2025 年广东工业大学 ACM 程序设计竞赛新生赛（决赛）题解

广东工业大学 ACM 集训队

2025 年 11 月 30 日

题目难度分布

题目难度分布：

- 签到题：A，L
- Easy：K，H，M
- Easy-Medium：C，E，J
- Medium：F，I，N，G
- Medium-Hard：B，D
- 好像没有高难防 ak!

A. 极差最大的区间

简明题意

给定一个长度为 n 的序列，找出一个连续子区间，使得该区间的极差（最大值减最小值）最大。

签到题，整个序列的极差一定是最大的极差。因为任何子区间的最大值不会超过全局最大值，最小值不会低于全局最小值。

因此，我们只需要找到极差 = 最大值 - 最小值，然后输出 1 n 极差。

时间复杂度 $O(n)$。

L. 气球采购

简明题意

给定 n 道题的通过率 p_i/q_i 和总人数 m，计算每道题所需气球数 $\lceil m \cdot \frac{p_i}{q_i} \rceil$，如果库存不足则补齐，求总购买量。

对于每一道题，计算需要的总气球数 $need = \lceil \frac{m \cdot p_i}{q_i} \rceil$。为了避免浮点误差，可以使用整数运算：$need = (m \cdot p_i + q_i - 1)/q_i$（整数除法）。

如果当前库存 $w_i < need$，则累加答案 $need - w_i$。最后输出总和。

时间复杂度 $O(n)$。

K. 线段覆盖

简明题意

给定数轴上 n 个点，允许使用至多 k 条线段覆盖所有点，求线段总长度的最小值。需输出 $k = 1 \ldots n$ 的所有答案。

用 1 条线段覆盖所有点，成本为 $x_n - x_1$。此时中间包含所有相邻点的间距。

如果我们允许增加 1 条线段（共 2 条），我们实际上可以在某个位置断开，即“省去”了某一段相邻点之间的距离。为了让总长度最小，我们应该省去最大的那个间距。

以此类推，使用 k 条线段等价于在 $x_n - x_1$ 的基础上，减去前 $k-1$ 大的相邻间距 $(x_{i+1} - x_i)$。

将所有 $n-1$ 个间距排序并计算前缀和即可，时间复杂度 $O(n \log n)$。

H. 能量汇聚

简明题意

从 0 跳到 n，每次可跳 1 步或 2 步，不能连续跳两次 1 步。每到一个点获得能量 E_i，求最大总能量。

简单的线性 DP。定义状态：

- $dp[i][0]$：到达位置 i，且最后一步是跳 1 步带来的最大能量。
- $dp[i][1]$：到达位置 i，且最后一步是跳 2 步带来的最大能量。

转移方程：

- $dp[i][0] = dp[i-1][1] + E_i$（因为不能连续跳 1 步，所以前一步必须是跳 2 步过来的）。
- $dp[i][1] = \max(dp[i-2][0], dp[i-2][1]) + E_i$（跳 2 步没有限制）。

最终答案为 $\max(dp[n][0], dp[n][1])$。时间复杂度 $O(n)$。

M. 敌人的敌人

简明题意

给定一棵树，定义敌人的敌人是朋友（即距离为 2 的节点对互为朋友），求朋友最多的节点及其朋友数量。

在树（或任何图）中，节点 u 的 “朋友” 就是它的邻居的邻居（不包括 u 自己）。

因此，对于节点 u，它的朋友数量等于 $\sum_{v \in neighbors(u)} (degree(v) - 1)$。

我们只需要遍历每个节点，枚举其邻居 v 并累加 $degree(v) - 1$ 即可。

由于树的边数为 $n - 1$，总时间复杂度为 $O(n)$。

C. 区间乘

简明题意

给定序列 a，多次查询是否存在连续子区间乘积等于 x。

由于 a_i 为正整数，区间乘积增长非常快。对于 $a_i \geq 2$，长度超过 30 的区间乘积就会超过 10^9。

我们可以忽略序列中的 1，记录所有非 1 元素的位置，然后枚举非 1 元素作为区间的起点，向后累乘，直到乘积达到或超过 10^9 ，在然后将所有可能的区间乘积存起来（使用 set 或普通数组）。

如果是 $x = 1$，只需判断序列中是否有 1 ；如果 $x > 1$ ，则检查保存的区间乘积是否包含 x（如果使用数组，可以排序后用二分判断）。

总复杂度 $O(n\log(\max x) + q\log(n\log(\max x)))$ 。

E. 质数变化

简明题意

求两个四位质数 A 到 B 的最短变换路径，每次变换改变一位数字且结果仍为质数。

标准的广度优先搜索（BFS）求最短路。

首先用筛法或试除法预处理出 10000 以内的所有质数。以 A 为起点进行 BFS，每次枚举 4 位数字中的某一位变为 $0 \dots 9$，检查变换后的数字是否为质数且未访问过。若是，则加入队列。时间复杂度 $O(10000T)$ 。

