入 b 或 c 中。
- 依次访问三个序列即可，访问序列过程中每段线段覆盖一个点，相邻两个序列之间需要额外的一条线段用于衔接。共 $n+3$ 条线段。

IOI2024 象形文字序列

题目描述

给定两个字符串 a, b，令 a, b 的所有公共子序列组成集合 S。定义 $c \in S$ 为 a, b 的“最全公共子序列”当且仅当 S 中所有元素均为 c 的子序列。
求 a, b 的最全公共子序列或报告无解。
$1 \le n, m \le 10^5$。

IOI2024 象形文字序列

解法

- 先考虑保证有解的情况。

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x, y)$ 次。将出现较少的一侧标记为关键位。

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x, y)$ 次。将出现较少的一侧标记为关键位。
- 相当于要将 a, b 中的相等元素匹配，使得每个关键位置都在匹配中，且匹配不交。

IOI2024 象形文字序列

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x, y)$ 次。将出现较少的一侧标记为关键位。
- 相当于要将 a, b 中的相等元素匹配，使得每个关键位置都在匹配中，且匹配不交。
- 考虑 a, b 中的第一个关键位置 a_p, b_q，有如下情况：

IOI2024 象形文字序列

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x, y)$ 次。将出现较少的一侧标记为关键位。
- 相当于要将 a, b 中的相等元素匹配，使得每个关键位置都在匹配中，且匹配不交。
- 考虑 a, b 中的第一个关键位置 a_p, b_q，有如下情况：
- 若 $a_p = b_q$ 则直接匹配。

IOI2024 象形文字序列

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x,y)$ 次。将出现较少的一侧标记为关键位。
- 相当于要将 a,b 中的相等元素匹配，使得每个关键位置都在匹配中，且匹配不交。
- 考虑 a,b 中的第一个关键位置 a_p,b_q，有如下情况：
- 若 $a_p=b_q$ 则直接匹配。
- a_p 对应的匹配点必在 b_q 之前。若 $b_1\dots b_{q-1}$ 中出现过 a_p，则 a_p 可匹配。

IOI2024 象形文字序列

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x,y)$ 次。将出现较少的一侧标记为关键位。
- 相当于要将 a,b 中的相等元素匹配，使得每个关键位置都在匹配中，且匹配不交。
- 考虑 a,b 中的第一个关键位置 a_p,b_q，有如下情况：
- 若 $a_p=b_q$ 则直接匹配。
- a_p 对应的匹配点必在 b_q 之前。若 $b_1\dots b_{q-1}$ 中出现过 a_p，则 a_p 可匹配。
- 同理，若 $a_1\dots a_{p-1}$ 中出现过 b_q，则 b_q 可匹配。

解法

- 先考虑保证有解的情况。
- 对于一种字符，若它在 a 中出现 x 次，在 b 中出现 y 次，则它在 c 中应出现 $\min(x, y)$ 次。将出现较少的一侧标记为关键位。
- 相当于要将 a, b 中的相等元素匹配，使得每个关键位置都在匹配中，且匹配不交。
- 考虑 a, b 中的第一个关键位置 a_p, b_q，有如下情况：
- 若 $a_p = b_q$ 则直接匹配。
- a_p 对应的匹配点必在 b_q 之前。若 $b_1 \dots b_{q-1}$ 中出现过 a_p，则 a_p 可匹配。
- 同理，若 $a_1 \dots a_{p-1}$ 中出现过 b_q，则 b_q 可匹配。
- 若两者均可匹配，则需要进一步分析决策。

解法

- 在先前定义可匹配的基础上，重新定义：若 $a_{p+1}\dots a_n$ 中 b_q 的出现次数不少于 $b_q\dots b_m$ 中 b_q 的出现次数，则 a_p 可匹配。

IOI2024 象形文字序列

解法

- 在先前定义可匹配的基础上，重新定义：若 $a_{p+1}\dots a_n$ 中 b_q 的出现次数不少于 $b_q\dots b_m$ 中 b_q 的出现次数，则 a_p 可匹配。
- 同理可重新定义 b_q 可匹配。

IOI2024 象形文字序列

解法

- 在先前定义可匹配的基础上，重新定义：若 $a_{p+1}\dots a_n$ 中 b_q 的出现次数不少于 $b_q\dots b_m$ 中 b_q 的出现次数，则 a_p 可匹配。
- 同理可重新定义 b_q 可匹配。
- 若在新定义下两者均可匹配，则考虑 $a_q b_p\dots b_p$ 和 $b_p a_q\dots a_q$，它们均为 a,b 的公共子序列，但不可能同时是 c 的子序列。

IOI2024 象形文字序列

解法

- 在先前定义可匹配的基础上，重新定义：若 $a_{p+1}\dots a_n$ 中 b_q 的出现次数不少于 $b_q\dots b_m$ 中 b_q 的出现次数，则 a_p 可匹配。
- 同理可重新定义 b_q 可匹配。
- 若在新定义下两者均可匹配，则考虑 $a_q b_p \dots b_p$ 和 $b_p a_q \dots a_q$，它们均为 a, b 的公共子序列，但不可能同时是 c 的子序列。
- 因此只有唯一决策，之后可以递归到子问题。时间复杂度 $O(n)$ 或 $O(n\log n)$。

IOI2024 象形文字序列

解法

- 在先前定义可匹配的基础上，重新定义：若 $a_{p+1}\dots a_n$ 中 b_q 的出现次数不少于 $b_q\dots b_m$ 中 b_q 的出现次数，则 a_p 可匹配。
- 同理可重新定义 b_q 可匹配。
- 若在新定义下两者均可匹配，则考虑 $a_q b_p \dots b_p$ 和 $b_p a_q \dots a_q$，它们均为 a, b 的公共子序列，但不可能同时是 c 的子序列。
- 因此只有唯一决策，之后可以递归到子问题。时间复杂度 $O(n)$ 或 $O(n\log n)$。
- 接下来需要判断上述方法求出的 c 是否为最全公共子序列。

