2024 国际大学生程序设计竞赛
亚洲区域赛（昆明）

题目讲解

林恺
omg_link@qq.com

黑冰茶命题组

2024 年 12 月 1 日

目录

1 简单题
1.1 M - 矩阵构造
1.2 J - 又一个排序问题
1.3 H - 扫描地平线
1.4 L - 绝望线缕
2 中等题
2.1 C - 金币
2.2 G - 最大公因数
2.3 E - 提取权值
2.4 D - 套娃
3 困难题
3.1 F - 花
3.2 A - 防毒
3.3 B - 括号
3.4 I - 物品
3.5 K - 密钥恢复

目录

1 简单题

1.1 M - 矩阵构造

1.2 J - 又一个排序问题

1.3 H - 扫描地平线

1.4 L - 绝望线缕

2 中等题

2.1 C - 金币

2.2 G - 最大公因数

2.3 E - 提取权值

2.4 D - 套娃

3 困难题

3.1 F - 花

3.2 A - 防毒

3.3 B - 括号

3.4 I - 物品

3.5 K - 密钥恢复

M - 矩阵构造

题目大意

构造一个 $n \times m$ 的矩阵，使得任意“相邻两个位置的数字的和”都互不相等。

现场情况

首次提交：想想 MJ 怎么做 @ 00:04
首次通过：少年的拼劲 @ 00:07
通过队数：449

M - 矩阵构造

题解

一种可能的构造方案是沿副对角线方向从小到大填入数字。例如：

$$\begin{bmatrix} 1 & 2 & 4 & 7 \\ 3 & 5 & 8 & 11 \\ 6 & 9 & 12 & 14 \\ 10 & 13 & 15 & 16 \end{bmatrix}$$

M - 矩阵构造

证明

下面证明这种构造方案的正确性。

首先，任意相邻的两个数字，都必然位于两个相邻的副对角线上。考虑任意两对相邻数字的和，假设它们所在的副对角线分别是 $(a, a+1)$ 和 $(b, b+1)$ 。

- 假如 $a \neq b$ ，不妨设 $a < b$ ，则第 a 个副对角线上的数字小于第 b 个副对角线，第 $a+1$ 个的也小于第 $b+1$ 个的。因此这两对和必然不可能相等。
- 假如 $a = b$ ，同一对副对角线上的和自右上到左下是单调递增的，因此也不可能相等。

综上，这种构造方法不会产生相同的和。

目录

1 简单题

1.1 M - 矩阵构造

1.2 J - 又一个排序问题

1.3 H - 扫描地平线

1.4 L - 绝望线缕

2 中等题

2.1 C - 金币

2.2 G - 最大公因数

2.3 E - 提取权值

2.4 D - 套娃

3 困难题

3.1 F - 花

3.2 A - 防毒

3.3 B - 括号

3.4 I - 物品

3.5 K - 密钥恢复

J - 又一个排序问题

题目大意

给定一个长度为 n 的排列，Alice 的目标是将排列排序，Bob 的目标是阻止 Alice 在有限步内排序。Alice 可以交换任意两个位置的数字，Bob 只能交换相邻两个位置的数字。

给定排列和先手，问：双方均采取最优策略的情况下，谁能达成目标?

现场情况

首次提交：签到失败 @ 00:05
首次通过：你怎么知道我在南京站直播抽到了袋鼠 @ 00:10
通过队数：448

J - 又一个排序问题

结论

当 $n \geq 4$ 时，除非 Alice 可以先手一步将排列排序，否则 Bob 必胜。

证明

假设排列中还有 k 个位置没有归位。当轮到 Bob 操作时，其总是能令 $k \leftarrow \min(k+1, n)$，操作后必然有 $k \geq 3$。当 $k \geq 3$ 时，Alice 显然不能在一步内将排列排好序。

因此只要轮到 Bob 操作了，Alice 就必然不可能胜利。

J - 又一个排序问题

结论

当 $n = 2$ 时，Alice 必胜。

证明

当 $n = 2$ 时，只能有 $p = [2, 1]$ 。此时，任意一个人操作都会使排列排好序，因此 Alice 必胜。

J - 又一个排序问题

结论

当 $n = 3$ 时，假如有 2 个位置没有归位，则先手必胜，否则后手必胜。

证明

当有 2 个位置没有归位时：

- 假如 Alice 先手，则 Alice 直接将排列归位，获得胜利。
- 假如 Bob 先手，则其必然可以通过操作使得有 3 个位置都没有归位。下一步，Alice 无论怎么操作都必然会归位 1 个位置，剩余 2 个位置没有归位，就回到了当前情况。因此 Bob 必胜。

