2023 年中国大学生程序设计竞赛全国邀请赛（湖南）暨第十三届湘潭市大学生程序设计竞赛 题解

2023 National Invitational of CCPC (Hunan) & The 13th Xiangtan Collegiate Programming Contest

电子科技大学

UESTC

2023 年 5 月 28 日

Problem A - Today's Word

题目大意

- 定义 S_n 的构造方式为：

$$S_n = S_{n-1}[0 \ldots \frac{l}{2} - 1] + S_{n-1} + \text{next}(S_{n-1}[\frac{l}{2} \ldots l - 1])$$

- 其中 $+$ 表示字符串拼接，next$(\cdot)$ 表示把字符串中每一个字符都换成下一个字符，z 换成 a。
- 给定一个长度为偶数且只包含小写字母的字符串 S_0，按如上方法构造字符串 $S_{10^{100}}$，问这个字符串长为 m 的后缀是什么。

Problem A - Today's Word

- 考虑当 S_k 长度超过 $2m$ 时，再进行一次操作后，其长度为 m 的后缀只会进行 $\texttt{next}(\cdot)$ 变换，把这个长为 m 的后缀的每个字符变成下一个字符。因此考虑暴力将 S_0 按构造方式操作，求出一个长度大于 $2m$ 的 S_k，取 S_k 长度为 m 的后缀进行 next$(\cdot)$ 变换即可。

Problem A - Today's Word

- 考虑当 S_k 长度超过 $2m$ 时，再进行一次操作后，其长度为 m 的后缀只会进行 $\texttt{next}(\cdot)$ 变换，把这个长为 m 的后缀的每个字符变成下一个字符。因此考虑暴力将 S_0 按构造方式操作，求出一个长度大于 $2m$ 的 S_k，取 S_k 长度为 m 的后缀进行 next$(\cdot)$ 变换即可。
- 容易观察到 next$(\cdot)$ 变换操作的循环节为 26，因此只需对这个后缀进行 $(10^{100} - k) \bmod 26$ 次 next$(\cdot)$ 变换即可，也就是将这个长为 m 的后缀中的每个字母变为它后面第 $(10^{100} - k) \bmod 26$ 个字母，经过简单计算即可求出。

Problem A - Today's Word

- 考虑当 S_k 长度超过 $2m$ 时，再进行一次操作后，其长度为 m 的后缀只会进行 $\texttt{next}(\cdot)$ 变换，把这个长为 m 的后缀的每个字符变成下一个字符。因此考虑暴力将 S_0 按构造方式操作，求出一个长度大于 $2m$ 的 S_k，取 S_k 长度为 m 的后缀进行 next$(\cdot)$ 变换即可。
- 容易观察到 next$(\cdot)$ 变换操作的循环节为 26，因此只需对这个后缀进行 $(10^{100} - k) \bmod 26$ 次 next$(\cdot)$ 变换即可，也就是将这个长为 m 的后缀中的每个字母变为它后面第 $(10^{100} - k) \bmod 26$ 个字母，经过简单计算即可求出。
- 考虑时间复杂度，由于操作一次将导致字符串长度翻倍，因此暴力构造的复杂度是可以接受的。实际上，时间复杂度可估计为 $\mathcal{O}(m \log m)$，空间复杂度 $\mathcal{O}(m)$。

Problem B - Honkai in TAIKULA

题目大意

- 给定一张 n 个节点 m 条边的有向带权图，对每个节点，若不存在奇权圈（可经过重复顶点、重复边，若经过重复边，边权计多次），输出 `Battle with the crazy Honkai`，否则，输出最小奇权圈的权值（负无穷输出 `Haha, stupid Honkai`）。

