构造选讲

Sol 1

December 6, 2024

为啥要讲这个？

构造题由其本身带有的非套路性导致它在考场上成为一种很难处理的存在，并且它在考场上还有不算很低的出现可能性。
因此我希望给出一些比较成套路的方式来处理这种题。
事实上，不仅仅是构造，对于更普遍的 ad-hoc，有一些方法也是适用的。
虽然很多题可能大家都做过，但是比起讲做法，其实这节课更重要的是还原一下每一个题的思维流程。

大多数时候需要知道的技巧

写 checker，以及使用写好的 checker 对拍：需要了解如何进行多文件读写，推荐使用 `std::ifstream` 和 `std::ofstream`。
利用给出的 checker：需要注意从题面复制编译选项时可能会出现部分符号变为非 ASCII 字符。

主要思想

遇到构造题的时候有一些常见的思考方向：

- 判定猜测一定有解，最优化猜测有一个很简单的形式；
- 手算小数据寻找规律；
- 转化为图论模型，使用欧拉回路、网络流、2-SAT 等算法；
- 先强化题目中的限制再逐渐引入没有用到的自由度，或限制先只利用一部分信息再加入未利用的信息；
- 增量法；
- 尝试构造出一个简单的变换 / 一种简单的信息之间的联系；
- ……

如果让你判断有解或最优化一个值

猜测有解条件 / 最优化值是一个很简单的形式，例如一定有解 / 最优值是一个显然的答案上界。

如果让你判断有解或最优化一个值

猜测有解条件 / 最优化值是一个很简单的形式，例如一定有解 / 最优值是一个显然的答案上界。
这是因为如果你需要一些复杂的算法来求出最优值 / 有解情况，那么通常构造方案不会显著难于判断 / 最优化（例如你需要把最优答案 dp 出来，那么构造方案只是记录一下 dp 转移过程的事），那么整个题也就不是个构造题了，可以使用处理正常的判定 / 最优化问题的方法。

如果让你判断有解或最优化一个值 (cont.)

但是给你一个你不会做的题，你不一定知道它的难点到底是不是构造方案……
为了判断一道题的难度是否在构造方案上，可以通过手动模拟一些较小/较特殊的情况来观察（对于大多数题，等价的情况在数据很小的时候不多，手工计算已经足够。OI 赛制下，必要的时候也可以写个暴力来辅助）。如果发现有解/最优值的分布没有明显规律，那么基本可以确定难度不在构造方案上。

Luogu P5811

给定一个 $a+b+c$ 个点，m 条边的连通图，要求将其划分为三个点集，大小分别为 a,b,c，且这三个点集的导出子图至少有两个连通，构造方案或判断无解。

$m \leq 2 \times 10^5$。

Luogu P5811

不妨假设 $a \leq b \leq c$，则让大小为 a, b 的点集不连通的方案是最容易构造的。

Luogu P5811

不妨假设 $a \le b \le c$，则让大小为 a, b 的点集不连通的方案是最容易构造的。
然后如果存在一个点，移除这个点之后图分裂为若干个 $< a$ 个点的连通块，那么显然无解：因为只要分出一个 a 个点的连通块，剩下的部分大小一定都 $< b$。

Luogu P5811

不妨假设 $a \le b \le c$，则让大小为 a, b 的点集不连通的方案是最容易构造的。
然后如果存在一个点，移除这个点之后图分裂为若干个 $< a$ 个点的连通块，那么显然无解：因为只要分出一个 a 个点的连通块，剩下的部分大小一定都 $< b$。
这个就是充要条件。构造方案的方法很复杂，略去。

QOJ5434

构造一个长度为 n 的 01 串，使得其拥有最大数量的本质不同子串数量。

$n \leq 2 \times 10^5$。

QOJ5434

枚举每一个长度 l，长度为 l 的本质不同子串个数有两个上界：2^l 和 $n-l+1$。

QOJ5434

枚举每一个长度 l，长度为 l 的本质不同子串个数有两个上界：2^l 和 $n-l+1$。直接加起来可以得到答案的显然上界：

$$\sum_{i=1}^{n} \min\{2^l, n-l+1\}$$

QOJ5434

枚举每一个长度 l，长度为 l 的本质不同子串个数有两个上界：2^l 和 $n-l+1$。直接加起来可以得到答案的显然上界：

$$\sum_{i=1}^{n} \min\{2^l, n-l+1\}$$

这个就是答案的值。构造方法涉及到另外一个套路，后面再讲。

寻找规律

就是如果输入只有一个数或者几个数，可以对于比较小的情况尝试若干种构造，并归纳出一种可以拓展到任意情况的方法。
这种通常都是要自己尝试构造的，样例在很多时候都会特殊构造使得你看不出规律。自己尝试构造也通常需要尝试很多次才能得到一个可以拓展的方案。