对于此题，经打表发现，任意 10000 以内的 A 到 B 的最小操作次数最大值为 7 ，运行速度非常快，因此不预处理质数（实时暴力判断）也可以通过。

J. 协会的实验

简明题意

动态构建字符串（支持前端和后端插入字符），实时维护子序列"acm" 的数量。

维护 6 个变量：cnt_a, cnt_c, cnt_m（单个字符数量），cnt_{ac}, cnt_{cm}（长度为 2 的子序列数量），cnt_{acm}（答案）。

- **后端插入'm'**：新增的'm' 可以与前面所有的"ac" 组成"acm"。更新 $cnt_{acm} += cnt_{ac}$，同时更新 cnt_m, cnt_{cm}。
- **前端插入'a'**：新增的'a' 可以与后面所有的"cm" 组成"acm"。更新 $cnt_{acm} += cnt_{cm}$，同时更新 cnt_a, cnt_{ac}。

其他字符插入同理维护上述六个变量即可，操作均为 $O(1)$。

F. 宇宙射线风暴

简明题意

在 $N \times N$ 区域内有若干 $y = x + b$ 和 $y = -x + c$ 的射线，寻找能量叠加最大的点。

如果斜率固定，则截距可以确定一条直线。

令 $b = y - x$（1 型射线的截距），$c = y + x$（2 型射线的截距）。

对于任意确定的 1 型射线截距 b 和 2 型射线截距 c，它们的交点在区域内的充要条件是 $|b| \le c \le 2N - |b|$。

令 $W_1[b]$ 为 1 型射线截距为 b 的能量和，$W_2[c]$ 为 2 型射线截距为 c 的能量和。

要最大化 $W_1[b] + W_2[c]$，我们可以枚举 1 型射线截距 b，然后需要在合法区间 $[|b|, 2N - |b|]$ 内查询 $W_2[c]$ 的最大值。

F. 宇宙射线风暴

显然可以使用 ST 表或线段树等数据结构查询 W_2 的区间最大值，但此题并没有那么复杂。

注意到随着 $|b|$ 的减小，对应的 c 的合法区间 $[|b|, 2N - |b|]$ 是逐渐向外扩张的（包含关系）。我们可以预处理 $pre[k]$ 表示 $c \in [k, 2N - k]$ 范围内 W_2 的最大值，递推式为 $pre[k] = \max(pre[k+1], W_2[k], W_2[2N - k])$。