当有 3 个位置没有归位时，先手必然会归位 1 个位置：

- 假如 Alice 先手，下一步回到了上面的第二种情况，Bob 必胜。
- 假如 Bob 先手，下一步回到了上面的第一种情况，Alice 必胜。

目录

1 简单题
- 1.1 M - 矩阵构造
- 1.2 J - 又一个排序问题
- **1.3 H - 扫描地平线**
- 1.4 L - 绝望线缕

2 中等题
- 2.1 C - 金币
- 2.2 G - 最大公因数
- 2.3 E - 提取权值
- 2.4 D - 套娃

3 困难题
- 3.1 F - 花
- 3.2 A - 防毒
- 3.3 B - 括号
- 3.4 I - 物品
- 3.5 K - 密钥恢复

H - 扫描地平线

题目大意

给定平面上的 n 个点，要求在原点放置一个雷达，使得雷达在旋转到任意角度时，都能照到至少 k 个点。最小化雷达照射的角度范围。

现场情况

首次提交：一路向北 @ 00:09
首次通过：一路向北 @ 00:09
通过队数：333

H - 扫描地平线

题解

使用 atan2 等方法，将点转化为极角。

此时问题转变为：求一个循环的极角序列上，相隔 k 个位置的极角差值的最大值。

枚举所有相隔 k 个位置的极角，取最大值即可。

目录

1 简单题

1.1 M - 矩阵构造

1.2 J - 又一个排序问题

1.3 H - 扫描地平线

1.4 L - 绝望线缕

2 中等题

2.1 C - 金币

2.2 G - 最大公因数

2.3 E - 提取权值

2.4 D - 套娃

3 困难题

3.1 F - 花

3.2 A - 防毒

3.3 B - 括号

3.4 I - 物品

3.5 K - 密钥恢复

L - 绝望线缕

题目大意

敌我双方分别有 n 和 m 个单位，我方每个单位可以攻击一次，攻击一个敌方单位会使战斗双方分别-1 生命值，当一个单位生命值清零后会发生一次爆炸，使所有单位生命值-1，询问是否存在一种攻击方案消灭所有敌方单位。

现场情况

首次提交：一路向北 @ 00:29
首次通过：一路向北 @ 00:29
通过队数：196

L - 绝望线缕

题解

首先观察到：如果一个操作方案中，在某次攻击之前发生过爆炸，那么我们将此次攻击移动到所有攻击之前，效果不变，同时也不会导致原本的其他操作失效。因此必然存在一种方案使得我方先进行所有攻击操作，此后不断发生爆炸直到对方所有单位清空。

考虑在这种条件下我们如何攻击，可以发现，所有生命值超过 1 的我方单位都拥有一次攻击机会，同时生命值等于 1 的我方单位加起来总共拥有一次攻击机会，且这次攻击必然发生在最后。只要敌方单位未被清空，那么我方单位必然能够攻击一次，使自己的生命值-1。因此连续发生的爆炸有两个触发条件，要么是我方单位的生命值-1 后低于已经发生的爆炸伤害，要么是敌方单位受到攻击后生命值低于已经发生的爆炸伤害。

L - 绝望线缕

题解

我们考虑贪心地分配我方的攻击次数，初始时爆炸伤害为 0，每次我们分别检查生命值最低的我方单位和敌方单位。若我方单位满足触发条件，则爆炸伤害直接 $+1$，否则就需要消耗攻击次数使敌方生命值最低的单位触发爆炸。若攻击次数已经不足以使敌方单位爆炸，则不存在消灭方案。

可以将双方单位分别按生命值排序之后，移动指针完成贪心检查过程。

目录

1 简单题

1.1 M - 矩阵构造

1.2 J - 又一个排序问题

1.3 H - 扫描地平线

1.4 L - 绝望线缕

2 中等题

2.1 C - 金币

2.2 G - 最大公因数

2.3 E - 提取权值

2.4 D - 套娃

3 困难题

3.1 F - 花

3.2 A - 防毒

3.3 B - 括号

3.4 I - 物品

3.5 K - 密钥恢复

目录

1 简单题
- 1.1 M - 矩阵构造
- 1.2 J - 又一个排序问题
- 1.3 H - 扫描地平线
- 1.4 L - 绝望线缕

2 中等题
- **2.1 C - 金币**
- 2.2 G - 最大公因数
- 2.3 E - 提取权值
- 2.4 D - 套娃

3 困难题
- 3.1 F - 花
- 3.2 A - 防毒
- 3.3 B - 括号
- 3.4 I - 物品
- 3.5 K - 密钥恢复

C - 金币

题目大意

将 n 个人排成一排，每轮淘汰下标模 k 余 1 位置上的人。问最后剩下的人一开始位于第几位?

$2 \le n, k \le 10^{12}$

现场情况

首次提交：铜牌已达两位数！@ 00:07
首次通过：击中月亮 @ 00:18
通过队数：138

题解

基本思路：从最后一轮开始，逆推胜者在每一轮中的位置。

这样就涉及两个子问题：

- 总轮数是几轮?
- 某一轮位于第 x 位的人，前一轮位于哪里?

C - 金币

子问题：总轮数是几轮?