Problem B - Honkai in TAIKULA

- 首先用 tarjan 算法将图分为若干个强连通子图，针对每个强连通子图分别进行计算。

Problem B - Honkai in TAIKULA

- 首先用 tarjan 算法将图分为若干个强连通子图，针对每个强连通子图分别进行计算。
- 给定一个强连通子图，构造新的图，每个点 x 拆分为两个节点 Odd_x 和 $Even_x$，对一条权值为 w 的边 (x, y)，若 w 是奇数，添加权值为 w 的 $(Odd_x, Even_y)$ 和 $(Even_x, Odd_y)$ 两条边，否则添加权值为 w 的 (Odd_x, Odd_y) 和 $(Even_x, Even_y)$ 两条边，在新图上用 Bellman ford 之类的算法判断是否存在负权圈：

Problem B - Honkai in TAIKULA

- 首先用 tarjan 算法将图分为若干个强连通子图，针对每个强连通子图分别进行计算。
- 给定一个强连通子图，构造新的图，每个点 x 拆分为两个节点 Odd_x 和 $Even_x$，对一条权值为 w 的边 (x, y)，若 w 是奇数，添加权值为 w 的 $(Odd_x, Even_y)$ 和 $(Even_x, Odd_y)$ 两条边，否则添加权值为 w 的 (Odd_x, Odd_y) 和 $(Even_x, Even_y)$ 两条边，在新图上用 Bellman ford 之类的算法判断是否存在负权圈：
  1. 如果存在负权圈，则对每个节点 x 判断 Odd_x 能否到达 $Even_x$，若能，输出 `Haha, stupid Honkai`，否则输出 `Battle with the crazy Honkai`

Problem B - Honkai in TAIKULA

- 首先用 tarjan 算法将图分为若干个强连通子图，针对每个强连通子图分别进行计算。
- 给定一个强连通子图，构造新的图，每个点 x 拆分为两个节点 Odd_x 和 $Even_x$，对一条权值为 w 的边 (x, y)，若 w 是奇数，添加权值为 w 的 $(Odd_x, Even_y)$ 和 $(Even_x, Odd_y)$ 两条边，否则添加权值为 w 的 (Odd_x, Odd_y) 和 $(Even_x, Even_y)$ 两条边，在新图上用 Bellman ford 之类的算法判断是否存在负权圈：
  1. 如果存在负权圈，则对每个节点 x 判断 Odd_x 能否到达 $Even_x$，若能，输出 `Haha, stupid Honkai`，否则输出 `Battle with the crazy Honkai`
  2. 如果不存在负权圈，使用 Johnson 算法跑全源最短路，对每个节点 x，若 Odd_x 能到达 $Even_x$，输出路径长度，否则输出 `Battle with the crazy Honkai`。

Problem B - Honkai in TAIKULA

- 首先用 tarjan 算法将图分为若干个强连通子图，针对每个强连通子图分别进行计算。
- 给定一个强连通子图，构造新的图，每个点 x 拆分为两个节点 Odd_x 和 $Even_x$，对一条权值为 w 的边 (x, y)，若 w 是奇数，添加权值为 w 的 $(Odd_x, Even_y)$ 和 $(Even_x, Odd_y)$ 两条边，否则添加权值为 w 的 (Odd_x, Odd_y) 和 $(Even_x, Even_y)$ 两条边，在新图上用 Bellman ford 之类的算法判断是否存在负权圈：
  1. 如果存在负权圈，则对每个节点 x 判断 Odd_x 能否到达 $Even_x$，若能，输出 `Haha, stupid Honkai`，否则输出 `Battle with the crazy Honkai`
  2. 如果不存在负权圈，使用 Johnson 算法跑全源最短路，对每个节点 x，若 Odd_x 能到达 $Even_x$，输出路径长度，否则输出 `Battle with the crazy Honkai`。
- 时间复杂度：$O(nm \log m)$