CF1916D

给定奇数 n，构造 n 个不含前导 0 的 n 位十进制数，使得每一个数都是完全平方数，并且它们包含相同的数字可重集。

$n \leq 99$。

$n = 3 : 169, 196, 961$；$n = 5 : 16384, 31684, 36481, 38416, 43681$。

CF1916D

做法 1：再构造一组 $n = 5$ 的：$10609, 16900, 19600, 90601, 96100$，然后规律就很明显了。

做法 2：写一个 $O(10^{n/2})$ 的暴力，可以发现 $n = 11$ 可以构造一个大小为 99 的集合。于是打表，对更大的数据往后加 0 即可。

CCPC 网络赛 A

给定 n, m, k，构造一个 $n \times m$ 的 01 矩阵，使得如果允许对矩阵做任意次（包括 0 次）将所有 1 上/下/左/右对齐的操作，则恰好可以得到 k 种不同的形态。
$n, m \leq 1000, k \leq 10^9$。

CCPC 网络赛 A

首先注意到本质不同的操作方式只有操作 $0 \sim 2$ 次，共 13 种。从而 $k \geq 14$ 则无解。
通过这个观察写一个暴力对所有 $n \times m \leq 25, k \leq 13$ 的情况求出构造。
对着它找规律即可。非常好找。至于为啥我们通过规律认为无解的那些位置确实无解……我不会证，官方题解也没证，我们就当它是无解吧。

涉及到图论模型的构造

如果你观察到一些限制/操作会涉及到恰好两个元素，那么通常可以考虑建图，并在限制/操作涉及到的两个元素之间连边。
即使不是这个形式，也可以枚举一些图论模型，例如欧拉回路和网络流。这些算法都是不太能直接从题面里面看出要用，但是把题目往这个算法上面套就可以得到做法。

P6185

一个长度为 n 的序列，可以对它进行 m 种操作：第 i 种操作由三元组 (t_i, u_i, v_i) 描述：如果 $t_i = 1$，则可以对序列的 u_i 位置和 v_i 位置同时 $+1$ 或同时 -1；如果 $t_i = 2$，则可以对序列的 u_i 位置和 v_i 位置中的一个 $+1$，另一个 -1。

给定长为 n 的序列 a, b 和 m 种操作，判定是否能够将 a 通过若干次操作变成 b。$n, m \leq 10^5$。

（虽然是判定，但是其实基本上就是构造：可以输出每个操作的操作次数。）

P6185

考虑将所有 $t = 2$ 的边建图，那么一个连通块内的所有值可以任意互相输送。于是我们可以将连通块缩成点，点权是连通块内所有点的点权和，从而转化为 $t = 1$ 的情况。

$t = 1$ 的情况涉及到另外一个 trick，后面再说。

QOJ5434

构造一个长度为 n 的 01 串，使得其拥有最大数量的本质不同子串数量。

$n \leq 2 \times 10^5$。

QOJ5434

之前已经提到过需要构造出的本质不同子串数量是：

$$\sum_{i=1}^{n} \min\{2^l, n-l+1\}$$

注意到存在 k 使得 $n-k+1 > 2^k$ 但是 $n-k \le 2^{k+1}$，同时只需要让 k 和 $k+1$ 同时取到上界，剩下的部分都自然取到上界。

QOJ5434

考虑建图：图包含 2^k 个点，每个点是一个长为 k 的 01 串。对于每一个长为 $k+1$ 的 01 串，从它的前 k 位到它的后 k 位连一条有向边。我们需要找到一条路径使得它恰好经过 $n-k$ 条边，不经过重复边，且经过所有点。我们定义这个图为 G_k。由于 $n-k \geq 2^k$，所以点数肯定是够用的。我们不妨先构造出原图的一个哈密顿路。但是求哈密顿路是 NPC 问题，需要一些其他的手法。

QOJ5434

考虑建图：图包含 2^k 个点，每个点是一个长为 k 的 01 串。对于每一个长为 $k+1$ 的 01 串，从它的前 k 位到它的后 k 位连一条有向边。我们需要找到一条路径使得它恰好经过 $n-k$ 条边，不经过重复边，且经过所有点。我们定义这个图为 G_k。

由于 $n-k \geq 2^k$，所以点数肯定是够用的。我们不妨先构造出原图的一个哈密顿路。但是求哈密顿路是 NPC 问题，需要一些其他的手法。

注意到：G_k 的哈密顿路接成的 0/1 串和 G_{k-1} 的欧拉路接成的 0/1 串是一样的，于是这个图上可以 $O(n)$ 求哈密顿路。

这样，我们构造出 G_k 的一个哈密顿路。我们考虑欧拉路径算法的本质：插入回路。我们将找到的哈密顿路上的所有边从 G_k 中删去，然后依次枚举 G_k 上的所有点。

