2023 年牛客多校第一场题解

出题人：华中科技大学

1 A Almost Correct

题意：给定长度为 n 的 01 串 s，构造一个排序网络，使得能够将除 s 之外的任意 01 序列正确排序，且 s 无法被正确排序。T 组测试，$1 \le T \le 10^4$，$2 \le n \le 16$。

解法：记 01 串 s 中 0 所在位置的下标集合为 S_0，1 所在位置的下标集合为 S_1（下标从 1 开始）。令 S_0 中最大值为 r，S_1 中最小值为 l（显然 $l \le |S_0| < r$，因为 s 没被排好序）。下简写一次操作为 (x, y) 表示对 a_x 与 a_y 的比较与交换。对 01 序列 t 排序方法如下：

- 将 S_0 中最大值挪到最右边，S_1 中最小值挪到最左边，即执行 $\forall x \in S_0, (x, r)$ 和 $\forall x \in S_1, (l, x)$。
- 对 $\{1, 2, 3, \cdots, n\} - \{l, r\} = \{1, 2, \cdots l-1, l+1, \cdots, r-1, r+1, \cdots, n\}$ 正常排序，即去掉下标 l, r 对其他元素排序。
- 依次执行 $(r, |S_0|+1), (r, |S_0|+2), \ldots, (r, r-1), (r, n), (r, n-1), \ldots, (r, r+1)$ ，$(l, |S_0|), (l, |S_0|-1), \cdots, (l, l+1), (l, 1), (l, 2), \cdots, (l, l-1)$。

记 t 中 0 所在位置的下标集合为 T_0，1 所在位置的下标集合为 T_1。上面的算法满足题目要求，证明如下：

- 当 $t = s$，第一步结束后 $t_l = 1$ 且 $t_r = 0$，显然第一步无效。第二步中 t_l 和 t_r 显然不变，其他元素已经排好序了。第三步只能将 t_l 交换到 $t_{|S_0|}$，将 t_r 交换到 $t_{|S_0|+1}$，因此排序后 t 形如 $0 \ldots 001011 \ldots 1$，在零一分界线 $|S_0|$ 处恰有一对 10，因而未被正确排序；
- 当 $|T_0| \ge |S_0|$ 且 $t \ne s$，S_1 中一定存在一个 0，因此经过第一步操作后 $t_l = 0$，第二步操作后 $|S_0|$ 及其左边全为 0。第三步操作中，若 $t_r = 0$，则 t_r 会被交换到 $t_{|S_0|+1}$；若 $t_r = 1$，则会被交换到 t_n。由于 $t_{|S_0|+1:r-1}t_{r+1:n}$ 在第二步时已经排好序，因此交换后 $t_{|S_0|+1:n}$ 有序，于是整个序列 t 有序；
- 当 $|T_0| < |S_0|$ 时同理。

2 B Anticomplementary Triangle

题意：平面上给一个 n 个点的凸多边形 $P_1P_2 \ldots, P_n$，要求找到 3 个点，使得这三个点构成的三角形的反互补三角形包含所有点（可以在边界上）。$1 \le n \le 10^6$。

解法：乱搞题，目标是找到一个三角形使得面积极大。有很多不知道对不对但是能过的搞法，一种能证明的做法如下：

- 固定 $r = 1$，找到两个点 P_s, P_t $(1 < s < t \le n)$ 使得 $S_{\triangle P_rP_sP_t}$ 最大；
- 尝试向左或者向右挪动 P_r，哪边能让 $S_{\triangle P_rP_sP_t}$ 变大就往哪边挪。P_r 挪到极大值后挪 P_t，P_t 挪到极大值后挪 P_s，P_s 挪到极大值后挪 P_r，直到没有一个点能动。

在第一步结束后，只有 r 能够移动。假设 r 逆时针移动，这之后 r, s, t 就只能逆时针移动，并且 r, s, t 中任何一个点无法重新移动到 P_1，因为 $S_{\triangle P_rP_sP_t}$ 在移动过程中递增，并且第一步得到的 $S_{\triangle P_1P_sP_t}$ 已经是最大的。因此时间复杂度不超过 $O(n)$。

3 C Carrot Trees

题意：给一个长为 n 的序列，初始状态全为 0。有两种操作，第一种将区间 $[l,r]$ 内的数均增加 x，第二种对区间 $[l,r]$ 内的每个数如果大于或等于 k 就减去 k（k 在所有操作中相等）。你需要执行 m 次操作，输出执行完所有操作后所有数一共被减了多少次。$1 \le n \le 10^6$，$1 \le m \le 2 \times 10^5$，$1 \le x,k \le 10^9$。

解法：考虑某个位置上的数。将第 i 次操作后的数表示成 $a_i = c_i + k \cdot b_i$ 的形式（$b_0 = c_0 = 0$），其中 b_i 代表了借位操作次数。考虑如下的一个暴力算法：对于操作一，直接令 $c_{i+1} = c_i + x$。对于操作二，若 $a_i \ge k$（即 $c_i + kb_i \ge k \Rightarrow c_i - k \ge -k\,b_i$），则令 $c_{i+1} = c_i - k$；否则令 $c_{i+1} = c_i - k$ 且 $b_{i+1} = b_i + 1$。

> 引理：使用上述算法计算，所有操作后 $b_m = \left\lceil -\dfrac{\min_i c_i}{k} \right\rceil$。
>
> 证明：经过一次操作一，$c_{i+1} \ge c_i$，则 $\min_{1 \le j \le i} c_j = \min_{1 \le j \le i+1} c_j$，因而 $b_i = b_{i+1}$，与上述算法执行过程相同。考虑操作二的两种情况：
>
> 1. 当 $a_i \ge k$ 时，有 $c_{i+1} = c_i - k \ge -kb_i$，因而 $b_{i+1} = b_i \ge -\dfrac{c_i}{k} + 1 \ge \left\lceil -\dfrac{c_i}{k} \right\rceil$。显然 $c_{i+1} < c_i$，由数学归纳法可得若 $b_i = \left\lceil \dfrac{\max_{0 \le j \le i}(-c_j)}{k} \right\rceil$ 成立则 $b_{i+1} = \left\lceil \dfrac{\max_{1 \le j \le i+1}(-c_j)}{k} \right\rceil$ 成立，即 $b_{i+1} = \left\lceil -\dfrac{\min_j c_j}{k} \right\rceil$。
> 2. 当 $a_i < k$ 时，$0 \le c_i + kb_i < k$ 可得 $c_i \ge -kb_i$。由此时的操作 $b_{i+1} = b_i + 1$ 可得 $c_i \ge -k - kb_{i+1}$，则与第一种情况证明方法相同。

由该引理可知，因此 a_i 减少的次数一共为 $t - b_m = t + \min_i \left\lfloor \dfrac{c_i}{k} \right\rfloor$（其中 t 为操作二的次数）。