Problem C - GG and YY's Game

题目大意

- 给定一张由若干不相交路径组成的图，两个玩家在图上交替进行博弈，在每轮操作时，对应玩家选择图中一个仍然存在的点，取走该点以及距离该点不超过 t 条边距离的所有的点。
- 令 cnt_{GG}、cnt_{YY} 分别为 GG 和 YY 在博弈结束时取走的节点数，双方均想取走尽可能多的节点，问双方均采用最优策略下的博弈结果：对 $t = 1$ 回答 $cnt_{GG} - cnt_{YY}$，对 $t > 1$ 回答胜者或平局。

Problem C - GG and YY's Game

- 首先判定博弈结果，令 $G = \{l_1, l_2, \ldots, l_n\}$ 为给定的图（这里用路径长度指代对应的路径），不难发现若 G 中存在平局局面 $\{l_{a_1}, l_{a_2}, \ldots, l_{a_k}\}$，去除这些子图后，$G \setminus \{l_{a_1}, l_{a_2}, \ldots, l_{a_k}\}$ 的博弈结果和 G 的博弈结果一致，所以考虑找出所有的平局局面从而完成判定。

Problem C - GG and YY's Game

- 首先判定博弈结果，令 $G = \{l_1, l_2, \ldots, l_n\}$ 为给定的图（这里用路径长度指代对应的路径），不难发现若 G 中存在平局局面 $\{l_{a_1}, l_{a_2}, \ldots, l_{a_k}\}$，去除这些子图后，$G \setminus \{l_{a_1}, l_{a_2}, \ldots, l_{a_k}\}$ 的博弈结果和 G 的博弈结果一致，所以考虑找出所有的平局局面从而完成判定。
- 对 $t = 1$，若 $G \in (\{\{a, a\} \mid a \in N^+\} \cup \{\{1, 3, 4\}, \{3, 5, 8\}, \{1, 4, 5, 8\}\})^*$，则打平，否则先手必胜，对 $t > 1$，若 $G \in (\{\{a, a\} \mid a \in N^+\} \cup \{\{1, 2t + 1, 2t + 2\}\})^*$，则打平，否则先手必胜。

Problem C - GG and YY's Game

对于 $t = 1$ 分析结果：

Problem C - GG and YY's Game

对于 $t = 1$ 分析结果：

- 如果 G 包含偶数个节点，若先手必胜，则 $cnt_{GG} - cnt_{YY} = 2$，否则 $cnt_{GG} - cnt_{YY} = 0$，

Problem C - GG and YY's Game

对于 $t = 1$ 分析结果：

- 如果 G 包含偶数个节点，若先手必胜，则 $cnt_{GG} - cnt_{YY} = 2$，否则 $cnt_{GG} - cnt_{YY} = 0$，
- 如果 G 包含奇数个节点，令 G' 为 G 去除所有平局子局面后剩余的图，不难发现，$cnt_{GG} - cnt_{YY} = 3$ 的充要条件是先手能够在取走 3 个节点后使得局面变为平局局面，枚举可得，当 $G' \in$ $\{\{2a+3\},\{4,7\},\{3,8\},\{1,10\},\{8,11\},\{5,14\},\{3,16\},\{a,3+a\},\{1,4,6\},\{1,3,7\},\{5,6,8\},\{3,5,11\},\{4,8,9\},\{4,5,12\},\{1,8,12\},\{$ $a+b\},\{1,5,7,8\},\{1,4,5,11\},\{3,4,a,4+a\},\{3,5,a,11+a\},\{1,4,a,6+a\},\{1,3,a,7+a\},\{5,8,a,6+a\},\{3,8,a,8+a\},\{4,5,8,a,4+a\},\{1,5,8,a,7+a\},\{1,4,8,a,8+a\},\{1,4,5,a,11+a\}\}_{a,b \in N}$ 时，$cnt_{GG} - cnt_{YY} = 3$。