解法

- 尝试构造一个字符串 $c' \in S$ 且不为 c 的子序列。

解法

- 尝试构造一个字符串 $c' \in S$ 且不为 c 的子序列。
- 依次加入 c' 的每一个字符，维护当前在 a, b 中的匹配位置 a_p, b_q，以及在 c 中的匹配对应 a, b 中的位置 $a_{p'}, b_{q'}$。

IOI2024 象形文字序列

解法

- 尝试构造一个字符串 $c' \in S$ 且不为 c 的子序列。
- 依次加入 c' 的每一个字符，维护当前在 a, b 中的匹配位置 a_p, b_q，以及在 c 中的匹配对应 a, b 中的位置 $a_{p'}, b_{q'}$。
- 显然有 $p \le p', q \le q'$。若某个时刻有 $p < p', q < q'$ 则一定可以构造出 c'。

解法

- 尝试构造一个字符串 $c' \in S$ 且不为 c 的子序列。
- 依次加入 c' 的每一个字符，维护当前在 a, b 中的匹配位置 a_p, b_q，以及在 c 中的匹配对应 a, b 中的位置 $a_{p'}, b_{q'}$。
- 显然有 $p \le p', q \le q'$。若某个时刻有 $p < p', q < q'$ 则一定可以构造出 c'。
- 因此我们只需要考虑 $p = p'$ 或 $q = q'$ 的情况。令 f_p 表示 $p = p'$ 时 q 的最小值，g_q 表示 $q = q'$ 时 p 的最小值。

解法

- 尝试构造一个字符串 $c' \in S$ 且不为 c 的子序列。
- 依次加入 c' 的每一个字符，维护当前在 a, b 中的匹配位置 a_p, b_q，以及在 c 中的匹配对应 a, b 中的位置 $a_{p'}, b_{q'}$。
- 显然有 $p \le p', q \le q'$。若某个时刻有 $p < p', q < q'$ 则一定可以构造出 c'。
- 因此我们只需要考虑 $p = p'$ 或 $q = q'$ 的情况。令 f_p 表示 $p = p'$ 时 q 的最小值，g_q 表示 $q = q'$ 时 p 的最小值。
- 经过简单的分类讨论可以发现，只需要分别做 f, g 内部的转移（即不考虑 f, g 相互转移），最后判断是否能通过增加一步操作达到不合法情况即可。

IOI2024 象形文字序列

解法

- 尝试构造一个字符串 $c' \in S$ 且不为 c 的子序列。
- 依次加入 c' 的每一个字符，维护当前在 a, b 中的匹配位置 a_p, b_q，以及在 c 中的匹配对应 a, b 中的位置 $a_{p'}, b_{q'}$。
- 显然有 $p \le p', q \le q'$。若某个时刻有 $p < p', q < q'$ 则一定可以构造出 c'。
- 因此我们只需要考虑 $p = p'$ 或 $q = q'$ 的情况。令 f_p 表示 $p = p'$ 时 q 的最小值，g_q 表示 $q = q'$ 时 p 的最小值。
- 经过简单的分类讨论可以发现，只需要分别做 f, g 内部的转移（即不考虑 f, g 相互转移），最后判断是否能通过增加一步操作达到不合法情况即可。
- 可以用单调栈维护后缀最小值加速转移。时间复杂度 $O(n \log n)$。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

题目描述

给定两棵 n 个点的树 T_1, T_2，每次选择两条边 $(u_1, v_1) \in T_1, (u_2, v_2) \in T_2$，在 T_1 中删除 (u_1, v_1) 加入 (u_2, v_2)，在 T_2 中删除 (u_2, v_2) 加入 (u_1, v_1)。
要求每次操作之后 T_1, T_2 依然为树。构造一种操作方案使得最终 T_1, T_2 中每条边均恰好被选择一次。
$1 \le n \le 10^3$。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

解法

- 选择在两棵树中度数之和最小的点 u。所有点度数和为 $4(n-1)$，因此 $\deg_1(u)+\deg_2(u)\le 3$。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

解法

- 选择在两棵树中度数之和最小的点 u。所有点度数和为 $4(n-1)$，因此 $\deg_1(u)+\deg_2(u)\leq 3$。
- 若 u 在两棵树中均为叶子，则交换对应的边并删除 u 即可递归子问题。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

解法

- 选择在两棵树中度数之和最小的点 u。所有点度数和为 $4(n-1)$，因此 $\deg_1(u)+\deg_2(u)\le 3$。
- 若 u 在两棵树中均为叶子，则交换对应的边并删除 u 即可递归子问题。
- 否则不妨令 $\deg_1(u)=1, \deg_2(u)=2, (u,a)\in T_1, (u,b), (u,c)\in T_2$，且 T_2 中 (u,b) 在 $u\to a$ 的简单路径上。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

解法

- 选择在两棵树中度数之和最小的点 u。所有点度数和为 $4(n-1)$，因此 $\deg_1(u)+\deg_2(u)\le 3$。
- 若 u 在两棵树中均为叶子，则交换对应的边并删除 u 即可递归子问题。
- 否则不妨令 $\deg_1(u)=1, \deg_2(u)=2, (u,a)\in T_1, (u,b),(u,c)\in T_2$，且 T_2 中 (u,b) 在 $u\to a$ 的简单路径上。
- 在 T_1, T_2 中将 u 删除，并向 T_2 中加入边 (b,c)，递归子问题。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

