山东理工大学第十六届 ACM 程序设计竞赛

解题报告

凌乱之风

山东理工大学 ACM 集训队

2024 年 5 月 12 日

出题组

- BKBB
- zjc
- SugarT
- m_rd
- 凌乱之风
- Ginger
- Pan_QY
- SRT
- BXT
- WXY

验题组

- BKBB
- zjc
- SugarT
- m_rd
- 凌乱之风
- Ginger
- Pan_QY
- SRT
- BXT
- WXY
- wudima
- ygq
- yuyansheng
- QLU-Leonard

A. 欢迎来到山东理工大学第十六届程序设计竞赛

出题人：BKBB

I/O

- 输出 YES 即可。

参考代码：https://paste.ubuntu.com/p/k64HFV78r7/

B.Q 的网课

出题人：zjc

模拟

- 用 map 存每一个网课的学时即可。

参考代码：https://paste.ubuntu.com/p/bVHtGN9Rqr/

C. 震惊！最好的走位竟然是...

出题人：SugarT

DP

- 考虑到二进制枚举只能操作到 $n = 20$ 的数据，无法通过全部数据。
- 称不会撞到墙壁的操作为有效操作。考虑到 n 最大为 100，那么保证它能够跑回来的最远路程为 50（即每一步都是有效操作）
- 所以我们只需要考虑距离起点上下左右 50 的距离即可。
- 将起点离散化至 $(50, 50)$ 的位置，其余障碍物同步移动，那么在 $(100, 100)$ 的二维矩阵以外的障碍物、以及 DP 走到的步数都不需要考虑。
- 考虑 DP，将三维优化为二维即可（不优化也可以）对于当前操作来说，有走与不走两种状态，对于走来说又有有效操作和无效操作两种状态，判断一下加在哪里即可。
- 时间复杂度 $O(n^3)$

参考代码：`https://paste.ubuntu.com/p/Xsms7XMtmK/`

D. 会编程的老师

出题人：m_rd

双指针

- 首先二进制枚举预处理查询，即枚举条件串的一个子集。
- 在累计满足条件的答案时，可以使用双指针，考虑将 x_i 中所有字母出现次数大于 1 次作为单调性。

参考代码：https://paste.ubuntu.com/p/9BgkVCpXWt/

E. 不是，哥们

出题人：凌乱之风

数论

- 对等式 $a_i^2 = k \times a_j$ 两边开根得到 $a_i = \sqrt{k \times a_j}$，考虑找到最小的 k 满足 $k \times a_j$ 为平方数，即将 a_j 中奇数次幂的质因子变为偶数次幂。
- 考虑倒序枚举 a_i，将所有后缀的 $\sqrt{k \times a_j}$ 在桶 f 中计数，枚举 a_i 时需要遍历累加 a_i 所有的约数。
- 即

$$\sum_{i=n}^{1} \sum_{d|a_i} f_d$$

参考代码：`https://paste.ubuntu.com/p/zfjXsjqsXn/`

F. 森林的奥秘

出题人：Ginger

并查集/LCA

- 考虑倒序遍历，相当于每次添加一条边。每次添加一条边，森林的直径答案有以下情况：
  1. 没加这条边时森林的直径。
  2. A 所在的树的直径。
  3. B 所在的树的直。
  4. A 和 B 所相连后形成的树的直径。
- 定义 $f(i)$ 为第 i 次查询的答案，开两个数组来标记每个树集合的直径所对应的端点 l 和 r
- 对于情况 1，答案为 $f(i+1)$
- 对于情况 2，答案为所在树集合对应端点的距离，就是 A 树的直径。
- 对于情况 3，答案为 B 所在树集合对应端点的距离，就是 B 树的直径。
- 对于情况 4，如果第一棵树直径两端点为 (u, v)，第二棵树直径两端点为 (x, y)，用一条边将两棵树连接，那么新树的直径一定是 u, v, x, y 中的两个点所以只需要两两取点求距离取最大值就是直径。

参考代码：`https://paste.ubuntu.com/p/3HdJGh8vGt/`

G.ArmyL 对比法

出题人：SugarT

模拟

- 按照题意模拟循环遍历即可。
- 若 $a_i > b_i$ 输出 WinWinWin!!!
- 若 $a_{\max} > b_{\min}$ 输出 HaHa
- 否则输出 AreYouOK?

参考代码：`https://paste.ubuntu.com/p/StMh68CgY9/`

H. 我最喜欢吃饭了

出题人：Pan_QY

模拟

- 读完可知所谓的排队打餐离开就是队列的入队/出队（为了照顾可能没学过队列的同学题面里稍微多讲了一下队列的性质）
- 每个打餐窗口排队的同学一定是编号递增的，因为后来的排在后面编号更大。
- 所以题目就是要把给定的序列拆成至多 m 个单调增序列问是否可以满足，这就是初学时最经典的贪心问题中的“导弹拦截系统”。
- 我们将给定的序列拆成尽可能少的单调增序列，查看最小的满足读入序列需要的打饭窗口数是否超过食堂有的窗口数就可以了。

参考代码：`https://paste.ubuntu.com/p/67jCHMPXsS/`

I. 树上跳棋

出题人：SRT

做法 1:ExGcd/LCA

- 设 p_1, p_2 为两枚棋子初始点在树中的深度，d_1, d_2 为跳跃距离。若两枚棋子能到达同一深度为 p 的点，有 $p_1 - k_1 \times d_1 = p_2 - k_2 \times d_2 = p$，移项得 $k_1 \times d_1 - k_2 \times d_2 = p_1 - p_2$
- 对于式子 $k_1 \times d_1 - k_2 \times d_2 = p_1 - p_2$，显然可以用 ExGcd 求出一组 k_1, k_2 的特解，找到第一个使得深度 p 小于等于两初始点 LCA 深度的 k_1 或 k_2 即可。
- 无解情况：$p_1 - p_2 \neq 0 \pmod{\gcd(k_1, k_2)}$ 或解出的深度小于树根深度。
- 找到 p 后倍增跳到答案点输出即可。