Problem C - GG and YY's Game

对于 $t = 1$ 分析结果：

- 如果 G 包含偶数个节点，若先手必胜，则 $cnt_{GG} - cnt_{YY} = 2$，否则 $cnt_{GG} - cnt_{YY} = 0$，
- 如果 G 包含奇数个节点，令 G' 为 G 去除所有平局子局面后剩余的图，不难发现，$cnt_{GG} - cnt_{YY} = 3$ 的充要条件是先手能够在取走 3 个节点后使得局面变为平局局面，枚举可得，当 $G' \in$ $\{\{2a+3\},\{4,7\},\{3,8\},\{1,10\},\{8,11\},\{5,14\},\{3,16\},\{a,3+a\},\{1,4,6\},\{1,3,7\},\{5,6,8\},\{3,5,11\},\{4,8,9\},\{4,5,12\},\{1,8,12\},$ $a+b\},\{1,5,7,8\},\{1,4,5,11\},\{3,4,a,4+a\},\{3,5,a,11+a\},\{1,4,a,6+a\},\{1,3,a,7+a\},\{5,8,a,6+a\},\{3,8,a,8+a\},\{4,5,8,a,4+a\},\{1,5,8,a,7+a\},\{1,4,8,a,8+a\},\{1,4,5,a,11+a\}\}_{a,b \in N}$ 时，$cnt_{GG} - cnt_{YY} = 3$。

最终，排序 + 分类讨论，时间复杂度为 $\mathcal{O}(n \log n)$。

Problem D - Star Rail

题目大意

- 二维平面上有 n 个点，设 $A_{i,j}$ 表示将第 i 个点删除后，将剩下 $n-1$ 个点划分为两个集合后，提取其中点数为 j 的集合，这个集合的种类数。
- 求出该矩阵。

Problem D - Star Rail

- 一个集合如果能被划分出来，那么一定是可以找到一条直线，将这个集合投影在这条直线上，集合内的点严格在集合外的点的左边，而且这条投影的直线的斜率一定是一个连续的区间。

Problem D - Star Rail

- 一个集合如果能被划分出来，那么一定是可以找到一条直线，将这个集合投影在这条直线上，集合内的点严格在集合外的点的左边，而且这条投影的直线的斜率一定是一个连续的区间。
- 那么我们考虑这条直线的斜率变化，会使得投影的点的排列产生变化。每次变化一定反转某几段连续的区间里的点，总的变化次数是 $O(n^2)$。

Problem D - Star Rail

- 考虑对每个位置 i 维护 cnt_i, 表示从一开始到现在产生了多少大小为 i 的不同的集合，那么只有反转的区间时候包含 i 才会增加 cnt_i，这个可以每次 O(1) 维护, 再考虑 cnt_i 对答案的贡献，会使左边的每个点 u 的 $ans_{u,i-1} + cnt_i$，右边每个点 v 的 $ans_{v,i} + cnt_i$, 这部分可以最后计算, 由于每次变化某些点会移动，因此移动的时候要把对应的 cnt_i 的影响加进答案，总复杂度 $O(n^2 \log n)$.

Problem E - LCM Plus GCD

题目大意

- 输入 x, k，求大小为 k 的 distinct 集合个数，其中所有数的 $\gcd + \operatorname{lcm} = x$。

Problem E - LCM Plus GCD

- 假设
  $\mathrm{GCD}(a_1, a_2, \ldots, a_k) = G, \mathrm{LCM}(a_1, a_2, \ldots, a_k) = L = A \times G$,
  那么 $(1 + A) \times G = x$, 枚举 G, 我们需要找到 k 个数
  $a_i' = \frac{a_i}{G}, \mathrm{GCD}(a_1', a_2', \ldots, a_k') = 1, \mathrm{LCM}(a_1', a_2', \ldots, a_k') = A$,
  分解因子，$A = \prod_{j=1}^{m} p_j^{b_j}, a_i' = \prod_{j=1}^{m} p_j^{c_{ij}}$, 于是有
  $\min\{c_{ij}\} = 1, \max\{c_{ij}\} = b_j$ 总共 $2m$ 个约束要满足。