QOJ5434

对于每一个点，我们可以在这个点所在的连通块里面找到一个欧拉回路。（初始所有点度数为偶数，删去一个环之后每个点度数一定仍为偶数，所以每个连通块都是欧拉图。）

QOJ5434

对于每一个点，我们可以在这个点所在的连通块里面找到一个欧拉回路。（初始所有点度数为偶数，删去一个环之后每个点度数一定仍为偶数，所以每个连通块都是欧拉图。）
然后我们考虑把它插进环里。如果插进去，总共的环长不超过 $n-k$，那么就直接插进去。

QOJ5434

对于每一个点，我们可以在这个点所在的连通块里面找到一个欧拉回路。（初始所有点度数为偶数，删去一个环之后每个点度数一定仍为偶数，所以每个连通块都是欧拉图。）

然后我们考虑把它插进环里。如果插进去，总共的环长不超过 $n-k$，那么就直接插进去。

否则我们只能插一部分进去。此时，我们将路径的起点设为这个点在环上的下一个点，然后在环上绕一圈（可能经过已经插入的回路）之后回到这个点，然后沿着这个新的回路走，直到长度达到 $n-k$，就得到了答案。

总复杂度 $O(n)$。

CF2026E

给定一个长度为 n 的序列 $a_{1\cdots n}$，选择一个子序列，使得这个子序列的长度减去子序列中所有数的 OR 和的 popcount 最小。

$n \leq 100$，$a_i < 2^{60}$。

CF2026E

序列建一列点，每一个二进制位建一个点，一个元素向所有二进制下值为 1 的位连边，求最大权闭合子图。

强化限制 / 只利用一部分信息

有些题目会给我们很大的操作自由度，或者很多的信息。这可能反而会导致我们无从下手。
遇到这些情况时，我们可以考虑强化限制，并尝试设计做法；当遇到问题时，再逐步移除增强的限制。

P6185

一个长度为 n 的序列，可以对它进行 m 种操作：第 i 种操作由三元组 (t_i, u_i, v_i) 描述：如果 $t_i = 1$，则可以对序列的 u_i 位置和 v_i 位置同时 $+1$ 或同时 -1；如果 $t_i = 2$，则可以对序列的 u_i 位置和 v_i 位置中的一个 $+1$，另一个 -1。
给定长为 n 的序列 a, b 和 m 种操作，判定是否能够将 a 通过若干次操作变成 b。
$n, m \leq 10^5$。

P6185

之前我们已经将问题转化为了只有 $t = 1$ 的形式，我们需要将一个序列调整成另一个序列。
仍然考虑建图，连通块之间还是独立的，所以我们对每一个连通块分别处理。

P6185

之前我们已经将问题转化为了只有 $t = 1$ 的形式，我们需要将一个序列调整成另一个序列。

仍然考虑建图，连通块之间还是独立的，所以我们对每一个连通块分别处理。

注意到：对于每一个连通块，如果我们只保留它的一棵生成树，那么我们已经能够将其调整到最多只有一个点不满足要求：取定一个根，从孩子向上调整，则只有根可能无法调整到目标值。

P6185

之前我们已经将问题转化为了只有 $t = 1$ 的形式，我们需要将一个序列调整成另一个序列。

仍然考虑建图，连通块之间还是独立的，所以我们对每一个连通块分别处理。

注意到：对于每一个连通块，如果我们只保留它的一棵生成树，那么我们已经能够将其调整到最多只有一个点不满足要求：取定一个根，从孩子向上调整，则只有根可能无法调整到目标值。

为了解决这个问题，我们考虑用不在树上的边预先调整权值，使得根最终恰好被调整到目标值。通过一些计算可以发现：如果存在一条非树边，其连接树上两个深度奇偶性相同的点，则它可以将根的权值 +2/-2。直接用这个判即可。复杂度 $O(n + m)$。

经典问题

对于一棵 n 个点，边带正权的树，对所有 $1 \leq u, v \leq n$ 给定 u, v 之间的距离，构造一棵符合条件的树。

$n \leq 1000$。

经典问题

信息很多，一点一点考虑。

经典问题

信息很多，一点一点考虑。
首先考虑整个矩阵里的最小值，它一定是一条直接连接两个点的边。

经典问题

信息很多，一点一点考虑。
首先考虑整个矩阵里的最小值，它一定是一条直接连接两个点的边。
再考虑次小值，它一定也是。

经典问题

信息很多，一点一点考虑。
首先考虑整个矩阵里的最小值，它一定是一条直接连接两个点的边。
再考虑次小值，它一定也是。
再大，它可能会是上面两条边的和，也可能是一条新边。注意到如果加入它改变连通性，那么它就是树边。

经典问题

信息很多，一点一点考虑。
首先考虑整个矩阵里的最小值，它一定是一条直接连接两个点的边。
再考虑次小值，它一定也是。
再大，它可能会是上面两条边的和，也可能是一条新边。注意到如果加入它改变连通性，那么它就是树边。
然后考虑更大的边，发现它满足相同的性质。于是我们直接求最小生成树，判断其是否合法即可。使用 Prim，复杂度 $O(n^2)$。