因而本题转化为转化为区间加、单点查询历史最小值，可以参看https://oi-wiki.org/ds/seg-beats/的处理方法，或者使用线段树分治的思想。总时间复杂度 $O(n \log m)$。

4 D Chocolate

题意：有一个 $n \times m$ 的网格，每格放了块巧克力。Walk Alone（懵哥）和 Kelin 轮流吃巧克力，Kelin 先吃。每轮一个人能选择一个左下角为 $(1,1)$ 的子矩形，把里面的巧克力吃光，且至少要吃一个。吃到最后一个巧克力的人输。问懵哥和 Kelin 谁赢。T 组测试数据，$1 \le T \le 10^5$，$1 \le n,m \le 10^9$。

解法：当 $n = m = 1$ 显然懵哥赢。当 $n > 1$ 或 $m > 1$，假设 Kelin 第一步吃 $(1,1)$ 并且懵哥赢了，那么 Kelin 可以抢先走懵哥的必胜态且不改变局面（因为 $(1,1)$ 被任意子矩形包含），因此 Kelin 必胜。

5 E Heap

题意：给定一个以 1 为根的有根树，点个数为 n，每个点有点权 a_i。现要求给每个点重新赋点权 b_i，且要求 $b_{p_i} \le b_i$，其中 p_i 为 i 的父亲。输出最小的 $\sum_{i=1}^{n}(a_i - b_i)^2$。$1 \le n \le 2 \times 10^5$，$0 \le a_i \le 10^9$。

解法：本题主要使用到了 Slope Trick 的方法。

首先我们研究一个简单一些的题目，并从该弱化版的题目中得到本题真正的做法。

弱化版题目：给定一个以 1 为根的有根树，点个数为 n，每个点有点权 a_i。现要求给每个点重新赋点权 b_i，且要求 $b_{p_i} \le b_i$，其中 p_i 为 i 的父亲。输出最小的 $\sum_{i=1}^{n} |a_i - b_i|$。$1 \le n \le 2 \times 10^5$，$0 \le a_i \le 10^9$。

用 $g_u(x)$ 表示保证以 u 为根的子树满足所有父节点权值小于等于子节点，并且使得 u 的权值**恰好**为 x 的最小代价；$f_u(x)$ 表示使得 u 的权值**至少**为 x 的最小代价，即 $f_u(x) = \min_{y \ge x} g_u(y)$。可以得到以下转移：

$$g_u(x) = |x - a_u| + \sum_{v \in \text{son}_u} f_v(x)$$

首先需要证明引理 1：

$f_u(x)$ 是一个可导且单调递增的函数，且形如 $f_u(x) = c_u + \sum_{i \in S_u} r(x - i)$，其中：

1. $r(x - a_u) = \text{ReLU}(x - a_u) = (x - a_u)[x \ge a_u]$，其中 $[x \ge a_u]$ 为艾弗森括号，表示当 $x \ge a_u$ 时值为 1，反之为 0。
2. S_u 为 u 节点子树中全部 a_u 构成的可重集合。
3. c_u 为一确定常数。

证明：递增性显然成立，下面使用数学归纳法证明该结论。

首先我们考虑叶节点 $g_u(x)$ 与 $f_u(x)$ 的形式——$g_u(x) = |x - a_u|$，
$f_u(x) = r(x - a_u) = (x - a_u)[x \ge a_u]$，因而 $f_u(x)$ 是一个单调递增且可导的函数，且符合归纳形式。
注：ReLU 函数在 $x = 0$ 处严格意义上不可导，但是为方便后文的表述，不妨定义 $r'(0^+) = 1$。

对其余非叶节点，$g_u(x) = |x - a_u| + \sum_{v \in \text{son}_u} f_v(x)$。由可导函数相加还是可导函数，$g_u(x)$ 也同样可导，且 $c_u + \sum_{i \in S_u} r(x - i)$ 形式具有可加性。对其做后缀 min 操作，则 $|x - a_u| \to r(x - a_u)$，因而 $f_u(x)$ 同样符合原形式。归纳得证。

考虑将 $f_u(x) = c_u + \sum_{i \in S_u} r(x - i)$ 带入原转移式：

$$\begin{aligned} g_u(x) &= |x - a_u| + \sum_{v \in \text{son}_u} c_v + \sum_{v \in \text{son}_u} \sum_{i \in S_v} r(x - i) \\ &= (-x + a_u + 2r(x - a_u)) + \sum_{v \in \text{son}_u} c_v + \sum_{v \in \text{son}_u} \sum_{i \in S_v} r(x - i) \\ &= -x + a_u + \sum_{v \in \text{son}_u} c_v + \sum_{i \in S'_u} r(x - i) \end{aligned}$$

其中 $S'_u = \bigcup_{v \in \text{son}_u} S_v \cup \{a_u, a_u\}$。不妨对 $g_u(x)$ 求导，当导函数第一次大于等于 0 时，$x = \min S'_u$。令 $x_0 = \min S'_u$，当 $x = x_0$ 时 $g_u(x)$ 取到最小值 $a_u - x_0 + \sum_{v \in \text{son}_u} c_v$，因此令

$$\begin{aligned} S_u &= S'_u - \{x_0\} \\ c_u &= a_u - x_0 + \sum_{v \in \text{son}_u} c_v \end{aligned}$$

当 $x \ge x_0$ 时有

$$
\begin{aligned}
f_u(x) &= g_u(x) \\
&= -x + x_0 + c_u + \sum_{i \in S'_u} r(x - i) \\
&= r(x - x_0) - x + x_0 + c_u + \sum_{i \in S_u} r(x - i) \\
&= c_u + \sum_{i \in S_u} r(x - i)
\end{aligned}
$$

当 $x < x_0$ 时，由于 $f_u(x)$ 是 $g_u(x)$ 的后缀 min，因而

$$
f_u(x) \equiv c_u + \sum_{i \in S_u} r(x - i) = c_u
$$

用优先队列维护 S_u 做启发式合并可以做到 $O(n \log^2 n)$。

接下来我们回到原问题，即目标函数为平方代价。

用 $g_u(x)$ 表示保证以 u 为根的子树满足所有父节点权值小于等于子节点，并且使得 u 的权值**恰好**为 x 的最小代价；$f_u(x)$ 表示使得 u 的权值**至少**为 x 的最小代价，即 $f_u(x) = \min_{y \geq x} g_u(y)$。可以得到以下转移：

$$
g_u(x) = (x - a_u)^2 + \sum_{v \in \mathrm{son}_u} f_v(x)
$$

用上面同样的方法，可以得到

$$
f_u(x) = c_u + \sum_{i \in S_u} (x - i)^2 [x \geq i]
$$

其中 S_u 仅为关于 u 的集合。这时形式较为复杂，对其做后缀 min 操作难度较大。但是我们可以研究它的导函数以研究它的极小值情况。对转移式两边求导有

$$
g'_u(x) = 2(x - a_u) + \sum_{v \in \mathrm{son}_u} f'_v(x)
$$

其中 son_u 表示 u 的儿子集合。根据上面同样的方法，显然 $f_u(x)$ 是一个连续、递增、下凸的分段函数，因此 $f'_u(x) \geq 0$，且由 $f_u(x) = \min_{y \geq x} g_u(y)$ 可得

$$
f'_u(x) = \max\{g'_u(x), 0\}
$$

用可重集合 $S_u \subseteq \mathbb{R} \times \mathbb{N}$ 维护 $f'_u(x)$ 的所有拐点以及拐点两侧的斜率差。下用同样的方法证明 $f'_u(x) = \sum_{(i,\delta) \in S_u} \delta \cdot \mathrm{ReLU}(x - i)$。