Problem E - LCM Plus GCD

- 假设
  $\mathrm{GCD}(a_1, a_2, \ldots, a_k) = G, \mathrm{LCM}(a_1, a_2, \ldots, a_k) = L = A \times G$,
  那么 $(1 + A) \times G = x$, 枚举 G, 我们需要找到 k 个数
  $a'_i = \frac{a_i}{G}, \mathrm{GCD}(a'_1, a'_2, \ldots, a'_k) = 1, \mathrm{LCM}(a'_1, a'_2, \ldots, a'_k) = A$,
  分解因子，$A = \prod_{j=1}^{m} p_j^{b_j}, a'_i = \prod_{j=1}^{m} p_j^{c_{ij}}$, 于是有
  $\min\{c_{ij}\} = 1, \max\{c_{ij}\} = b_j$ 总共 $2m$ 个约束要满足。
- 考虑容斥，2^{2m} 枚举至少违反了某些约束集合的方案数, 每一次是找 k 个 a'_i 满足 $y_j \le c_{ij} \le z_j$ 的方案数, 假设
  $B = \prod_{j=1}^{m}(z_j - y_j + 1)$，那么方案数为 C_B^k。

Problem E - LCM Plus GCD

- 假设
  $\mathrm{GCD}(a_1, a_2, \ldots, a_k) = G, \mathrm{LCM}(a_1, a_2, \ldots, a_k) = L = A \times G$,
  那么 $(1 + A) \times G = x$, 枚举 G, 我们需要找到 k 个数
  $a'_i = \frac{a_i}{G}, \mathrm{GCD}(a'_1, a'_2, \ldots, a'_k) = 1, \mathrm{LCM}(a'_1, a'_2, \ldots, a'_k) = A$,
  分解因子，$A = \prod_{j=1}^{m} p_j^{b_j}, a'_i = \prod_{j=1}^{m} p_j^{c_{ij}}$, 于是有
  $\min\{c_{ij}\} = 1, \max\{c_{ij}\} = b_j$ 总共 $2m$ 个约束要满足。
- 考虑容斥，2^{2m} 枚举至少违反了某些约束集合的方案数, 每一次是找 k 个 a'_i 满足 $y_j \le c_{ij} \le z_j$ 的方案数, 假设
  $B = \prod_{j=1}^{m}(z_j - y_j + 1)$，那么方案数为 C_B^k。
- 总时间复杂度 $O(Fm2^{2m})$, 其中 F 是 x 的因子数，在 $x \le 10^9$ 时，$F \le 1344, m \le 9$, 实际中由于只有极少的 A 能使得 $m = 9$, 可以满足时限要求.

Problem F - Timaeus

题目大意

你有 A 个原料，每 B 个合成一个产品。每一次合成你可以选择下列两种 buff 的任意一种：

1. $P\%$ 的概率合成双倍产物
2. $Q\%$ 的概率返还一个原料

求期望最多合成多少产物。

Problem F - Timaeus

- 设 $dp(i)$ 表示还剩下 i 个原料时的最大期望。根据题意，容易得到如下 dp 方程：

Problem F - Timaeus

- 设 $dp(i)$ 表示还剩下 i 个原料时的最大期望。根据题意，容易得到如下 dp 方程：

$$
\begin{aligned}
dp(i) = \max\{ & P \times (dp(i-B)+2) + (1-P) \times (dp(i-B)+1), \\
& Q \times (dp(i-B+1)+1) + (1-Q) \times (dp(i-B)+1)\}
\end{aligned}
$$

Problem F - Timaeus

- 设 $dp(i)$ 表示还剩下 i 个原料时的最大期望。根据题意，容易得到如下 dp 方程：

$$\begin{aligned} dp(i) = \max\{&P \times (dp(i-B)+2) + (1-P) \times (dp(i-B)+1), \\ &Q \times (dp(i-B+1)+1) + (1-Q) \times (dp(i-B)+1)\} \end{aligned}$$

