第 1 届哈尔滨工业大学一校三区程序设计竞赛

命题报告

出题人

2025 年 12 月 28 日

Hello, Harbin Institute of Technology!

题意

- 输出 hello 和哈尔滨工业大学的英文缩写
- 输出”hello hit”

Bingos!

题意

- 现有一个 5×5 的网格图，每个位置有 0/1
- 若任意一行，列或对角线有 5 个 1，输出 Bingo!；否则按照下面规则计算每个 1 的价值和
- 1 的价值定义为，这个 1 与其他 1 所构成的最长连线长度。

- 枚举行，列，对角线，判断是否填满 5 个。是则输出 Bingo!
- 否则枚举每个 1，在 8 个方向上枚举计算能延伸多远，然后计算最长连线长度。最后求和

约数游戏

题意

- 给定两个正整数 a, b。
- 两人轮流操作，每次可以选择其中一个数，将其修改为它的一个真约数（小于自身的约数）。
- 无法进行操作的一方判负（即当 $a = 1, b = 1$ 时）。
- 判断先手是否必胜。$a, b \leq 10^9$。

约数游戏

- **SG 函数推导：**考虑单个数字 n 的游戏。
  - 设 $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$，定义 $\Omega(n) = \sum e_i$ 为 n 的质因子指数之和（即质因子的总个数）。
  - 操作规则是将 n 变为真约数 d，这等价于从 n 的质因子集合中移除至少一个质因子。
  - 因此，对于任意真约数 d，有 $\Omega(d) < \Omega(n)$。且对于任意 $k < \Omega(n)$，总存在一个真约数 d 使得 $\Omega(d) = k$。
  - 这完全等价于一个大小为 $\Omega(n)$ 的 Nim 堆取石子游戏。故 $SG(n) = \Omega(n)$。

约数游戏

- **结论：**计算 $\Omega(a)$ 和 $\Omega(b)$，若 $\Omega(a) \oplus \Omega(b) \neq 0$ 则先手必胜，否则必败。
- **实现：**对 a 和 b 进行试除法分解质因数即可，单次复杂度 $O(\sqrt{a}+\sqrt{b})$。
- 总复杂度 $O(T \times (\sqrt{a}+\sqrt{b}))$。
- **注意：**如果 #define int long long 会因为 long long 计算速度慢而 TLE。

小丑牌

题意

- 你有 n 张手牌和 m 张小丑牌，你一回合可以打出至多 5 张牌和 5 张小丑牌，且决定小丑牌的生效顺序问你这一回合能获得的最大分数是多少。
- $n, m \leq 15$

- 只有 6 种基础倍率和筹码，对这六种计算牌面筹码能带来的最大值。然后暴力枚举小丑牌的选择和生效顺序
- 复杂度 $6 * A_{15}^{5}$

不是小丑牌

题意

- 给定三种颜色的牌，数量分别为 C,S,D。每次选取数量最多的两种牌（若有多种方案则随机选一种），移除这两种牌各一张，并增加一张第三种牌。问能否确定最后剩下的牌是什么，并回答是什么。
- $-T \leq 10, C, S, D \leq 8$。

- 数据范围极小，暴力搜索即可，复杂度 $O(CSD)$。
- 能够发现每次操作都不改变场上三种牌的相对奇偶性，比如原先是 1 奇 2 偶，操作完还是 2 奇 1 偶，原先和其他牌奇偶性不同的牌还是奇偶性不同。最终剩下的一定是原先与其他牌奇偶性不同的那张。判断一下直接输出复杂度 $O(1)$.

二维码生成器

题意

- 给定 26 个分区的数据码字，按照题目给定的规则，还原出二维码

- 给出一种好的实现方式：手动对每个位置处于哪个分区或者不属于分区进行预处理；对每个分区是升序排列还是降序排列进行预处理。这样我们顺序遍历每一个位置时，就可以直接得到它应该填写什么信息。

回文串魔法

题意

- 给定一个字符串。把任意一个字符变成字符 c 的代价为 v_c。你一开始可以交换字符串上的两个位置（最多交换一次），问把字符串变成回文串的最小代价是多少
- $n \leq 10^6 \quad v_a, \ldots, v_z \leq 10^9$