假设队列中还有 x 个人，则下一轮会淘汰 $\lceil \frac{x}{k} \rceil$ 人。

直接模拟是不可行的：当 k 比较大时，轮数会接近 n 。

考虑对 k 分治：

- 当 $k \le \sqrt{n}$ 时：每一轮淘汰的人数大于 $\sqrt{n}$ 人，因此总轮数不会超过 $\sqrt{n}$ 轮，可以直接模拟。
- 当 $k > \sqrt{n}$ 时：注意到 $\lceil \frac{x}{k} \rceil$ 的变化不会超过 $\sqrt{n}$ 次。可以计算这个值在多少轮之后会变化，$O(1)$ 的模拟变化之前的轮次。

因此，我们可以在 $O(\sqrt{n})$ 的时间内计算出轮数。

C - 金币

子问题：某一轮位于第 x 位的人，前一轮位于哪里？

考虑所有位置小于等于 x 的人，将这些人按每 $k-1$ 个人分组。每有一组，就意味着上一轮有一个编号比胜者小的人被淘汰了。

因此，某一轮位于第 x 位的人，前一轮位于第 $x+\lceil\frac{x}{k-1}\rceil$ 位。

但是，总轮数可能接近 n ，还需要对 k 分治：

- 当 $k \le \sqrt{n}$ 时：总轮数不会超过 $\sqrt{n}$ 轮，可以直接模拟。
- 当 $k > \sqrt{n}$ 时：注意到 $\lceil\frac{x}{k-1}\rceil$ 的变化次数同样是 $O(\sqrt{n})$ 的。依然可以计算这个值在多少轮之后会变化，$O(1)$ 的模拟变化之前的轮次。

因此，我们可以在 $O(\sqrt{n})$ 的时间内逆推出胜者一开始的位置。

C - 金币

花絮

- 在验题过程中，不少队伍的做法中都出现了 log 相关的代码段，导致轻微的 TLE。考虑到比赛的整体难度，我们略微放宽了时限，允许大部分带一些小 log 的做法通过。但如果您的代码常数过大，依然可能无法通过。

目录

1 简单题
1.1 M - 矩阵构造
1.2 J - 又一个排序问题
1.3 H - 扫描地平线
1.4 L - 绝望线缕
2 中等题
2.1 C - 金币
2.2 G - 最大公因数
2.3 E - 提取权值
2.4 D - 套娃
3 困难题
3.1 F - 花
3.2 A - 防毒
3.3 B - 括号
3.4 I - 物品
3.5 K - 密钥恢复

G - 最大公因数

题目大意

给定两个正整数 a 和 b ，每次可以选一个数并减去 $gcd(a,b)$ 。

问：至少要多少次操作，才能使 a 和 b 都变成 0 ？

$a \leq 5000, b \leq 10^{18}$

现场情况

首次提交：签到失败 @ 00:08
首次通过：w4p3r Black-fan Club @ 00:32
通过队数：144

G - 最大公因数

题解

首先观察到以下两个性质：

- 假设其中一个数变成了 0 ，则另一个数可以在下一步也变成 0 。
- 将 a 变成 0 只需要至多 25 步。

于是，最优解不超过 26 。

性质二证明

考虑 a 和 b 的二进制最低位：

- 如果都是 0 ，则此后的 GCD 中必然包含 2 这个因子，此时将 a 和 b 除以 2 不会影响答案。
- 如果某一个数的二进制最低位不是 0 ，则 GCD 必然是一个奇数。此时将最低位不是 0 的那个数减去 GCD ，最低位会变为 0 。

于是，只需要至多 24 步，就能将 a 除以 4096 ，而 a 的最大值只有 5000 。此时 a 最大只能为 1 ，再操作一次即可使 a 归零。

G - 最大公因数

题解

既然保证了答案不超过 26 ，就可以枚举每一步做了什么，使用 $O(2^{26})$ 的复杂度搜索出最优解。

理论上，搜索过程中需要使用 $O(1)$ 复杂度的 GCD ，但实际上很难将答案构造到 26 步。事实上，命题组只生成出了答案为 16 的数据。所以使用 log 的 GCD 算法也可以通过。

G - 最大公因数

题解

本题还有一种 $O(n^2)$ 的动态规划做法，但是需要使用 $O(1)$ 的 GCD。使用 $O(\log n)$ 的 GCD 会超时。

目录

1 简单题
- 1.1 M - 矩阵构造
- 1.2 J - 又一个排序问题
- 1.3 H - 扫描地平线
- 1.4 L - 绝望线缕

2 中等题
- 2.1 C - 金币
- 2.2 G - 最大公因数
- **2.3 E - 提取权值**
- 2.4 D - 套娃

3 困难题
- 3.1 F - 花
- 3.2 A - 防毒
- 3.3 B - 括号
- 3.4 I - 物品
- 3.5 K - 密钥恢复

E - 提取权值

题目大意

给定一棵树，每个点有权值。已知根节点的权值为 0 。

你可以提出 n 个问题，每个问题询问距离为 k 个两个点的简单路径上所有点权值的异或和。

已知根节点权值为 0 ，问能否通过询问计算所有点的权值？如果能，需要通过交互证明。

$n, k \leq 250$

现场情况

首次提交：WF 在逃三人团 @ 00:46
首次通过：呜呜呜嘤嘤嘤哇哇哇 @ 00:53
通过队数：49

E - 提取权值

题解

枚举所有可能的询问，将这些询问表示成方程的形式。只要能找到 n 个线性无关的方程，即可通过这些方程对应的询问计算出每个点的权值。

需要注意的是，你需要将 $w_1 = 0$ 也作为一个方程插入方程组中，而不是先找到 $n-1$ 个方程，然后直接加入 $w_1 = 0$ 这个方程。因为 $w_1 = 0$ 可能可以在解方程的过程中解出来。