- 注意 $B = 1$ 时需要特判。此时只选择其中一种 buff 是最优的，可以分别算出各自的期望值然后取 max 即可。
- 时间复杂度：$O(A)$

Problem G - Moving Boxes

题目大意

- n 个物品在一条直线上，每个需要从 x_i 移动到 y_i，有一个机器人移动，机器人每次只能拿一个物品，速度为 1，转向需要 c 秒，最后需要回到起点，方向回到初始方向。求运输完所有物品需要的最小时间，$n \leq 10^5$

Problem G - Moving Boxes

- 首先，由于物品可以中途放下，所以显然一条长路径可以看做若干单位长度的路径组成。由于范围很大，离散化所有坐标。

Problem G - Moving Boxes

- 首先，由于物品可以中途放下，所以显然一条长路径可以看做若干单位长度的路径组成。由于范围很大，离散化所有坐标。
- 先不考虑转向代价的情况。

Problem G - Moving Boxes

- 首先，由于物品可以中途放下，所以显然一条长路径可以看做若干单位长度的路径组成。由于范围很大，离散化所有坐标。
- 先不考虑转向代价的情况。
- 我们考虑对于每个单位长度，对于考虑有多少路径经过这个长度，设从左往右经过第 i 个段路径个数为 a_i，从右往左的个数为 b_i。一个显然的结论是此时最优答案中，从左往右和从右往左都会经过 $\phi(i) = max(a_i, b_i, 1)$ 次。

Problem G - Moving Boxes

- 再考虑有转向代价的情况。

Problem G - Moving Boxes

- 再考虑有转向代价的情况。
- 考虑第 i 和第 $i+1$ 和第 $i+2$ 个段路径之间，设 $\phi(i) > \phi(i+1) < \phi(i+2)$，如果转向代价为 0，那么这之间会转向 $\phi(i) - \phi(i+1) + \phi(i+2) - \phi(i+1)$ 次（考虑多余的那些路径，会选择转向回去，和反方向的路径接起来）。

Problem G - Moving Boxes

- 再考虑有转向代价的情况。
- 考虑第 i 和第 $i+1$ 和第 $i+2$ 个段路径之间，设 $\phi(i) > \phi(i+1) < \phi(i+2)$，如果转向代价为 0，那么这之间会转向 $\phi(i) - \phi(i+1) + \phi(i+2) - \phi(i+1)$ 次（考虑多余的那些路径，会选择转向回去，和反方向的路径接起来）。
- 如果转向代价无穷大，对于 $i+1$ 这段，会选择多走 $\min(\phi(i+2), \phi(i)) - \phi(i+1)$ 次，使得 $\phi(i+1) = \min(\phi(i), \phi(i+2))$，这样只需要转向 $|\phi(i) - \phi(i+2)|$ 次，但是会多走 $\min(\phi(i+2), \phi(i)) - \phi(i+1)$ 次。

Problem G - Moving Boxes

- 对于一般情况，显然只需要比较两种方案的代价，即比较 $len(i)$ 和 C 的值，选择较小的方案即可。

Problem G - Moving Boxes

- 对于一般情况，显然只需要比较两种方案的代价，即比较 $len(i)$ 和 C 的值，选择较小的方案即可。
- 具体的，对于所有的 $\phi(i)$ 做单调栈，求出左右第一个大于 i 的位置，然后进行比较选择方案计算代价即可。

Problem G - Moving Boxes

- 对于一般情况，显然只需要比较两种方案的代价，即比较 $len(i)$ 和 C 的值，选择较小的方案即可。
- 具体的，对于所有的 $\phi(i)$ 做单调栈，求出左右第一个大于 i 的位置，然后进行比较选择方案计算代价即可。
- 复杂度 $O(n \log n)$

