构造题选做

目录

Preface

本文为对《构造题方法总汇》进行大学习的成果。

大部分内容都是贺的。只有少数混杂了我自己的拙见。

构造一定不要害怕尝试。本文中很多构造方案的每一步都不一定有充足的理由，纯粹只是抱着「**试一试这样构造行不行得通，如果行得通就赢了，行不通就回到这里尝试别的构造方案**」的策略，经过好几次尝试后试出来的成功道路。别忘了，在这个成功的构造方案之外，还有很多失败的构造方案，只是本文不一定会将它们写出来罢了。

勇敢尝试的同时，根据**解题思想**获得启发是很有必要的。做构造题时的解题思想就好比做 OI 题时的 trick。本文会用 思路 tag 标识每道题用到的解题思想。

胡出一道题并不是终点。真正的终点是，**编织出一条尽可能细、切实可行的思考路径，以使得下次遇到相似的题时能做出**。

最后放个

> **Content**
>
> $$A=\begin{bmatrix}不&挑&战\\怕&战&胜\\困&困&困\\难&难&难\end{bmatrix}$$
>
> $$A^T=\begin{bmatrix}不&怕&困&难\\挑&战&困&难\\战&胜&困&难\end{bmatrix}$$
>
> 不要做 A，要做 A^T！如果你是 A，请把自己的心态转置一下！

Definitions

- 「精确值限制」指的是「要求某个数恰好等于什么」的限制，区别于「要求某个数大于等于 / 小于等于什么」的限制。例如，P5419 [CTSC2016] 单调上升序列 中，题面要求的「使得最长的单调上升路径最短」是「最优化条件」，转化后的「使得最长的单调上升路径长度恰好为 $n-1$」是「精确值限制」。

常用的解题思想

> NOTE
>
> 所有解题思想之间大多没有明确的分界，往往结合使用。

a. 等价表述条件和观察模型（4 题）

使用不同的方式和图形去表示问题模型，可能会突出问题的某一部分特征，从而带来新的启发。有一个经典的例子可以说明换一种方式描述会产生多大的区别（一道 dp）：CF1368H1。原问题（最大流模型）和最小割模型都看似无法解决，但如果转变到平面图最短路的角度去分析出路径的简化性质（每行或每列均相同），再回到最小割模型，就可以 dp 了。因此，不要认为等价模型没有意义，否则会错失动机。

Warning

在转述过程中要注意可能会不小心加强模型或丢性质，反而导致无解，这个后面会提到。

等价表述条件和观察模型 这个 tag 包含了以下几层意思：

- 分析操作性质，以等价表述条件、转化条件
- 观察模型，以转化模型，规约到其他问题

CF1680F Lenient Vertex Cover

撰写时间 2025.9.25

标签 思维题 生成树 二分图 黑白染色 枚举特殊对象 等价表述条件和观察模型 将无解和最优化条件转成已知条件 分讨

首先，由「无向图」想到生成树。

思考路径：看到无向图就要想到生成树。

想到生成树后，因为树是二分图，可以想到经典的黑白染色，令黑色点的点集为点覆盖的点集。我们发现，如果是二分图，那么黑白染色直接就是对的；有环的部分下面讨论。

思考路径：「点覆盖」和「树」都可以引导想到黑白染色。（其实可能更直观的思考路径是，题中所给的条件是适配偶环的，而奇环必定会出现一条边两端颜色相同，故由此想到二分图黑白染色并得出「二分图直接黑白染色就是对的」的结论）

有奇环的情况，注意到题目中能容忍至多一条边两端都在点集中。考虑怎么利用这个容忍度，先钦定一条边用掉这个容忍度（即钦定一条边两端都在点集中），然后从图中移除这条边，在剩下的图中考虑。由于容忍度已经被用掉了，剩下的图必须是二分图。对于钦定的这条边，它的两个端点必须都是黑色，且能发现在剩下的图是二分图的时候，必定存在一种方案将其染为黑色。

分析至此，我们总结出在有奇环的情况下，有解的充要条件：**找到一条边，使得移除这条边后剩下的图是二分图**。找到这条边后，获得染色的构造是简单的。

思考路径：题目中能容忍至多一条边两端都在点集中，那么这条边必然是特殊的，不妨先枚举这条边是哪个边，剩下的问题即可转换为没有特殊边的情况。

现在，问题变为了「找到一条边，使得移除这条边后剩下的图是二分图」。

接下来的问题就比较 OI 了。上分讨：

- 当奇环数量恰好为 1 时，将奇环中的非树边作为这条特殊边即可。
- 当奇环数量大于 1 时，显然只能选择树边作为特殊边了（选择非树边只能去掉一个奇环）。经过思考后，我们发现一条树边能作为特殊边当且仅当：
  - 所有的奇环都覆盖了它；
  - 不存在偶环覆盖了它。

  那么树上差分即可。

思考路径：显然应当对树边和非树边进行分讨。总结「一条树边能作为特殊边」的时候要仔细一点、想全一点，我自己思考的时候就漏掉了「不存在偶环覆盖了它」这个条件。

至此，我们成功解决了这道题。注意到如果生成树用的是 DFS 生成树，那么非树边都是返祖边，在树上差分的时候也就不需要求 LCA，写起来会简单一点。

总时间复杂度 $O(n)$。

CF1110E Magic Stones

撰写时间 2025.9.25

标签 思维题 差分 等价表述条件和观察模型

经典 trick，设 $d_i = a_i - a_{i-1}$，发现在 i 处做操作等价于 swap d_i, d_{i+1}。

思考路径：要总结各种经典 trick。如果忘了 trick，考场上遇见这样的操作，要多想想前缀和、差分什么的；且不一定就是一阶前缀和或差分了，也可能是二阶、三阶，等等。

故答案为 YES 当且仅当

- $a_1 = b_1$
- a 的差分数组和 b 的差分数组构成相同

第二个条件，将两者的差分数组排序后判断是否相等即可。

思考路径：总结条件要总结全！我就漏掉了 $a_1 = b_1$ 这个条件。

时间复杂度 $O(n \log n)$。

> **Content**
>
> 两个经典 trick：
>
> - $(a_{i-1}, a_i, a_{i+1}) \to (a_{i-1}, a_{i-1} + a_{i+1} - a_i, a_{i+1})$ 作差分解决。
> - $(a_{i-1}, a_i, a_{i+1}) \to (a_{i-1} + a_i, -a_i, a_{i+1} + a_i)$ 作前缀和解决。

