2025 “钉耙编程”中国大学生算法设计暑期联赛（5）题解

A. 一个更无聊的游戏

题目描述

给一棵 n 个点的树，q 次询问，每个点有 a_i, b_i，每次询问给定初始点 x 和等级 y，每次可以向相邻点移动，如果第一次移动到一个点 u，进行比较，如果等级 $y \geq a_u$，那么 y 增加 b_u，否则结束。问结束前的最高等级是多少。

数据范围：$n, q \leq 10^5$。

题解

首先可以考虑数组上的情况。如果经过了某个点 i，那么接下来只需要考虑所有 $a_j > a_i$ 的 j，因为显然这时候遇到更小的 a 都一定可以胜利。所以可以考虑按照 a 的最大值分治建立一棵笛卡尔树，那么我们移动的过程显然就变成了在树上不断往上走，每走到一个点，这个点的所有子树就一定都可以访问，设 $S_x = \sum_{y \in subtree(x)} b_y$，那么当前在 x，能走到笛卡尔树上 x 的父亲的条件就是 $a_{fa_x} - S_x \leq y$，即打败所有 a 较小的儿子们之后能通过 fa_x，这里的 y 是初始等级。

所以建立笛卡尔树后，对于每次询问，只需要通过树上倍增或者树链剖分等方法维护，找到第一个大于 $a_{fa_x} - S_x > y$ 的点 x，那么 $y + S_x$ 就是答案。

那么对于树上的情况，如果我们能对树建出笛卡尔树，后面的步骤就是一样的了。

我们要做的是找到一棵树中的最大值，然后递归各个子树。

一种方法是可以做类似点分治，用线段树维护 dfs 序上的最值，找到当前树中的最大值所在点 rt，对于 rt 的子树可以直接递归，问题是 rt 向上到根的部分，并不是 dfs 序的一个区间，一个解决方式是先将所有子树递归处理，处理之后赋值为 $-\infty$，这样就不会对 rt 到根的部分产生影响。

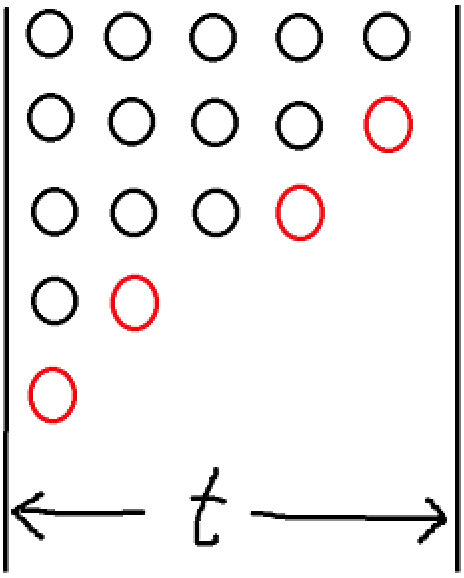
另一个更简单的方法是将建树过程倒过来，直接按照 a 的值从小到大枚举每个点 u，用并查集维护每个笛卡尔树和合并笛卡尔树的过程。

两种方法的时间复杂度都是 $O(n \log n)$。

B. "合理"避税

二分答案。设当前二分到的月份数为 t，那么对于第 i 个人而言，其最多只能收 a_i 元钱，可以拆成这个人先收若干次 k 元，最后再收 $a_i \bmod k$ 元。

如下图，每一行为同一个人，每个黑色点表示收 k 元，红色点表示收 $a_i \bmod k$ 元:



如果这个人对应的 $a_i/k \geq t \cdot p$，那么他每个月都可以有 k 元钱的贡献。

如果在规定时间内，所有人能贡献的黑点数量至少为 $t \cdot p$，那么我们总是能找到一种方案，使得所有人的收款至少为 $t \cdot p \cdot k$。这是因为：每个人在时间范围内最多只能贡献 t 个黑点，而黑点总数 $\geq t \cdot p$，那么每个月我们都能找到 p 个至少还有一个黑点的人。那么此时只需要比较 $t \cdot p \cdot k$ 与 m 之间的关系即可。

如果黑点数量不足 $t \cdot p$，那么黑点全部贡献完之后，必然会贡献剩余的部分红点。此时能够贡献的红点的最大个数为 $c_{red} = t \cdot p - \sum_i \min(a_i/k, t)$ （暂不考虑实际红点数量）。我们可以发现：必然会选择最大的 c_{red} 个红点。这是因为：