复杂度为 $O(\frac{n^4}{64} + n^3)$ 。

目录

1 简单题
  1.1 M - 矩阵构造
  1.2 J - 又一个排序问题
  1.3 H - 扫描地平线
  1.4 L - 绝望线缕

2 中等题
  2.1 C - 金币
  2.2 G - 最大公因数
  2.3 E - 提取权值
  2.4 D - 套娃

3 困难题
  3.1 F - 花
  3.2 A - 防毒
  3.3 B - 括号
  3.4 I - 物品
  3.5 K - 密钥恢复

D - 套娃

题目大意

给定一排 n 个套娃，套娃的大小互不相同。你可以将相邻两个套娃套在一起，问最多能套几次?

$n \leq 10^5$

现场情况

首次提交：小甜甜 @ 01:42
首次通过：一路向北 @ 02:20
通过队数：3

D - 套娃

题解

考虑一个更简单的问题：给定一个套娃序列，能否将这些套娃全部合并成一个?

该问题可以贪心解决：当任意相邻的两个套娃可以“紧密”的合并时，就立即合并它们。

D - 套娃

题解

回到原问题。

首先观察到如下性质：假如一段区间可以合并成一个套娃，则其任意子区间都可以合并成一个套娃。

推广上述性质：如果一个区间不能合并成一个套娃，则任意包含该区间的区间都不能合并成一个套娃。

从左到右决定将哪些套娃合并成一个套娃。由上述性质，可以贪心地将右侧的套娃合并到左侧的区间中，而不会使答案变差。

原问题转化为：指定一个区间的左端点，其最多能合并右侧多少套娃?

D - 套娃

题解

问题：指定一个区间的左端点，其最多能合并右侧多少套娃？

由上述性质，右端点的位置是可以二分的。使用一开始讲的方法判断整个区间是否能合并成一个套娃。判断方法要求先对区间离散化，因此假设待判断的区间长度为 n ，时间复杂度为 $O(n\log n)$ 。

需要注意的是，要保证二分的初始区间总长度在 $O(n)$ 级别。这样总体复杂度才能控制在 $O(n\log^2 n)$ 。为了实现这一点，二分的初始右端点不能直接设成 n ，而是应当先倍增的查找第一个不能合并成一个套娃的右端点。

D - 套娃

花絮

- 另一种可能的做法是：每次尝试加入右侧的一个套娃，但是其解法比较复杂，可能占用较多的机时。

目录

1 简单题
1.1 M - 矩阵构造
1.2 J - 又一个排序问题
1.3 H - 扫描地平线
1.4 L - 绝望线缕
2 中等题
2.1 C - 金币
2.2 G - 最大公因数
2.3 E - 提取权值
2.4 D - 套娃
3 困难题
3.1 F - 花
3.2 A - 防毒
3.3 B - 括号
3.4 I - 物品
3.5 K - 密钥恢复

目录

1 简单题
1.1 M - 矩阵构造
1.2 J - 又一个排序问题
1.3 H - 扫描地平线
1.4 L - 绝望线缕
2 中等题
2.1 C - 金币
2.2 G - 最大公因数
2.3 E - 提取权值
2.4 D - 套娃
3 困难题
3.1 F - 花
3.2 A - 防毒
3.3 B - 括号
3.4 I - 物品
3.5 K - 密钥恢复

F - 花

题目大意

计算满足以下条件的图的数量：

- 图中包含 n 个点，编号 $1 \sim n$ 。
- 与 1 号点相连的所有点的编号互质，称为关键点。
- 其余点都位于关键点往外延伸出的一条链上，链上点的编号是最近关键点的编号的倍数，且单调递增。

$n \leq 10^{10}$

现场情况

首次提交：思路打开 @ 00:59
首次通过：思路打开 @ 00:59
通过队数：9

F - 花

题解

由于限制三“链上的点必须是关键点的倍数”，所有的质数都只能是关键点。

由于所有的质数都是关键点，加上限制二“关键点的编号互质”，任意的合数都不是关键点。

由于链上的点必须按编号升序排列，只需要考虑每个合数位于哪个质因数的链上即可。

F - 花

题解

对于合数 x ，其放置方式一共为 $d(x)$ 种，$d(x)$ 表示 x 本质不同的质因数个数。题目所求的答案即为 $\prod_{i=2}^{n} d(i)$ 。