CF1458D Flip and Reverse

撰写时间 2025.9.25 ~ 2025.9.26

标签 思维题 欧拉路径 前缀和 等价表述条件和观察模型

首先，看到 01 序列，还有要求 0 的个数和 1 的个数相等，想到：令 0 的权值为 -1，1 的权值为 $+1$，求出前缀和数组。

思考路径：看到 01 序列，还有要求 0 的个数和 1 的个数相等的限制，就要想到转为 $-1, +1$ 序列前缀和，利用前缀和数组来刻画。

下面是我的想法：

> **Content**

分析操作性质，画出前缀和数组的折线图，我们发现：这个操作相当于：对于 $s_l = s_r$ 的 l, r，将折线图中 $[l, r]$ 这段折线 reverse。

那么我们发现了对于一个 k，序列中所有 $s_i = k$ 的 i 将序列分成若干段，这些 i 夹着的段之间可以任意排列，对于单个段内则可以 reverse。

那么到这里已经可以得出一个递归解法了，当递归求解子区间 $(l, r]$（保证 $s_l = s_r$ 且 $\min_{i=l+1}^{r-1} \geq s_r$）所能产生的字典序最小的 01 序列时，若 $\min_{i=l+1}^{r-1} > s_r$ 则递归到 $(l+1, r-1]$ 的情况，否则那些 $s_i = s_r$ 的 i 将区间分为若干段，递归求解每段的最小字典序，再将每段的结果从小到大排序。

这个做法时间复杂度不知道。但是有很大的启发意义。

思考路径：看到 $-1, +1$ 的前缀和序列，就要想到折线图。

下面是题解做法：

Content

有一张有向图，每一个值 k 都有一个对应的点。对于前缀和数组 s，对于所有 $i \in [1, n]$，从 s_{i-1} 的点向 s_i 的点连一条边。

然后，对于如果某一次操作了区间 $(l, r]$，由于 $s_l = s_r$，故 $(l, r]$ 里的边必然形成**环**。并且可以发现操作的效果相当于把这个环中的所有边反转方向。

然后我们发现，这张图的任意欧拉路径都是可以通过若干次操作获得的。（形式化表述就是说，对于每一条欧拉路径 P，存在一个操作序列，使得：设操作后的 01 序列建出的前缀和数组为 s，则「$i = 1 \to n$，顺次走 $s_{i-1} \to s_i$ 这条边形成的路径」是 P 这条路径）

从而，问题转化为了求这张图的字典序最小的欧拉路径。套个板子即可。

思考路径：看到 $-1, +1$ 的前缀和序列，就要想到建出值域的有向图，用图论知识刻画。

具体实现时，其实不需要用 dfs。你可以在走每一步的时候，设当前在点 x，执行以下操作：

- 若不存在 $x \to x-1$ 的边，则只能走 $x \to x+1$ 的边，在答案序列末尾加入 `1`；
- 否则：
  - 若没有 $x \to x+1$ 的边，则可以放心地走 $x \to x-1$ 的边，在答案序列末尾加入 `0`；
  - 否则：
    - 若有 $x-1 \to x$ 的边，则也可以放心地走 $x \to x-1$ 的边，因为之后一定能通过 $x-1 \to x$ 的边走回来，然后再走 $x \to x+1$ 的边。在答案序列末尾加入 `0`；
    - 否则，则必须走 $x \to x+1$ 的边，在答案序列末尾加入 `1`。

时间复杂度 $O(n)$。

思考路径：不要什么东西都急着套板子，可以思考一下，在特殊的条件下，某些经典算法有没有更容易的实现。（当然，有时候还是套板子更保险一点）

CF1054G New Road Network

撰写时间 ？

标签 思维题 树 bitset 生成树 MST 贴紧上界或下界 等价表述条件和观察模型

未完待续

b. 将无解和最优化条件转成已知条件（2 题）

> Tip
>
> 首先是经验：CF 中如果 Output 里说无解输出 -1 但 Example 里没有 -1，那么就不会有无解的情况。

部分构造题有一个很大的特点，就是在解题开头就确定无解条件或最优解公式，也就是先确定充要条件的一边，去构造另一边，或是先给出不等式的一边，再构造取等。可以说这是一种“先猜后证”，也为构造增加了提示。当然这种猜测要足够谨慎，尽量思考通过题目条件能推导出的必然性，以及从数学角度去分析模型（找到模型的某个示性数，或守恒量）。

> Warning
>
> 不是所有构造题都可以使用这个技巧，同时要避免将非构造题误判成构造（例如，有时无解和最优化情况必须通过 dp 求出）。关键是通过分析条件来感知题目模型的解是可以用一种特定的规律描述，还是要根据输入数据，去遍历所有可能的解才能确定。

将无解和最优化条件转成已知条件 这个 tag 包含了以下几层意思：

- 判定无解条件（例：CF1543E The Final Pursuit）
- 将最优化条件转为精确值限制（例：P5419 [CTSC2016] 单调上升序列）

CF1543E The Final Pursuit

撰写时间 2025.9.26

标签 思维题 构造题 BFS 将无解和最优化条件转成已知条件 发现等价对象

这个题分两问。把两问分开来做。

第一问

容易发现所有的点都是等价的，可以随意找一个点赋予 0 的 id，从这个点出发推断所有点的 id。假设这个起点为 x。

> **思考路径**：容易发现所有的点都是等价的，可以随意找一个点赋予 0 的 id。这也是本题 发现等价对象 这个 tag 的来源。

然后，容易想到将 id 按照 popcount(id) 从小到大分配。

先分配 popcount(id) $= 1$ 的 id。对于 x 的所有邻居，分别赋予 $1, 2, 4, \ldots, 2^{n-1}$ 的 id，由于对称性，什么 id 分配给什么点是不重要的，随意分配即可。

然后，按照 popcount(id) 从小到大，分配 popcount(id) ≥ 2 的 id。容易发现这是一个 bfs 的过程，且发现如果我们令一个点的 id 为「所有拓展到它的点的 id 的 bitwise or」，这样构造出来的方案一定是合法的。于是，第一问就做完了，写个 bfs 就行。

思考路径：钦定起点后，显然要找到一些方式来依次递推出所有点的 id。这个递推的过程显然是 bfs，故能想到在 bfs 的过程中递推求出所有点的 id。

起点的邻居的 id 没有任何已有的信息可以参照，这时因为有对称性，不妨直接任意分配 $1, 2, 4, ..., 2^{n-1}$ 的 id，由对称性最后一定能构造出一种方案。

然后剩余的点的 id 就可以使用递推出它的点的 id 来参照了。构造一个点的 id 为「所有拓展到它的点的 id 的 bitwise or」是容易想到的。

第二问

首先尝试判定无解。

为了方便表述，把无向图的每条无向边拆成两条有向边。对于一个颜色 col，由于每个点都有一个颜色为 col 的邻居，故「终点的颜色为 col 的有向边」共有恰好 2^n 条。由此可以得到，「起点的颜色为 col 的有向边」共有恰好 2^n 条。而每个颜色为 col 的点都会产生 n 条「起点的颜色为 col 的有向边」，故颜色为 col 的点共有恰好 $\frac{2^n}{n}$ 个。当 $n \nmid 2^n$ 时，必然无解。故有解的情况只有 n 为 2 的幂次时。