- 除了全选择黑点的月份之外，最后一个月份可能是若干黑点+若干红点的策略。此时可以通过跟之前全是黑点的选择进行调整，从而将红点换作最大的那几个。

那么就首先将 a_i 按照 $a_i \bmod k$ 从大到小排序，这样二分 check 的时候就不需要排序了。时间复杂度 $O(n(\log n + \log m))$。

C. 黑白球

先考虑给的那个问题怎么做。

注意到我们一次操作后颜色只跟选择的 y 有关，所以统计一个球覆盖了哪些球。

同时我们可以从概率的角度考虑。

一个球不被上面的操作覆盖的概率为 $\frac{m-1}{m}$。

枚举向下覆盖的长度 l，那么一个球的贡献有两种：

1. 不被覆盖且向下覆盖，概率为 $\frac{m-1}{l!m^{l+1}}$
2. 被覆盖且在被覆盖之前向下覆盖，概率为 $\frac{l}{(l+1)!m^{l+1}}$

显然可以前缀和优化，$n-1$ 位置的贡献特判，这样每个位置就有一个贡献 c_i，我们的答案只跟每个位置的黑球个数有关。

然后考虑交换相邻两个数，可以设 $f_{i,j}$ 表示交换 i 次，所有情况下 j 位置上的黑球个数和。

这样就得到了一个 $\Theta(n^2)$ 的做法。

考虑 f 的转移方程 $f_{i,j} = f_{i-1,j-1} + f_{i-1,j+1} + (n-3)f_{i-1,j}$。

我们需要特判 $0, n-1$ 的转移，这是不好的，考虑如何将特判去掉。

考虑一个构造，我们令 $f_{i,j} = f_{i,2n-1-j}\ (j \geq n)$，然后变成一个环，此时我们就不需要特判了。

因此令 $F_i(x) = \sum_{j=0}^{2n-1} f_{i,j}x^j, P(x) = x + n - 3 + \frac{1}{x}$。

转移就是 $F_i(x) = F_{i-1}(x) \times P(x) \pmod{x^{2n} - 1}$，就是一个循环卷积。

接下来的多项式乘法都定义为循环卷积。

令 $C(x) = \sum_{i=1}^{n} c_{i-1}x^i$。

对于一个 k，所求即为 $[x^0]F_0(x) \times C(x) \times P(x)^k$。

接下来有两种做法。

一个是注意到 $P(x)$ 的次数很小，可以分治 NTT。

一个是利用循环卷积的性质，求出点值，答案变为 $\frac{1}{2n}\sum_{i=0}^{2n-1} f_i c_i p_i^k$，也就是要求 $[x^k]\frac{1}{2n}\sum_{i=0}^{2n-1}\frac{f_i c_i}{1-p_i x}$，分治 NTT 然后求逆即可。

复杂度 $\Theta(n \log^2 n)$。

D. 信赖跃迁：置换轨迹计算

题目描述

给定长度为 n 的置换 q 和一个长度为 m 的整数序列 $k_1, k_2, \ldots, k_m$，计算存在多少个长度为 $m+1$ 的置换序列 $p_0, p_1, \ldots, p_m$ 能够满足:

$$p_0 \circ p_1^{k_1} \circ p_2^{k_2} \circ \cdots \circ p_m^{k_m} = q$$

其中 $\circ$ 是置换的复合运算。结果对 $10^9 + 7$ 取模。

题解

无论 $p_1, \ldots, p_m$ 是何种置换序列，p_0 都可以唯一确定地将置换复合结果变成 q。因此答案为 $(n!)^m \pmod{10^9 + 7}$。

E. 四角洲行动

题目描述

有若干个 $1 \times 1, 1 \times 2, 1 \times 3, 2 \times 2$ 的物品和格子，物品有价值，问最多能把多少价值的物品放入格子内。每个大小的物品个数和格子个数都小于等于 100。

题解

首先肯定对每个大小的物品按价值排序，考虑格子和物品的尺寸非常特殊。可以考虑直接枚举格子放物品的情况，求出每个大小的物品要几个，然后计算价值，最后对所有情况取最大值。