F - 花

题解

在 10^{10} 内 $d(i)$ 的至多取到 10，于是我们考虑如何对每个整数 w 求出 $d(i)=w(w\in[2,10])$ 的 i 数量。

一个朴素的做法是从小到大对 n 以内所有的质数进行暴力搜索，不断地在当前搜索到的值后面添加质因数，同时传递当前搜索到的数有多少个本质不同的质因子。这个做法需要我们筛出 n 以内的所有质数，然后暴力搜索的复杂度，由于每个 n 以内的数只会被搜索一次，复杂度为 $O(n)$ 。

然后我们考虑如何优化这个做法。设 $w(x)$ 为 x 的最大质因数，p_i 为第 i 个质数，$c(x)$ 为 x 以内的质数个数。我们注意到，在搜索的最后一层，当 $w(x)*x>n$ 时，我们其实不必递归进入这一层，可以直接在外层统计。那么我们考虑 x 是除掉一个最大质因子后得到的值，当搜索到第 i 个质数时，有 $x*p_i*p_i>n$ ，则 x 乘上第 i 到第 $c(\lfloor\frac{n}{\lfloor n/p_i\rfloor}\rfloor)$ 个质数的方案都可以在搜索到 x 时统计，其复杂度和 $x*w(x)<n$ 的数的个数同级，在 10^{10} 内大约需要搜索 6.2×10^6 次。

F - 花

题解

剩下的问题就是如何迅速求出 $c(\lfloor\frac{n}{\lfloor n/p_i\rfloor}\rfloor)$ 。

我们需要求得 n 以内的质数个数，其实就是 $g(x)=1$ 这个函数前缀和质数部分的贡献，可以使用 min_25 筛的前半部分求质数贡献的 dp 方程得到。

设 $G_{k,n}$ 为第 k 轮埃氏筛后剩下的数的 g 值之和，我们可以得到如下递推方程：

$$G_{k,n}=G_{k-1,n}-[p_k^2\le n]*g(p_k)(G_{k-1,n/p_k}-G_{k-1,p_{k-1}})$$

筛完之后，我们可以获得任何 $\lfloor n/d\rfloor$ 以内的质数个数。

复杂度为 $O(\frac{n^{3/4}}{\log n}+n^{1-\epsilon})$，最后一项至多取到 6.2×10^6 。

目录

1 简单题
1.1 M - 矩阵构造
1.2 J - 又一个排序问题
1.3 H - 扫描地平线
1.4 L - 绝望线缕
2 中等题
2.1 C - 金币
2.2 G - 最大公因数
2.3 E - 提取权值
2.4 D - 套娃
3 困难题
3.1 F - 花
3.2 A - 防毒
3.3 B - 括号
3.4 I - 物品
3.5 K - 密钥恢复

A - 防毒

题目大意

给定一张 n 个点的有向图。流感季共经历 q 天，第 i 天会有病毒从城市 a_i 出发，向城市 1 传播。如果存在一条从 a_i 到 1 的路径，则产生 b_i 的经济损失代价。

为了减小代价，你每天可以选择一个城市部署病毒过滤器。在第 i 个城市部署的代价为 c_i 。但同时只能存在一个病毒过滤器，下一个部署后上一个自动失效。

问：前 i 天最小的代价总和是多少？你需要对于每个 i 都回答。

$n, q \leq 10^5$

现场情况

首次提交：一路向北 @ 02:26
首次通过：哥哥来了 @ 02:57
通过队数：3

A - 防毒

题解

病毒过滤器只有部署在支配点上才能生效，因此首先对原图运行支配树算法，转化为一个支配树上的问题。

设 $f_{i,j}$ 表示经历了前 i 天，且当前病毒过滤器部署在点 j 的情况下，最小的代价总和，转移方程为：

$$f_{i,j} = \min\{f_{i-1,j}, f_{i-1,k} + c_j\} + w_{i,j}$$

式中，$w_{i,j}$ 表示第 i 天中午病毒过滤器位于点 j 时的经济损失代价。若 j 位于 a_i 到 1 的路径上，则 $w_{i,j} = 0$ ，否则 $w_{i,j} = b_i$ 。

A - 防毒

题解

$$f_{i,j} = \text{min}\{f_{i-1,j}, f_{i-1,k} + c_j\} + w_{i,j}$$

转移时，可以将 i 相同的部分统一转移。对状态的第二维建线段树，状态的转移可以表达为线段树上的修改。令 t_j 表示病毒过滤器部署在节点 j 时的最小代价。线段树上的每个节点需要维护所覆盖区间内 t_j 的最小值。

转移可以分为两步：第一步，令所有的 t_j 对 $\text{min}\{t_j\} + c_j$ 取 min 。第二步，将除了 a_i 到 1 的路径上以外的所有 t_j 加上 b_i 。

对于第一步：由于 $\text{min}\{t_j\}$ 在一次操作中是一个固定值，在对一个区间做这样的操作时，只有 c_j 最小的那个位置可能影响区间最小值。因此维护了区间中最小的 c_j ，就能实现这一操作。