思考路径：判定各种条件，可以从数量入手。

当 $n \mid 2^n$ 时，考虑如何构造。

考虑每个点的邻居的 id 是自己的 id 异或上 $1, 2, 4, ..., 2^{n-1}$ 所得到的。这里直接给出构造：令 id 为 x 的节点的颜色为 $\bigoplus_{i=0}^{n-1} [\mathrm{bit}(x, i) = 1] \times i$，其中 $\mathrm{bit}(x, i)$ 表示 x 的第 i 个二进制位是 0 还是 1。

思考路径：我们想要确定一个点的邻居的颜色，可以发现如果我们让邻居的颜色与「自己的 id 与邻居的异或值」相关，使得对于不同的「自己的 id 与邻居的异或值」，邻居的颜色也是两两不同的，就可以获得一个合法的构造。

因为「自己的 id 与邻居的异或值」必定为 2 的幂次，不难想到使用 $\log_2$ **自己的** id **与邻居的异或值** 参与颜色的构造。这个值的值域恰好为 $[0, n)$，符合颜色的值域。

如何让这个值参与颜色的构造呢？不难想到异或，因为异或的性质很好。这时发现由于 n 是 2 的幂次，对 $\log_2$ **自己的** id **与邻居的异或值** 任意做异或不会超出 $[0, n)$ 的值域，感到很舒服。

这时，我就想到了一种构造：设 x 的颜色为 col_x，则令 x 的邻居 v 的颜色为 $\mathrm{col}_x \oplus (x \oplus v)$，则这样构造出的所有邻居 v 的颜色必定两两不同。

于是这时就可以套用第一问的 bfs 递推做法了，随便钦定一个点的颜色，使用上述式子进行递推染色。容易发现这样染色不会出现矛盾。于是第二问这样就做完了。

但是实际上不需要递推。令 id 为 x 的节点的颜色为 $\bigoplus_{i=0}^{n-1} [\mathrm{bit}(x, i) = 1] \times i$，容易发现这样就符合「设 x 的颜色为 col_x，则令 x 的邻居 v 的颜色为 $\mathrm{col}_x \oplus (x \oplus v)$」的条件。

至此，两问全部做完了，时间复杂度 $O(2^n n)$。

P5419 [CTSC2016] 单调上升序列

撰写时间 2025.9.28

标签 思维题 构造题 构造 $n(2 \mid n)$ 个点的完全图的 $n-1$ 个两两不交的完美匹配 将无解和最优化条件转成已知条件 分析数量关系 从题面中获取提示 根据精确值限制来构造

首先，题面中给了一个例子：

> QUOTE
>
> 结论：假设带权无向图 G 有 100 个节点 1000 条边，且所有权值各不相同。那么，G 中一定存在一个单调上升路径，它的长度大于等于 20。
>
> 证明：假设每个节点上有一个探险家。我们按权值从小到大枚举所有的边，每次将该边连接的节点中的探险家的位置进行对调。可以知道，每个探险家都走的是一条单调上升路径。另外，由于共有 100 个探险家，而探险家一共走了 2000 步，所以有人走了 20 步。证毕。

OI 题目给这样的例子，还给出完整证明是非常少见的。而题目要求使得最长的单调上升路径最短。

考虑将这个例子中的证明套用到题目中，得出大小为 $n(2 \mid n)$ 的无向完全图的最长的单调上升路径的长度不小于 $n-1$。

观察两个样例，发现两个样例都恰好取到了这个下界。故而先猜测，所有情况都能取到这个下界，也就是我们需要构造一张图使得其最长的单调上升路径的长度**恰好**为 $n-1$。

> **思考路径：** 看到题面中不同寻常给出有完整证明（或其他各种奇怪的东西）就要想它有什么用。
>
> 要大胆猜测。先猜测答案一定取到下界，并以此为方向构造。如果后面发现了反例再补救/回来重新猜测，没发现反例就成功了。

然后，我们就要顺着这个 $n-1$ 思考了。

容易想到，把图的边集拆分为 $n-1$ 个集合 $S_1, S_2, \ldots, S_{n-1}$，使得：对于任意的路径，$\forall S_i$，在只经过 S_i 里的边的情况下，只能经过至多一条边。

然后，我们按照集合顺序依次 $S_1, S_2, \ldots, S_{n-1}$ 里的边从小到大赋权。这样就可以保证，当一条路径经过 S_i 里的某条边后，不可能再经过 $S_1, S_2, \ldots, S_{i-1}$ 里的边。从而，任意的路径都只能依次经过 $S_1, S_2, \ldots, S_{n-1}$ 里的边。并且由于前面保证了「对于任意的路径，$\forall S_i$，在只经过 S_i 里的边的情况下，只能经过至多一条边」，任意的路径最多只能经过 $n-1$ 条边 in total，从而达成了构造目标。

> **思考路径：** 首先要想到按边权分层。如何想到按边权分层，有两种方法：
>
> - 观察样例。第一个小样例是符合的（虽然第二个小样例不符合）。
> - 在「最长的单调上升路径的长度**恰好**为 $n-1$」的限制下，要由此想到按边权分层。

现在来考虑如何构造 $S_1, S_2, \ldots, S_{n-1}$。如果我们令每个 S_i 都是完全图的一个完全匹配，且 S_i 之间两两无交，那么就恰好满足了限制。

> **思考路径：** 如何想到用完全匹配来构造 S_i 有两种方法：

- 第一种：先试试如果让每个 S_i 的大小都相等怎么样，这样 $|S_i| = \frac{n(n-1)}{2} \div (n-1) = \frac{n}{2}$。由 $|S_i| = \frac{n}{2}$ 容易想到使用完全匹配。
- 第二种：「对于任意的路径，$\forall S_i$，在只经过 S_i 里的边的情况下，只能经过至多一条边」这个限制，显然要求了一个 S_i 里的边两两没有重复端点。那么这个限制很容易想到完全匹配。并且如果任意一个 S_i 不是完全匹配，$\sum |S_i|$ 就会小于 $\frac{n(n-1)}{2}$，无法把边分配完。从而只能构造每个 S_i 都是一个完全匹配，且 S_i 之间两两无交。

另外在构造时，也要有意识的往自己学过的典构造方法上靠近。

现在的问题就变成了，构造无向完全图的 $n-1$ 个完全匹配，使得它们两两无交。这是典，构造如图所示，图中的结构是单个完全匹配，将图中的结构旋转 $n-1$ 次（中间的品红色的点是特殊点，不参与旋转）就得到了 $n-1$ 个完全匹配。