增量法 / 递归构造

增量法是一个非常有用的方法，在合适的时候可以极大简化问题。
类似于数学归纳法的思想，我们每次考虑问题的一部分，每次向其中引入一个新的元素。
递归构造就是反过来，通过调整 $O(1)$ 个元素 / 分治等方法，转化为规模更小的子问题。

EI 的一道题

给定一个 n 个点的无向完全图，每条边带 0/1 权值。
构造一条路径，使得它经过所有 n 个点恰好一次，且它经过的所有 0 权边都在路径的一个前缀内。
$n \leq 2000$。

EI 的一道题

考虑逐渐加点。

EI 的一道题

考虑逐渐加点。
加点时，考虑新点和原路径的 0/1 交界处的边。如果这条边权值是 0，那么断掉与交界点相邻的 1 边，将新点在切断处插入路径。
如果权值是 1 也是同理的。
使用链表维护即可 $O(n)$ 对一个起点构造路径。

CF1558C

给定一个长度为奇数的排列 $a_1, a_2, \ldots, a_n$，你需要构造一组长度不超过的 $\frac{5}{2}n$ 的操作序列 $s_1, s_2, \ldots, s_k$，使得：

- $1 \le s_i \le n$，s_i 为奇数；
- 按从前往后的顺序，对于每个 s_i，反转排列的前 s_i 项，最后得到的排列中 $a_i = i$。

无解输出 -1。

$n \le 2021$。

CF1558C

考虑每次复位 n 和 $n-1$，并对剩下的部分递归。

CF1558C

考虑每次复位 n 和 $n-1$，并对剩下的部分递归。
首先我们把 n 和 $n-1$ 放到一起，也就是把 $n-1$ 转到头，再转到 n 左边。
然后只需要把 $n-1$ 和 n 转到头，再整体翻转到最后，就完成了 $n-1, n$ 的复位，接下来递归即可。

CF1558C

考虑每次复位 n 和 $n-1$，并对剩下的部分递归。
首先我们把 n 和 $n-1$ 放到一起，也就是把 $n-1$ 转到头，再转到 n 左边。
然后只需要把 $n-1$ 和 n 转到头，再整体翻转到最后，就完成了 $n-1, n$ 的复位，接下来递归即可。
上面有任何一步做不到即无解，因为每个元素最终位置的奇偶性不会改变。

如果题目中的变换方式 / 信息形式很复杂

尝试构造一些更简单的变换（例如交换两个元素），或者组合一些信息以得到更简单、更容易处理的限制形式。

CF1696F

对于一棵 n 个点、边权均为 1 的树，对所有 $1 \leq i, j, k \leq n$，已知 $\mathrm{dis}(i, j) = \mathrm{dis}(i, k)$ 是否成立。
构造一棵符合要求的树或判断无解。$n \leq 100$。

CF1696F

如果已知 u, v 之间有边，并且满足 $\mathrm{dis}(u, v) = \mathrm{dis}(v, w)$，那么 v, w 之间有边。

CF1696F

如果已知 u, v 之间有边，并且满足 $\mathrm{dis}(u, v) = \mathrm{dis}(v, w)$，那么 v, w 之间有边。
所以我们如果已知一条边 u, v，那么我们可以利用这个信息来通过类似 BFS 的方式求出整个树。

CF1696F

如果已知 u, v 之间有边，并且满足 $\mathrm{dis}(u, v) = \mathrm{dis}(v, w)$，那么 v, w 之间有边。
所以我们如果已知一条边 u, v，那么我们可以利用这个信息来通过类似 BFS 的方式求出整个树。
于是我们枚举 1 的一条出边 $1, x$，从它开始搜出一棵树。如果可以搜出一棵树，就 $O(n^3)$ 暴力检查它是否满足所有的约束。总复杂度即为 $O(n^4)$。

CF1764F

较复杂，见原题面。

CF1764F

看到这个形式其实就可以猜到这个 $f(i,j)$ 应该可以通过一些方式推出树上 i,j 之间的距离。

CF1764F

看到这个形式其实就可以猜到这个 $f(i,j)$ 应该可以通过一些方式推出树上 i,j 之间的距离。

事实上也是这样的：

$$\mathrm{dis}(i,j)=\frac{f(i,i)+f(j,j)-2f(i,j)}{n}$$

然后就变成了上面讲过的经典问题，直接做即可。

杂项

一些其他类型的题的解题思路也是可以使用的。
例如从简单情况入手、感性理解、调整法等。

CF1444D

给定 $a_1, a_2, \cdots, a_n$ 和 $b_1, b_2, \cdots, b_n$。
将长度分别为 $a_1, a_2, \cdots, a_n$ 的平行于 x 轴的线段和长度分别为 $b_1, b_2, \cdots, b_n$ 的平行于 y 轴的线段首尾相接形成一个环，要求线段之间只在线段端点上相交，并且横向线段只和纵向线段相交，纵向线段只和横向线段相交。
$n, a \leq 1000$。