对于第二步：对支配树进行树链剖分，然后执行区间加即可。

A - 防毒

复杂度分析

求支配树可以 $O(n)$ 的实现。

线段树的操作一可以 $O(q\log n)$ 实现，操作二可以 $O(q\log^2 n)$ 实现。

因此，总体复杂度为 $O(q\log^2 n)$ 。

A - 防毒

花絮

- 另一种可能是思路是：使用 f_i 表示在第 i 天前的晚上将病毒过滤器部署在城市 a_i 的最小代价，使用一些技巧快速选出最优的转移点。这种方法可能是可行的，但是比较复杂，命题组暂时没有找到这样的“技巧”。

目录

1 简单题
- 1.1 M - 矩阵构造
- 1.2 J - 又一个排序问题
- 1.3 H - 扫描地平线
- 1.4 L - 绝望线缕

2 中等题
- 2.1 C - 金币
- 2.2 G - 最大公因数
- 2.3 E - 提取权值
- 2.4 D - 套娃

3 困难题
- 3.1 F - 花
- 3.2 A - 防毒
- 3.3 B - 括号
- 3.4 I - 物品
- 3.5 K - 密钥恢复

B - 括号

题目大意

给定 m 个由一个长括号序列节选出的子括号序列，将它们两两配对，构造出尽可能多的合法括号序列。

$m \le 5 \times 10^5$

现场情况

首次提交：一路向北 @ 01:10
首次通过：一路向北 @ 01:10
通过队数：7

B - 括号

题解

每个子括号序列在化简后，要么是一段左括号，要么是一段右括号（或是为空）。其余的情况均可以直接忽略，因为其余的子括号序列不可能拼出合法的括号序列。

对于一段左括号，其必须要匹配顺序和类型一致的右括号，可以用哈希解决这一问题。

首先考虑如何计算左端点的 hash 值：维护一个括号栈，从左到右依次插入括号，并消除匹配上的括号。如果遇到右括号无法匹配左括号的情况，则将栈清空，并标记后续所有左端点小于该括号的区间无效。

当一个区间的所有括号都已经被处理过时，考虑此时区间左端点的位置：

- 如果左端点位于某对已经闭合的括号内部，则该区间是一个无效的区间，因为其左侧必然包含一段右括号。
- 否则，计算栈中所有位于该段区间内的左括号的哈希值，即为该括号子序列的哈希值。

对于右括号，只需要将整个序列和询问全部翻转，再做一次上面的过程即可。

B - 括号

题解

上述过程可以在 $O(n \log n)$ 的时间复杂度下实现。

如果您被卡常了，可以尝试消除一些 log 。此题时限较紧是因为可能有莫队能过，但我们依然将时限开到了标程的 5 倍。具体的说，每个部分的 log 消除方法为：

- 标记左端点位于某个区间内无效：使用链表消除 log 。
- 查找栈内位于左端点左侧的第一个字符的位置：每个下标对应的栈内左侧第一个字符的位置其实是固定的，可以预先计算出来，不需要二分。

B - 括号

花絮

本题的做法比较多。为了计算子序列的哈希值，还可以使用莫队、整体二分等做法，但是：

- 莫队的做法需要极小的常数，出题组没有写出能够通过该题的莫队解法（最优运行时长为时限的 1.5 倍），但是不排除莫队存在通过的可能性。
- 整体二分的做法常数同样较大，但经过一定的优化，可以在时限内通过。但是这种做法比较复杂，需要考虑的情况比较多，可能占用较多的机时。

目录

1 简单题
- 1.1 M - 矩阵构造
- 1.2 J - 又一个排序问题
- 1.3 H - 扫描地平线
- 1.4 L - 绝望线缕

2 中等题
- 2.1 C - 金币
- 2.2 G - 最大公因数
- 2.3 E - 提取权值
- 2.4 D - 套娃

3 困难题
- 3.1 F - 花
- 3.2 A - 防毒
- 3.3 B - 括号
- **3.4 I - 物品**
- 3.5 K - 密钥恢复

I - 物品

题目大意

给定 n 种物品，每种物品有无限个，重量在 $0 \sim n$ 中。

你需要判断能否选出恰好 n 个物品，使得总重量恰好为 m 。

$n \leq 10^5, m \leq n^2$

现场情况

首次提交：樱花泪 @ 00:13
首次通过：思路打开 @ 01:42
通过队数：1

I - 物品

题解

设 b_i 表示是否存在重量为 i 的物品，定义生成函数：

$$f(x) = \sum_{i=0}^{n} b_i x^i$$

那么答案为 $[x^m]f(x)^n > 0$ 。

直接计算 $f(x)^n$ 显然是不可行的。

不妨令 $B = \lfloor \frac{m}{n} \rfloor$ ，$w_i \leftarrow w_i - B$ ，$m \leftarrow m - nB$ 。显然，对 w 和 m 变换后的问题与原问题是等价的。此时，$w_i \in [-B, n - B]$ ，$m \in [0, n]$ 。

I - 物品

题解

注意到：假设存在一种取物品的顺序 $a_1, a_2, \ldots, a_n$ 使得 $(\sum_{i=1}^{n} a_i) = m$ ，则可以对 a 重新排序，使得 $\forall p \in [1, n], (\sum_{i=1}^{p} a_i) \in [0, n]$ 。