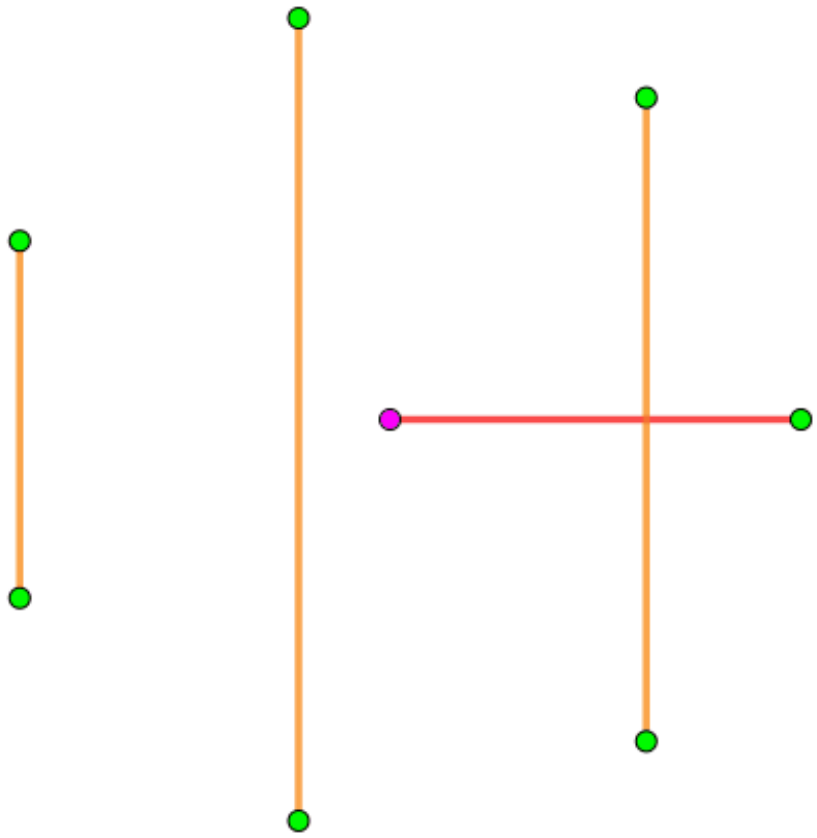

图 1　完全图的单个完全匹配

思考路径：「构造 $n(2 \mid n)$ 个点的完全图的 $n-1$ 个两两不交的完美匹配」是典，我是看 sjy APIO 2024 课件学到的。背 trick 吧。

NOTE

另外，我在想这题时，想到了按边权分层，但是我想偏了，想成构造 $\frac{n}{2}$ 棵两两不交的生成树，因为第一个样例的答案同样符合这个构造。但是这样只能做到路径长度为 n，我对着如何优化到 $n-1$ 想了好久，看了题解才恍然大悟。

这告诉我们，要多找几个方案想想！例如这个例子中，相当于把 $\frac{n(n-1)}{2}$ 拆分成 $\frac{n}{2} \times (n-1)$ 还是 $(n-1) \times \frac{n}{2}$ 的问题，而我只思考了一个。

c. 从特殊情况获取提示（5 题）

特殊情况包括特殊输入和特殊解，这一条侧重于特殊输入（主要指 corner case 和极端情况）。这样思考的帮助是：

- 通过特殊情况的解获取一般构造的线索；
- 通过将所有情况用某些手段归约到特殊情况，从而只需解决特殊情况；
- 通过构造极端输入确认无解性/答案上下界，或指出构造中面临的主要限制并排除不可能的构造方法。

从特殊情况获取提示 这个 tag 包含了上面提到的几层意思。

> Warning
>
> 但是不是所有题的特殊情况都能给出有效的提示，有时候反而会起到误导作用。要很警惕的一点是一些不成熟的思路可能会被构造的极端情况排除掉，但是实际上改一改就对了。下面的例题中会举出几例这样的误导实例。

CF1311E Construct the Binary Tree

撰写时间 2025.9.28

标签 思维题 构造题 调整法 从特殊情况获取提示

考虑极端情况。考虑对于固定的点数 n，能构造出的 d 的最小值 $d_{\min}$ 和最大值 $d_{\max}$ 分别是由什么形态的树构造出的。

容易发现，完全二叉树能达成 $d_{\min}$，而链能达成 $d_{\max}$。故如果输入的 d 不在 $[d_{\min}, d_{\max}]$ 内则必定无解。

> Tip
>
> 对于这种「输入恰为两个数，一个表示希望构造的结构的大小 n（在本题中为二叉树的节点数 n），一个表示构造的结构需满足某个权值为 m（在本题中为所有点的深度之和 d）」的题，可以考虑固定 n 和 m 其中一维，思索另一维能在什么范围内变动，既能获得无解条件，又能以此结合 调整法 来构造方案。下文的 CF1158B The minimal unique substring 也是这种类型的题。

现在考虑怎么通过调整法构造解。不妨从完全二叉树出发，每次构造使得 $d \xleftarrow{+} 1$，直至达成目标的 d。（从链出发每次使 $d \xleftarrow{-} 1$ 应该也能做，但是我写的是完全二叉树出发）

考虑由于是朝着链的方向调整，故我们要有一条链伸出来，下面称这条链为「主链」，为方便要求保证「主链」上每个点都是左儿子。「主链」初始为完全二叉树的左链。

当 $d \xleftarrow{+} 1$ 时，我们从剩余的二叉树中取出一个不在「主链」上且深度最大的点，设它为 x，把 x 放到「「主链」上与它深度相同的节点」的右儿子处（令这个操作为操作 1）。

此后的每次 $d \xleftarrow{+} 1$，我们把 x 再放到它在「主链」的兄弟的右儿子处（令这个操作为操作 2），直到 x 没有兄弟时把 x 改为左儿子，并返回执行操作 1。

Note

调整法的主旨是「每次调整的量最小」，以使得能遍历到并构造出所有的情况。像本题就让 d 每次只加 1。

调整的过程中，可以朝着终极的目标状态调整，同时尽量让结构同时具有起始和终止状态的特点。像本题中，就是从完全二叉树出发，朝着链调整，调整的过程中尽量让结构同时具有完全二叉树和链的特点（完全二叉树是因为我们每次只剥一个叶子，剩下的部分仍然是完全二叉树；链是因为我们构造了「主链」这一结构并逐步增长之）