参考代码：`https://paste.ubuntu.com/p/4VJbSXF7BF/`

I. 树上跳棋

出题人：SRT

做法 2:CRT/LCA

- 我们假设 k 是最终要跳到的最近点的反向深度（这个反向深度由最大深度减去 k 的深度得到），p_1 是点 1 的深度，d_1 是点 1 的跳跃距离，p_2 是点 2 的深度，d_2 是点 2 的跳跃距离。那么就有
- $p_1 \equiv k \pmod{d_1}$
- $p_2 \equiv k \pmod{d_2}$
- 这个 k 可以用中国剩余定理 $O(\log n)$ 求解，如果无法求出则无解。
- 其次我们还要求得两个点的最近公共祖先来确定答案的下限，这个可以用 LCA 算法 $O(\log n)$ 求解。
- 如果我们算出的 k 比下限还要低，那么在 k 的基础上用 $\mathrm{lcm}(d_1, d_2)$ 为跳跃距离跳到下限之上为最优答案，但如果跳出了树根则无解，若 k 跳离了树根也是无解。
- 最后的时间复杂度为 $O(\max(n, q) \log n)$

参考代码：`https://paste.ubuntu.com/p/mynqXGPTkM/`

J. 这不是 RMQ 问题捏:)

出题人：BKBB

做法 1：思维

- 构造几组样例，手玩一下可以发现答案区间一定等价于取二维数组中和最大的那一列。
- 我们不妨假设取得的答案区间包含多列，我们任取答案区间中的某列，区间左右端点分别向左和向右移动。
- 在移动过程中每一行中的最小值不会增大，只会不变或者减小。故当区间扩大到我们取得的答案区间，S 的取值只会小于等于我们选取的中间某列。
- 所以答案区间一定等价于取二维数组中和最大的那一列。时间复杂度 $O(nm)$

参考代码：`https://paste.ubuntu.com/p/XvYS9Cj7xX/`

J. 这不是 RMQ 问题捏:)

出题人：BKBB

做法 2：二维数点/扫描线

- 我们将每个区间抽象成二维平面上的点，对于点 (x, y) 处的值可以认为是选择了区间 $[x, y]$ 后 S 的值。
- 考虑每个数的贡献，也就是每个数会对二维平面中哪些点产生贡献。对于一行中一个数 a_i ，其左边第一个小于等于其数值的数为 a_j，右边第一个小于等于其数值相同的数为 a_k
- 那么区间的左端点选在 $j+1$ 至 i 之间，右端点选在 i 至 $k-1$ 之间的任何一种选择都可以得到 a_i 的贡献。映射到二维平面上，a_i 可以对 x 取值在 $j+1$ 至 i 之间，y 取值在 i 到 $k-1$ 之间的矩形产生贡献（矩形中的每个点权值均加 a_i）
- 对于处理每个数一侧最近的小于等于其值的数可以用单调栈维护。对于二维平面的矩形增删的问题可以想到扫描线维护。时间复杂度 $O(nm\log(m))$

参考代码：`https://paste.ubuntu.com/p/Y4JscR4sCr/`

K. 猜猜数？

出题人：BXT、WXY

思维/模拟

- 现约定：对于 B 来说，拆分指将一个数分成两个不同的数，使得这两个数的积为 B 知道的数，而对于 C 则是分成两个不同的数的和。
- 对于 B 的第一句话，可以得知如果一个数只有一种拆分方式，那么必然不成立，这样 B 就可以确定两个数的具体数值，那么可以将拆分方式数量小于 2 的排除。
- 对于 C 的第一句话，可以知道，对于 C 的拆分方式也至少为两种，并且如果 C 通过假设一种拆分方式得到两数的积，那么如果这两数的积如果只有一种拆分方式，那么与 B 的上一句矛盾，于是排除掉拆分方式小于 2 的。
- 对于 B 的第二句话，当前 B 剩下推断的数应该至少有两种拆分方式，B 假设一种拆分方式，得出两个数的和，然后假设自己为 C 做出判断，若至少有两种拆分情况使得 C 做出未知的判断，那么现在 B 也只能做出未知的判断，与自己当前已知矛盾，即只有一种拆分的情况使得 C 不知道，即当前排除拆分情况大于 1 的。
- 对于 C 的第二句话，排除前面已知推断后，此时剩下判断的两数之和的拆分方式必有两种以上的可能，若有多种拆分方式使得 B 已知，那么与当前 C 已知矛盾，故当前也只能有一种拆分方式使得 B 已知。于是排除拆分方式不等于 1 的。
- 由以上过程即可排除掉所有不合法方案。

K. 猜猜数？

出题人：BXT、WXY

思维/模拟

- 由于 n 最大为 70，两个数的积最大为 4900，且为了使得计算方便，我们将一个 32 位数代表一个数对，即前 16 位和后 16 位数组成一个数对。
- 于是每次首先将所有的范围内的数对放入数组，随后每次按照上述分析过程将不合法的数对排除掉。每次排除在上次排除过的基础上在进行排除。最后数组里留下的数对即为所有合法的数对。
- 时间复杂度 $O(n^4)$

参考代码：`https://paste.ubuntu.com/p/jFvWj5ZNBj/`