解法

- 选择在两棵树中度数之和最小的点 u。所有点度数和为 $4(n-1)$，因此 $\deg_1(u)+\deg_2(u)\le 3$。
- 若 u 在两棵树中均为叶子，则交换对应的边并删除 u 即可递归子问题。
- 否则不妨令 $\deg_1(u)=1,\deg_2(u)=2,(u,a)\in T_1,(u,b),(u,c)\in T_2$，且 T_2 中 (u,b) 在 $u\to a$ 的简单路径上。
- 在 T_1,T_2 中将 u 删除，并向 T_2 中加入边 (b,c)，递归子问题。
- 在子问题的解中，必有一步为交换 $(b,c),(d,e)$，将这一步扩展为：交换 $(u,a),(u,b)$，交换 $(u,c),(d,e)$ 即可得到原问题的解。

The 3rd Universal Cup. Stage 21: Ōokayama G. Diverge and Converge

解法

- 选择在两棵树中度数之和最小的点 u。所有点度数和为 $4(n-1)$，因此 $\deg_1(u)+\deg_2(u)\le 3$。
- 若 u 在两棵树中均为叶子，则交换对应的边并删除 u 即可递归子问题。
- 否则不妨令 $\deg_1(u)=1, \deg_2(u)=2, (u,a)\in T_1, (u,b),(u,c)\in T_2$，且 T_2 中 (u,b) 在 $u\to a$ 的简单路径上。
- 在 T_1, T_2 中将 u 删除，并向 T_2 中加入边 (b,c)，递归子问题。
- 在子问题的解中，必有一步为交换 $(b,c),(d,e)$，将这一步扩展为：交换 $(u,a),(u,b)$，交换 $(u,c),(d,e)$ 即可得到原问题的解。
- 暴力实现上述过程，时间复杂度 $O(n^2)$。

The 3rd Universal Cup. Stage 20: Kunming I. Items

题目描述

有 n 种物品，第 i 种物品重量为 a_i，有无穷多个。其中 $0 \le a_i \le n$。判断是否能选出 n 个物品使得重量之和为 m。

$1 \le n \le 10^5, 0 \le m \le n^2$。

解法

- 先将每个物品重量减少 $\lfloor m/n \rfloor$。

解法

- 先将每个物品重量减少 $\lfloor m/n \rfloor$。
- 令选出的物品为 $b_1 \dots b_n$，则必存在 b 的一个重排满足所有前缀和均在 $[-n, n]$ 内。

解法

- 先将每个物品重量减少 $\lfloor m/n \rfloor$。
- 令选出的物品为 $b_1 \dots b_n$，则必存在 b 的一个重排满足所有前缀和均在 $[-n, n]$ 内。
- 倍增求多项式幂即可，每一步只需要保留次数为 $[-n, n]$ 的项。总时间复杂度 $O(n \log^2 n)$。

The 3rd Universal Cup. Stage 7: Warsaw C. Price Combo

题目描述

有 n 种物品，第 i 种物品在商店 A,B 中的价格分别为 a_i,b_i。每次可以在一个商店中购买两个物品，花费的代价为本商店中较贵者的价格。求每种物品都至少购买一个的最小总代价。

$1\le n\le 2\times 10^5, 1\le a_i,b_i\le 10^9$。

解法

- 相当于将每个物品放在两个商店之一。一个商店的代价为本商店中的物品代价从大到小排序后奇数位置和。

解法

- 相当于将每个物品放在两个商店之一。一个商店的代价为本商店中的物品代价从大到小排序后奇数位置和。
- 初始将每个物品放在价格较低的商店中，需要选择一些物品换到另一商店中使得总代价最小。

The 3rd Universal Cup. Stage 7: Warsaw C. Price Combo

解法

- 相当于将每个物品放在两个商店之一。一个商店的代价为本商店中的物品代价从大到小排序后奇数位置和。
- 初始将每个物品放在价格较低的商店中，需要选择一些物品换到另一商店中使得总代价最小。
- 若两个被选择的物品原价格为 x_1, x_2，换后价格为 y_1, y_2 且 $\max(x_1, x_2) \le \min(y_1, y_2)$，则每个商店中的价格集合均变劣，因此最优解中不应当出现这种情况。

解法

- 相当于将每个物品放在两个商店之一。一个商店的代价为本商店中的物品代价从大到小排序后奇数位置和。
- 初始将每个物品放在价格较低的商店中，需要选择一些物品换到另一商店中使得总代价最小。
- 若两个被选择的物品原价格为 x_1, x_2，换后价格为 y_1, y_2 且 $\max(x_1, x_2) \le \min(y_1, y_2)$，则每个商店中的价格集合均变劣，因此最优解中不应当出现这种情况。
- 考虑一个被选择的物品，令原价格 x，换后价格为 y，则连一条线段 $x \to y$。上述条件等价于线段两两不交。

The 3rd Universal Cup. Stage 7: Warsaw C. Price Combo

解法

- 将所有 a_i, b_i 放在数轴上，令 $f_{i,j,k}$ 表示考虑后 i 个点，当前两个商店中元素总数奇偶性分别为 j, k 时，最小的代价。

The 3rd Universal Cup. Stage 7: Warsaw C. Price Combo

解法

- 将所有 a_i, b_i 放在数轴上，令 $f_{i,j,k}$ 表示考虑后 i 个点，当前两个商店中元素总数奇偶性分别为 j, k 时，最小的代价。
- 转移时只需要考虑当前元素是否被换到另一商店中，只需支持快速计算一段区间的代价，用线段树维护转移信息即可。时间复杂度 $O(n \log n)$。

题目描述

定义一个 01 串为好的，当且仅当可以通过删除形如 0100 的子串。
给定一个包含 $0, 1, ?$ 的串，求将 $?$ 替换为 0 或 1 后能得到多少种好的串。答案对 998244353 取模。
$1 \le n \le 500$。