时间复杂度 $O(N + Q)$。

I. 比较大小

简明题意

交互题。n 个未知数，每次查询返回子集的最小值模 p。求最大值模 p。

首先，我们可以用 n 次查询得到每个数模 p 后的值。

如果两个数模 p 后的值不同，我们可以查询这两个数，返回的是较小值。

每次选择两个模 p 后不同的数查询，然后删去较小值，直到剩余所有数模 p 后相等即可。

查询次数至多 $2n-1$ ，满足要求。

N. 最大化仿射变换

简明题意

n 个操作 $x := a_i x + b_i$，初始 $x = 0$，安排顺序使得最终 x 最大。

贪心题，考虑相邻的两个操作 i 和 j：

- 若 i 先执行，贡献为 $a_j(a_i X + b_i) + b_j$；
- 若 j 先执行，贡献为 $a_i(a_j X + b_j) + b_i$。

比较两式差异，化简得：若 $\frac{b_i}{(a_i - 1)} > \frac{b_j}{(a_j - 1)}$，则应先执行 i。

按此比较函数进行排序，然后模拟即可。注意 $a_i = 0$ 或 $a_i = 1$ 的边界情况要特殊处理。

时间复杂度 $O(nlogn)$ 。

G. 万能矩阵

简明题意

构造 $2n \times 2n$ 矩阵，使得子矩阵和能覆盖 $1 \ldots n^4$ 的所有整数。

将目标值分解为 $A + Bn + Cn^2 + Dn^3$ 的形式，其中系数 A, B, C, D 的范围为 $[0, n-1]$。

按下图十字形构造即可，未填入位置均为 0：

			n				
			n				
			n				
1	1	1		n^2	n^2	n^2	
			n^3				
			n^3				
			n^3				

B. 魔法棋盘

简明题意

构造一个连通块，恰好包含 a 个白格和 b 个黑格。

构造题，当 a 和 b 中较大的不超过较小的三倍加一，即 $max(a, b) <= min(a, b) \times 3 + 1$ 时，存在合法构造方案。

下面假设白格子数量小于等于黑格子数量，即 $a \leq b$。

必要性：

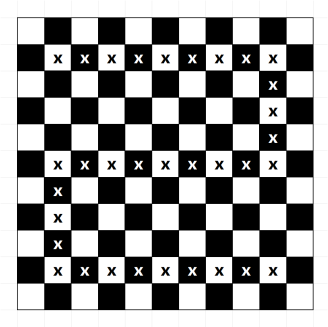
- 显而易见，1 个白格子最多与 4 个黑格子直接相连。
- 新白格子还需要与 1 个旧的黑格子连接以保证和旧的白格子的连通性。
- 因此，当有多个白格子时，除了第一个白格子可以拥有 4 个黑格子，每新增 1 个相连的白格子，最多只能新增 3 个相连的黑格子。

因此，对于 a 个白格子，最多允许 $b \leq 3a + 1$ 个黑格子与其相连，才能形成连通块。

B. 魔法棋盘

充分性：

存在构造方案。先按蛇形型构造拥有 a 个白格子和 a 个黑格子的“骨架”，然后再在”骨架”的白格子上把剩下 $b-a$ 个黑格子连接上去，“骨架”的大致样子如下图：



B. 魔法棋盘

可以看到“骨架”每 4 行可以容纳 $\frac{\text{列数}}{2}$ 个白格子，而由于 $a+b \le 2\times10^5$，即白格子数量最多 10^5 个，棋盘又有 1000 列，因此总共最多需要 $\frac{10^5}{\frac{1000}{2}} \times 4 = 800$ 行即可。而棋盘有 1000 行，因此该方案可行。

对于 $a > b$ 的情况，只需要交换 a 和 b 后进行构造，并将最终结果全部下移一格即可。

D. 环球旅行商

简明题意

在圆柱面网格上，从北极到南极，只能向南、东、西移动。每行需访问若干指定点，求最少步数。

首先发现向下移动的步数固定为 $n+1$，因此只计算横向移动的步数即可。显然可以只考虑从一个物资点移动到另一个物资点（代价为曼哈顿距离），而不是一格一格地移动。

由于不能向北走，必须逐行处理。对于第 i 行，我们需要从第 $i-1$ 行下来的位置出发，访问该行所有物资点，最后停在某个位置准备下楼。

把处于同一行的物资点一起计算，只考虑有物资点的行。先按行进行离散化分层，并在层内按从左到右排序，并设 dp 状态如下：

- $dp_{i,j,0}$ 表示：当前在第 i 层的第 j 个物资点，且没完成第 i 层的最小步数。
- $dp_{i,j,1}$ 表示：当前在第 i 层的第 j 个物资点，且已经完成了第 i 层，即经过了第 i 层的所有物资点至少一次的最小步数。

D. 环球旅行商

考虑不同层之间的转移:

一种显而易见的想法是,枚举这一层的点 (i,j),再枚举上一层的点 $(i-1,l)$ 来转移,即 $dp_{i,j,0} = \min\{dp_{i-1,l,1} + dis((i,j),(i-1,l))\}$。

但这样的转移是 $O(k^2)$ 的,考虑优化。

可以发现 $dis((i,j),(i-1,l))$ 由逆时针和顺时针两部分取 min 组成,而同一个方向的距离的 min,在 (i,j) 到 $(i,j+1)$ 时,会增加 (i,j) 到 $(i,j+1)$ 的一段距离。

因此可以在第 i 层和第 $i-1$ 层跑双指针来解决,考虑完 (i,j) 再考虑 $(i,j+1)$ 时,就将这个方向上的上一个 min 增加一段距离,而上一层的 $(i-1,l)$ 介于 (i,j) 和 $(i,j+1)$ 之间时,也将其计入到 min 的计算中去。

于是不同层之间的转移时间复杂度为 $O(k)$。

D. 环球旅行商

考虑同层之间的转移:

注意到，当从点 (i, j) 进入第 i 层时，向东或向西转一圈并从 $(i, j-1)$ 或 $(i, j+1)$ 离开（也就是只需要考虑从相邻的两个点离开的情况）即可最优。如果有从其他点离开更优的方案，则这个方案将在不同层转移时被考虑进去。

因此转移方程为
$dp_{i,j,1} = \min(dp_{i,j-1,0} + dis((i,j),(i,j-1)), dp_{i,j+1,0} + dis((i,j),(i,j+1)))$。

于是同层的转移时间复杂度为 $O(k)$。

算上一开始对必经点离散化的排序的复杂度，总时间复杂度 $O(k \log k)$。

结语

Thank you!