



牛客竞赛

AC.NOWCODER.COM

2025 牛客 暑期多校训练营

cfz 出题组

HR.NOWCODER.COM



概况

- 入门: FDJ
- 简单: A
- 较简单: EH
- 中等: BKI
- 较难: CG
- 难题: 好像没有难题!

F - Flower

题意

- 有一朵 n 片花瓣的花。Yuki 每轮会先摘下 a 片花瓣，再摘下 b 片花瓣，直到摘完为止。
- 若 Yuki 摘下的最后一片花瓣属于某轮中的前 a 片，则 Yuki 会离开。
- 你需要先摘下一些花瓣，但不能摘下所有花瓣，使得 Yuki 最后会留下。
- 最小化摘下花瓣的个数，或者判断无解。
- 多组数据， $1 \leq t \leq 100$ ， $1 \leq n, a, b \leq 10^9$ 。

AC.

F - Flower

题解

- 考虑什么情况能使 Yuki 留下。每轮操作总共会摘下 $a + b$ 片花瓣，所以她能否留下只跟 $r = n \bmod (a + b)$ 的值有关：当且仅当 $r \notin [1, a]$ 时，她会留下（此时答案为 0）；否则，在 $1 \leq r \leq a$ 时，至少需要摘下的花瓣个数即为 r ，此时余数变为 0。特别地，若 $n \leq a$ 则无解。

D - Distant Control

题意

- 有 n 个机器人排成一排，初始时部分机器人开机，其余关机。
- 给定常数 $1 \leq a \leq n - 1$ ，每次你可以做以下两种操作：
 - 选择连续 a 个处于开机状态的机器人，将其关机；
 - 选择连续 $a + 1$ 个处于关机状态的机器人，将其开机。
- 求经过若干次操作后，处于开机状态的机器人数量的最大值。
- 多组数据， $\sum n \leq 4 \cdot 10^5$ 。

D - Distant Control

题解

- 如果初始时存在连续的至少 $a + 1$ 个关机机器人，可以把他们开机。而一旦存在连续的 a 个开机机器人，就可以不断拓展开机的连续段，使得所有机器人都开机。
- 于是若初始能进行任何合法操作，答案为 n ；否则答案为初始时 1 的个数。

J - Jetton

题意

- 初始时 Yuki 有 x 个筹码，Ena 有 y 个筹码。
- 每回合会确定该回合的获胜方，确定方式如下：
 - 若二人筹码量不相等，则筹码量较少的一方为获胜方；
 - 若二人筹码量相等，则 Yuki 为获胜方。
- 另一方需要向获胜方支付等同于获胜方持有量的筹码。某方筹码量变为 0 游戏结束。
- 求游戏是否会在有限回合内结束。若是，则求游戏历经的回合数。
- 多组数据， $1 \leq t \leq 10^5$ ， $1 \leq x, y \leq 10^9$ 。

J - Jetton

题解

- 结论：若游戏会结束，则回合数不会超过 $\lfloor \log_2(x + y) \rfloor$ 。证明如下：
 - 若游戏会结束，则在最后一回合中，两方的筹码量相等。
 - 考虑倒推。设两方的筹码量都为 n 。则上一回合，两方的筹码量必然为 $\frac{n}{2}$ 和 $\frac{3n}{2}$ ；如此类推，我们发现在 t 回合之前，双方的筹码量一定形如 $\frac{pn}{2^t}$ 和 $\frac{qn}{2^t}$ ，其中 p, q 均为奇数。又由于筹码量必然为整数，上述结论得证。
- 于是模拟即可，时间复杂度 $\mathcal{O}(t \log(x + y))$ 。此外，容易证明游戏能在有限回合内结束，充要于 $\frac{x+y}{\gcd(x,y)}$ 是 2 的正整数次幂。

A - Ad-hoc Newbie

题意

- 给定长度为 n 的正整数序列 f_1, \dots, f_n , 保证对每个 i 都有 $1 \leq f_i \leq i$ 。
- 构造一个 n 阶方阵 A 满足:
 - 对任意 $1 \leq i, j \leq n$, 都有 $0 \leq A_{i,j} \leq n$;
 - 对任意 $1 \leq i \leq n$,
 $\text{mex}(A_{i,1}, A_{i,2}, \dots, A_{i,n}) = \text{mex}(A_{1,i}, A_{2,i}, \dots, A_{n,i}) = f_i$ 。
- 多组数据, $\sum n^2 \leq 2 \cdot 10^6$ 。

A - Ad-hoc Newbie

题解

- 不失一般性地，假设 $f_1 \leq f_2 \leq \dots \leq f_n$ 。这是因为如果 f 数组不单调不降，我们可以将其排序，容易证明仍然满足 $1 \leq f_i \leq i$ 的性质，在最终输出时交换对应行列编号即可。
- 那么考虑构造一个非负整数序列 b_1, \dots, b_n ，满足对于方阵 A 有 $A_{i,j} = b_{\min(i,j)}$ 。这时，我们将原先的二维限制条件转化为了一维的：只需要满足对每个 $1 \leq i \leq n$ 都有 $\text{mex}(b_1, \dots, b_i) = f_i$ 即可。

A - Ad-hoc Newbie

题解 (cont'd)