解法

- 将 1 看作分隔符，记录一个序列 $a_1 \dots a_k$ 依次表示相邻两个 1 之间 0 的个数。

解法

- 将 1 看作分隔符，记录一个序列 $a_1 \dots a_k$ 依次表示相邻两个 1 之间 0 的个数。
- 操作变为：选择 i 满足 $a_i \geq 1, a_{i+1} \geq 2$，将它们合并为 $a_i + a_{i+1} - 3$。

解法

- 将 1 看作分隔符，记录一个序列 $a_1 \dots a_k$ 依次表示相邻两个 1 之间 0 的个数。
- 操作变为：选择 i 满足 $a_i \geq 1, a_{i+1} \geq 2$，将它们合并为 $a_i + a_{i+1} - 3$。
- 令 $b_i = a_i - 3$，显然要求 $\sum b_i = -3$。
- 操作变为：选择 i 满足 $b_i \geq -2, b_{i+1} \geq -1$，将它们合并为 $b_i + b_{i+1}$。

解法

- 将 1 看作分隔符，记录一个序列 $a_1 \dots a_k$ 依次表示相邻两个 1 之间 0 的个数。
- 操作变为：选择 i 满足 $a_i \geq 1, a_{i+1} \geq 2$，将它们合并为 $a_i + a_{i+1} - 3$。
- 令 $b_i = a_i - 3$，显然要求 $\sum b_i = -3$。
- 操作变为：选择 i 满足 $b_i \geq -2, b_{i+1} \geq -1$，将它们合并为 $b_i + b_{i+1}$。
- 若 $|b| \geq 2$，且存在 $b_i \leq -3$，则这个元素无法再被操作，不合法。

解法

- 将 1 看作分隔符，记录一个序列 $a_1 \dots a_k$ 依次表示相邻两个 1 之间 0 的个数。
- 操作变为：选择 i 满足 $a_i \geq 1, a_{i+1} \geq 2$，将它们合并为 $a_i + a_{i+1} - 3$。
- 令 $b_i = a_i - 3$，显然要求 $\sum b_i = -3$。
- 操作变为：选择 i 满足 $b_i \geq -2, b_{i+1} \geq -1$，将它们合并为 $b_i + b_{i+1}$。
- 若 $|b| \geq 2$，且存在 $b_i \leq -3$，则这个元素无法再被操作，不合法。
- 考虑 b 中最后一个 -2，令其为 b_x。若 $x \geq 2$，则必须有 $y \geq x$ 满足 $\sum\limits_{i=x}^{y} b_i \geq -1$。否则 b_x 永远无法与前面的元素合并。

解法

- 满足上述条件一定合法!

解法

- 满足上述条件一定合法！
- 先用每个 -2 后面对应的区间将它合并为 ≥ -1 的元素，此时序列中每个元素均 ≥ -1。若长度 ≥ 4 则必存在非负数，将它与相邻数合并后可归纳。若长度 ≤ 3 则显然合法。

The 3rd Universal Cup. Stage 13: Sendai N. 0100 Insertion

解法

- 满足上述条件一定合法！
- 先用每个 -2 后面对应的区间将它合并为 ≥ -1 的元素，此时序列中每个元素均 ≥ -1。若长度 ≥ 4 则必存在非负数，将它与相邻数合并后可归纳。若长度 ≤ 3 则显然合法。
- 将 0 看作 $+1$，1 看作 -3，则限制变为：总和为 -3，不存在相邻的 -3，且每个后缀的最大前缀和均 ≥ -1。dp 即可。时间复杂度 $O(n^3)$。

The 2nd Universal Cup Semifinals H. Exchanging Kubic 2

题目描述

数轴上有 n 个人，初始第 i 个人在坐标 a_i 处。每次可以选择一个人 i，其他所有人会向 a_i 移动一格。定义初始状态 $a_1 \dots a_n$ 的权值为，按照某种顺序选择每个人恰好一次后，坐标极差的最小值。

给定 n 个集合 $S_1 \dots S_n$，求所有不降且 $a_i \in S_i$ 的序列权值之和。答案对 998244353 取模。

$1 \le n \le 400, 0 \le S_{i,j} \le 800$。

解法

- 结论：等价于每个人向左右分别至多匹配一条长度为 1 的线段，要求线段处于 $[a_1, a_n]$ 中，求最大匹配数量。权值即为 $a_n - a_1 -$ 匹配数量。

The 2nd Universal Cup Semifinals H. Exchanging Kubic 2

解法

- 结论：等价于每个人向左右分别至多匹配一条长度为 1 的线段，要求线段处于 $[a_1, a_n]$ 中，求最大匹配数量。权值即为 $a_n - a_1 -$ 匹配数量。
- 由于删除过程的特性，需要对匹配方式有所限制。假设第 i 人匹配 l_i, r_i 两条线段，实际上要求 $[l_i, r_i]$ 两两不交或包含。但经过分类讨论可以发现，若出现相交则必可调整。因此可以忽略这个限制。

The 2nd Universal Cup Semifinals H. Exchanging Kubic 2

解法

- 结论：等价于每个人向左右分别至多匹配一条长度为 1 的线段，要求线段处于 $[a_1, a_n]$ 中，求最大匹配数量。权值即为 $a_n - a_1 -$ 匹配数量。
- 由于删除过程的特性，需要对匹配方式有所限制。假设第 i 人匹配 l_i, r_i 两条线段，实际上要求 $[l_i, r_i]$ 两两不交或包含。但经过分类讨论可以发现，若出现相交则必可调整。因此可以忽略这个限制。
- 向左匹配最大数量为 $(n-1) - \max((i-1)-(a_i - a_1))$，向右匹配最大数量为 $(n-1) - \max((n-i)-(a_n - a_i))$，最大匹配数量即为两者之和对 $a_n - a_1$ 取 $\min$。

