XJTUPC2024

西安交通大学程序设计竞赛校队

2024.5.19

预期题目难度顺序

$$ACBF < DMKEI < ON < HGLJ$$

A 交小西的礼物

输出 $a + b + 2c + 3d$ 即可。

C 榕树之心

按照题示公式计算即可。

B 转呀转

题意即为给定半径和圆心角弧度求弦长。

答案是 $2\sin((tv - \lfloor tv \rfloor)\pi)\sqrt{x^2 + y^2}$。

F Everyone's ALL IN

第 x 号骑士团和第 y 号骑士团进行比赛，收益为：

$$\sum_{a_i=x}\sum_{a_j=y} i \times j$$

即

$$\sum_{a_i=x} i \times \sum_{a_j=y} j$$

提前计算好 $\sum_{a_i=x} i$ 即可。

由于 $1 \le a_i \le 10^6$，可以使用桶去计算。

D 瑟莉姆的宴会

考虑任意构造一条长为 n 的链，如 $1 \to 2 \to 3 \to \dots \to n$，容易计算出这条链的分数为 val。

如果这条链的分数非负则直接输出。

否则将序列翻转后输出，

如 $1 \to 2 \to 3 \to \dots \to n$ 变为 $1 \leftarrow 2 \leftarrow 3 \leftarrow \dots \leftarrow n$，

由于可以发现反转后所有之前 $lca(u, v) = u$ 的将会变为 $lca(u, v) = v$，反之亦然，故分数将会变成 $-val$ 为非负整数，反转后的输出是一定符合要求的。

在暴力的时候我们每轮都要查询每个点的度数，但其中很多点的度数其实并没有改变，所以我们考虑每轮只访问度数改变的节点。

在最开始时，将所有度数为 k 的节点取出，可以用队列等方式存储。

接下来每一轮，我们只需在删除点时，枚举每条和这个点相连的边，并将边的另一端的点度数减一，而当另一端的点度数**恰好**减为 k 时，就将其加入一个新的队列，表示这个点可能在下一轮被删除。

在本轮删点结束后，所有新队列中的点就可能在下一轮中被删除，但也可能在本轮中度数被减到小于 k。所以只需再扫一遍新队列，把现在度数仍为 k 的点取出，即为下一轮所需要删除的点。

M 生命游戏

考虑这样的复杂度，首先对于一棵 n 个点的树，其每个点的度数之和为 $2(n-1)$ 而我们的删点操作，对于单点来说其复杂度为这个点的度数，所以删去所有点的复杂度最多为 $O(n)$。

而每个点最多进入一次 ‘新队列’，故我们遍历新队列检验每个点是否需要被删的复杂度也是 $O(n)$。

故总复杂度可以做到 $O(n)$。

但由于读入常数有点大，直接用优先队列维护每个点的度数然后一批批取出删除理论上也能通过，复杂度 $O(n\log n)$。

K 崩坏：星穹铁道

考虑当 n 很小时，我们可以设 $f(i,j)$ 表示当角色总共行动 i 次，当前行动点为 j 时的方案数。我们可以根据不同的角色类型很自然地写出状态转移方程。

n 很大，不难想到矩阵快速幂。容易发现这个转移方程能够写成矩阵乘法的转移式。

K 崩坏：星穹铁道

设 $f(n)$ 为 n 次行动后的向量，设四名角色对应的矩阵分别为 A_0, A_1, A_2, A_3 。令 $P = \prod_{i=0}^{3} A_i$，

则有

$$f(n) = f(0) P^{\lfloor \frac{n}{4} \rfloor} \prod_{i=0}^{n \bmod 4} A_i$$

通过矩阵快速幂求出 $f(n)$，则最终答案 $Answer = \sum_{i=0}^{5} f(n)_i$

E 雪中楼

令题目中给出的数组为 a_i，还原出的高度数组的排列为 p_i。

首先可以通过反证法证明，当 $a_i = a_j$，$i < j$ 时，必然有 $p_i > p_j$。

同样的，可以证明，若 $a_i = a_j$，$a_k = j$，满足 $i < j < k$，那么必然有 $p_i > p_k > p_j$。

那么根据这样几个条件，我们可以把 a_i 相同的位置放在一起，然后按照 a_i 的值逐一合并入序列。

考虑维护一个链表，链表中记录每个位置在全局前一个比它高的位置在哪，然后顺序维护这个链表，若新来的数为 a_i，那么把它插在节点 a_i 的前面，它的编号为 i。若 $a_i = 0$，那么插入到链表头即可。