CF1444D

直觉是尽量往外凸，但是只是一个直觉。

CF1444D

直觉是尽量往外凸，但是只是一个直觉。
手玩一下可以发现：对于一个图形，可能的调整是将一段切出来旋转 180 度再放回原位。
从而最终的答案一定是 4 段线，每一段分别只向左上/左下/右上/右下走。

CF1444D

下面我们考虑如何构造一段线，以右上为例。
加强一下限制，假设左边是竖线，右边是横线，我们不妨让整个线必须在两个端点的连线上面。
那么我们的一个很简单的直觉是尽量先向上走再向右走。为了实现这个，我们把竖线长度降序排序，横线长度升序排序接起来。

CF1444D

下面我们考虑如何构造一段线，以右上为例。
加强一下限制，假设左边是竖线，右边是横线，我们不妨让整个线必须在两个端点的连线上面。
那么我们的一个很简单的直觉是尽量先向上走再向右走。为了实现这个，我们把竖线长度降序排序，横线长度升序排序接起来。
然后我们严谨分析一下，发现这样是对的……？

CF1444D

下面我们考虑把线段划分到折线里面。
首先我们要把横线和竖线划分为两个总长相等的集合。
假设横线划分为 $\{a_1, a_2, \cdots, a_p\}$ 和 $\{a_{p+1}, \cdots, a_n\}$；竖线划分为 $\{b_1, b_2, \cdots, b_q\}$ 和 $\{b_{q+1}, \cdots, b_n\}$。不妨假设 $p \leq q$，另一个方向对称。
那么首先我们把 $\{a_1, \cdots, a_p\}$ 和 $\{b_1, \cdots, b_p\}$ 按照上面的排序连接起来成为一段凸的折线，$\{a_{q+1}, \cdots, a_n\}$ 和 $\{b_{q+1}, \cdots, b_n\}$ 也按照上面的排序连接起来成为一段凸的折线，然后我们发现空出来的区域就允许我们将 $a_{p+1}, \cdots, a_q$ 和 $b_{p+1}, \cdots, b_q$ 任意连接起来了。
划分为和相等的两个集合一步使用 bitset 优化背包，复杂度 $O(n^2v/w)$。

Luogu P5811

给定一个 $a+b+c$ 个点，m 条边的连通图，要求将其划分为三个点集，大小分别为 a,b,c，且这三个点集的导出子图至少有两个连通，构造方案或判断无解。

$m \leq 2 \times 10^5$。

Luogu P5811

先推一些转化。注意到整个图连通，所以原问题等价于我们要将图划分为两个连通块 V_1, V_2，使得一个的大小至少为 a，另一个的大小至少为 b。
推不等式，转化为 $a \le |V_1| \le n - b$ 或 $b \le |V_1| \le n - a$。又由于 $b \le c < n - b$，故等价于 $a \le |V_1| \le n - a$。

Luogu P5811

先推一些转化。注意到整个图连通，所以原问题等价于我们要将图划分为两个连通块 V_1, V_2，使得一个的大小至少为 a，另一个的大小至少为 b。
推不等式，转化为 $a \le |V_1| \le n - b$ 或 $b \le |V_1| \le n - a$。又由于 $b \le c < n - b$，故等价于 $a \le |V_1| \le n - a$。
建圆方树，那么在树上任意断一条边都可以将图划分为两个连通块。如果可以满足大小限制，那么我们就做完了。

Luogu P5811

如果不能满足大小限制，则存在一个点，使得以这个点为根时，所有子树都只含有 $< n - a$ 个圆点。
如果这个点本身是一个圆点，则我们分析一下容易发现这对应无解情况。
如果这个点是一个方点，那么我们考虑将它定为根然后继续细分。

Luogu P5811

如果不能满足大小限制，则存在一个点，使得以这个点为根时，所有子树都只含有 $< n - a$ 个圆点。
如果这个点本身是一个圆点，则我们分析一下容易发现这对应无解情况。
如果这个点是一个方点，那么我们考虑将它定为根然后继续细分。
我们考虑这个方点对应的 vBCC。我们对于每一个点赋予权值，一个点的权值为它子树内圆点的个数。然后我们在该 vBCC 内建 DFS 树。

Luogu P5811

那么这样我们又有了一次断边的机会。如果仍然不能满足大小限制，那么相当于我们在 dfs 树里面存在一个点，这个点的所有子树的点权和 $< a$，但是这个点的子树点权和 $> n - a$，设这个点是 x。