证明：同样引入 $r(x) = x \cdot [x > 0]$，假设 u 的所有儿子节点 $f'_v(x)$ 都能表示成以下形式：

$$
f'_v(x) = \sum_{(i,\delta) \in S_v} \delta \cdot r(x - i)
$$

代入 $g'_u(x)$ 的转移式可得

$$
\begin{aligned}
g'_u(x) &= 2(x - a_u) + \sum_{v \in \mathrm{son}_u} \sum_{(i,\delta) \in S_v} \delta \cdot r(x - i) \\
&= -2a_u + 2x + \sum_{(i,\delta) \in S'_u} \delta \cdot r(x - i)
\end{aligned}
$$

其中 $S'_u = \bigcup_{v \in \mathrm{son}_u} S_v$。显然 $g'_u(x)$ 单调递增，且存在唯一零点 x_0。考虑由于求后缀 min 使得 x_0 处增加的拐点 $(x_0, g''_u(x_0^+))$，令

$$
S_u = \{(i, \delta) \in S'_u \mid i > x_0\} \cup (x_0, g''_u(x_0^+))
$$

则 $f'_u(x)=\sum\limits_{(i,\delta)\in S_u}\delta\cdot r(x-i)$，归纳假设成立。

记 $f_u(x)$ 与 $g_u(x)$ 的最小值为 C_u。不难注意到，当子树中存在拐点 (i,δ) 有 $i\le x_0$ 时，由于已经在 x_0 极其左侧取得最小值，因而 x_0 左侧的全部拐点都可以删去。在寻找这样的 x_0 时，可以依次枚举 S'_u 中相邻的拐点，找到零点存在的区间然后求出 x_0，再删去左侧的拐点。

由于 $g(x)$ 的极小值在 x_0 处取到，代入 $g_u(x)$ 的转移式可得

$$\begin{aligned}C_u&=(x_0-a_u)^2+\sum_{v\in\mathrm{son}_u}f_v(x_0)\\&=(x_0-a_u)^2+\sum_{v\in\mathrm{son}_u}C_v+\sum_{v\in\mathrm{son}_u}\int_{-\infty}^{x_0}f'_v(x)\,\mathrm{d}x\end{aligned}$$

通过启发式合并可以在 $O(n\log^2 n)$ 的时间内求出所有集合 S_u 与每个函数对应的最小值 C_u。总时间复杂度 $O(n\log^2 n)$。

6 F Intersection

题意：平面上有 n 个圆 $C_1,C_2,\ldots,C_n$，要求画一个圆 O（或直线）经过尽可能多的圆。$1\le n\le 150$，坐标范围 $0\le|x|,|y|\le 10^3$。

解法：当 $n\le 3$ 时答案为 n。当 $n>3$ 时，最优解一定能通过调整使得与至少两个圆外切。枚举 $i,j\,(i\ne j)$，假设圆 O 与 C_i,C_j 外切，求 O 最多能与多少 C_k 相交。

记 C_i,C_j 的半径分别为 r_i,r_j，圆心分别为 o_i,o_j（用复数表示）。假设对复平面进行反演映射 $f(z)=1/(z-o)$，则反演后 C_i 的圆心和半径分别变为

$$\begin{aligned}o'_i&=\frac{1}{2}\Big(\frac{1}{o_i-r_ie-o}+\frac{1}{o_i+r_ie-o}\Big)\\&=\frac{o_i-o}{(o_i-o)^2-(r_ie)^2}\\r'_i&=\frac{1}{2}\Big|\frac{1}{o_i-r_ie-o}-\frac{1}{o_i+r_ie-o}\Big|\\&=\frac{r_i}{(o_i-o)^2-(r_ie)^2}\end{aligned}$$

其中 $e=(o_i-o)/|o_i-o|$。类似地可以计算 C_j 的圆心和半径。令反演后 C'_i 与 C'_j 圆心位置相同，可以得到

$$\frac{o_i-o}{(o_i-o)^2-(r_ie)^2}=\frac{o_j-o}{(o_j-o)^2-(r_je)^2}$$

解得 $Ao^2+Bo+C=0$，其中

$$\begin{aligned}A&=o_i-o_j\\B&=(r_i^2-r_j^2)\,e^2-(o_i^2-o_j^2)\\C&=(o_ir_j^2-o_jr_i^2)\,e^2+o_io_j(o_i-o_j)\end{aligned}$$

取圆 C_i 内部的点 o，则反演后除 C_i,C_j 外的所有圆都在 C_i 内部、C_j 外部，且 O 与 C_i 内切，与 C_j 外切。对于每个圆 C_k，可以计算出 O 与 C_k 相交时 C_iO 的辐角取值区间，通过扫描线可以求出最多相交的圆的数量。总复杂度 $O(n^3\log n)$。

7 G LCRS Transform

题意：给出一种对 n 个点的二叉树有根树（树根在 1）的操作：

- 由深到浅的遍历树上的每个节点 u，将 u 的右儿子 r_u 变为 u 的左儿子 ℓ_u 的右儿子。
- 然后对于所有只有右儿子没有左儿子的节点，将其右儿子变为左儿子。

问进行 k 次操作后有多少棵树能变成给定的树。$1 \le n \le 10^5$，$0 \le k \le 10^5$。

解法：注意到每次操作后树的前序遍历不变，因此可以用前序遍历对所有点重新标号。这样标号后点 u 的左儿子 ℓ_u 满足 $\ell_u = u+1$，且每次操作后边 (u, r_u) 会变为 $(\min\{u+1, r_u-1\}, r_u)$——$u+1$ 对应第一个操作，r_u-1 对应第二个操作（若执行第二个操作此时满足 $r_u - 1 = u$）。k 次操作后变为 $(\min\{u+k, r_u-1\}, r_u)$。

考虑 k 次操作后的树最开始的样子。考虑两种边：

1. 对于操作完成之后的树上的每条满足 $r_u > u+1$ 的边 (u, r_u)，最开始的树中一定存在边 $(u-k, r_u)$——每一次变化都是执行了 $u \leftarrow u+1$。因而需要保证此时的 $u > k$ 且这些边对应的“区间”不相交（这里定义两条边 (x, y) 与 (ℓ, r) 相交为 $\ell \le x < r < y$ 或 $x < \ell < y \le r$），否则答案为 0，因为这样就无法建立一条 $u \to r_u$ 的边。
2. 对于满足 $\ell_u = u+1$ 条件的边 (u, ℓ_u)，最开始的边可以为 $(u-i, \ell_u)$ $(0 \le i \le k)$，表示这条边在第 i 步开始就从连接右儿子变成连接左儿子，然后相对关系不再变化。