综上，我们的构造方法是，在 $d \xleftarrow{+} 1$ 时，如果能执行操作 2 就执行操作 2，否则执行操作 1。容易发现这个构造方法能够遍历 $[d_{\min}, d_{\max}]$ 里的每个 d，从而所有的 $d \in [d_{\min}, d_{\max}]$ 都有解，都能通过这个方法构造出。

这个维护过程直接不带脑子暴力维护也不会比 $O(n^2)$ 更慢，且容易做到 $O(n)$，足以通过本题。（出题人很良心的开了 $n, d \le 5000$，让不带脑子暴力维护也能通过）

CF1515F Phoenix and Earthquake

撰写时间 2025.9.28

标签 思维题 生成树 鸽巢原理 从特殊情况获取提示 将无解和最优化条件转成已知条件 分析数量关系

首先，能使所有城市连通的必要条件是 $\sum a_i \ge (n-1)x$（显然）。

我们大胆猜测这个条件也是充分的。容易构思出一种构造方案：每次找出一个 $a_u \ge x$ 的点 u，将 u 与它的任意一个邻居 v 连通，并将 u 和 v 缩成一个点。这个方案的正确性证明如下：

Proof

若保证了 $\sum a_i \ge (n-1)x$，那么由鸽巢原理，必然存在 $a_u \ge x$ 的点 u。

故我们说明了这个条件是充要的，并给出了一种构造方案。这个构造方案直接做是 $O(n^2)$ 的，但是我们把所有点的 a 放入一个堆里维护就可以做到 $O(n \log n)$，成功解决本题。

上面的是我想的做法，下面是题解的另一种构造方式：

Solution

~~找出任意一棵生成树。从下往上 dfs，对于一个点 u 和它的父亲 fa，做以下事情：~~

- ~~若 $a_u \ge x$，则立即把 u 和 fa 连通并缩成一个点。~~
- ~~若 $a_u < x$，由于我们证明了~~

贺一下洛谷题解：

在 n 个点的树中，我们先随便选出一个叶子，考虑其点权 a_u：

- 若 $a_u \ge x$，那么我们最早修建 u 和其父亲的边，将它们缩成一点，然后转化成 $n-1$ 个点的树，仍然满足条件。
- 若 $a_u < x$，那么我们如果从树中删除这个叶子和其父边，因为 $(n-1) \times x$ 减少了 x 而点权减少了 a_u，所以剩余树仍然满足条件，所以我们把这条边最后修建即可。

所以做法就很简单了！

我们先随便选出一个生成树，进行一次 DFS，如果一个点 u 的父亲为 v，当前 u 的子树访问完毕，那么分类讨论：

- 如果当前 $a_u \geq x$，那么令 $a_v \leftarrow a_u + a_v - x$，将 (v, u) 加入一个队列中。
- 如果当前 $a_u < x$，那么将 (v, u) 加入一个栈中。

最后先按序输出队列中的边，再按序输出栈中的边，即可。

时间复杂度 $O(n)$。

CF618F Double Knapsack

撰写时间 2025.9.29

标签 思维题 前缀和 鸽巢原理 分析数量关系 从特殊情况获取提示 加强构造条件

本题的限制看起来是比较弱的，手模几组发现都有解，于是大胆猜测所有情况都有解。

本题有两种解法：

SOLUTION

解法 1

见 这篇题解。

看到“选子集”，考虑弱化条件，改成选子段。

设 $\text{suma}_i, \text{sumb}_i$ 表示 a, b 的前缀和，则我们需要选出 $[l_1, r_1], [l_2, r_2]$，使得 $\text{suma}_{r_1} - \text{sumb}_{r_2} = \text{suma}_{l_1-1} - \text{sumb}_{l_2-1}$。

注意到 $a_i, b_i \in [1, n]$，这启发我们：

- 假设 $\text{suma}_n < \text{sumb}_n$。
- 对于每个 i，选出最小的 j 使得 $\text{sumb}_j \geq \text{suma}_i$，此时 $\text{sumb}_j - \text{suma}_i \in [0, n)$。由于这样的 (i, j) 共有 $n + 1$ 对，由鸽巢原理一定会产生重复。
- 我们把这样的 (i, j) 拿去当上面的 $(r_1, r_2), (l_1 - 1, l_2 - 1)$，选两对前缀和之差相同的作为答案即可。

上面的构造方式同样证明了随意排列后一定存在至少一组合法子段。$\text{suma}_n \geq \text{sumb}_n$ 时 swap a, b 即可。

双指针即可。时间复杂度为 $O(n)$。

思考路径：由本题的限制情况比较弱，可以想到弱化条件，变为子段。然后再往鸽巢原理上面靠。

不过感觉「选出最小的 j 使得 $\text{sumb}_j \geq \text{suma}_i$」有点难以想到。下面是《构造题方法总汇》给出的一种思考路径：

> **Content**

考虑 $B=\underbrace{\{n,n,...,n\}}_{n个n}$ 的情况，因为要使 A,B 中的和相等，A 中的和就必须是 n 的倍数，容易想到将 A 作前缀和后对模 n 余数鸽巢。这样就把正解的提示都集齐了鸽巢、前缀和。

但是这还不至于直接想到 A 的前缀和减去 B 的前缀和，我给出的思考路径如下：

尝试把它拓展到更一般的情况，此时不能简单的求模 n 余数了。考虑修改 $B=\underbrace{\{n,n,...,n\}}_{n个n}$ 的情况的做法，让其更能适配一般的情况。发现 suma_i 模 n 的余数，同时也是：「令 j 为最大的 j 使得 $\text{sumb}_j\le\text{suma}_i$，return $\text{suma}_i-\text{sumb}_j$」。发现这个很容易拓展到一般情况，尝试一下，发现竟然是正解！于是就成功切掉这题了。

Solution

解法 2

参考的是这篇题解。

与 ZR3333 好东西一定要送！很像。

令 x 为 A 集合选的数的和减去 B 集合选的数的和.

如果 $x<n$，那么从 A 中任意选一个数，否则从 B 中任意选一个数。容易发现这样能使得任意时刻都有 $x\in[0,2n)$。

如果存在两个时刻的 x 相等，那么选取「在这两个时刻中间被选取的数」作为答案集合，容易发现是合法的。

可以证明一定存在两个时刻的 x 相等。证明如下：

Proof

算上最初没有选任何数时 $x=0$ 的时刻，总共有 $2n+1$ 个时刻，而 $x\in[0,2n)$，由鸽巢原理可得一定会有两个相同的 x。

另外，还有一个问题是，当 $x<n$ 时，如果 A 集合被取光了怎么办?