E 雪中楼

例如，当 a_i 为 $0, 0, 1, 0$ 的时候：

$a_1 = 0$，链表结构为 1。

$a_2 = 0$，链表结构为 $2 \to 1$。

$a_3 = 1$，将 3 插入到 1 的前面，链表结构为 $2 \to 1 \to 3$。

$a_4 = 0$，链表结构为 $4 \to 2 \to 1 \to 3$。

这个链式结构说明了，4 所在的位置最小，2 其次，3 最大，也就是离散化后的高度应该是 $3, 2, 4, 1$。

容易发现，这个链式结构完美地刻画了上面两个性质。时空复杂度 $O(n)$。

其余的 $O(n \log n)$ 做法也完全可以通过。

命令行

建立命令串的 Trie 树，对于输入串 s，其可以表示为 (pos, len)，代表 s 在 Trie 树上行走最后到达的节点编号和到达 pos 之后，不在 Trie 树上的长度。

a-z,E,B 的实现比较简单，考虑如何实现 T：

在每个节点维护 $next_i$，代表在此匹配的跳转节点。对于 i，如果其有且仅有一个孩子 son，则 $next_i = next_{son}$，否则 $next_i = i$。特殊的，若当前节点表示的字符串是一条指令的末尾，那么 $next_i = i$。

O 筛法

本题目的是让有敏锐观察能力，或有强大推公式能力，或敢于打表找规律的选手通过。

打表不难发现结果为 n^2。

下面给出若干种证明。

首先可以暴力推公式。

$$\sum_{i=1}^{n}\sum_{j=1,gcd(i,j)=1}^{n}\lfloor\frac{n}{max\{i,j\}}\rfloor = 2\sum_{i=1}^{n}\sum_{j=i+1,gcd(i,j)=1}^{n}\lfloor\frac{n}{j}\rfloor + n$$

$$= 2\sum_{i=1}^{n}\lfloor\frac{n}{i}\rfloor\cdot\phi(i) - n$$

令 $f(n) = \sum_{i=1}^{n}\lfloor\frac{n}{i}\rfloor\cdot\phi(i)$

$$f(n) - f(n-1) = \sum_{i=1}^{n}\left(\lfloor\frac{n}{i}\rfloor - \lfloor\frac{n-1}{i}\rfloor\right)\phi(i)$$

$$= \sum_{d|n}^{n}\phi(d) = n$$

故 $f(n) = \frac{1}{2}n(n+1)$，原式 $= n^2$。

O 筛法

从几何角度出发：

考虑二维平面上的 n^2 个点 (i,j) $1 \le i,j \le n$.

从原点向每个满足 $i \perp j$ 的点 (i,j) 引出一条射线，可以发现这 n^2 个点均唯一存在于其中一条射线上，因为当 $\gcd(i,j) \ne 1$ 时，(i,j) 会被 $(\frac{i}{\gcd(i,j)}, \frac{j}{\gcd(i,j)})$ 引出的射线覆盖。

我们再对每个 $i \perp j$ 的点对 (i,j) 考虑其引出射线覆盖的点数，不难发现恰好就是 $\lfloor \frac{n}{\max(i,j)} \rfloor$，因为对所有 $1 \le k \le \lfloor \frac{n}{\max(i,j)} \rfloor$，点对 (ik, jk) 在这条射线上，当 k 再大时 ik, jk 中至少有一维超过 n.

故 $\sum_{i=1}^{n}\sum_{j=1}^{n}[i \perp j]\lfloor \frac{n}{\max(i,j)} \rfloor$ 可以理解为对所有互质的 (i,j) 求上述射线所覆盖的点对数量，在上述结论中可知该和为 n^2.

O 筛法

从映射角度出发:

考虑两个集合 $A = \{(i,j) \in \mathbb{Z}^2_{\geq 1} : 1 \leq i,j \leq n\}, B = \{(i,j,k) \in \mathbb{Z}^3_{\geq 1} : i \perp j, 1 \leq k \leq \lfloor \frac{n}{\max(i,j)} \rfloor\}$

再考虑映射 $f(i,j) = (\frac{i}{\gcd(i,j)}, \frac{j}{\gcd(i,j)}, \gcd(i,j))$ 是 A 到 B 的映射。

首先证明这个映射是单射:

$\forall f(i_1,j_1)=f(i_2,j_2),(i_1,j_1)\in A,(i_2,j_2)\in A$,

为了方便，下面在运算中用 (i,j) 代表 $gcd(i,j)$。

$(\frac{i_1}{(i_1,j_1)},\frac{j_1}{(i_1,j_1)},(i_1,j_1))=(\frac{i_2}{(i_2,j_2)},\frac{j_2}{(i_2,j_2)},(i_2,j_2))$，故 $(i_1,j_1)=(i_2,j_2)$，