于是这道题等价于：一个长度为 n 的线段中给定一些长度大于 k 的区间，要求在其中添加一些长度不超过 k 的区间，使得所有区间互不相交，问一共有多少种方案。

显然新添加的区间不可能跨过给定的区间，因此可以认为这些给定的区间把整个线段划分为了若干段，单独考虑每一段的贡献。令 f_i 表示长度为 i 的区间的方案数，其中 $f_0 = f_1 = 1$。对于 $i \ge 2$ 有

$$f_i = f_{i-1} + \sum_{l=2}^{\min\{k+1, i\}} f_{i-l} \cdot f_{l-2}$$

令 $g = \sum_{i=0}^{k+1} f_i x^i$。当 $i \le k+1$ 时 g 有生成函数等式：

$$g = xg + x^2 g^2 + 1$$

解得

$$g = \frac{1 - x - \sqrt{1 - 2x - 3x^2}}{2x^2}$$

此时通过多项式求逆求出 f 数组的前 $k+1$ 项，即得到了递推的系数项。考虑对于 $i > k+1$ 时递推式可化简为 $f_i = f_{i-1} + \sum_{l=2}^{k+1} f_{i-l} \cdot f_{l-2}$，转写成多项式形式为：

$$f = fx + fgx^2 + 1$$

即 $f = \dfrac{1}{1 - x - gx^2}$。可以用多项式求逆和多项式开方在 $O(n \log n)$ 时间内求解。

8 H Matches

题意：给定两个长度为 n 的序列 $\{a\}_{i=1}^{n}$ 和 $\{b\}_{i=1}^{n}$，现在可以选择其中一个序列交换其中的两个数字，问经过至多一次操作后最小的 $\sum_{i=1}^{n}|a_i-b_i|$。$1\le n\le 2\times 10^5$，$0\le |a_i|,|b_i|\le 10^{12}$。

解法：考虑交换 a_i,a_j ，不妨设 $a_i<a_j$ ，则交换前后的贡献差 $\Delta=|a_i-b_i|+|a_j-b_j|-|a_j-b_i|-|a_i-b_j|$ ，即最大化 Δ 。

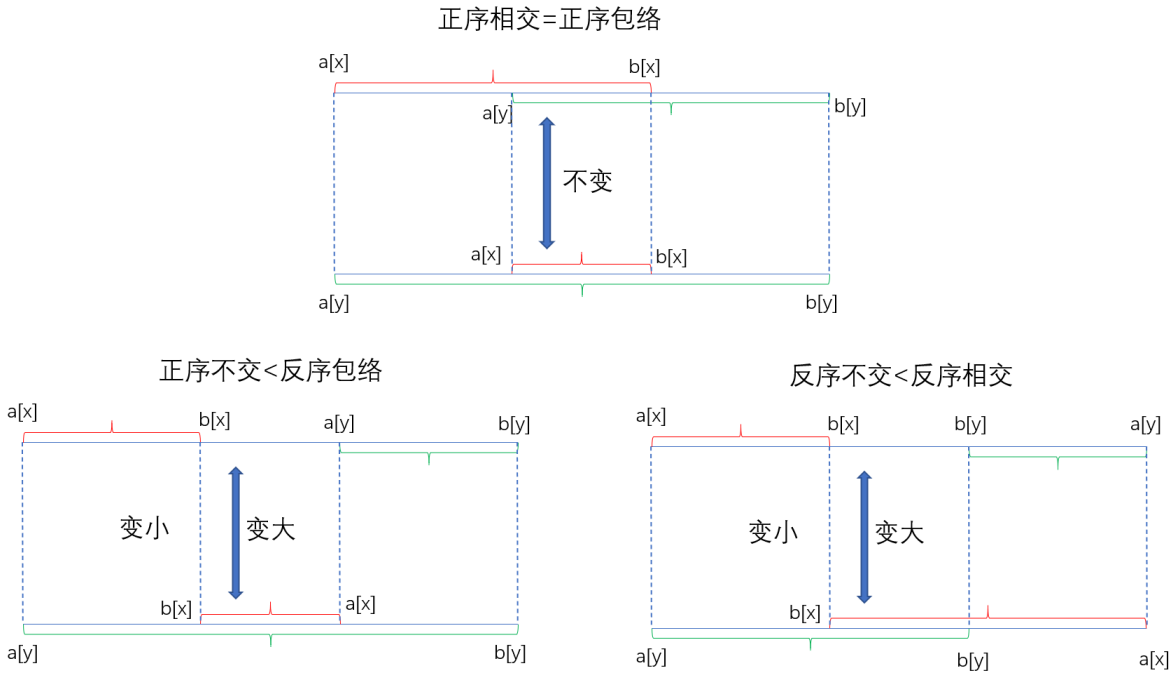
下面使用一张图来给出最大化 Δ 的条件。首先定义：

1. 对于一对 (a_x,b_x) 和 (a_y,b_y)，不妨设 $a_x<b_x$，若 $b_y<a_y$ 则称他们成反序关系，若 $a_y<b_y$ 则为正序。
2. 若一对 (a_x,b_x) 与 (a_y,b_y) 在数轴上覆盖范围有重叠，且存在独占一段的情况，称为相交。
3. 若一对 (a_x,b_x) 与 (a_y,b_y) 在数轴上覆盖范围有重叠，一段完全包络另一段，称为包络。
4. 若一对 (a_x,b_x) 与 (a_y,b_y) 在数轴上覆盖范围没有重叠，称为不交。

不难注意到：

1. 正序相交=正序包络
2. 正序不交<反序包络
3. 反序不交<反序相交

即将反序相交、反序包络进行交换即可。



不难注意到，我们经过一次交换不同种类（正序和反序）的区间，可以消除两倍的重叠（包络）长度。

具体到做法，首先考虑将所有区间一起按右端点（$\max(a_i,b_i)$）排序，按**右端点递减**的顺序依次枚举每个区间。假设当前枚举的区间满足 $a_i<b_i$，这个时候用一个变量 l_T 维护**已经枚举过**的区间里面所有满足 $a_j>b_j$ 的区间中最小的**左端点** b_j，不难发现如果 $b_j<a_i$，则现在的重叠区间长度为 $b_i-\max(a_i,b_j)$；对于 $a_i>b_i$ 的同理可以维护 l_S。

最后答案即为原始答案减去二倍的最大重叠区间长度。总时间复杂度为 $\mathcal{O}(n\log n+n)$。

9 I Random

题意：给定对 x 进行 m 次左移/右移并异或的函数 $\mathrm{rand}(x)$，问期望对 $[0,2^n-1]$ 上均匀随机分布的 x 执行多少次 rand 可以变回 x 本身。$1\le n\le 10^5$，$1\le \log_2 x\le 32$。