- 因为只要求回文串，我们仅需考虑对称的两个位置的字符情况，且不对称的位置间是独立的。发现本质上只有 $O(26 * 26)$ 种字符对
- 交换操作最多只涉及两对，可以枚举这两对还有他们的交换情况，计算出改变量后贡献给答案。

追忆

题意

- 给定序列 a，求一个区间 $[l, r]$，使得（区间内不同元素个数 $+$ 区间外不同元素个数）最大。
- $N \leq 10^6$。

- 设 D 为整个序列的不同元素总数，S_{in} 为区间内出现的元素集合，S_{out} 为区间外出现的元素集合。
- 目标是最大化
  $|S_{in}| + |S_{out}| = |S_{in} \cup S_{out}| + |S_{in} \cap S_{out}| = D + |S_{in} \cap S_{out}|$。
- 问题转化为：最大化“既在区间内出现，又在区间外出现”的元素个数。
- 一个元素 x 对答案有贡献，当且仅当它被区间 $[l, r]$ “切断”（即部分在内，部分在外）。

追忆

- 使用扫描线，枚举区间的左端点 l，然后维护每一个可能的右端点 r 对应的贡献值。
- 设左端点元素 x 的出现位置序列为 $p_1, p_2, \ldots, p_k$。
- 对于固定的 l，若 $p_i < l \leq p_{i+1}$，则 x 在区间外已有分布，无需更多考虑
- 若 $l == p_1$，说明可能有某个 $[l, r]$ 把 $p_1, p_2 \ldots p_k$ 完全包含，应当对右端点在 $[p_k, n]$ 的贡献值减 1。
- 通过线段树动态维护区间的加减操作。
- **复杂度**：$O(N \log N)$，足以通过 10^6 数据。

子树 mex

题意

- 给定一棵树，求每个节点的子树中，点权集合的 mex（未出现的最小非负整数）。
- $N \leq 5 \times 10^5, a_i \leq 10^9$。

- **转化：**显然 mex 的值不会超过子树大小，也就不会超过 N。大于 N 的权值视为 $N+1$。
- **按秩合并**为每个子树的 set 维护一个 mex 指针，当两棵子树合并时采用按秩合并，当合并时，mex 指针更新为二者的较大值，然后不断向后移动直到找到第一个不在集合中的值。
- **复杂度分析**每个节点被合并的次数不超过 $O(\log N)$ 次，且每个点每次合并至多只会让一个 mex 向大的方向移动一次，因此总复杂度为 $O(N \log N)$。
- 时间复杂度 $O(N \log N)$。

过程淘汰 II

题意

- 给定长为 n 的序列 A 和整数 K，计算 ans：

$$ans = \left(\sum_{\substack{p \in \mathbb{P} \\ p \mid K}} \sum_{1 \le i \le j \le n} (A_i \bmod p) \oplus (A_j \bmod p) \times \min_{t=i}^{j} (A_t \bmod p) \right) \pmod{998244353}$$

- $n \le 5 \times 10^5, K \le 10^8$

过程淘汰 II

- 首先在根号复杂度对 K 进行质因数分解，然后建立笛卡尔树。只需要知道每个子树中在二进制下第 i 位的 0 和 1 有多少个
- 合并的时候记录贡献即可。因为笛卡尔树的性质，满足区间 min 恰好是这个子树的根。
- 复杂度 $O(nlog^2 n)$

餐券设计

题意

- 构造一个长度最小的非降序列 a，使得其子集和能覆盖 $1 \ldots n$。
- 输出最小长度 k，以及字典序最小和最大的方案。
- $n \leq 10^9$。

- **最小长度 k：** 显然由二进制原理可知 $k = \lceil \log_2(n+1) \rceil$。

餐券设计

- **字典序最小：**
  - 策略：**逆向贪心**。为了让字典序最小（即前面的数小），我们需要让后面的数尽可能大。
  - 在覆盖 $1 \ldots n$ 的序列中，最大的数 a_k 最多只能是 $\lceil n/2 \rceil$（否则无法通过子集和组合出 n）。
  - 算法：每次取 $val = \lceil n/2 \rceil$ 加入序列，令 $n \leftarrow n - val$（即 $\lfloor n/2 \rfloor$），重复直到 $n = 0$。最后排序输出。
  - 例如 $n = 10 \rightarrow 5, 3, 1, 1$，排序后为 $1, 1, 3, 5$。