所以 $i_1=\frac{i_1}{(i_1,j_1)}\times(i_1,j_1)=\frac{i_2}{(i_2,j_2)}\times(i_2,j_2)=i_2$。

同理可得 $j_1=j_2$，故 f 是单射。

再证明这个映射是满射:

$\forall(i,j,k)\in B$，有 $(ik,jk)\in A$ 且 $f(ik,jk)=(i,j,k)$。故 f 是满射。

O 筛法

综上 f 是双射，$|B| = |A| = n^2$。

而和 $\sum_{i=1}^{n}\sum_{j=1}^{n}[i \perp j]\lfloor \frac{n}{\max(i,j)} \rfloor = \sum_{\substack{i \perp j \\ 1 \le i,j \le n}} \sum_{k=1}^{\lfloor \frac{n}{\max(i,j)} \rfloor} 1$ 就是在求 B 中的元素个数，故其值为 n^2。

O 筛法

验题队使用了杜教筛等方法，对于本题的数据范围也可以通过。

$$\sum_{i=1}^{n}\sum_{j=1,gcd(i,j)=1}^{n}\lfloor\frac{n}{max\{i,j\}}\rfloor = 2\sum_{i=1}^{n}\lfloor\frac{n}{i}\rfloor\cdot\phi(i) - n$$

由于除法分块的性质，只需要求出 $O(\sqrt{n})$ 个对应点值处的 ϕ 前缀和，显然可以使用杜教筛。

时间复杂度 $O(n^{\frac{2}{3}})$。

N 圣诞树

考虑一个经典的树上联通块贪心，即在 DFS 中，回溯时一个子树满足答案要求，就可以把这个子树计入答案。

问题变成如何快速统计子树内颜色数。子树颜色有一个经典的离线做法 DSU On Tree。

当一个子树满足条件的时候就直接截断。显然被截断的子树不会再被遍历到。

树上启发式合并（可能）也可以通过。前者时间复杂度 $O(n\log n)$，后者 $O(n\log^2 n)$。

H 图上操作

考虑无向图瓶颈路的做法，是求一个最大生成树然后在上面找边权最小的一条边。

转化到有向图，同样的我们可以按照边权从大到小加进这张图，若加进来一条边以后存在一条从 1 到 i 的有向路径，说明 i 的瓶颈路大小为这条边的边权。

由于边权范围很小，所以可以直接建 100 张图，第 i 张图中所有的边均满足边权 $w \geq i$。如果边权下降，那么相当于在一些图中删除这些边，然后维护可达性。

但是有向图由于有环的存在，删边维护可达性难以在线维护。考虑离线，由删边改为加边。

H 图上操作

在每张图中维护每个点是否能从 1 出发可达，假设增加了一条 $x \to y$ 的有向边，那么如果 x 能够从 1 到达，那么 y 以及它的后继也就可达，直接遍历维护，若访问到可达点就可以直接跳过。

可以发现每个点只会被入边遍历到，出边被访问当且仅当这个点变成可达。所以每张图的维护总复杂度是 $O(n+m)$ 的。

若一个点的可达性发生变化，那么它的答案也会发生改变。显然暴力的枚举图，找一张最小边权最大的那张图使得它能从 1 出发可达。

总复杂度 $O((n+m)maxd+q)$。

G 循环移位

首先计算 $\max \sum a_i \oplus i$

考虑 a_i 第 $j(0 \le j < n)$ 个二进制位在循环移位 k 次时产生的贡献，显然，其关于 k 存在大小为 2^{j+1} 的周期。

我们记 $x[j]$ 为 x 的第 $j(0 \le j < n)$ 个二进制位代表的值。

G 循环移位

考虑

$$(a_i \oplus (i+k))[j]$$

当 $a_i \oplus (i+k) \bmod 2^{j+1} \in [0, 2^j)$，上式为 0。

当 $a_i \oplus (i+k) \bmod 2^{j+1} \in [2^j, 2^{j+1})$，上式为 2^j。

当 $(i+k)[j] \neq a_i[j]$ 时，有 $a_i \oplus (j+k) \bmod 2^{j+1} \in [2^j, 2^{j+1})$。

记 $b_{ij} = a_i[j] \oplus 2^j$

所以当 k 满足 $(j+k) \in [b_{ij}, b_{ij}+2^j) \bmod 2^{j+1}$ 时，也即当 k 满足 $k \in [b_{ij}-i, b_{ij}-i+2^j) \bmod 2^{j+1}$ 时，会对答案产生 2^j 的贡献。