对于 2×2 的格子，要么放一个 2×2，要么可以直接看成 2 个 1×2 的格子进行考虑（1×2 的格子的拆分可以在后面枚举到 1×2 的时候再进行考虑）。

对于 1×3 的格子，要么放一个 1×3，要么可以看成 1 个 1×2 和 1 个 1×1 的格子。

对于 1×2 的格子，要么放一个 1×2，要么可以看成 2 个 1×1 的格子。

那么对于每种格子都可以直接枚举几个放同样大小的物品，剩下的直接拆分成更小的格子。

这样就可以在 $O(n^3)$ 的时间枚举所有的物品需求情况，那么对于每个大小的物品，显然只要价值最大的那些，提前从大到小排序做前缀和即可 $O(1)$ 的时间计算物品最大价值之和。因此总时间复杂度为 $O(n^3)$。

F. 支配游戏

题目描述

给定一个由若干连通块组成的无向图，任意两个节点间最多只有一条边，每个连通块是简单路径或者简单环，称一个节点 u 支配另一个节点 v 当且仅当 u 和 v 相邻。两个玩家交替在图上选点博弈，每一轮，对应玩家选取一个能够支配至少一个未被支配节点的节点，并将该节点和所有被其支配的节点标记为已支配。若当前玩家无法选取节点，则该玩家输掉游戏。现在需要判定先手是否必胜。

题解

该问题为公平博弈问题，所以考虑采用 SG 定理来求解问题。

令 $P_{i,n}$ $(n \geq 2, i \in \{0,1,2\})$ 为长度为 n，左右端点中的 i 个端点被支配且其余中间节点均未被支配的路径，C_n $(n \geq 3)$ 为长度为 n 且所有节点均未被支配的环，则通过打表和归纳法可得所有这些图的 SG 值如下所示。

- 当 $i = 2$ 时。对于 $n \leq 2$，$\mathrm{SG}(P_{2,n}) = 0$；对于 $n > 2$ 有

$$\mathrm{SG}(P_{2,n}) = \mathrm{mex}_{i=1}^{n}(\mathrm{SG}(P_{2,i-1}) \oplus \mathrm{SG}(P_{2,n-i})) = \begin{cases} 2 & n \bmod 4 = 0 \\ 3 & n \bmod 4 = 1 \\ 0 & n \bmod 4 = 2 \\ 1 & n \bmod 4 = 3 \end{cases}$$

- 当 $i = 1$ 时。对于 $n \leq 1$，$\mathrm{SG}(P_{1,n}) = 0$；对于 $n > 1$ 有

$$\mathrm{SG}(P_{1,n}) = \mathrm{mex}_{i=1}^{n}(\mathrm{SG}(P_{2,i-1}) \oplus \mathrm{SG}(P_{1,n-i})) = \begin{cases} 3 & n \bmod 4 = 0 \\ 0 & n \bmod 4 = 1 \\ 1 & n \bmod 4 = 2 \\ 2 & n \bmod 4 = 3 \end{cases}$$

- 当 $i = 0$ 时。对于 $n = 3$，$\mathrm{SG}(P_{0,n}) = 2$；对于 $n \neq 3$ 有

$$\mathrm{SG}(P_{0,n}) = \mathrm{mex}_{i=1}^{n}(\mathrm{SG}(P_{1,i-1}) \oplus \mathrm{SG}(P_{1,n-i})) = \begin{cases} 0 & n \bmod 4 = 0 \\ 1 & n \bmod 4 = 1 \\ 1 & n \bmod 4 = 2 \\ 3 & n \bmod 4 = 3 \end{cases}$$

- 对于 $n \geq 3$ 有

$$\mathrm{SG}(C_n) = \mathrm{mex}(\mathrm{SG}(P_{2,n-1})) = \begin{cases} 1 & n \bmod 4 = 3 \\ 0 & \text{otherwise} \end{cases}$$

最后把所有连通块的 SG 值异或起来即可得到整个图的 SG 值。若 SG 值为 0，则先手必败；否则先手必胜。

G. k-MEX

设 $P(i)$ 表示 $MEX = i$ 对应的概率，那么答案就是 $\sum_{i=1}^{k} i \cdot P(i)$。

将其展开后竖向观察：

$$\begin{aligned}
&\sum_{i=1}^{k} i \cdot P(i) = \\
&P(1) + \\
&P(2) + P(2) + \\
&P(3) + P(3) + P(3) + \\
&\dots \\
&P(k) + P(k) + P(k) + \dots + P(k)
\end{aligned}$$