可以通过构造的方式证明存在这样的方案。

性质证明

考虑从 1 到 n 依次确定 a_i 的值。设 pre 表示已经确定的前缀的前缀和，可重集合 S 为排序前的 a_i 构成的集合，则：

- 当 $pre = B$ 时：随意选择 S 中任意的数加入 a 的末尾均可。
- 当 $pre < B$ 时：若 S 中存在大于 0 的数，则取出一个加入到 a 的末尾。否则，剩余的数都小于等于 0 ，且和恰好为 $m - pre$ ，以任意顺序全部加入到 a 的末尾即可。
- 当 $pre > B$ 时：若 S 中存在小于 0 的数，则取出一个加入到 a 的末尾。否则，剩余的数都大于等于 0 ，且和恰好为 $m - pre$ ，以任意顺序全部加入到 a 的末尾即可。

I - 物品

题解

因此，可以通过倍增的方式计算 $f(x)^n$ ，且每次倍增后只需要保留 $x^{-n} \sim x^n$ 中的项即可。

上述过程使用 FFT 或 NTT 实现均可。但在使用 FFT 时，需要每次倍增后都将系数二值化，否则会导致浮点溢出。NTT 理论上也需要这么做，否则会有极小的概率出错，但是出题人没有卡~~因为不会卡~~。

时间复杂度为 $O(n \log^2 n)$ 。

目录

1 简单题
1.1 M - 矩阵构造
1.2 J - 又一个排序问题
1.3 H - 扫描地平线
1.4 L - 绝望线缕

2 中等题
2.1 C - 金币
2.2 G - 最大公因数
2.3 E - 提取权值
2.4 D - 套娃

3 困难题
3.1 F - 花
3.2 A - 防毒
3.3 B - 括号
3.4 I - 物品
3.5 K - 密钥恢复

K - 密钥恢复

题目大意

给定一个加密算法。通过测试某些输入对应的输出，反推该算法使用的密钥。

现场情况

首次提交：都市狂少 @ 03:19
首次通过：? @ ??:??
通过队数：0

K - 密钥恢复

符号约定

下列每个符号都代表一个 4bit 向量的取值列表，每个列表的长度为 16。

- $\mathcal{A}$：表示该位置的取值为全部可能的取值各一个，即该列表是一个 $0 \sim 15$ 的排列。
- $\mathcal{B}$：表示该位置的取值的异或和为 0 ，即 $\bigoplus_{i=0}^{15} v_i = 0$ 。
- $\mathcal{C}$：表示该位置的取值为某一个常数，即 $\mathcal{C} = \{c, c, c, \ldots, c\}$ 。注意，下文中不同的 $\mathcal{C}$ 对应的常数值 c 可以不一样。
- $\mathcal{X}$：表示该位置的取值不具有任何性质。

K - 密钥恢复

性质分析

下面，分析每个符号在经过每种结构后具有的性质。

对于 xor 结构：$\mathcal{A}, \mathcal{B}, \mathcal{C}$ 在经过该结构后均保持原有的性质不变，即依然可以表达为 $\mathcal{A}, \mathcal{B}, \mathcal{C}$ (尽管列表内的值改变了，其性质依然保持)。

对于 perm 结构：$\mathcal{A}, \mathcal{C}$ 在经过该结构后均保持原有的性质不变，即依然可以表达为 $\mathcal{A}, \mathcal{C}$ 。而 $\mathcal{B}$ 在经过该结构后会变为 $\mathcal{X}$ 。

对于 mix 结构，其事实上是两两混合了 4bit 向量，并打乱了它们的位置。考虑如下四种情形：

- $\mathcal{C}$ 与 $\mathcal{C}$ 混合：结果为 $\mathcal{C}$ 与 $\mathcal{C}$ 。
- $\mathcal{A}$ 与 $\mathcal{C}$ 混合：结果为 $\mathcal{A}$ 与 $\mathcal{A}$ 。
- $\mathcal{A}$ 与 $\mathcal{A}$ 混合：结果为 $\mathcal{B}$ 与 $\mathcal{B}$ 。
- $\mathcal{B}$ 与 $\mathcal{B}$ 混合：结果为 $\mathcal{X}$ 与 $\mathcal{X}$ 。