由于 A 集合都被取光了，迄今为止（包含现在这个时刻）的所有尝试取 A 集合（意即 $x<n$）的时刻共有 $n+1$ 个，也就是说 x 总共在 $n+1$ 个时刻 $\in[0,n)$，由鸽巢原理可得一定出现过两个相同的 x，故应当在之前就得出一组解了，故不会有这种情况。

思考路径：看到值域为 n 和 $2n$ 个物品，可以想到 ZR3333 好东西一定要送！这个典。

另外也可以朝着鸽巢原理的方向想一想，构造什么东西来使其符合鸽巢原理从而成功构造。

CF1019C Sergey's problem

未完待续

AT_agc064_b [AGC064B] Red and Blue Spanning Tree

未完待续

d. 分析数量关系（4 题）

CF1667C Half Queen Cover

撰写时间 2025.9.29

标签 思维题 构造题 分析数量关系

参考的这篇题解。

设答案为 k，考虑在 n 阶棋盘左上角的 k 阶棋盘中不重行不重列地放置 k 个棋子，于是 n 阶棋盘中的前 k 行均能被同行棋子攻击，前 k 列均能被同列棋子攻击。

于是还剩下右下角的 $n-k$ 阶棋盘，它们只能被同对角线的棋子攻击。

$n-k$ 阶棋盘可被划分为 $2(n-k)-1$ 条对角线，为使它们被覆盖，需要有 $k \geq 2(n-k)-1$。

解得 $k = \lceil \frac{2n-1}{3} \rceil = \lfloor \frac{2n+1}{3} \rfloor$。

思考路径：这种比较诡异的「半皇后」，首先要想到假设半皇后不能攻击对角线的话会怎么样，即只能攻击行和列，显然这种情况必须使用 n 个半皇后。然后再把对角线的攻击加回来，我们想到，放置 k 个半皇后覆盖 k 行 k 列，将剩余的部分用半皇后的斜线攻击来覆盖。

至于为什么一定是放在左上角的 k 行 k 列？这个我也不知道，我也不会证，可能是因为不聚集在左上角的话，一部分格子会更加难以用斜线攻击到。实际想题时应当是先尝试一下这个方案（因为这种比较容易想到），由此计算出皇后个数的最小值，并用暴力验证皇后个数的最小值的正确性，从而给自己的构造提供强有力的论据支持，得知自己的构造很大概率是正解。然后再试图以此构造具体的放置 k 行 k 列皇后的方案，暴力也可以对这个构造方案提供灵感。

接下来考虑放置方案。我们想要把 $2k-1$ 个对角线全部覆盖。

直接上图。类似马走日的移动方式，每次 $x \xleftarrow{+} 1$、$y \xleftarrow{+} 2$，$y > k$ 了就把 y 设为 2。

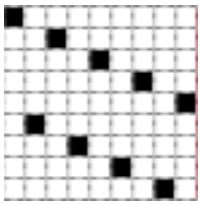

图 2　半皇后放置方案的构造

思考路径：这个构造还是比较好想的，首先尝试在 $(1,1)$ 放置一个皇后（因为看上去比较舒服），覆盖了一条对角线。然后这个 $(1,1)$ 的皇后会 ban 掉一行一列和一条对角线，下一个皇后明显最近也只能放在 $(2,3)$ 了。以此类推，可以构造出所有皇后的放置方案，发现满足条件，就成功了。

CF1158B The minimal unique substring

撰写时间 2025.9.29

标签 思维题 构造题 分析数量关系

观察样例发现有循环节。故往循环节上面思考。

具体分析，发现：设 s 为最小的循环节，l 为循环节的长度，则 $\forall i \geq l-1, s+\text{repeat}(s,i)$ 的最短唯一子串的长度为 $i-l+2$（其中 $\text{repeat}(s,i)$ 表示 $\underbrace{s+s+...+s}_{\lfloor \frac{i}{l} \rfloor}+s[1, i \bmod l]$）。如果继续缩短则最短唯一子串的长度保持为 1。

得出这个结论后，考虑对于给定的 n,k 推算 l。有

$$\begin{cases} l+i=n(s+\text{repeat}(s,i)\text{ 的长度为 }n) \\ i-l+2=k(\text{最短唯一子串的长度 }k\text{为}i-l+2) \end{cases} \Rightarrow l=\frac{n-k}{2}+1$$

由于题面中保证了 $(x \bmod 2)=(k \bmod 2)$，故 l 一定为整数。

故构造方案即是：令 $s=\underbrace{000...0}_{l-1\text{个}0}1$，令答案为 $s+\text{repeat}(s,n-i)$ 即可。容易验证这样构造的正确性。

> **思考路径**：这道题首先观察样例，往循环节上思考；
>
> 再要手模样例和小数据，冷静分析总结，得出「设 s 为最小的循环节，l 为循环节的长度，则 $\forall i \geq l-1, s+\text{repeat}(s,i)$ 的最短唯一子串的长度为 $i-l+2$」的结论；
>
> 最后尝试把这个结论应用在给定的 n,k 上，发现能利用方程组唯一解出 l。解出 l 后，构造也是显然的了。

> Tip
>
> 可以通过题面中给的 $(x \bmod 2)=(k \bmod 2)$ 来从侧面验证自己的构造是否很可能是正解。
>
> 详细地说，因为我们给出的构造方法当且仅当 $(x \bmod 2)=(k \bmod 2)$ 时有解，而 $(x \bmod 2)=(k \bmod 2)$ 恰好是题面给的条件，而题面中又保证有解。这就从侧面说明了，我们给出的构造方法很有可能是正解。

樱花抄

暂无题面，是《构造题方法总汇》的作者的校内题，笔者无法获取。

P8416 [THUPC 2022 决赛] 拯救还是毁灭

未完待续

> Content
>
> P9961 [THUPC 2024 初赛] 排序大师

e. 寻找诈骗条件（6 题）

「诈骗条件」指的是一些「看上去很复杂，实际上可以经过转化变为比较 simple 的条件」的条件。因其以复杂的条件劝退选手、让选手一头雾水，实则是一些 simple 的结论，故名之「诈骗」。因此 寻找诈骗条件 这个 tag 常与 等价表述条件和观察模型 这个 tag 一起出现。

除了上面这种，「诈骗题」还有可能是「藏着一些对解题至关重要，又很 simple、无脑的结论」的题，典型的如题面中说答案是非负整数，实际上的答案只可能是 0 和 1 其中之一。

主要是要有寻找诈骗条件的意识。有些题目的条件或性质可以极大程度地减小可能性，不要忽视。往往可以通过思考极端情况获得启发。

CF425B Sereja and Table

撰写时间 2025.9.29

标签 思维题 数据范围分治 等价表述条件和观察模型 寻找诈骗条件

题目中给出的条件有点难以用来刻画。尝试思考这个条件的本质。

注意到 a 的值只有 0/1。经过思考后发现，设操作后达成题目条件的 a 为 b，则 b 一定成网格状。形式化地说，一定存在两个 0/1 数组 $\{r_n\},\{c_m\}$ 使得 $b_{i,j}=r_i\oplus c_j$。