Luogu P5811

那么这样我们又有了一次断边的机会。如果仍然不能满足大小限制，那么相当于我们在 dfs 树里面存在一个点，这个点的所有子树的点权和 $< a$，但是这个点的子树点权和 $> n - a$，设这个点是 x。

考虑调整，注意到现在我们处理的是一个 vBCC 的 DFS 树，因此 x 的每一个子树里面都至少有一条连向 x 的祖先的返祖边（否则 x 就是割点）。那么我们的做法就是每次把一个子树从 x 下面移动到 x 上面。

Luogu P5811

那么这样我们又有了一次断边的机会。如果仍然不能满足大小限制，那么相当于我们在 dfs 树里面存在一个点，这个点的所有子树的点权和 $< a$，但是这个点的子树点权和 $> n - a$，设这个点是 x。

考虑调整，注意到现在我们处理的是一个 vBCC 的 DFS 树，因此 x 的每一个子树里面都至少有一条连向 x 的祖先的返祖边（否则 x 就是割点）。那么我们的做法就是每次把一个子树从 x 下面移动到 x 上面。

注意到这样每次 x 的子树的点权和至多减少 $a - 1 < \frac{1}{3}n$，而 x 的子树的合法点权和范围大小是 $n - 2a + 1 > \frac{1}{3}n$，因此在调整过程中一定有一个时刻合法。

这样我们就做完了，复杂度 $O(n + m)$。

ARC175E

给定 n, k，考虑一个正方体，其被均匀划分为 $n \times n \times n$ 个小正方体。构造一种方案恰好保留其中的 k 个，使得保留的图形的三视图面积均为 k 且形状相同。

$n \leq 500, k \leq n^2$。

ARC175E

从简单情况入手。我们先不管 k，我们该如何构造出一个不存在遮挡且三视图形状相同的图形？

ARC175E

k 的取值范围有点大，我们考虑一个递归构造的方法。注意到如果 $k \leq (n-1)^2+1$，那么我们可以在右上角放一个转化为 $n-1, k-1$ 的问题。
那么我们只需要考虑 $k > (n-1)^2+1$ 的情况，考虑用之前构造出来的东西覆盖它。

试看看!

现在我们可以来尝试真正在考场上出现过的有区分度的构造了。

Final Boss 1：NOIP 2022 T2 meow

见 Luogu P8866。

meow

猜测一定有解，首先考虑 $k = 2n - 2$。
加强限制，尝试让每一个栈里面最多只有 2 个元素，且每一种颜色最多只有一个在栈里。

meow

猜测一定有解，首先考虑 $k = 2n - 2$。
加强限制，尝试让每一个栈里面最多只有 2 个元素，且每一种颜色最多只有一个在栈里。
注意到我们可以钦定一个栈是空的，此时如果新的元素的颜色在栈中没有出现，则直接随便扔到没钦定的位置里面；否则不管栈里面那个在栈底还是栈顶都可以被消去。

meow

猜测一定有解，首先考虑 $k = 2n - 2$。
加强限制，尝试让每一个栈里面最多只有 2 个元素，且每一种颜色最多只有一个在栈里。
注意到我们可以钦定一个栈是空的，此时如果新的元素的颜色在栈中没有出现，则直接随便扔到没钦定的位置里面；否则不管栈里面那个在栈底还是栈顶都可以被消去。
接下来是 $k = 2n - 1$。我们从这个思路开始推广，仍然尝试满足上面那个条件，但是在栈里面已经出现了 $2n - 2$ 种颜色，再加入最后一种颜色的时候会有问题。

meow (cont.)

分别尝试两种方案：将新元素放到钦定的空栈里面，以及将新元素放到一个栈顶（使这个栈里有 3 个元素）。

meow (cont.)

分别尝试两种方案：将新元素放到钦定的空栈里面，以及将新元素放到一个栈顶（使这个栈里有 3 个元素）。
首先尝试第一种，但是如果在产生一个空栈之前我们遇到另外一个在栈底且上面有其他元素的元素就会彻底爆掉。这时用第二种方案可以构造出来。

meow (cont.)

分别尝试两种方案：将新元素放到钦定的空栈里面，以及将新元素放到一个栈顶（使这个栈里有 3 个元素）。
首先尝试第一种，但是如果在产生一个空栈之前我们遇到另外一个在栈底且上面有其他元素的元素就会彻底爆掉。这时用第二种方案可以构造出来。
于是再尝试第二种，但是如果在遇到包含 3 个元素的这个栈的栈顶或者栈底元素之前遇到了中间的那个元素也会彻底爆掉。

meow (cont.)

分别尝试两种方案：将新元素放到钦定的空栈里面，以及将新元素放到一个栈顶（使这个栈里有 3 个元素）。
首先尝试第一种，但是如果在产生一个空栈之前我们遇到另外一个在栈底且上面有其他元素的元素就会彻底爆掉。这时用第二种方案可以构造出来。
于是再尝试第二种，但是如果在遇到包含 3 个元素的这个栈的栈顶或者栈底元素之前遇到了中间的那个元素也会彻底爆掉。
但是你发现此时用第一种方案就可以构造出来……？

meow (cont.)