解法：整体做法：对映射矩阵的极小多项式的阶枚举因数统计答案，进行莫比乌斯反演。

首先考虑下面的引理 1：

> 引理 1：这个随机数生成器是一个 k 维模 2 （$\mathbb{F}_2$）的向量空间 V 中的线性映射 A，且 A 可逆。
>
> 证明：由于题目给出的 $0<|a_i|$ ，因而容易证明每一步单步操作对应一个由列初等变换得到的可逆矩阵，而 A 可以看作每一步操作对应的可逆矩阵的乘积。

由于矩阵 A 可逆，因而有引理 2：

> 引理 2：对于任意输入 $\boldsymbol{x}$，周期 t 必然存在，即 $A^t\boldsymbol{x}=\boldsymbol{x}$，即这个线性映射意义下的阶，下记为 $\mathrm{ord}(\boldsymbol{x})$。
>
> 证明：如果把向量 $\boldsymbol{x}$ 看成一个节点，向量空间 V 看成一张图 G，那么显然每个点都有一个后继 $A\boldsymbol{x}$ ，也都有一个前驱 $A^{-1}\boldsymbol{x}$。这就是说 G 一定是由若干个环构成的。于是我们证明了每个 $\boldsymbol{x}$ 的周期 $t_{\boldsymbol{x}}$ 必然存在，且为它的阶 $\mathrm{ord}(\boldsymbol{x})$。

因而最后的答案为 $\dfrac{1}{2^k}\displaystyle\sum_{i=0}^{2^k-1}\mathrm{ord}(i)$，其中 i 表示 i 的二进制表示对应的向量 $\boldsymbol{x}_i$。当然，直接统计的时间复杂度是不可接受的。但是以 t 为周期（不一定是最简周期）的向量 $\boldsymbol{x}$ 的数量却可能是容易的。考虑引理 3：

> 引理 3：以 t 为周期（不一定最简）的向量数量为 $2^{k-\mathrm{rank}(A^t-E)}$ 。
>
> 证明：立刻能发现符合条件的向量数量本质上是方程 $A^t\boldsymbol{x}=\boldsymbol{x}$ 的解集大小。也就是求 $|\ker(A^t-E)|$ 或者说 $2^{\mathrm{nul}(A^T)}$，其中 ker 指的是零空间，nul 指的是零化度，E 是单位阵。根据秩-零化度定理，解个数就是 $2^{k-\mathrm{rank}(A^t-E)}$ 。

考虑通过容斥求出每个向量具体的阶，以 s （$s|t$）为阶的向量的答案可以通过对以 t 为周期的向量数量容斥求出。这是因为到每一个以 t 的因子为最小周期的 $\boldsymbol{x}$ 也会以 t 为周期，故而我们需要容斥掉 t 的因子的贡献。设 $f(t)=t\times 2^{k-\mathrm{rank}(A^t-E)}\times 2^{-k}$，阶为 s 的向量对答案的贡献为 $g(s)$，则由上知 $f(t)=\displaystyle\sum_{s|t}g(s)$。若知 $f(t)$ 的值，莫比乌斯反演即可线性求得 $g(s)$。问题可转化为求 $f(t)$。

然而 s,t 的量级是难以确定的。上述操作仍不足以解决此题，甚至不一定会使得时间更优。我们注意到不少 s 贡献均为 0 ，因此尝试寻找 s 的性质和量级。有引理 4：

> 引理 4：所有 s 均是方程 $A^r=E$ 的解 r 的因子。
>
> 证明：由于 A 本质有限且可逆，不难发现 $\{A^r\}$ 构成一个循环群。根据群论的拉格朗日定理，有限群的循环子群的阶是其母群的阶的因数。此处母群即是整个群 $\langle\{A^0=E,A^1,\cdots,A^{r-1}\},\times\rangle$，而子群表示由阶为 s 的元素构成的子群 $\langle\{A^0,A^s,A^{2s},\cdots,A^{r-s}\},\cdot\rangle$，显然有 $s|r-s$。

利用引理 4，我们知道最小周期 s 是循环子群的阶。在这里，我们可以认为其母群的阶即是最小的 r 使得 $A^r=E$ 。

引理 5：母群的阶 r 是方程 $X^r \equiv E \pmod{p(X)}$ 的最小正整数解。其中 $p(X)$ 是极小多项式，即一个 $\mathbb{F}_2$ 上的最低次首一多项式 $p(X) \in \left\{ \sum_{i=0}^{l} p_i X^l \middle| p_i \in \mathbb{F}_2, 1 \le l < r \right\}$ 满足 $p(A) = 0$。

证明：由于 A 满秩，因而 $\forall k, A^k \neq 0$，因而 $p(X) \nmid A^r$。所以不难注意到在模 $p(x)$ 的矩阵乘法运算作用下的代数系统 $\langle \{A^0, A^s, A^{2s}, \cdots, A^{r-s}\}, \cdot \pmod{p(X)} \rangle$ 也是一个群。由于 $A^r = E$，考虑对 $A^r - E = 0$ 进行因式分解，则 $p(A)$ 为 $A^r - E$ 中的一项或若干项的乘积。假设 r 是极小多项式 $p(X)$ 的阶，即最小的正整数满足 $\forall X, p^r(X) = E$ 的 r，则必然有 $X^r - E = 0$，则可推出 $X^r \equiv E \pmod{p(X)}$。而显然如果母群的阶 $r' < r$，则 $p^{r'}(X) \neq E$，则 $X^{r'} \neq E$ 矛盾。因而该方程的解就是原问题的解。

方程 $X^r \equiv 1 \pmod{p(X)}$正是标准的离散对数问题，可以使用 BSGS 求解出 r，然后对 r 分解出 t 计算答案，最后莫比乌斯反演即可解决。

接下来，倘若我们能够证明 r 的范围，便可以解决此题。接下来的部分是对这样做的时间复杂度正确性的证明。

引理 6：$r < 2^k$。

证明：对于 A 的特征多项式 $q(\lambda) = \prod_{i=1}^{k} q_i^{\alpha_i}(\lambda)$ ，由卡莱-哈密顿定理可得 $q(A) = 0$。因而利用引理 5 的结论，方程 $A^x \equiv E \pmod{q(X)}$ 的解一定是 r 的倍数，可以通过它框定 r 的范围。考虑 $q(\lambda)$ 的结构，考虑进行一个类似于 excrt 的过程——将 $q(\lambda)$ 根据每个不可约多项式进行分解，首先研究每个不可约质多项式下同余多项式的解范围，再升次，最后合并这些解。它在线性空间中的意义是，每一个因式 $q_i^{\alpha_i}(\lambda)$ 都对应一个不变子空间，也就是 $q_i^{\alpha_i}(A)$ 的零空间，记作 $\ker(q_i^{\alpha_i}(A))$ 。可以验证，所有 $\ker(q_i^{\alpha_i}(A))$ 的直和就是 k 维全空间，即它们的基向量彼此线性无关。

