大连大学 2021 年 ACM 程序设计竞赛校内选拔赛（正式赛）题解

Problem A. Made in Abyss

签到题，按照题目要求细心输出即可。

Problem B. Best Match

观察到：因为 p 为一个排列，因此对于数组 a 的某个位置 i，若 $a_i = p_j$，则对于位置 i 只有 2 种可能：

1. $a[i:i+n-1]$ 与 $(p_j, p_{j+1}, \dots, p_n, p_1, p_2, \dots, p_{j-1})$ 完全相等。

2. $a[i:i+n-1]$ 与 $(p_j, p_{j+1}, \dots, p_n, p_1, p_2, \dots, p_{j-1})$ 至少有一位不相等。

于是我们可以考虑预处理数组 b，有 $b[a[i]] = j$，这样就能 $O(1)$ 找到排列 p 的下标 j 并向后遍历，验证是两种情况中的具体哪一种。当可以判定是两种情况中的第一种时，则可以将位置 i 记录在成功数组中；否则不予记录，注意前面失配的位置不重复遍历，因为若对于位置 i，在 $i+k$ 时发现$a_{i+k} \neq p_{((j+k)\%n+1)}$ 时，则 $[i, i+k]$ 都不可能被记录在成功数组中。

然后我们对于每个询问，贪心地二分查找第一个大于等于左端点的，在成功数组中的位置 pos，并检查$pos + n - 1$是否已经超过右端点，若未超过则证明该区间内存在排列 p 作为子数组。

时间复杂度 $O(n + qlogn)$。

Problem C. 5 Centimeters Per Second

我们先将每个位置 i 的花瓣的高度转化成还剩多长时间落地，也就是将高度除以 5。接着因为远野贵树从第一棵树移动到第二棵树下需要一秒，从第一棵树移动到第三棵树下需要两秒…因此我们将位置 i 的时间减去 i，这样数组代表的含义就变成了“若不是马不停蹄的向后赶路，最多可以在前面耽误停留的时间”。

这样，我们就可以列出 dp 方程 $dp[i] = max(dp[i], dp[j] + 1)$，其中$a_i \leq a_j$，状态含义为到达位置 i 时最

多可以获得的花瓣，可以发现这是在求最长不下降子序列的长度，利用数据结构或者二分优化一下即可通过本题，注意开始的位置必须为非负数。

时间复杂度 $O(nlogn)$。

Problem D. No Netherite No PG

注意到这是一个有向无环图，且 W 较小，所以在不能回头的情况下，若设 dp 状态为 $dp[w][x]$，代表到 x 结点时已经击败的怪物总量为 w，则有 $dp[w+j][y] = dp[j][x] + w[y]$。但是需要注意转移顺序：只有在一个点的前面所有点都向这个点转移完成后，这个点才能用作转移，不难想到拓扑排列，因此本题就是用拓扑排列的顺序更新 dp 数组，最后输出 $max(dp[w][x])$ 即可。

时间复杂度 $O(W(n+m))$。

Problem E. Rotating Calipers

在多边形旋转的过程中，我们其实只需要考虑离转轴最远的，多边形外边框上的一点，是否比给定的固定点离转轴更远即可，因为若最远的一点比固定点离转轴更远，则必定有一条边会扫到固定点。

因此只需要用距离公式预处理出最远的点离转轴的距离，然后对于每个询问计算固定点离转轴的距离，比较后输出答案即可。

时间复杂度 $O(n+q)$。

Problem F. Concise description

对于数字 a 和 b ，假设 $a \le b$，若 b 确定，则有 $\frac{b}{a}$ 个 $a \in [1,b]$ 满足 $a|b$，因此题目转化成求 $\sum_{i=1}^{n}\frac{n}{i}$。

数论分块算法可以做到 $O(\sqrt{n})$ 计算该式子，因此最后的答案就是 $\frac{(\sum_{i=1}^{n}\frac{n}{i}-n)}{2}+n$。

时间复杂度 $O(\sqrt{n})$。

Problem G. Diamond Forest

对于一个连通块进行深度优先搜索，若访问了已经访问过的结点，则该连通块不是一棵树；若每个点的度都为该联通块大小减一，则其为钻石，否则该连通块为石头。

时间复杂度 $O(n+m)$。

Problem H. Palindrome Mahjong

使用双指针，一开始将左指针放在字符串最左端，右指针放在字符串最右端，接着左指针右移，右指针左移，可以证明若左右指针指向的字符相同，则不需要进行任何操作，否则有以下 8 种可能：

1. 左指针做操作 1，右指针不做操作
2. 左指针仅做操作 2，右指针不做操作
3. 左指针做操作 1 和操作 2，右指针不做操作
4. 左指针不做操作，右指针做操作 1
5. 左指针不做操作，右指针做操作 2
6. 左指针不做操作，右指针做操作 1 和操作 2
7. 左指针做操作 1，右指针做操作 2
8. 左指针做操作 2，右指针做操作 1

因为两指针只能做 1 次，因此都枚举一遍即可。

时间复杂度 $O(n)$。

Problem I. The Hand

想要做到字符串最小，必定从前向后贪心，让第一个字符尽可能小。因此遍历字符串，若对于位置 i，有 $s_i > s_{i+1}$，则必定是删去 s_i；若到最后都没有删除字符，即字符串单调不减，则删除最后一个字符。

时间复杂度 $O(n)$。

Problem J. Befriend distant states while attacking those nearby

观察到建交和出兵的对象是一定的，因此考虑先预处理出每个国家的建交和出兵对象。使用线段树可以做到在 $O(logn)$ 的时间复杂度内查找区间内第一个/最后一个大于/小于/大于等于/小于等于值 x 的下标，因此可以使用线段树确定建交和出兵对象。

确定后将国家按照战力值排序，则按升序排序的顺序出兵，有存活国家数量最小；按照降序排序的顺序出兵，有存活国家数量最大，证明略。

时间复杂度 $O(nlogn)$。

Problem K. TARDIS

共有三种情况：起点与终点相同，起点与终点在同一行/列，起点与终点不在同一行也不在同一列，分别输出 0，1，2。

时间复杂度 $O(1)$。

Problem L. Nim Type Zero

设你赢的局数为 A，露娜赢的局数为 B，则有

$$\begin{cases} 2A + B = a \\ 2B + A = b \end{cases}$$

转化后可得:

$$\begin{cases} A = \frac{2a-b}{3} \\ B = \frac{2b-a}{3} \end{cases}$$

因为 A，B，a，b 都为自然数，所以有

$$\begin{cases} 3|(2a-b) \\ 3|(2b-a) \\ 2a \geq b \\ 2b \geq a \end{cases}$$

判断一下是否都满足，如果都满足就是可能出现的最终得分。

时间复杂度 $O(1)$。