我们再随意尝试一些情况，会发现第一种方案和第二种方案里面总有一个可以构造出来。于是我们不难想到把两个方案拼在一起。

meow (cont.)

我们再随意尝试一些情况，会发现第一种方案和第二种方案里面总有一个可以构造出来。于是我们不难想到把两个方案拼在一起。

具体来说，我们考虑这个元素后面第一个在栈底或颜色与当前元素相同的元素 x。如果 x 与当前元素相同，那么我们直接把当前元素放到空栈里面，然后中间的元素都在栈顶从而都可以消去，x 和当前元素消去之后保持每个栈大小 ≤ 2，栈内元素互不相同，且存在空栈。

meow (cont.)

如果 x 与当前元素不同，我们需要再考虑 x 所在的栈的栈顶元素 y：如果中间的元素里面 y 有奇数个，那么将当前元素放进空栈，中间元素放完之后 x 所在的栈仅有一个元素，从而在加入另一个 x 之后得到空栈。

meow (cont.)

如果 x 与当前元素不同，我们需要再考虑 x 所在的栈的栈顶元素 y：如果中间的元素里面 y 有奇数个，那么将当前元素放进空栈，中间元素放完之后 x 所在的栈仅有一个元素，从而在加入另一个 x 之后得到空栈。
如果 y 有偶数个，那么将当前元素放在 y 上面。后面遇到其他栈顶元素就放到对应栈顶，遇到 y 放进空栈。这样最后空栈一定还是空的，放一个 x 进去之后可以消掉之前的 x，使得最终状态保持上面的条件。
总复杂度可以做到 $O(n+m)$。

Final Boss 2：联合省选 2021 matrix

见 Luogu P7515。

matrix

如果我们没有这个 $[0, 10^6]$ 的限制，那么我们直接给最后一行和最后一列任意赋值即可推出整个矩阵。
所以实际上这个题要求我们做的是合理设定最后一行和最后一列的值，使得最终递推出的所有值在 $[0, 10^6]$ 范围内。

matrix

如果我们没有这个 $[0, 10^6]$ 的限制，那么我们直接给最后一行和最后一列任意赋值即可推出整个矩阵。
所以实际上这个题要求我们做的是合理设定最后一行和最后一列的值，使得最终递推出的所有值在 $[0, 10^6]$ 范围内。
不等式形式的限制，不难想到差分约束。但是这里每一个位置涉及到三个变量，不能直接差分约束。

matrix (cont.)

$$\begin{bmatrix} z & x_1 & x_2 \\ y_1 & C - x_1 - y_1 - z & C - x_2 + y_1 + z \\ y_2 & C + x_1 - y_2 + z & C + x_2 + y_2 - z \end{bmatrix}$$

matrix (cont.)

$$\begin{bmatrix} z & x_1 & x_2 \\ y_1 & C - x_1 - y_1 - z & C - x_2 + y_1 + z \\ y_2 & C + x_1 - y_2 + z & C + x_2 + y_2 - z \end{bmatrix}$$

注意到对于同一个 i，x_i 与 z 带的符号要么对所有 j 全部相同，要么对所有 j 全部不同。从而我们可以想到合并 x_i 和 z：设 $x'_1 = z + x_1, x'_2 = z - x_2$，引入限制 $x'_1 - z \in [0, 10^6]$ 和 $z - x'_2 \in [0, 10^6]$。均可用差分约束处理。

matrix (cont.)

$$\begin{bmatrix} z & x'_1 - z & z - x'_2 \\ y_1 & C - x'_1 - y_1 & C + x'_2 + y_1 \\ y_2 & C + x'_1 - y_2 & C - x'_2 + y_2 \end{bmatrix}$$

但是我们发现这样还不能用差分约束做，因为有一些地方是同号的。

matrix (cont.)

$$\begin{bmatrix} z & x'_1 - z & z - x'_2 \\ y_1 & C - x'_1 - y_1 & C + x'_2 + y_1 \\ y_2 & C + x'_1 - y_2 & C - x'_2 + y_2 \end{bmatrix}$$

但是我们发现这样还不能用差分约束做，因为有一些地方是同号的。
但是我们注意到：对于每一行，要么所有元素都是同号的，要么所有元素都是异号的!
从而我们不难想到设 $y'_1 = -y_1$，$y'_2 = y_2$，改为限制 $y'_1 \in [-10^6, 0]$，仍然可以用差分约束处理。

matrix (cont.)

$$\begin{bmatrix} z & x'_1 - z & z - x'_2 \\ -y'_1 & C - x'_1 + y'_1 & C + x'_2 - y'_1 \\ y'_2 & C + x'_1 - y'_2 & C - x'_2 + y'_2 \end{bmatrix}$$

