[YDRS007] 云斗五月月赛题解

大家看看自己和 std 有没有心有灵犀呀~

A 礼物

考虑取 $(n+1)!+2$ 至 $(n+1)!+(n+1)$。

我们知道：

$$(n+1)!=1\times 2\times\cdots\times(n+1)$$

因此 $(n+1)!$ 一定为 $(i+1)$ 的倍数，其中 $1\le i\le n$。故此，有：

$$\begin{aligned}(n+1)!+(i+1)&=(i+1)\times\frac{(n+1)!}{i+1}+(i+1)\\&=(i+1)\times\left(\frac{(n+1)!}{i+1}+1\right)\end{aligned}$$

由于 $\dfrac{(n+1)!}{i+1}$ 为正整数，故 $i+1$ 和 $\dfrac{(n+1)!}{i+1}+1$ 均为大于 1 的正整数。根据定义，$(n+1)!+(i+1)$ 为合数。

观察到当 $n=20$ 时，$(n+1)!=21!\approx 5.1\times 10^{19}$ 超出了 `long long` 的范围。考虑上述构造可行的本质是 2 至 $n+1$ 均为 $(n+1)!$ 的一个正因数，因此不妨取 2 至 $n+1$ 的最小公倍数，这是必定可行的。使用这种构造，即使当 $n=41$ 时，$\left(\operatorname{lcm}_{i=1}^{n+1} i\right)\approx 2.2\times 10^{17}$ 也是绰绰有余的。

因此，构造 $\left(\operatorname{lcm}_{i=1}^{n+1} i\right)+2$ 至 $\left(\operatorname{lcm}_{i=1}^{n+1} i\right)+(n+1)$ 即可。

时间复杂度 $O(n\log n)$。

事实上，在 `int` 范围内 n 可取 281。

B 小清新构造题

先给出构造方法：对于第 i 行第 j 列，填入 $C_{i,j}=(a_i+j-1)\bmod n+1$。

下面证明其正确性。

首先看第一个要求，其 $\max$ 值是显然满足的。而对于同一个 i 值若 j 分别取 1 至 n 所组成的序列必定为一个 $1\sim n$ 的排列，且当且仅当 $j=n$ 时才有 $C_{i,j}=a_i$，又本题中仅有 $n-1$ 列，故 $C_{i,1}$ 至 $C_{i,n-1}$ 中必定有 1 至 a_i-1 且必定没有 a_i，即 mex' 值为 a_i。

然后是第二个要求，由于 a_i 两两不同，故对于固定的 j 值有 $C_{i,j}$ 两两不同，即为 $1\sim n$ 的一个排列。

综上，得证。

时间复杂度 $O(n^2)$。

思考：如何构造使得字典序最小?

C 词汇测试

我们不难发现，对于一个等差数列，只要其中两项确定了，整个数列就确定了。具体地，设你知道了第 p, q 项，那么公差就是 $\frac{a_p - a_q}{p - q}$。所以答案肯定是不太多的。为了优化枚举，我们可以选出做测试最少的两天进行枚举并检验，这样只需要检验 $O((\frac{L}{n})^2)$ 个等差数列。

每次检验如果暴力 `lower_bound`，总时间复杂度为 $O((\frac{L}{n})^2 n \log L) = O(\frac{L^2}{n} \log L)$（然而非常不满，但是我暂时还没有更精确的估计），可以通过。

假设枚举的是 a_p, a_q，在 a_p 不动时，a_q 越大，你 `lower_bound` 结果也会越大。使用单调性可以在 $O(L)$ 时间内处理出同一个 a_p 对应的所有答案，总时间复杂度 $O(\frac{L^2}{n})$，更加安全。

D 在银河中孤独摇摆

考虑每次操作之后立刻翻转整个序列（选择 $k = n, A_i = (p_i)$ 即可），那么此时操作等效于将每个段内部翻转。

考虑有这个操作之后怎么做，先考虑怎么将 01 序列排序，把极长的相同连续段缩起来，设当前序列中有 m 个极长连续段，且以 0 开头，以 1 结尾（为此可能需要在开头或结尾添加长为 0 的连续段）。我们考虑对每个 `0101`（这里 `0/1` 代表一个极长的 $0/1$ 连续段）这样的段，进行操作 `0101 -> [0][10][1] -> 0011`，于是段数就会除掉 2。

于是只需要 $O(\log n)$ 次操作就可以将一个 01 序列排序。

考虑一般的排列怎么做，我们把 $> \frac{n}{2}$ 的数视为 1，其余的视为 0，对其排序之后，再对两边递归下去分别做。这里递归时每一层的区间可以同步操作，于是每一层的操作数都是 $O(\log n)$，一共有 $O(\log n)$ 层，故总次数为 $O(\log^2 n)$。

实现的时候没必要每次操作的时候都立刻翻转，可以假装序列翻转了，对下标做一些变换，最后如果序列变成了 $n, n-1, \cdots, 1$ 就再整体翻转一遍，可以少两倍常数。

综上，我们在 $O(\log^2 n)$ 次操作（大约为 $\frac{1}{2}\log^2 n$ 次）内成功将序列排序。

E 希望有羽毛和翅膀

首先，$F(S)$ 可以以如下方式计算：找到最大的 k，满足 $\sum_{x \in S} [x < k] \geq k$，也就是说用 S 中 $< k$ 的数可以铺满 $[0, k-1]$。证明考虑首先如果 k 符合这个条件，那么显然可以构造方案用 $< k$ 的数铺满 $[0, k-1]$；另一方面，设 p 是最大的满足此条件的数，如果存在 $q > p$ 使得我们可以构造 $\operatorname{mex} = q$ 的方案，那么我们构造的时候一定要用到 $\geq q$ 的数，用它们来往下铺。那么此时一定存在 $r > q$ 满足 r 也满足 $\sum_{x \in S} [x < r] \geq r$，导出矛盾。

此时实现莫队算法配合数据结构维护可以得到 $O(n\sqrt{n \log n})$ 或 $O(n\sqrt{n})$ 做法，最高可以获得 72 分。

现在考虑怎么处理区间询问，扫描线扫值域维度，从大到小枚举答案，对每个 $[l, r]$，一旦它符合条件了，我们就将其删除。注意到如果 $[l_1, r_1] \subseteq [l_2, r_2]$，那么 $[l_1, r_1]$ 的答案一定比 $[l_2, r_2]$ 小。于是我们同一时刻只需要维护互不包含的若干区间，每次删除时再插入若干新的未被包含的区间。

由于区间互不包含，每次修改某个 a_i 时，一定是对 l 在一个连续范围内的区间产生影响，于是我们可以使用线段树维护区间加，以及删除所有值 ≥ 0 的位置。

由于每个区间只会被插入和删除一次，因此总复杂度 $O((n+q)\log n)$。