可以发现所求等于 $\sum_{i=1}^{k} \mathbb{P}[MEX \geq i]$。根据概率定义可知，$\mathbb{P}[MEX \geq i]$ 等于所有 $MEX \geq i$ 的情况数除以所有选择方案数（也就是 $\binom{n}{k}$）。

要使得 $MEX \geq i$，那么 $0 \sim i-1$ 都至少必须被选中才行，而 $i \sim n-1$ 是否被选中是不需要关心的。那么也就可以计算出对应的式子：

$$\mathbb{P}[MEX \geq i] = \frac{\binom{n-i}{k-i}}{\binom{n}{k}}$$

则所求为

$$\sum_{i=1}^{k} \frac{\binom{n-i}{k-i}}{\binom{n}{k}}$$

下面化简上式。观察杨辉三角，有

$$\begin{aligned}
&\sum_{i=1}^{k} \binom{n-i}{k-i} \\
&= \binom{n-k}{0} + \binom{n-k+1}{1} + \binom{n-k+2}{2} + \dots + \binom{n-1}{k-1} \\
&= \binom{n-k+1}{0} + \binom{n-k+1}{1} + \binom{n-k+2}{2} + \dots + \binom{n-1}{k-1} \\
&= \binom{n-k+2}{1} + \binom{n-k+2}{2} + \dots + \binom{n-1}{k-1} \\
&= \dots \\
&= \binom{n-1}{k-2} + \binom{n-1}{k-1} \\
&= \binom{n}{k-1}
\end{aligned}$$

则

$$\begin{aligned}
&\sum_{i=1}^{k} \frac{\binom{n-i}{k-i}}{\binom{n}{k}} \\
&= \frac{1}{\binom{n}{k}} \sum_{i=1}^{k} \binom{n-i}{k-i} \\
&= \frac{\binom{n}{k-1}}{\binom{n}{k}} \\
&= \frac{k}{n-k+1}
\end{aligned}$$

H. 双端魔咒

对于 pre 集合，如果一个串 p_1 是另一个串 p_2 的前缀，那么显然我们可以去掉 p_2。因为如果一个串以 p_2 为前缀，那么必然以 p_1 为前缀。同理，对于 suf 也可以类似去掉冗余后缀。这一操作可以用 Trie 来实现。以 pre 为例，把所有字符串都插入 Trie 之后，从根节点进行一次 dfs 或者 bfs 直到遇到第一个串就停止，这样就去掉了冗余的前缀。

这样去掉之后的好处是：对于查询串 S 的任意一个位置 i 而言，最多只有一个 pre 中的前缀以 i 开头。这样，我们对 pre 建立出 AC 自动机，再将 S 放到 AC 自动机上进行匹配，就可以找到每个前缀出现的位置了。所有位置形成的集合大小就是 $O(|S|)$ 的。对于 suf 而言也同理。

那么接下来就可以枚举每个位置 i，找到 i 之后有多少个位置是后缀就行。

但还有些细节问题。如下图所示，红色表示我们枚举到的一个前缀，蓝色表示 i 之后的一个后缀，但显然这样的串是不符合要求的，因为蓝色部分并非实际形成的子串后缀。

不过对于每个前缀而言，其对应在 suf 上的蓝色串都是相同的，所以可以首先将 pre 中的每个串都在 suf 集合对应的 AC 自动机上进行一次匹配，统计有多少蓝色串即可。

时间复杂度：$O(|\Sigma|(|S|+|T|))$。

I. 切排列

从 $A_1, A_2, \ldots, A_n$ 顺序依次考虑。找到 A_i 在 B 中出现的位置（设为 P_{A_i}），如果 $P_{A_{i+1}} \neq P_{A_i}+1$，那就说明 A_i 与 A_{i+1} 在 B 中不是连续的，此时必然分段。