The 2nd Universal Cup Semifinals H. Exchanging Kubic 2

解法

- 结论：等价于每个人向左右分别至多匹配一条长度为 1 的线段，要求线段处于 $[a_1, a_n]$ 中，求最大匹配数量。权值即为 $a_n - a_1 -$ 匹配数量。
- 由于删除过程的特性，需要对匹配方式有所限制。假设第 i 人匹配 l_i, r_i 两条线段，实际上要求 $[l_i, r_i]$ 两两不交或包含。但经过分类讨论可以发现，若出现相交则必可调整。因此可以忽略这个限制。
- 向左匹配最大数量为 $(n-1) - \max((i-1) - (a_i - a_1))$，向右匹配最大数量为 $(n-1) - \max((n-i) - (a_n - a_i))$，最大匹配数量即为两者之和对 $a_n - a_1$ 取 min。
- 化简可得权值为 $\max(\max(a_i - i) - \min(a_i - i) - n + 1, 0)$。

解法

- 将 $\min, \max$ 两项拆开算贡献。

解法

- 将 $\min, \max$ 两项拆开算贡献。
- 对于 min，枚举 $x = \min(a_i - i)$，并进行 dp，过程中记录当前是否存在 i 使得 $a_i - i = x$ 以及是否存在 i 使得 $a_i - i \geq x + n$。即可得知 x 的贡献次数。同理可计算 max 的贡献。时间复杂度 $O(nm^2)$。

题目描述

有一张 n 个点的无向图，初始 $i, i+1$ 之间有边，形成一条链。进行 m 次操作，每次加入一条边，在每次操作后求出有多少种选择两条边 e_1, e_2 的方式使得将 e_1, e_2 删除后图不连通。

$1 \leq n, m \leq 2.5 \times 10^5$。

解法

- 先选出一棵生成树，由图论中的经典结论，应当选择一组“非树边集合”在异或意义下线性相关的边。本题中可以选取初始链作为生成树，两个集合线性相关当且仅当它们相同或其中一者为空集。

解法

- 先选出一棵生成树，由图论中的经典结论，应当选择一组“非树边集合”在异或意义下线性相关的边。本题中可以选取初始链作为生成树，两个集合线性相关当且仅当它们相同或其中一者为空集。
- 令经过树边 $(i, i+1)$ 的非树边集合为 S_i，将 S_i 相同的边放入同一等价类中。相当于要在加边的过程中维护等价类。

解法

- 先选出一棵生成树，由图论中的经典结论，应当选择一组“非树边集合”在异或意义下线性相关的边。本题中可以选取初始链作为生成树，两个集合线性相关当且仅当它们相同或其中一者为空集。
- 令经过树边 $(i, i+1)$ 的非树边集合为 S_i，将 S_i 相同的边放入同一等价类中。相当于要在加边的过程中维护等价类。
- 令 $a_{i,j} = [(i, i+1) \subseteq (u_j, v_j)]$。则 i 与 j 在 t 时刻属于同一等价类当且仅当 $t \le \mathrm{LCP}(a_i, a_j)$。

The 1st Universal Cup. Stage 0: Nanjing L. Proposition Composition

解法

- 先选出一棵生成树，由图论中的经典结论，应当选择一组“非树边集合”在异或意义下线性相关的边。本题中可以选取初始链作为生成树，两个集合线性相关当且仅当它们相同或其中一者为空集。
- 令经过树边 $(i, i+1)$ 的非树边集合为 S_i，将 S_i 相同的边放入同一等价类中。相当于要在加边的过程中维护等价类。
- 令 $a_{i,j} = [(i, i+1) \subseteq (u_j, v_j)]$。则 i 与 j 在 t 时刻属于同一等价类当且仅当 $t \le \mathsf{LCP}(a_i, a_j)$。
- 因此将所有 $a_1 \dots a_{n-1}$ 按照字典序排序得到 $b_1 \dots b_{n-1}$，同时将过程倒置变为合并等价类，在 $\mathsf{LCP}(b_i, b_{i+1})$ 时刻将 $i, i+1$ 合并即可。

解法

- 先选出一棵生成树，由图论中的经典结论，应当选择一组“非树边集合”在异或意义下线性相关的边。本题中可以选取初始链作为生成树，两个集合线性相关当且仅当它们相同或其中一者为空集。
- 令经过树边 $(i, i+1)$ 的非树边集合为 S_i，将 S_i 相同的边放入同一等价类中。相当于要在加边的过程中维护等价类。
- 令 $a_{i,j} = [(i, i+1) \subseteq (u_j, v_j)]$。则 i 与 j 在 t 时刻属于同一等价类当且仅当 $t \leq \mathsf{LCP}(a_i, a_j)$。
- 因此将所有 $a_1 \dots a_{n-1}$ 按照字典序排序得到 $b_1 \dots b_{n-1}$，同时将过程倒置变为合并等价类，在 $\mathsf{LCP}(b_i, b_{i+1})$ 时刻将 $i, i+1$ 合并即可。
- 排序需要支持 $O(n \log n)$ 次比较两个串字典序大小，可以主席树维护哈希做到 $O(\log n)$ 比较。时间复杂度 $O(n \log^2 n)$。

UOJ Round 25 C. 装配序列

题目描述

给定一个 $1 \sim n$ 的排列 a。设 a' 为 a 重复 $+\infty$ 次得到的序列。给再定 m 组询问，每次给定一个 x，求 a' 中长度为 x 的前缀的 ** 最长严格上升子序列 ** 的长度。

$1 \leq n \leq 2 \times 10^5, 1 \leq m \leq 10^6, 1 \leq x \leq 10^{18}$。

UOJ Round 25 C. 装配序列

解法

- 考虑 LIS 的一种贪心求法：维护一个递增序列 b。假设当前考虑的数是 x，找到最小的 y 满足 $b_y \geq x$ 并将 b_y 改为 x。如果不存在这样的 y 那么就在 b 的末尾插入一个 x。