现在我们考察每一个 $\ker(q_i^{\alpha_i}(A))$ 。对于全空间中的某一个向量 $\boldsymbol{x}$ ，它的周期即是它在每一个子空间里的分量的周期的 lcm。因此，我们只需要考察每一个子空间里的分量的周期的量级，求取 lcm 即可得出全空间里的每一个分量的周期的量级。对于 $q_i^{\alpha_i}(X)$ ，我们考察它的不可约因子 $q_i(X)$。因为 $\mathbb{F}_2$ 上 d 次不可约多项式 $q(X)$ 是 $X^{2^d} - X$ 的因子，故而不可约多项式 q_i 的周期 $r_{i,1}$ 不会超过 2^d 。在这里，周期指的是最小正整数 $r_{i,1}$ 使得 $X^{r_{i,1}} \equiv E \pmod{q_i(X)}$ 。

为了籍此推出 $q_i^{k_i}(X)$ 的周期的量级，我们需要证明引理 6.1：

> 引理 6.1：若 $\mathbb{F}_2$ 上的不可约多项式 $p(X)$ 的周期是 d，那么 $p^k(X)$ 的周期的上界（可以证明是紧上界）是 $d \times \mathrm{bitceil}(k)$ 。
>
> 证明：因为 $p(X) | (X^d - 1)$，故而 $p^2(X) | (X^d - 1)^2$。在 $\mathbb{F}_2$ 上，$(X^d - 1) = (X^d + 1)$，于是 $p^2(z) | (X^d + 1)(X^d - 1) = X^{2d} - 1$。故而 $p^{\mathrm{bitceil}(k)}(X)$ 的周期是 $d \times \mathrm{bitceil}(k)$ 。又显然若 $i < j$ 则 $p^i(z)$ 的周期不大于 $p^j(X)$ 的周期，故而 $p^k(X)$ 的周期的上界是 $d \times \mathrm{bitceil}(k)$ 。

由此我们得到了 r 的规模为 $\mathrm{lcm}(2^{d_i} \times \mathrm{bitceil}(k_i))$，在这里，$d_i$ 是不可约多项式 $p_i(X)$ 的次数。这个值是显著小于 $O(2^k)$ 的。

最后，考虑复杂度：

结论：这样做的复杂度的级别是 $O\left(nk^2 + k^3 + k\sqrt{2^k} + \max_{1 \le i \le 2^k} d^2(i)\right)$。

证明：梳理一遍整个算法流程：

1. 首先我们预处理出映射矩阵 A ，这需要 n 次 $\mathbb{F}_2$ 上的矩阵乘法，复杂度是 $O(nk^2)$ 。

2. 我们首先求 A 的极小多项式 $p(X)$：我们找到最小的 m 使得 $\exists\{a\}_m \in \{\mathbb{F}_2\}_m, \sum_{i=0}^{m-1} a_i A^i = A^m$，即 $p(X) = X^m - \sum_{i=0}^{m-1} a_i X^i$。根据凯莱-哈密顿定理，$A$ 一定是其特征多项式 $q(\lambda)$ 的根即 $q(A) = 0$，而特征多项式为 k 次，因而 $m \le k$。故而这个部分的复杂度是 $O(k^3)$。
3. 接下来，我们使用 BSGS 求出阶 r，r 的规模在前文已经证明为 $O(2^k)$，故而这里需要 $O\left(k\sqrt{2^k}\right)$ 的复杂度。
4. 然后，我们以 $O\left(\sqrt{r}\right) = O\left(\sqrt{2^k}\right)$ 的复杂度分解 r，并枚举它的因数。考虑到 $d(n)$ 的规模足够小，可以不使用莫比乌斯反演，$d^2(n)$ 的暴力容斥也是可以接受的。

这样，我们就完成了此题。

10 J Roulette

题意：Walk Alone 初始有 n 块钱，如果每次投 x 元，有一半的概率输掉这 x 元，另一半概率赢得 $2x$ 元。现在 Walk Alone 采取下述策略投注：

1. 如果上一把赢了，这一把投 $x_i = 1$ 元
2. 如果上一把输了，这一把投 $x_i = 2x_{i-1}$ 元。

问 Walk Alone 有多大概率拿到 $n+m$ 元离开。$1 \le n, m \le 10^9$。

解法：不难注意到，以“输输输……输赢”为一个周期，可以赢到一块钱。

如：输输输赢输赢赢赢赢 可以划分为：输输输赢/输赢/赢/赢/赢/，即以一次赢结束，找到上一次赢的地方，划分成一块。对于每一个周期内，假设前面输了 n 轮，那么整个周期赚的钱为 $-1-2-4-\cdots-2^n+2^{n+1}=1$。

因而需要经过 m 个这样的周期。考虑每个周期不会输光的条件：如果当前有 x 元，那么不能连输超过 r 次，其中 r 是最大满足 $2^r-1 \le x$ 的正整数，否则就将没有钱继续投入。这样成功的概率为 $1-\left(\frac{1}{2}\right)^r$，即不会连续输 r 次。

由于我们在赢钱的过程中是一轮赢一块钱，因而随着赢钱轮数增加，r 也会发生变化，但是不难注意到 r 的级别和个数都是 $\log x$ 级别的。因而可以考虑枚举 r，对于每个 r，该段的 x 获胜的概率均相等——$[2^r-1, 2^{r+1}-2]$。同时这样的 r 只有 $O(\log(n+m))$ 段，配合快速幂可以在 $O(\log^2 n)$ 的复杂度内解决本题。

本题需要逆元的知识点。由于在取模意义下没有除法运算，为了表示除法，和矩阵的逆一样，我们找到另一个元素以乘法来代替除法运算。

定义：在模 m 意义下，x 的逆元 y 为满足 $xy \equiv 1 \pmod m$ 的解。这样除以 x 的运算就可以使用乘以 y 来替代。

考虑如何求出这样的 y。对于求某一个 x 的逆元 y，通法是使用 exgcd（扩展欧几里得算法）。即将原同余方程转化为一般的丢番图方程——$xy+km=1$，其中 (x,m) 为已知数求 (y,k)，其中 y 为代求量。

扩展欧几里得算法的核心在于 $ax_1+by_2=\gcd(a,b)=\gcd(a,b)$ 与 $bx_2+(a \bmod b)y_2=\gcd(b,a \bmod b)$ 方程的等价性。由欧几里得算法，$\gcd(a,b)=\gcd(b,a \bmod b)$。将 $a \bmod b = a-\left\lfloor\frac{a}{b}\right\rfloor \times b$ 带入整理上式有

$$ax_1+by_1=ay_2+b\left(x_2-\left\lfloor\frac{a}{b}\right\rfloor y_2\right)$$

即 $x_1=y_2$，$y_1=x_2-\left\lfloor\frac{a}{b}\right\rfloor y_2$。

使用下面的代码即可：

```cpp
// x y 为引用，传入待解的参数；a b 为已知量。求解方程ax+by=gcd(a,b)的一组解（x,y）
void exgcd(int a, int b, int& x, int& y) {
  if (b == 0) {
    x = 1, y = 0;
    return; // 此时的a为原始gcd(a,b)
  }
  exgcd(b, a % b, y, x);
  y -= a / b * x;
}
```