G 循环移位

注意到在 $[0, 2^{j+1})$ 这是一段（或者两段）区间，可以通过前缀和计算所有 k 的结果，求出最大值。

我们又注意到：

$$\sum a_i \oplus i + 2 \times \sum a_i \& i = \sum (a_i + i)$$

$$-\sum a_i \oplus i + 2 \times \sum a_i | i = \sum (a_i + i)$$

等号右边容易计算，可以计算所有 k 的与/或结果，求出最大值。

时间复杂度 $O(n2^n)$，空间复杂度 $O(2^n)$。

L 勘探队

容易猜测答案一定是若干条纵坐标递增的折线连接 $(0,0)$ 与 $(0,y)$ 。

证明比较显然，即假设安装设备的位置是 (x_i, y_i)，那么在两个设备间，为了保证最小化体力消耗，显然会走直线。

那么每一段路的横坐标变化量 Δx_i 就已知了。设第 i 段路所需携带的重量为 M_i ，那么现在问题变成了，有 $n+1$ 段纵向路径，每一段的长度为 $y_i > 0$ ，约束条件为 $\sum y_i = y$ ，最小化 $\sum M_i\sqrt{\Delta x_i^2 + y_i^2}$。

考虑拉格朗日乘数法，令 $F = \sum M_i \sqrt{\Delta x_i^2 + y_i^2} + \lambda(\sum y_i - y)$，对每个变量求偏导：

$$\frac{\delta F}{\delta y_i} = \frac{M_i y_i}{\sqrt{\Delta x_i^2 + y_i^2}} + \lambda = \frac{M_i}{\sqrt{(\frac{\Delta x_i}{y_i})^2 + 1}} + \lambda = 0$$

$$\frac{\delta F}{\delta \lambda} = \sum y_i - y = 0$$

并且可以发现，$\frac{\delta F}{\delta y_i}$ 是随着 y_i 增大而增大的，所以可以通过二分 λ 的取值求出每一个 y_i 的值来检查，进而求解最终的答案。

L 勘探队

也可以考虑转化为物理模型中的光路折射，由于光在两点之间传播走时间最短路径，可以类比到这个题目中。

我们的目的是最小化 $\sum M_i\sqrt{\Delta x_i^2+y_i^2}$，不妨认为每段路径长度是 $\sqrt{\Delta x_i^2+y_i^2}$，速度是 $\frac{1}{M_i}$，时间就是 $M_i\sqrt{\Delta x_i^2+y_i^2}$，于是我们将问题规约为了光路折射。

折射定律告诉我们，$\frac{\sin i_a}{\sin i_b}=\frac{v_a}{v_b}$，代入速度得到 $M_a\sin i_a=M_b\sin i_b$，即所有路段的 $M\sin i$ 都一样。利用这个条件二分初始的 $\sin i_a$ 即可。

两个解法的复杂度均为 $O(n\log V)$。建议开启 long double 以及使用掉精度较小的算法。

J 最后一块石头的重量

原题为 leetcode 原题 [1049. 最后一块石头的重量 II]，但是数据范围很小。题意相当于给每个石子赋上正号或者负号，使得最终答案非负且尽可能小。

朴素的 01 背包复杂度为 $O(n^2m)$，显然无法通过本题。

考虑两个优化方法:

1. 使用 bitset。
2. 将输入打乱。

此时时间复杂度为 $O(\frac{n^{\frac{3}{2}}m}{w})$

关于优化 2 的证明，参考：随机变量前缀和的控制 - EntropyIncreaser (https://www.cnblogs.com/Elegia/p/16922216.html)

J 最后一块石头的重量

关于优化 2 的另外一种证明:

考虑最优解的一种方案，给定每个数的符号，并随机打乱后，我们需要求前缀和绝对值小于等于给定的值的概率。

考虑简化的情况，数组为 n 个 -1 和 n 个 1.

前缀和绝对值 $\leq d$ 的概率是从 $(0,0)$ 走到 (n,n)，只允许向右和向上移动，不越过 $y=x+d$ 和 $y=x-d$ 的方案数除以总方案数。

J 最后一块石头的重量

用容斥可计算出，记

$$f(n,d) = P(\max\{|y-x|\} \le d) = 1 - \frac{2\sum_{i=1}^{n/(d+1)}(-1)^{i+1}\binom{2n}{n-i\cdot(d+1)}}{\binom{2n}{n}}.$$

考虑原本的情况，设有 x 个正号，y 个负号，由于 $|a_i| \le m$，有最优解方案的 $|\sum a_i| \le m$，

$$P(Prefixsum \le d \cdot m) \le \min(f(x,d-1), f(y,d-1)) \le f(n/2,d-1).$$

计算可以得出取 $n = 10000, d = 200$ 时，一个数据点出错概率不超过 $1e-3$。