UOJ Round 25 C. 装配序列

解法

- 考虑 LIS 的一种贪心求法：维护一个递增序列 b。假设当前考虑的数是 x，找到最小的 y 满足 $b_y \geq x$ 并将 b_y 改为 x。如果不存在这样的 y 那么就在 b 的末尾插入一个 x。
- 这个算法通常的实现是按照下标从小到大，每次加入一个值。但本题中我们按照值从小到大，每次加入一个下标。

UOJ Round 25 C. 装配序列

解法

- 考虑 LIS 的一种贪心求法：维护一个递增序列 b。假设当前考虑的数是 x，找到最小的 y 满足 $b_y \geq x$ 并将 b_y 改为 x。如果不存在这样的 y 那么就在 b 的末尾插入一个 x。
- 这个算法通常的实现是按照下标从小到大，每次加入一个值。但本题中我们按照值从小到大，每次加入一个下标。
- 考虑维护出这个长度不超过 n 的序列，每次询问的答案即为序列中 $\leq x$ 的数的个数。这是因为这个序列可以看作是在维护 dp 数组的差分。

解法

- 归纳地认为任何一个时刻每种数保留的都是一个前缀。设当前考虑到的值在排列中的下标为 x，下标为 i 的数保留了 c_i 个。那么进行如下过程：

解法

- 归纳地认为任何一个时刻每种数保留的都是一个前缀。设当前考虑到的值在排列中的下标为 x，下标为 i 的数保留了 c_i 个。那么进行如下过程：
- 初始 $c_x = 0$。
- 依次访问 $(x, n]$ 中的每一个 i，如果 $c_x < c_i$，那么交换 c_x 和 c_i。
- 依次访问 $[1, x)$ 中的每一个 i，如果 $c_x < c_i - 1$，那么先交换 c_x 和 c_i，然后 $c_x \leftarrow c_x - 1, c_i \leftarrow c_i + 1$。
- 最后 $c_x \leftarrow c_x + 1$。

UOJ Round 25 C. 装配序列

解法

- 在上面的过程中，每次访问到的 c_i 是严格单调递增的。

UOJ Round 25 C. 装配序列

解法

- 在上面的过程中，每次访问到的 c_i 是严格单调递增的。
- 设一个阈值 B，对于 $\leq B$ 中的每一种数用一个 set 维护它们的出现位置，对于 $> B$ 的数直接维护它们的出现位置。

UOJ Round 25 C. 装配序列

解法

- 在上面的过程中，每次访问到的 c_i 是严格单调递增的。
- 设一个阈值 B，对于 $\le B$ 中的每一种数用一个 set 维护它们的出现位置，对于 $> B$ 的数直接维护它们的出现位置。
- 每次操作先枚举所有 $\le B$ 的数，并在对应 set 中找后继，再暴力枚举所有 $> B$ 的数是并更新。

UOJ Round 25 C. 装配序列

解法

- 在上面的过程中，每次访问到的 c_i 是严格单调递增的。
- 设一个阈值 B，对于 $\leq B$ 中的每一种数用一个 set 维护它们的出现位置，对于 $> B$ 的数直接维护它们的出现位置。
- 每次操作先枚举所有 $\leq B$ 的数，并在对应 set 中找后继，再暴力枚举所有 $> B$ 的数是并更新。
- 取 $B = \sqrt{n}$，前半部分时间复杂度为 $\sum \log_{d_i+1}$，其中 d_i 表示 i 的出现次数，有 $\sum i \times d_i \leq n$。可证明这个值为 $O(\sqrt{n})$。后半部分时间复杂度为 $\frac{n}{B}$，同样为 $O(\sqrt{n})$。总时间复杂度 $O(n\sqrt{n} + m\log n)$。

UOJ NOI Round 7 Day1 C. 璀璨宝石

题目描述

游戏中有 n 张发展卡构成一个牌堆。同时，共有 5 种不同类型的宝石。
每张发展卡上都标记了购买这张发展卡所需的 5 种宝石的数量 $a_{i,0}, b_{i,0}, c_{i,0}, d_{i,0}, e_{i,0}$，以及抵扣数量 $a_{i,1}, b_{i,1}, c_{i,1}, d_{i,1}, e_{i,1}$。
当购买发展卡 i 时，若已购买的发展卡为集合 S，则只需要消耗 $\max\left(a_{i,0} - \sum_{j \in S} a_{j,1}, 0\right)$ 个 A 类宝石。对于其余 4 类宝石同理。
游戏初始时会将牌堆的前两张发展卡放到桌上，每买走一张桌上的发展卡时，就从牌堆顶拿出下一张发展卡放到桌上。
初始你没有宝石。游戏会进行很多轮，每一轮中，你可以进行两种操作之一：

- 选择两种不同的宝石，拿取这两种宝石各一个。
- 买走一张桌上的发展卡。

给定牌堆，求最少进行几轮操作可以把 n 张发展卡全部买下。
$1 \le n \le 50, \sum a_{i,0}, \sum a_{i,1} \le m \le 2 \times 10^3$，$b, c, d, e$ 同理。

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 桌上的两张发展卡编号可以唯一确定已购买发展卡集合。dp 过程可以看作在一个 $O(n^2)$ 大小，每个点出度 ≤ 2 的 DAG 上移动。

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 桌上的两张发展卡编号可以唯一确定已购买发展卡集合。dp 过程可以看作在一个 $O(n^2)$ 大小，每个点出度 ≤ 2 的 DAG 上移动。
- 假设已经确定了每种宝石的总花费数量，则需要的轮数为 $\max(\max, \lceil \mathsf{sum}/2 \rceil)$。