注意：逆元不一定存在，当且仅当 $\gcd(x,m)=1$ 时逆元存在。考察方程 $xy+km=1$，如果 $\gcd(x,m)\neq 1$，则方程左侧一定是 $\gcd(x,m)$ 的倍数，而右侧不是，等式不成立。这就是裴蜀定理。

还可以使用费马小定理或欧拉定理求解。对于模数为质数（如常见的 998 244 353 和 10^9+7 等大质数）的情况，有 $a^p\equiv a\pmod p$。当 $a\neq p$ 时有 $a^{p-1}\equiv 1\pmod p$。因而不难注意到 a 的逆元为 a^{p-2}。使用快速幂求解即可。

还可以在线性时间（均摊 $O(1)$）内求出 $[1,i]$ 的逆元。

显然对于任意的模数 m，1 的逆元都是 1——$1\times 1\equiv 1\pmod m$。考虑已知前 $i-1$ 个数的逆元，如何求出第 i 个数的逆元。

设 $p=ki+r$，则有 $ki\equiv -r\pmod p$。方程两边同时乘以 i^{-1} 和 r^{-1} 可得：

$$kr^{-1}+i^{-1}\equiv 0\pmod p$$

因而 $i^{-1}\equiv\left\lfloor\frac{p}{i}\right\rfloor(p\bmod i)^{-1}\pmod p$。其中 $p\bmod i<i$，因而其逆元必然已经求出。所以根据该线性递推式可得所有存在逆元元素的逆元。

11 K Subdivision

题意：给定一个 n 个点 m 条边的无向图 G，可以将其中任意一条边分裂成一条长度为任意的链（向边中插任意多个点），可以操作任意多次（也可以不操作）。问经过这样处理之后，从 1 号节点出发，至多走 k 步最多可以到多少个节点。$1\leq n\leq 10^5$，$0\leq m\leq 2\times 10^5$，$0\leq k\leq 10^9$。

解法：由于需要考察从 1 出发走 k 步所能到的点，那么显然需要首先构造出 1 出发的 bfs 树（最短路树）。由于此题中所有边权均为 1，因而最短路树退化为 bfs 树。其中最短路树是通过 Dijkstra 算法得到的，满足 $d_u=d_v+w$ 的边 (u,v,w) 构成的树状子图。

考虑以下两种情况：

1. 该边不在 bfs 树上。这种情况存在两个子类——该边连接了同层的两点、该边连接了不同层的两点，但是该边去除不影响图深度的计算。则这种情况下该边都可以分裂为无穷多个点，并且不影响其他点是否可以到达。
2. 该边在 bfs 树上。首先需要证明一个引理：

   > 对于从 1 号节点开始的路径 $P_n=\{1,v_1,v_2,\cdots,v_n\}$，如果需要分裂边以增加点数，那么在增加点数相同的情况下，分裂 (v_{n-1},v_n) 最优。

证明：不失一般性的，考虑仅增加一个点。假设在 v_i 处有 k_i 条不在 bfs 树上的边，其中 $i \in [1, n-1]$。如果这时修改 (v_j, v_{j+1})，则会让 $i \in [j+1, n-1]$ 所有的点连接的所有非 bfs 树边所能到达的点数减少一。那么显然当 $j = n-1$ 时这个损失的量最小。

考察对于 bfs 树的叶节点 u 及根节点到它的路径 $P_n = \{1, v_1, v_2, \cdots, v_{n-1}, u\}$，首先我们对该路径上所有边不做处理，如果这种情况下 u 就已经到不了了，那么这条路径上不用分裂——因为显然分裂一次创造出来的新点不过是置换了路径中更深的一个点，并且由引理可知这样还会损失非树边的答案。如果 u 可以到达，那么需要考虑 u 连接的非树边条数。如果 u 没有连接非树边，那么通过分裂 (v_{n-1}, u) 让 u 变为最深的节点可以让答案增加；否则则不要执行边分裂操作，因为这样增加了 (v_{n-1}, u) 路径上的答案会损失 u 连接的非树边的答案，哪怕只有一条边也不过是打平手。

因而建立 bfs 树，模拟该过程即可，注意没有边的情况。时间复杂度 $O(m)$。

12 L Three Permutations

题意：给定三个长度为 n 的排列 a, b, c，(x, y, z) 最开始为 $(1, 1, 1)$，每过一秒变为 (a_y, b_z, c_x)。q 次询问求变成 (x', y', z') 的最短时间。$1 \le n \le 10^5$，$1 \le q \le 10^5$。

解法：注意到每过 3 秒 (x, y, z) 变为 $((a \circ b \circ c)_x, (b \circ c \circ a)_y, (c \circ a \circ b)_z)$，其中 $a \circ b$ 表示置换的复合运算。这时就将三个位置相互独立开来，只需要研究每一个位置的答案。

一个置换是一个长度为 n 的数列，其中 $[1, n]$ 各出现一次。如果不断重复 $x \to f_x$ 的操作，从任意一个元素 i 出发，一定可以回到原始的 i，这个操作次数称为 i 的周期。如果把置换想象成一张图，即连接 $i \to f_i$ 的有向边，那么整个置换是若干个有向环构成的图。

置换可以复合，复合后仍然是置换。置换 a 和置换 b 复合，即给定 i，返回 a_{b_i}。

此时研究这三个复合置换的置换环性质，枚举初值（即从 1 秒、2 秒、3 秒开始计算）可以分别用 excrt 计算在 $3t, 3t+1, 3t+2$ 秒时变成 (x', y', z') 的最短时间，取最小值。时间复杂度 $O(q \log n)$。

中国剩余定理是用来解如下的同余方程组的：

$$\begin{cases} x \equiv \alpha_1 \pmod{\beta_1} \\ x \equiv \alpha_2 \pmod{\beta_2} \\ \vdots \\ x \equiv \alpha_n \pmod{\beta_n} \end{cases}$$

其中 β_i 之间互质。这时一定有解：

$$x \equiv \sum_{i=1}^{n} \alpha_i \cdot \frac{M}{\beta_i} \cdot \left(\left(\frac{M}{\beta_i} \right)^{-1} \bmod \beta_i \right) \pmod{M}$$

当模数之间不互质的情况下，我们可以考虑对每个方程进行质因子分解。这样原问题的方程可以变成若干个质因子的方程，这些分解出来的质因子方程可以按照原来的中国剩余定理合并到原方程。