下文中，受页面宽度限制，只展示会对性质产生影响的结构，即忽略 xor 和大部分的 perm 。

K - 密钥恢复

题解

考虑以下过程：

$$
\begin{array}{|c|}\hline \mathcal{A}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \hline\end{array}
\xrightarrow{\text{M}}
\begin{array}{|c|}\hline \mathcal{A}\\ \mathcal{C}\\ \mathcal{A}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \mathcal{C}\\ \hline\end{array}
\xrightarrow{\text{M}}
\begin{array}{|c|}\hline \mathcal{A}\\ \mathcal{C}\\ \mathcal{A}\\ \mathcal{C}\\ \mathcal{A}\\ \mathcal{C}\\ \mathcal{A}\\ \mathcal{C}\\ \hline\end{array}
\xrightarrow{\text{M}}
\begin{array}{|c|}\hline \mathcal{A}\\ \mathcal{A}\\ \mathcal{A}\\ \mathcal{A}\\ \mathcal{A}\\ \mathcal{A}\\ \mathcal{A}\\ \mathcal{A}\\ \hline\end{array}
\xrightarrow{\text{M}}
\begin{array}{|c|}\hline \mathcal{B}\\ \mathcal{B}\\ \mathcal{B}\\ \mathcal{B}\\ \mathcal{B}\\ \mathcal{B}\\ \mathcal{B}\\ \mathcal{B}\\ \hline\end{array}
\xrightarrow{\text{M}}
\begin{array}{|c|}\hline \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \hline\end{array}
\xrightarrow{\text{MXPM}}
\begin{array}{|c|}\hline \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \mathcal{X}\\ \hline\end{array}
$$

经历了四轮 `mix` 和一轮 `perm` 后，性质已经变成了 $\mathcal{X}$ 。而题目后续还有四个结构，结果更是不具有任何性质。

K - 密钥恢复

题解

对上述过程进行一些改进：

$$
\boxed{\begin{array}{c}\mathcal{A}_1\\\mathcal{A}_2\\\mathcal{C}\\\mathcal{C}\\\mathcal{C}\\\mathcal{C}\\\mathcal{C}\\\mathcal{C}\end{array}}
\xrightarrow{\mathsf{M}}
\boxed{\begin{array}{c}16\mathcal{A}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\end{array}}
\xrightarrow{\mathsf{M}}
\boxed{\begin{array}{c}16\mathcal{A}\\16\mathcal{C}\\16\mathcal{A}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\\16\mathcal{C}\end{array}}
\xrightarrow{\mathsf{M}}
\boxed{\begin{array}{c}16\mathcal{A}\\16\mathcal{C}\\16\mathcal{A}\\16\mathcal{C}\\16\mathcal{A}\\16\mathcal{C}\\16\mathcal{A}\\16\mathcal{C}\end{array}}
\xrightarrow{\mathsf{M}}
\boxed{\begin{array}{c}16\mathcal{A}\\16\mathcal{A}\\16\mathcal{A}\\16\mathcal{A}\\16\mathcal{A}\\16\mathcal{A}\\16\mathcal{A}\\16\mathcal{A}\end{array}}
\xrightarrow{\mathsf{M}}
\boxed{\begin{array}{c}16\mathcal{B}\\16\mathcal{B}\\16\mathcal{B}\\16\mathcal{B}\\16\mathcal{B}\\16\mathcal{B}\\16\mathcal{B}\\16\mathcal{B}\end{array}}
\xrightarrow{\mathsf{P}}
\boxed{\begin{array}{c}\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\end{array}}
\xrightarrow{\mathsf{MX}}
\boxed{\begin{array}{c}\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\\\mathcal{X}\end{array}}
$$

令首轮的前两个向量取遍全部 256 种可能的组合。在第一轮 `mix` 后，这些组合可以被分为 16 个小组，每组都符合上一页中描述的初始性质。这样，在第五轮 `mix` 后，输出具有性质 $\mathcal{B}$ 。

题解

接下来，有两种方法可以找出密钥。

第一种方案是将最后一轮的 `xor` 操作等价地移动到 `mix` 操作之前，这用到了 `mix` 操作是线性变换的性质。在这种情况下，最后一步 `mix` 操作可以被直接反向计算。接下来，可以对等价密钥的每 4bit 分别进行尝试。从输出出发，反向计算出每组数据在第五轮 `mix` 之后的值。如果其不符合性质 $\mathcal{B}$ ，则此次猜测一定是错误的。

为了正确找出密钥，我们希望在密钥猜测错误的情况下，反推出的第五轮 `mix` 之后的值不符合性质 $\mathcal{B}$ 。在猜测错误的情况下，反推结果依然具有性质 $\mathcal{B}$ 的概率可以被近似的看作 $\frac{1}{16}$ 。因为这相当于将输出又异或了一个密钥，再经过一个 `perm` 结构，结果事实上变得“更乱了”。选手们也可以写一个程序实验验证该结论。

如果不想推演等价密钥的变换过程，也可以一次直接枚举位于同一个 `mix` 组内的 8bit 密钥。这两种方法是完全等价的，后者的枚举代价高于前者，但在本题中不影响复杂度。

K - 密钥恢复

题解

使用第一种方法，我们在期望下可以用 256 个询问将每 4bit 可能的密钥值都排除 $\frac{15}{16}$ 。对于 4096 个询问，一个 4bit 位置上的错误密钥依然存活的概率为 2^{-64} 。需要指出，第二种方法的过滤概率与第一种方法是一致的，受篇幅所限，在此不做赘述。

如果你对这个概率还是不放心，还可以逐个验证剩余所有可能的密钥与你提出的询问是否相符。不过，这已经没有什么必要了。