容易证明这样分段的段数是最少的。

时间复杂度：$O(n)$。

J. 随机反馈

考虑期望 dp，设 f_i 表示从 i 分钟开始提交的最小罚时期望。那么有两个决策：是否在第 i 分钟提交。转移方程为：

$$f_i = \min(f_{i+1}, (1-p_i)\times i + p_i \times (f_{i+1}+20))$$

直接从后往前 dp 即可。时间复杂度 $O(n)$。

注意：读入浮点数相较于读入整数存在性能差距。此差距无关于平台，原因在于字符串与 IEEE 754 标准浮点数之间的近似转化需要特定算法，此处提供一组体现性能差距的测试数据。本题中不涉及读入浮点数，采用正确的类型读入与输出并不会导致严重的性能降级。

K. 营火

考虑双指针。首先把所有黑点加入进去，然后不断按照 y 坐标从大到小删除黑点，然后另一边按照 y 坐标从上到下加入白点，维护中间的所有火种是否连通。判断是否连通可以使用 LCT，把火种的 $size$ 设为 1，其他黑白点设为 0，那么加入点和删除点可以用 LCT 维护，判断整棵树 $size$ 是否等于 n 即可。

唯一的问题是现在的将所有火种连通时，可能构成的是图而不是一棵树，我们需要时刻维护 LCT 为一棵树，所以可以化边为点，每次当连边的时候，删去坐标最大的一条边。注意双指针的时候对于同一个坐标的点特殊处理。

时间复杂度 $O(n\log n)$。

L. 最小值

题目描述

求

$$\min_{p\neq q}||a_p-a_q|-|b_p-b_q||$$

数据范围：$n\leq 10^5$。

题解

首先我们证明 $|(a_p+b_p)-(a_q+b_q)|$ 和 $|(a_p-b_p)-(a_q-b_q)|$ 中的一个是 $||a_p-a_q|-|b_p-b_q||$。我们分类讨论四种情况来证明：

- 当 $a_p\geq a_q,b_p\geq b_q$ 时，$||a_p-a_q|-|b_p-b_q||=|(a_p-b_p)-(a_q-b_q)|$；
- 当 $a_p\geq a_q,b_p<b_q$ 时，$||a_p-a_q|-|b_p-b_q||=|(a_p+b_p)-(a_q+b_q)|$；
- 当 $a_p<a_q,b_p\geq b_q$ 时，$||a_p-a_q|-|b_p-b_q||=|(a_q+b_q)-(a_p+b_p)|$；
- 当 $a_p<a_q,b_p<b_q$ 时，$||a_p-a_q|-|b_p-b_q||=|(a_q-b_q)-(a_p-b_p)|$。

实际上，后两种情况结果与前两种对应相同，由此命题得证。

从讨论中可以整理得到：

- 如果 $(a_p-a_q)(b_p-b_q)\geq 0$，则 $||a_p-a_q|-|b_p-b_q||=|(a_q-b_q)-(a_p-b_p)|$；
- 如果 $(a_p-a_q)(b_p-b_q)<0$，则 $||a_p-a_q|-|b_p-b_q||=|(a_q+b_q)-(a_p+b_p)|$。

接下来我们证明 $||a_p-a_q|-|b_p-b_q||$ 是 $|(a_p+b_p)-(a_q+b_q)|$ 和 $|(a_p-b_p)-(a_q-b_q)|$ 中较小的一个。我们仍然讨论上述两种情况来证明。

当 $(a_p-a_q)(b_p-b_q)\geq 0$ 时，讨论 $|(a_p-a_q)+(b_p-b_q)|$ 与 $|(a_p-a_q)-(b_p-b_q)|$ 的大小关系，令 $x=a_p-a_q$，$y=b_p-b_q$，之后讨论 x,y 的大小关系与 x,y 与 0 的大小关系后可知，此时 $|(a_p-a_q)-(b_p-b_q)|\leq|(a_p-a_q)+(b_p-b_q)|$。

同理，当 $(a_p-a_q)(b_p-b_q)<0$ 时进行类似讨论，此时 $|(a_p-a_q)+(b_p-b_q)|\leq|(a_p-a_q)-(b_p-b_q)|$。

综上命题得证。

由此，要求的值为

$$\min_{p\neq q}||a_p-a_q|-|b_p-b_q||=\min_{p\neq q}\{|(a_p+b_p)-(a_q+b_q)|,|(a_p-b_p)-(a_q-b_q)|\}$$

令 $A_i=a_i+b_i,B_i=a_i-b_i$，对两数组分别排序后分别差分，取最小差即可。

时间复杂度：$\mathcal{O}(n\log n)$，空间复杂度：$\mathcal{O}(n)$。