Problem H - Neil's Machine

题目大意

- 有两个长度为 n, 数值为 $0-25$ 的数组 S, T。每次操作可以选择一个正数 k $(1 \leq k \leq 25)$, 并选择 S 的一个后缀加 k 模 26，问至少几次操作可以令 $S=T$。

Problem H - Neil's Machine

- 设 $S_0 = T_0 = 0$, $DS_i = (S_i - S_{i-1} + 26) \bmod 26, DT_i = (T_i - T_{i-1} + 26) \bmod 26$ $(1 \le i \le n)$，则 $S = T$ 等价于 $DS = DT$。每次操作可以将一个 DS_i 修改为 $0 - 25$ 的任意数，故统计 $DS_i \neq DT_i$ 的数量即可。时间复杂度 $O(n)$。

Problem I - Elevator

题目大意

- 电梯里有 n 个人（你是其中一个），这些人想去 m 个楼层，每个人只想去一个楼层。到达一个楼层，所有想去这个楼层的人都会下电梯。问在你想去的楼层最多有多少人下电梯。

Problem I - Elevator

- 考虑鸽巢原理，其中人数最多的情况为：在你不想去的楼层每层只下一个人，在你想去的楼层能下的都下。考虑在 $m-1$ 个不想去的楼层每层分配一个人下梯，那么在想去的楼层一起下梯的就有 $n-(m-1)$ 人。考虑调整法，可以证明这是最多情况。
- 单组数据时间复杂度 $\mathcal{O}(1)$，空间复杂度 $\mathcal{O}(1)$。

Problem J - Similarity (Easy Version)

题目大意

- 给定 n 个串，求两两之间最长公共子串的最大值。

- 直接暴力对 n^2 个二元组 (s_i, s_j) 跑最长公共子串即可，或者考虑暴力枚举匹配也可以通过。
- 复杂度 $O(T \times n^4)$ 或 $O(T \times n^5)$

Problem K - Similarity (Hard Version)

题目大意

- 构造 n 个不同的长度为 k 的只由小写字符组成的字符串，使得其两两之间最长公共子串的最大值等于 m。

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：
  1. $m = 0, n > 26$ 时无解
  2. $m \neq 0, m \geq k$ 时无解

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：
  1. $m = 0, n > 26$ 时无解
  2. $m \neq 0, m \geq k$ 时无解
- 剩下的有解情况可以按如下方式构造：

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：
  1. $m = 0, n > 26$ 时无解
  2. $m \neq 0, m \geq k$ 时无解
- 剩下的有解情况可以按如下方式构造：
  1. $m = 0$ 时，直接输出长度为 k 的只由第 i 个字母组成的小写字母组成的串即可。

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：
  1. $m = 0, n > 26$ 时无解
  2. $m \neq 0, m \geq k$ 时无解
- 剩下的有解情况可以按如下方式构造：
  1. $m = 0$ 时，直接输出长度为 k 的只由第 i 个字母组成的小写字母组成的串即可。
  2. $m \neq 0$ 时，我们考虑构造如下形式的字符串：

$$a_1 a_2 a_1 a_2 a_1 a_2 a_1 a_2 \cdots$$

其中 a_1 和 a_2 分别表示一种不同的小写字母，且 $a_1 < a_2 < z$

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：
  1. $m = 0, n > 26$ 时无解
  2. $m \neq 0, m \geq k$ 时无解
- 剩下的有解情况可以按如下方式构造：
  1. $m = 0$ 时，直接输出长度为 k 的只由第 i 个字母组成的小写字母组成的串即可。
  2. $m \neq 0$ 时，我们考虑构造如下形式的字符串：

$$a_1 a_2 a_1 a_2 a_1 a_2 a_1 a_2 \cdots$$

     其中 a_1 和 a_2 分别表示一种不同的小写字母，且 $a_1 < a_2 < z$
- 那么对于这种类型的串一共有 $\frac{25\times(25-1)}{2} = 300$ 种，两两之间的最长公共子串长度均不超过 1。