思考路径：当题目中的条件比较诡异的时候，要想想是不是被诈骗了，有没有什么更加 simple 的条件与题目中的条件等价。

本题中的「b 一定成网格状」的结论是容易发现的。具体的，就手模一下，发现一个较大的长方形旁边不能贴着一个较小的同色的长方形，然后就发现该结论了。

得出上面的结论后，我们可以得出，b 的每一行要么是 $\{c_m\}$ 这个数组，要么是 $\{c_m\}$ flip 后的结果。对于列同理。

但是后面我不会做了。我只想到了这个做法：

Solution

做法 1

暴力枚举在 a 的第一行中修改 $\le k$ 个位置作为 $\{c_m\}$ 数组。对于一个给定的 $\{c_m\}$ 数组，计算它能产生的最小修改代价也是容易的，因为每一行之间是独立的，对于一行，在不 flip 和 flip 中选择代价较小的那个即可。

时间复杂度 $\sum_{i=0}^{k}\binom{m}{k}nm$。无法通过。

直觉上感觉做法 1 有很多可以剪枝的地方，但是得不出一个确切复杂度的做法。

看了题解后发现，竟然是数据范围分治！

下面是另一个做法：

Solution

做法 2

因为总代价 $\le k$，故 $\{c_m\}$ 必定等于 a 的前 $k+1$ 行其中的某一个。不然的话前 $k+1$ 行每行至少造成 1 的代价，总代价就 $\ge k+1$ 了。

因此暴力枚举 $\{c_m\}$ 是 a 的前 $k+1$ 行的哪一行，再暴力 $O(nm)$ 计算总代价即可。时间复杂度 $O(nmk)$。

做法 2 的时间复杂度看起来很优秀，但是它有一个前提：$n\ge k+1$，否则总行数都 $<k+1$ 了，「$\{c_m\}$ 必定等于 a 的前 $k+1$ 行其中的某一个」也就不成立了。

而 $n \le k$ 的情况我们可以拼做法 1，此时做法 1 的时间复杂度为 $O(2^k nm)$，可以通过。

而在 $n > k$ 的情况我们拼做法 2 即可。时间复杂度 $O(nmk)$。

思考路径：当觉得做法中有很多可以剪枝的地方时，要用于尝试剪枝，或者往**数据范围分治**上想。尤其是题目中像这样给出一个区别于 n, m 的量 k 时，要想到数据范围分治。

CF1689E ANDfinity

未完待续

CF1685C Bring Balance

未完待续

> Note
>
> 要尽可能缩小操作次数，而非获得一个自己以为不能再优化的构造方案就摆了，要证明自己的构造方案是最优的。

CF1781G Diverse Coloring

未完待续

CF1852E Rivalries

撰写时间 2025.9.30

标签 思维题 等价表述条件和观察模型 寻找诈骗条件

算是第一个自己独立想出来的 *3400 了。虽然我怕自己假掉，没有在看题解之前自己写代码尝试，但是我的思路应该是正确的吧。

首先看到题目的条件感觉非常复杂。于是开始大眼观察样例。

发现样例似乎也不简单，看起来有很多种情况。而且还有新数组中的数小于原数组中的数的情况（样例中的 $(1,1,1,1,4,3,3,3) \to (1,2,2,1,4,3,2,3)$）。

想到这里，我有点想摆烂了。但是我还没怎么分析过性质，于是我打算从性质入手。

我之前已经通过观察样例发现了，对于一个区间有：设区间最大值为 k，若这个区间包含了 k 在原序列中的所有出现，那么这个区间的 power 一定为 k，从而剩下的数看起来会比较能修改一些。

这个结论并不是很通用。尝试思考其本质。

然后我尝试思考每个数对 power 造成的贡献。设一个数 k 在原序列中出现的最左边的位置为 l_k，最右边的位置为 r_k。发现一个数 k 会让一个区间 $[L,R]$ 的 power 对 k 取 max，当且仅当 $[l_k, r_k] \subseteq [L,R]$。

并且，我们之前观察样例发现，更改一个数的时候不会更改其最左边的出现和最右边的出现，如 $(2,1,1,1,2) \to (2,1,2,1,2)$。结合上面这个结论，容易发现，**一个数 k 的除了 l_k, r_k 以外的出现位置上的数都是可以更改的**。这是因为 k 的贡献只与 l_k, r_k 有关，中间那些数都是无关紧要的。下面称所有「一个数 k 的除了 l_k, r_k 以外的出现位置」为「可修改的位置」。

于是我们在修改这些「可修改的位置」的时候，自然贪心的希望越大越好。但是修改的过程中不能影响每个区间的 power 的值。思考后容易发现，如果将位置 p 上的数改为 k，只要不影响 k 能造成的贡献，也就是 $l_k \le p \le r_k$，那么一定是合法的。于是容易对每个位置 p 分别求出最大能修改成什么。

这个做法看起来很对，尝试去放到样例上手模试试。这时我们发现，还漏掉了一种情况：修改后的数可能没有出现在原序列中，这时我们上面的讨论对其都无效了（因为基于 l_k, r_k 的结论的基础是存在 l_k, r_k，也就是 k 必须要在原序列中出现过）。若一个修改后的数可能没有出现在原序列中，我们把这样的数叫做「横空出世数」。

进行思考。假设 k 为「横空出世数」，那么不应当有任何区间的 power 为 k。这个条件能否达成呢？答案是可以的。通过观察样例或思考都可以发现，设修改后的序列中 k 的最左边和最右边的出现位置分别为 l_k, r_k，如果存在一个 $v > k$ 使得 $[l_v, r_v] \subseteq [l_k, r_k]$，那么 k 就会被 v **支配**，不产生任何贡献（因为它能贡献到的区间，v 也一定会贡献到，并覆盖掉 k 的贡献）。

再次观察样例，我们发现，「横空出世数」最多只会有 1 种值。这个性质也是可以证的，结合上面的结论，显然 $[l_k, r_k]$ 越大，被其他数支配的可能性也越大。如果有 2 种「横空出世数」，设它们分别为 u, v，则将它们全部改为 $\max(u, v)$ 一定不劣，因为显然 $[l_u, r_u] \subseteq [l_u, r_u] \cup [l_v, r_v]$, $[l_v, r_v] \subseteq [l_u, r_u] \cup [l_v, r_v]$，故若同时存在 u, v 时合法，改为 $\max(u, v)$ 后也一定合法。**所以最多只会有 1 种「横空出世数」**。