形式化的，考虑方程 $x \equiv \alpha \pmod{\beta}$，若 $\beta = \prod_{i=1}^{k} p_i^{a_i}$，则可以分解为：

$$\begin{cases} x \equiv (\alpha \bmod p_1^{a_1}) \pmod{p_1^{a_1}} \\ x \equiv (\alpha \bmod p_2^{a_2}) \pmod{p_2^{a_2}} \\ \vdots \\ x \equiv (\alpha \bmod p_k^{a_k}) \pmod{p_k^{a_k}} \end{cases}$$

当我们把原方程组分解到如上的形式之后，这个时候各个模数要么互质，要么形如 p^{k_1} 与 p^{k_2}，即同一个质因子的不同次数的方程 $x \equiv \alpha_1 \pmod{p^{k_1}}$ 和 $x \equiv \alpha_2 \pmod{p^{k_2}}$。这个时候显然是低次服从高次，即若 $k_1 \le k_2$，则必须有 $\alpha_1 \equiv \alpha_2 \pmod{p^{k_1}}$。如果满足，则可删去低次方程。

当同一质因子的各方程合并完成后，现在剩下的方程的模数就都互质了，直接套用初始中国剩余定理的结论即可。

13 M Water

题意：给两个容积分别为 A, B 的水杯，每次可以执行以下的四种操作之一：

1. 把其中一个水杯装满水。
2. 把其中一个水杯中的全部水倒掉。
3. 把其中一个水杯中现有的水全部喝完。
4. 把一个杯子中的水尽可能转移到另一个水杯中，水不溢出。

为让懵哥喝掉恰好 x 体积的水，问最少要操作几次。T 组询问，$1 \le T \le 10^5$，$1 \le A, B \le 10^9$。

解法：记 $(r, s) = rA + sB$，其中 $r, s \in \mathbb{Z}$。显然两个杯子和喝掉的水在任意时刻都能表示成 (r, s) 的形式，因此由裴蜀定理当 $\gcd(A, B) \nmid x$ 时答案为 -1。当 A, B 中分别装有 $(r_A, s_A), (r_B, s_B)$ 水时，可以进行的操作如下：

1. 装水：将 A 变为 $(1, 0)$ 或将 B 变为 $(0, 1)$；
2. 倒水：将 A 或 B 变为 $(0, 0)$；
3. 喝水：将 A 或 B 变为 $(0, 0)$ 且喝掉的水 $+(r_A, s_A)$ 或 $+(r_B, s_B)$；
4. 将 A 转移到 B（B 转移到 A 同理）：
   - 若 $(r_A + r_B, s_A + s_B - 1) < 0$，则 A 变为 $(0, 0)$ 且 B 变为 $(r_A + r_B, s_A + s_B)$；
   - 否则 A 变为 $(r_A + r_B, s_A + s_B - 1)$ 且 B 变为 $(0, 1)$；

下面证明：$\text{ans} = \max\{2(r_0 + s_0), 2|r_0 - s_0| - 1\}$。

1. 假设 x 可以表示成 (r_0, s_0)，下证至多需要 $\max\{2(r_0 + s_0), 2|r_0 - s_0| - 1\}$ 次操作 。
   (a) 当 $r_0 \ge 0, s_0 \ge 0$ 时，$\text{ans} \le 2(r_0 + s_0)$。重复进行 r_0 次给 A 倒水、从 A 喝水，以及 s_0 次给 B 倒水、从 B 喝水即可。
   (b) 当 $r_0 s_0 < 0$ 时，$\text{ans} \le 2|r_0 - s_0| - 1$。不妨设 $r_0 > 0, s_0 < 0$。进行以下操作：

```
cntB = 0
重复 r_0 次:
    装满 A（操作 1）
    while B != (0,1) 且 A != (0,0):
        A 转移到 B（操作 4）
        if B == (0,1) and cntB < |s_0|-1:
            倒空 B（操作 2）
            cntB += 1
    if A != (0,0):
        喝 A（操作 3）
```

2. 下证至少需要 $\max\{2(r_0+s_0),2|r_0-s_0|-1\}$ 次操作。

   (a) $\text{ans}\geq 2(r_0+s_0)$。

   假设某次操作前已经喝了 (r_M,s_M) 水，此时两个杯子中有水量 $(r_A,s_A),(r_B,s_B)$，记 $m=r_M+s_M$，$a=r_A+s_A$，$b=r_B+s_B$，构造函数 $f(m,a,b)=2m+\max\{2a-1,0\}+\max\{2b-1,0\}$，注意该函数仅需要满足操作一次后函数值增大 1，且最后可以变化到 $2m$，这里仅给出其中一个符合条件的示例函数。可以证明在当前局面下进行任意操作后 $f(m',a',b')\leq f(m,a,b)+1$。由于初始状态为 $f(0,0,0)=0$，目标状态 $f(r_0+s_0,a,b)_{\min}=2(r_0+s_0)$，因此操作次数不少于 $2(r_0+s_0)$。

   (b) $\text{ans}\geq 2|r_0-s_0|-1$。

   当 $r_0-s_0\geq 0$。记 $m=r_M-s_M$，$a=r_A-s_A$，$b=r_B-s_B$，构造函数 $f(m,a,b)=2m+\max\{2a-1,0\}+\max\{2b,-1\}$。注意该函数仅需要满足操作一次后函数值增大 1，且最后可以变化到 $2m-1$，这里仅给出其中一个符合条件的示例函数。可以证明进行任意操作后 $f(m',a',b')\leq f(m,a,b)+1$。由于初始状态为 $f(0,0,0)=0$，目标状态 $f(r_0+s_0,a,b)_{\min}=2(r_0-s_0)-1$，因此操作次数不少于 $2(r_0-s_0)-1$。

   当 $r_0-s_0<0$ 时同理可证操作次数不少于 $2(s_0-r_0)-1$。

   一种更加简单的证明是，将 (r_0,s_0) 理解为一种操作。如果存在另一种操作比现有的 $\max\{2(r_0+s_0),2|r_0-s_0|-1\}$ 操作更少，那么一定是

   (a) 如果 r_0,s_0 都是正数，则一定让 A,B 中大的那一个的系数更大（贪心的增加一次多喝水的量可以减少总次数）。

   (b) 如果 r_0 为正 s_0 为负，则一定减少负数的系数（即被倒出的水量减少，这样让正的系数也可以减少）。

   但是不难发现，变化了 r_0 和 s_0 对应的次数和上面这个表达式的次数是相同的。即，A,B 系数变化前后的次数表达式仍然是关于 r_0,s_0 相同的函数表达式。因而不存在另一种更小的方法使得操作次数比 $\max\{2(r_0+s_0),2|r_0-s_0|-1\}$ 小。

综上 $\text{ans}=\max\{2(r_0+s_0),2|r_0-s_0|-1\}$。用 exgcd 求出离原点较近的 (r_0,s_0)，并对附近的 (r_0+kB,s_0-kA) 取 min 即可。每组数据复杂度 $O(\log x)$。