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 桌上的两张发展卡编号可以唯一确定已购买发展卡集合。dp 过程可以看作在一个 $O(n^2)$ 大小，每个点出度 ≤ 2 的 DAG 上移动。
- 假设已经确定了每种宝石的总花费数量，则需要的轮数为 $\max(\max, \lceil \mathsf{sum}/2 \rceil)$。
- 若 $\max \geq \lceil \mathsf{sum}/2 \rceil$。提前钦定 $\max$ 所对应的宝石种类 x，dp 过程中只需记录 x 和非 x 的个数之差。时间复杂度为 $O(n^2 m)$。

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 桌上的两张发展卡编号可以唯一确定已购买发展卡集合。dp 过程可以看作在一个 $O(n^2)$ 大小，每个点出度 ≤ 2 的 DAG 上移动。
- 假设已经确定了每种宝石的总花费数量，则需要的轮数为 $\max(\max, \lceil \text{sum}/2 \rceil)$。
- 若 $\max \ge \lceil \text{sum}/2 \rceil$。提前钦定 $\max$ 所对应的宝石种类 x，dp 过程中只需记录 x 和非 x 的个数之差。时间复杂度为 $O(n^2 m)$。
- 否则我们考虑将过程分为两个阶段：

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 桌上的两张发展卡编号可以唯一确定已购买发展卡集合。dp 过程可以看作在一个 $O(n^2)$ 大小，每个点出度 ≤ 2 的 DAG 上移动。
- 假设已经确定了每种宝石的总花费数量，则需要的轮数为 $\max(\max, \lceil \mathsf{sum}/2 \rceil)$。
- 若 $\max \geq \lceil \mathsf{sum}/2 \rceil$。提前钦定 $\max$ 所对应的宝石种类 x，dp 过程中只需记录 x 和非 x 的个数之差。时间复杂度为 $O(n^2 m)$。
- 否则我们考虑将过程分为两个阶段：
- 每次随便选两个不同的元素匹配，直到某一种数出现至少一半（实际上此时必有 $\max = \lceil \mathsf{sum}/2 \rceil$）。
- 钦定 $\max$ 对应的宝石种类 x，每次匹配一个 x 和一个非 x 的宝石。

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 对于两个阶段分别 dp。第一阶段中匹配完全后只会剩余一种宝石，只需记录种类与个数。第二阶段中只需记录 x 和非 x 的个数之差。

UOJ NOI Round 7 Day1 C. 璀璨宝石

解法

- 对于两个阶段分别 dp。第一阶段中匹配完全后只会剩余一种宝石，只需记录种类与个数。第二阶段中只需记录 x 和非 x 的个数之差。
- 每加入一张发展卡就依次考虑它花费的每一种宝石。若不切换阶段，则转移显然为 $O(1)$。可能在考虑一种宝石的过程切换阶段，这种情况可以用一些技巧（单调队列，前缀和等）将转移复杂度优化至每个状态 $O(1)$。总时间复杂度 $O(n^2m)$。

ROI 2017 Day 2 学习轨迹

题目描述

给定 n, m，两个长度为 n 的序列 a, x，两个长度为 m 的序列 b, y。
保证 a_i 互不相同，b_i 互不相同。但可能有 $a_i = b_j$。
你需要选择 l_1, r_1, l_2, r_2 满足：

- $0 \le l_1 - 1 \le r_1 \le n, 0 \le l_2 - 1 \le r_2 \le m$。
- $\forall l_1 \le i \le r_1, l_2 \le j \le r_2$，有 $a_i \ne b_j$。

求 $\sum\limits_{l_1 \le i \le r_1} x_i + \sum\limits_{l_2 \le i \le r_2} y_i$ 的最大值。
$1 \le n, m \le 5 \times 10^5$。

ROI 2017 Day 2 学习轨迹

解法

- 区间可以为空，因此答案至少为 $s = \max(\sum x_i, \sum y_i)$。

ROI 2017 Day 2 学习轨迹

解法

- 区间可以为空，因此答案至少为 $s=\max(\sum x_i, \sum y_i)$。
- 因此最优解中至少有一个区间和 $\geq \frac{s}{2}$，不妨令 a 中 $[l_1, r_1]$ 区间和 $\geq \frac{s}{2}$。

解法

- 区间可以为空，因此答案至少为 $s = \max(\sum x_i, \sum y_i)$。
- 因此最优解中至少有一个区间和 $\geq \frac{s}{2}$，不妨令 a 中 $[l_1, r_1]$ 区间和 $\geq \frac{s}{2}$。
- 令 p 为 x 中第一个 $\geq \frac{s}{2}$ 的前缀和的下标，则 $[l_1, r_1]$ 一定与 $[p, p+1]$ 有交。

ROI 2017 Day 2 学习轨迹

解法

- 区间可以为空，因此答案至少为 $s = \max(\sum x_i, \sum y_i)$。
- 因此最优解中至少有一个区间和 $\geq \frac{s}{2}$，不妨令 a 中 $[l_1, r_1]$ 区间和 $\geq \frac{s}{2}$。
- 令 p 为 x 中第一个 $\geq \frac{s}{2}$ 的前缀和的下标，则 $[l_1, r_1]$ 一定与 $[p, p+1]$ 有交。
- 对于一个区间 $[l_2, r_2]$，将 a_i 中所有在 $b_{l_2} \dots b_{r_2}$ 中出现过的值标记。并选择相邻两个标记之间的一段区间作为 $[l_1, r_1]$。