Problem K - Similarity (Hard Version)

- 首先特判如下情况无解：
  1. $m = 0, n > 26$ 时无解
  2. $m \neq 0, m \geq k$ 时无解
- 剩下的有解情况可以按如下方式构造：
  1. $m = 0$ 时，直接输出长度为 k 的只由第 i 个字母组成的小写字母组成的串即可。
  2. $m \neq 0$ 时，我们考虑构造如下形式的字符串：

$$a_1 a_2 a_1 a_2 a_1 a_2 a_1 a_2 \cdots$$

  其中 a_1 和 a_2 分别表示一种不同的小写字母，且 $a_1 < a_2 < z$
- 那么对于这种类型的串一共有 $\frac{25\times(25-1)}{2} = 300$ 种，两两之间的最长公共子串长度均不超过 1。
- 那么答案构造即可为，取上述形式的串前 $k - m + 1$ 位，拼上 $m - 1$ 位 z 即可得到满足题目要求的 n 个串。

Problem L - Architect

题目大意

- 给出 n 个长方体，询问其是否拼接成一个 $W \times H \times L$ 的立方体，没有重叠和空隙。

Problem L - Architect - 解法一

- 对于每一个不同的 z 坐标，去看所有长方体在这个坐标下的上下表面的关系，要满足所有上表面之间没有重叠，所有下表面之间没有重叠，上表面的并 $=$ 下表面的并，这些可以通过扫描线 $+$ 线段树判断，复杂度 $O(n\log n)$.

Problem L - Architect - 解法二

- 考虑每个长方体是将整个长方体内 $1 \times 1 \times 1$ 的小格子值 $+1$, 大长方体是 -1, 那么最终每个小格子的值都是 0, 对于任意一个点去求 $(0,0,0) \to (x,y,z)$ 这个长方体的三维前缀和也是 0，对于长方体的 8 个顶点标号 1 或者-1，比如 $a(x_l, y_l, z_l) = 1, a(x_l, y_l, z_r) = -1 \dots$

Problem L - Architect - 解法二

- 考虑每个长方体是将整个长方体内 $1 \times 1 \times 1$ 的小格子值 $+1$, 大长方体是 -1, 那么最终每个小格子的值都是 0, 对于任意一个点去求 $(0,0,0) \to (x,y,z)$ 这个长方体的三维前缀和也是 0，对于长方体的 8 个顶点标号 1 或者-1，比如 $a(x_l, y_l, z_l) = 1, a(x_l, y_l, z_r) = -1 \ldots$
- 考虑求三维前缀和的过程，$S(x,y,z) = \sum_{i=0}^{x} \sum_{j=0}^{y} \sum_{k=0}^{z} a(i,j,k)$, 这种标号方式能恰好使这个长方体内部所有格子 S 增加了 1，于是因为最终所有点的 S 应该为 0，所以所有的 a 应该为 0，判断这个就好，复杂度 $O(n \log n)$.

Problem L - Architect - 解法三

- 考虑每个坐标轴离散化，将大长方体划分成 $L \times W \times H$ 个小格子，对每个格子的尺寸随机赋值 x_i, y_j, z_k，那么大长方体的体积为 $\sum_{i=1}^{L} x_i \times \sum_{j=1}^{W} y_j \times \sum_{k=1}^{L} z_k$ 小长方体的体积和为 $\sum_{c=1}^{n}(\sum_{i=x_{cl}}^{x_{cr}} x_i \times \sum_{j=y_{cl}}^{y_{cr}} y_j \times \sum_{k=z_{cl}}^{z_{cr}} z_k)$，因为 x, y, z 都是随机赋值，那么两个式子相等一定大概率是多项式每一项都相等，类似 hash，因此可以随机 K 次尺寸，如果每次体积都相等，那么就是合法的，复杂度 $O(n \log n + Kn)$