Genshin Round 3 Solution

a

- 如果你马上观察大样例，你就会发现序列经过一次变换与经过三次变换的结果相同。
- 所以若 $m \mod 2 = 1$，则可以只对原序列进行一次变换，否则对原序列进行两次变换。
- 考虑证明。设 b 为一次变换后的序列，$c_{i,j}$ 表示经过一次操作后前 i 个数中 j 的个数，则由定义得 $c_{i,j+1} \le c_{i,j}$。则经过三次变换后序列的第 i 个位置为 $\sum_{k \le i}[c_{k,b_k} = c_{i,b_i}] = \sum[c_{i,b_i} \le c_{i,j}] = \sum[j \le b_i] = b_i$ 。

b

- 设 $f_{i,a,b}$ 表示前 i 组留下 a,b 的最大贡献。设第 i 组 3 个数为 u,v,w。
- 若 $u=v=w$，则显然贪心地把这三个取走即可。
- 有转移 $f_{i,a,b} \leftarrow f_{i-1,a,b}$，即把 u,v,w 丢掉。若我们用滚动数组省去 i 一维，则这相当于不改变 dp 数组。
- 剩下的情况 a,b 与 u,v,w 有交。这样的状态只有 $O(n)$ 个，可以直接更新，维护行最大值和全局最大值即可。
- 时间复杂度 $O(n^2)$。

C

- 设 $f_{l,r,u}$ 表示左边界为 l，右边界为 r，上边界为 u 时下边界最大为多少。
- 有 $f_{l,r,u} \le \min(f_{l+1,r,u}, f_{l,r-1,u}, f_{l,r,u+1})$
- 想想还有哪些限制没考虑到，发现只有 $a_{u,l} \neq a_{i,r}$ 和 $a_{u,r} \neq a_{j,l}$ 的限制。
- 枚举 l, r，u 从下往上扫，开两个桶维护 l 列和 r 列即可。$O(n^3)$。

d

- 考虑容斥。若我们钦定一条路径上经过 x 个非法点，且排列中有 y 个 -1 没有填，则贡献系数为 $(-1)^x \times y!$。
- 非法点有两种，一种是在输入中确定了的，一种没有在输入中确定。没有在输入中确定的非法点不能与任何一个非法点在同一行或同一列。
- 因此考虑 dp，设 $f_{i,j,k,0/1,0/1}$ 表示现在走到 (i,j)，排列已经填了 k 个 -1，当前行与当前列有没有非法点。从 $(i-1,j)$ 和 $(i,j-1)$ 转移过来即可，在转移过程中算出 $(-1)^x$ 的贡献，最后乘上 $y!$。
- 复杂度 $O(n^3)$。似乎有不需要容斥的做法。