餐券设计

- **字典序最大：**
  - 策略：**正向贪心**。为了让字典序最大，我们需要让前面的数尽可能大（增长快）。
  - 算法：优先使用 2 的幂次 $1, 2, 4, 8 \ldots$，直到剩余的数 rem 不足以下一个幂次。
  - **修正：**此时序列末尾可能是 $\ldots, 2^k, rem$。如果 $rem < 2^k$，这会导致字典序变小（如 $1, 2, 4, 1$ 排序成 $1, 1, 2, 4$）。
  - 代码做法是将最后两个数 2^k 和 rem 合并求和，然后尽可能平均分配（$\lfloor sum/2 \rfloor, \lceil sum/2 \rceil$）。
  - 例如 $n = 8 \to 1, 2, 4, 1 \xrightarrow{\text{修正}} 1, 2, 2, 3$。这样保证了字典序最大。

Jump Jump

题意

- 构造一个排列，使其最长上升子序列 (LIS) 的个数恰好为 K。
- $K < 2^{30}$，排列长度 $n \leq 1500$。

- **构造策略：** 利用二进制拆分 $K = \sum 2^{p_i}$。

Jump Jump

- **单元构造 (Block)：**对于二进制第 i 位 $(0 \le i < 30)$，构造一个长度为 $30 + i$ 的序列：
  - **前半部分：**长度 $2i$，每两个数交换 (如 $2, 1, 4, 3 \ldots$)。这部分 LIS 长度为 i，数量为 2^i。
  - **后半部分：**长度 $30 - i$，严格递增。这部分 LIS 长度为 $30 - i$，数量 1。
  - **Block 性质：**每个 Block 的 LIS 总长恒为 $i + (30 - i) = 30$，数量为 2^i。
- **整体拼接：**
  - 按二进制位从高到低 (29 到 0) 输出 Block。
  - **值域分配：**代码中先累加总长度，输出时倒序分配值域。即：先输出的 Block 使用**较大**的数值，后输出的 Block 使用**较小**的数值。
  - **隔离性：**整个序列在宏观上是数值递减的 (Block A 值域 $\gg$ Block B 值域)。因此 LIS 不能跨越 Block，只能在 Block 内部产生。
  - 总数量即为各 Block 数量之和：$\sum 2^{bit} = K$。

门不能从这一侧打开

题意

- 给定一张带权无向图。存在 K 个限制，形如 k 个 (a_i, b_i, c_i) 的三元组，要求路径上不能存在相邻的三个点 a_i, b_i, c_i。分别计算出在次条件下，从 1 号点走到 2,3...n 号点的最短距离。无法到达则输出 *infinity*
- $n, m, k \leq 5 \times 10^5$

- 本题要求在带权无向图中求最短路，但增加了一个特殊的约束：不能连续经过给定的三元组。这需要我们知道每个节点 b_i 的前驱是什么。
- 由于是最短路问题，考虑使用打上补丁的 dijkstra。

门不能从这一侧打开

- 定义 $dist[u][v]$ 表示经过边 (u, v) 到达节点 v 的最短路径长度。因为只有 m 条边，因此只有 $O(m)$ 个有效状态
- 当 dijkstra 取出这个状态 (u, v) 时, 利用前驱的信息来更新 v 每一条出边，此时可以直接判断是否会违背约束。
- 但这样更新的复杂度可能会飙升到 $O(m^2 logm)$。
  - 发现一条重要的性质，由于 dijkstra 的单调性，若点 v 的某条出边 (v, k) 在之前被一个状态 (u, v) 合法更新，那么在这之后任意的 $(u^{'}, v)$ 不会再更新它。
  - 启发我们对每个点存一个 map，储存下还未被更新的边。当我们合法更新一条边之后，就将其从 map 中删除。
  - 不合法当且仅当 (u, v, k) 是一个条件三元组，由于 (u, v) 只会被用来更新一次，所以出现不合法的总次数不会超过 K
- 总复杂度 $O((m + k)log)$
- 如果三元组的限制使用 map 来判断，会多出来一个 log。

Thanks!