至此，这个题差不多就做完了。

对于不存在「横空出世数」的情况，求出每个位置能变成的数的最大值即可，这个是区间 chkmax 查询单点值，随便做。

对于存在「横空出世数」的情况，枚举是哪个数把「横空出世数」给偏序了，设这个数为 k，令「横空出世数」为 $k-1$ 即可。首先把「可修改的位置」中，小于等于 $k-1$ 的位置全部修改为 $k-1$；若还未满足 $[l_k, r_k] \subseteq [l_{k-1}, r_{k-1}]$ 的限制，选择能使得条件满足且代价最小的「可修改的位置」改为 $k-1$ 即可。

总时间复杂度 $O(n \log n)$。

~~思考路径在上文中已经体现的够多了，就不写了。~~

> NOTE
>
> 上述的思考过程体现了「观察样例与推导结论相辅相成」。
>
> 具体的：
>
> 1. 观察样例 以 猜出核心性质和结论。
> 2. 猜出结论后，用已经推导的结论来验证新的结论。
> 3. 再把新的结论用在其他样例里，看看是否奏效。
>   - 奏效的话就是对的结论。
>   - 不奏效的话，再回到第 1 步，但是观察的是 hack 数据，以试图弥补结论的漏洞。
>
> ~~（我这段话的道法风味怎么这么浓啊啊啊）~~

CF1375F Integer Game

未完待续

f. 加强构造条件（8 题）

CF1599A Weights

撰写时间 ？

未完待续

CF1270G Subset with Zero Sum

未完待续

CF1835C Twin Clusters

未完待续

CF1391E Pairs of Pairs

未完待续

AT_arc153_c [ARC153C] ± Increasing Sequence

未完待续

CF1218G Alpha planetary system

未完待续

g. 构造不依赖于输入的通解（3 题）

对于输入非 $O(1)$ 的情况，如果对构造的限制较小，可能性较大，但难以基于题目要求条件进行讨论，可以尝试直接构造一个解，无论输入如何都满足条件。这其实也是一种加强，一种寻找模型当中的“必然性”的思维。有的时候只考虑根据输入推导出解也是一种错误的思维惯性。

AT_agc052_a [AGC052A] Long Common Subsequence

撰写时间 2025.9.29

标签 思维题 构造题 诈骗题 构造不依赖于输入的通解

注意到同时满足三**个**串的限制使得难以进行 dp。

经过一段时间的构造尝试无果后，我们猜测这个题可能是诈骗，开始向着

构造不依赖于输入的通解 上思考。

注意到，$\underbrace{000\ldots0}_{n\text{个}0}\underbrace{111\ldots1}_{n\text{个}1}$ 必定合法，且它的长度为 $2n$，相较于 $2n+1$ 仅差 1 个字符。我的直觉告诉我 $\underbrace{000\ldots0}_{n\text{个}0}\underbrace{111\ldots1}_{n\text{个}1}$ 这个串还有一点没发挥尽的潜力，似乎可以再塞一个字符进去。

于是我们就尝试从 $\underbrace{000\ldots0}_{n\text{个}0}\underbrace{111\ldots1}_{n\text{个}1}$ 上调整。经过一些尝试后，我们发现，似乎无论怎么构造都无法 hack 掉 $\underbrace{000\ldots0}_{n\text{个}0}\underbrace{111\ldots1}_{n\text{个}1}0$ 这个串。

然后你把 $\underbrace{000\ldots0}_{n\text{个}0}\underbrace{111\ldots1}_{n\text{个}1}0$ 提交上去，发现过了。~~不是哥们，这个题真是诈骗啊~~

其实可以证明 $\underbrace{000\ldots0}_{n个0}\underbrace{111\ldots1}_{n个1}0$ 是所有串的通解。证明如下：

PROOF

第 n 个 0 和第 $2n$ 个 0 之间是一个完整的串（「完整的串」指「由 n 个 0 和 n 个 1 组成的串」），故一定有 n 个 1。

选取

- 第 1 到第 n 个 0
- 第 n 个 0 和第 $2n$ 个 0 之间的所有 1
- 第 $2n$ 个 0

作为子序列，就得到了 $\underbrace{000\ldots0}_{n个0}\underbrace{111\ldots1}_{n个1}0$。

（注：上述的思考过程基本上都是我口嗨出来的，我虽然自己想到了正解，但我实际上没有在看题解前进行提交验证正确性，且我在思考的开始就因为知道 构造不依赖于输入的通解 这个 tag 而没有往正经的思路上尝试）

思考路径： 本题最大的难点在于意识到这其实是一道诈骗。

至于如何意识到，一是要求同时满足三**个**串的限制（如果是两个串的话会看起来更加像正经题一点），二就是经过足够多的尝试无果后就要想想有没有可能是诈骗，往一些 simple 的做法上尝试。

CF1495C Garden of the Sun

撰写时间 ？

未完待续

CF1485D Multiples and Power Differences

撰写时间 2025.9.29

标签 思维题 构造题 网格图黑白染色 构造不依赖于输入的通解 加强构造条件

这个倍数的要求看起来很烦（至少在相邻的两个数都受到倍数限制时是很烦的），于是我们想要尽可能去除掉倍数的限制。

注意到 a 很小。容易想到让一些数填 2 到 16 的 lcm，这样就不用管倍数的限制了。令 $L = \mathrm{lcm}(2,3,\ldots,16) = 720720$，惊喜地发现 $L \le 10^6$，且恰好是这个量级。于是我们认为这个做法很有可能是正解。

思考路径： 由「倍数的限制很烦」想到 构造不依赖于输入的通解，进一步地由「a 很小」想到填 lcm。

接下来我们的任务就是想办法利用 L 去除掉倍数的限制。可是仍然不能所有数都填 L，因为相邻格子差不能为 0。

显然如果一个ペア中恰有一个数为 L，那么显然容易构造出另一个数（令它为 $L-a^4$，其中 a 是另一个数的位置的 a）。

我们对格子黑白染色，黑色的格子一律填 L，白色的格子填 $L-a_{i,j}^4$，容易发现这是一个合法的构造。

> **思考路径**：如何想到黑白染色：
>
> 我们已知恰有一个数为 L 的ペア是容易构造的。那么顺着它，现在我们想要让任意一对相邻的格子组成的ペア，都能满足「恰有一个数为 L」的条件。由这个可以想到黑白染色。
>
> （其实我在想这道题的时候是脑电波一下子想到黑白染色的，上面的思考路径只是我胡的一个可能的思考路径）

h. 研究操作能实现的功能（6 题）

i. 逆操作与中继状态（3 题）

j. 外围排除法（7 题）

k. 归约法和增量法（6 题）

l. 调整法（4 题）

m. 做法拼合（3 题）

n. 分组与分治（2 题）

实战演练

P3514 [POI 2011] LIZ-Lollipop