这样我们就终于可以用差分约束做了。图中包含 $O(n)$ 个点和 $O(n^2)$ 条边，从而时间复杂度为 $O(n^3)$。

Final Boss 3：NOIP 2020 T3 ball

见 Luogu P7115。

ball

有个 $n = 2$ 的部分分，那么我们不妨先来做这个部分分。我们考虑把两个 $0/1$ 序列分别排序，然后可以直接用 $O(m)$ 次操作将一个全部变成 0，另一个全部变成 1。

ball

有个 $n = 2$ 的部分分，那么我们不妨先来做这个部分分。我们考虑把两个 0/1 序列分别排序，然后可以直接用 $O(m)$ 次操作将一个全部变成 0，另一个全部变成 1。排序也是好做的：假设我们要将序列 1 排序，设序列 1 里面有 x 个 1，则从另一个序列（称为序列 2）里面移动 x 个元素到空序列（称为序列 3）内，然后依次考虑序列 1 的每一个元素。如果是 1 则放入序列 2，如果是 0 则放入序列 3。清空序列 1 后，再将刚刚移出的所有序列 1 内元素依次倒回到序列 1 里面得到一个有序的序列 1，再将序列 3 内的元素倒回到序列 2 里面，序列 2 不会发生改变。

ball (cont.)

$$\begin{matrix}[0,1,0,0,0,1]\\[0,1,1,1,0,1]\\[]\end{matrix} \Rightarrow \begin{matrix}[0,1,0,0,0,1]\\[0,1,1,1]\\[1,0]\end{matrix} \Rightarrow \begin{matrix}[]\\[0,1,1,1,1,1]\\[1,0,0,0,0,0]\end{matrix}$$

$$\Rightarrow \begin{matrix}[0,0,0,0,1,1]\\[0,1,1,1]\\[1,0]\end{matrix} \Rightarrow \begin{matrix}[0,0,0,0,1,1]\\[0,1,1,1,0,1]\\[]\end{matrix}$$

ball (cont.)

推广到 $n > 2$，我们不难想到分治。每次将颜色集合折半，一半记为 0，另一半记为 1。每次将所有 0 和所有 1 分开，再对 $0, 1$ 分别递归，直到只剩一种颜色。
排序部分是完全一样的，我们只需要考虑如何将 k 个已经排序好的 $0/1$ 序列分离。注意到任选两个序列并使用原来的分离方式，即使 $0, 1$ 的个数不同，我们仍然可以让其中一个序列变为全 0 或全 1，并使得问题规模减少 1。于是分离 k 个序列只需要 $O(km)$ 次操作。
总操作次数 $O(nm \log n)$，无需卡常即可通过。

Final Boss 4：ICPC 上海站 H

给定 n。使用一个夹角为 120° 的 V 形覆盖一个边长为 n 的三角形，并使用 26 种颜色对于所有 V 形染色，要求相邻的 V 形颜色不同。需要放入尽量多的 V 形。$n \leq 1000$。

ICPC 上海站 H

注意到一个 V 不会和很多个 V 相邻，猜测染色是简单的。

画几个小的试一试。注意到 $\frac{n(n+1)}{2}$ 模 3 余 0 或 1，猜测余 0 时有 3 个空位，余 1 时有 4 个空位。

ICPC 上海站 H

注意到一个 V 不会和很多个 V 相邻，猜测染色是简单的。

画几个小的试一试。注意到 $\frac{n(n+1)}{2}$ 模 3 余 0 或 1，猜测余 0 时有 3 个空位，余 1 时有 4 个空位。

同时我们注意到，在构造的过程中，填满原来的三角形的一个 60 度的角是一个不太好操作的事情，同时我们有 3 或 4 个空位，所以我们不妨把所有角空掉，这样要填的图形都是对称的。

ICPC 上海站 H

注意到一个 V 不会和很多个 V 相邻，猜测染色是简单的。

画几个小的试一试。注意到 $\frac{n(n+1)}{2}$ 模 3 余 0 或 1，猜测余 0 时有 3 个空位，余 1 时有 4 个空位。

同时我们注意到，在构造的过程中，填满原来的三角形的一个 60 度的角是一个不太好操作的事情，同时我们有 3 或 4 个空位，所以我们不妨把所有角空掉，这样要填的图形都是对称的。

考虑增量法。由于模 3 的值构成长为 3 的周期，我们考虑一次加三行。n 小的情况比较特殊，我们考虑从 $n = 4, 5, 6$ 分别向后加。

ICPC 上海站 H

注意到两个 V 可以拼成一个大小为 2×3 的平行四边形。经过多次尝试可以得到构造方式。
染色可以直接贪心。复杂度 $O(n^2)$。

更多真题

联合省选 2022 academic、联合省选 2023 game、CTT 2023 若干个题……
大家可以自行挑战。