解法

- 区间可以为空，因此答案至少为 $s = \max(\sum x_i, \sum y_i)$。
- 因此最优解中至少有一个区间和 $\geq \frac{s}{2}$，不妨令 a 中 $[l_1, r_1]$ 区间和 $\geq \frac{s}{2}$。
- 令 p 为 x 中第一个 $\geq \frac{s}{2}$ 的前缀和的下标，则 $[l_1, r_1]$ 一定与 $[p, p+1]$ 有交。
- 对于一个区间 $[l_2, r_2]$，将 a_i 中所有在 $b_{l_2} \dots b_{r_2}$ 中出现过的值标记。并选择相邻两个标记之间的一段区间作为 $[l_1, r_1]$。
- 又 $[l_1, r_1]$ 必与 $[p, p+1]$ 有交，只需要考虑 $\leq p$ 的部分中最靠右的标记和 $> p$ 的部分中最靠左的标记。

ROI 2017 Day 2 学习轨迹

解法

- 区间可以为空，因此答案至少为 $s = \max(\sum x_i, \sum y_i)$。
- 因此最优解中至少有一个区间和 $\geq \frac{s}{2}$，不妨令 a 中 $[l_1, r_1]$ 区间和 $\geq \frac{s}{2}$。
- 令 p 为 x 中第一个 $\geq \frac{s}{2}$ 的前缀和的下标，则 $[l_1, r_1]$ 一定与 $[p, p+1]$ 有交。
- 对于一个区间 $[l_2, r_2]$，将 a_i 中所有在 $b_{l_2} \dots b_{r_2}$ 中出现过的值标记。并选择相邻两个标记之间的一段区间作为 $[l_1, r_1]$。
- 又 $[l_1, r_1]$ 必与 $[p, p+1]$ 有交，只需要考虑 $\leq p$ 的部分中最靠右的标记和 $> p$ 的部分中最靠左的标记。
- 扫描 r_2，用单调栈和线段树对于每个 l_2 分别维护两侧的标记即可。修改形如后缀取 min，单调栈后转化为后缀赋值。时间复杂度 $O(n \log n)$。

题目描述

有一个长度为 n 的 01 序列 a_0，和一个 n 阶排列 p。
共进行 m 轮操作，第 k 轮中：

- 若 k 为奇数，则对于每个 i，令 $a_{k,i} = f(k, i, a_{k-1,1} \dots a_{k-1,i}, p_1 \dots p_n)$。
- 若 k 为偶数，则对于每个 i，令 $a_{k,i} = f(k, i, a_{k-1,i} \dots a_{k-1,n}, p_1 \dots p_n)$。

你的任务是设计合适的函数 f，使得 $a_{m,i} = a_{p_i}$。
$1 \le n \le 500, m = 3$。

EGOI2024 Garden Decorations

解法

- 过程中不能丢失信息，考虑线性变换。相当于构造 $\mathcal{F}_2$ 下的矩阵 A, B, C 满足：A, C 下三角，B 上三角，$CBA = P$。其中 $P_{i,j} = [j = p_i]$。

解法

- 过程中不能丢失信息，考虑线性变换。相当于构造 $\mathcal{F}_2$ 下的矩阵 A, B, C 满足：A, C 下三角，B 上三角，$CBA = P$。其中 $P_{i,j} = [j = p_i]$。
- $CBA = P \Rightarrow C^{-1}PA^{-1} = B$，而 A^{-1}, C^{-1} 也为下三角。P 左乘 C^{-1} 为行变换，右乘 A^{-1} 为列变换。

解法

- 过程中不能丢失信息，考虑线性变换。相当于构造 $\mathcal{F}_2$ 下的矩阵 A, B, C 满足：A, C 下三角，B 上三角，$CBA = P$。其中 $P_{i,j} = [j = p_i]$。
- $CBA = P \Rightarrow C^{-1}PA^{-1} = B$，而 A^{-1}, C^{-1} 也为下三角。P 左乘 C^{-1} 为行变换，右乘 A^{-1} 为列变换。
- 相当于可以对 P 执行以下操作，要求将其变为上三角矩阵：

EGOI2024 Garden Decorations

解法

- 过程中不能丢失信息，考虑线性变换。相当于构造 $\mathcal{F}_2$ 下的矩阵 A, B, C 满足：A, C 下三角，B 上三角，$CBA = P$。其中 $P_{i,j} = [j = p_i]$。
- $CBA = P \Rightarrow C^{-1}PA^{-1} = B$，而 A^{-1}, C^{-1} 也为下三角。P 左乘 C^{-1} 为行变换，右乘 A^{-1} 为列变换。
- 相当于可以对 P 执行以下操作，要求将其变为上三角矩阵：
- 选择 $i < j$，将第 i 行加到第 j 行上。
- 选择 $i > j$，将第 i 列加到第 j 列上。

EGOI2024 Garden Decorations

解法

- 按照行从小到大依次操作，假设当前对于 $i = 1 \dots k-1$ 均满足 $P_{i,i} = 1$，且 $\forall i > j$，有 $P_{i,j} = 0$。

EGOI2024 Garden Decorations

解法

- 按照行从小到大依次操作，假设当前对于 $i = 1 \ldots k-1$ 均满足 $P_{i,i} = 1$，且 $\forall i > j$，有 $P_{i,j} = 0$。
- 从小到大枚举每个 $i = 1 \ldots k-1$，若 $P_{k,i} \neq 0$，则将第 i 行加到第 k 行上。

EGOI2024 Garden Decorations

解法

- 按照行从小到大依次操作，假设当前对于 $i = 1 \dots k-1$ 均满足 $P_{i,i} = 1$，且 $\forall i > j$，有 $P_{i,j} = 0$。
- 从小到大枚举每个 $i = 1 \dots k-1$，若 $P_{k,i} \neq 0$，则将第 i 行加到第 k 行上。
- 若 $P_{k,k} = 0$，则必有 $i > k$ 满足 $P_{k,i} \neq 0$（否则第 k 行全为 0，不满秩），将第 i 列加到第 k 列上。此时 $i = 1 \dots k$ 均满足归纳条件，继续归纳即可。时间复杂度 $O(n^3)$。

- 谢谢大家！