- 令 $f_0 = 0$ 。如果 $f_i \neq f_{i-1}$, 根据定义, 这说明了 $b_i = f_{i-1}$; 设已经确定的 b 的位置有 k 个, 这说明 f_1, \dots, f_n 恰好有 k 种不同的元素, 对于其它未确定的 $n - k$ 个位置, 我们依次从小到大填入 $[1, n]$ 内所有未在 f_1, \dots, f_n 内出现的值即可。由于 $f_i \leq i$, 可以归纳说明, 我们的构造方式一定满足对每个 i , $[0, f_i)$ 内的整数均出现, 保证了其正确性。

E - Equal

题意

- 给定一个长度为 n 的正整数序列 a_1, \dots, a_n , 你可以做以下两种操作任意次:
 - 选择正整数 i, j, d 满足 $1 \leq i < j \leq n$ 且 $d | a_i, d | a_j$, 然后将 $a_i \leftarrow \frac{a_i}{d}$,
 $a_j \leftarrow \frac{a_j}{d}$;
 - 选择正整数 i, j, d 满足 $1 \leq i < j \leq n$, 然后将 $a_i \leftarrow a_i \cdot d$, $a_j \leftarrow a_j \cdot d$ 。
- 判断若干次操作后, 能否使 $a_1 = a_2 = \dots = a_n$ 。
- 多组数据, $1 \leq n \leq 10^6$, $1 \leq a_i \leq 5 \cdot 10^6$, $\sum n \leq 2 \cdot 10^6$ 。

AC.

E - Equal

题解

- 让我们分别考虑每个质因子 p^k , 因为它们总是独立的。取 \ln 使我们能够将乘法和除法视为加法和减法。可以观察到 $(\sum k) \bmod 2$ 是一个不变量, 能够发现一个必要条件: 要么 $2 \nmid n$, 要么 $2 \mid \sum k$ 且 $n > 2$, 要么 $n = 2$ 且 $k_1 = k_2$ 。

E - Equal

题解 (cont'd)

■ 这个必要条件是充分的，我们可以做一个构造性证明：

■ 当 $(\sum k) \bmod 2 = 0$ 时：

- 当 $n = 2$ 时，当 $k_1 \neq k_2$ ，同时加减均不可能使 $k_1 = k_2$ 。
- 当 $n > 2$ 时：考虑反复选择两个位置 $k_i, k_j > 0$ ，并同时将它们减去 1，直到无法选出。然后，就最多会有一个位置满足 $k_i > 0$ （如果没有，我们已经达到了目标）。我们选择两个位置 i_1, i_2 使得 i, i_1, i_2 两两不同，并依次执行以下操作：将 k_{i_1}, k_{i_2} 加 1；将 k_i, k_{i_1} 减 1；将 k_i, k_{i_2} 减 1。由于 $(\sum k) \bmod 2$ 是不变的， a_i 在此过程中始终为偶数，最终一定能减少到 0。这使得所有 k_i 相等。

E - Equal

题解 (cont'd)

■ 这个必要条件是充分的，我们可以做一个构造性证明：

■ 当 $2 \nmid n$ 时：

- 当 $n = 1$ 时，所有数字已经相等。
- 当 $n > 1$ 时：如果 $(\sum k) \bmod 2 = 0$ ，我们可以执行与上述相同的过程。否则，我们仍然可以使用相同的方法让数组 k 包含恰好 $n - 1$ 个 0 和 1 个 1。通过将 $n - 1$ 个 0 配对，我们可以将它们全部增加到 1，从而使所有 k_i 相等。

AC.

E - Equal

题解 (cont'd)

- 设 V 为 $\max(a_i)$ 。朴素实现的瓶颈在于质因数分解，时间复杂度为 $\mathcal{O}(n\sqrt{V})$ 。然而，我们只需要检查所有质数出现的次数是否为偶数；因此，可以使用 XOR Hashing。我们首先为每个质数设置一个随机哈希值，然后使用线性筛初始化所有合数的哈希值，时间复杂度为 $\mathcal{O}(n + V)$ 。

H - Head out to the Target

题意

- 给定一棵 n 个结点的树。有一枚棋子初始位于 1 号结点。
- 按序给定 k 个时间段。第 i 个时间段 $[l_i, r_i]$ 中，树上会出现一个目标 u_i 。保证时间段两两不交。
- 每一时刻，若目标存在，棋子与目标位置不同且连通，则棋子向目标移动一步；否则棋子不动。若此时二者位置相同，则称该时刻二者重合。
- 你可以在任意时刻切断树上任意数量的边。每条边被切断后不会恢复。
- 判断棋子能否与目标重合，若能则最小化重合发生的时刻。
- $1 \leq n \leq 10^6$, $1 \leq k \leq 10^6$, $1 \leq u_i \leq n$, $1 \leq l_i \leq r_i \leq 10^9$ 。

H - Head out to the Target

题解

- 观察到任意时刻可达的点是一个包含根的极大连通块。
- 考虑直接维护这个连通块，当目标在 $[l_i, r_i]$ 时刻内出现在 u_i 时，相当于将这个连通块向 u_i 方向扩张 $(r_i - l_i + 1)$ 步。
- 考虑使用倍增，从 u_i 开始向上找到最接近的可达点，然后暴力向下逐步标记为可达，若在 $(r_i - l_i + 1)$ 步前 u_i 已经被标记可达即可输出答案。
- 时间复杂度 $\mathcal{O}(n \log n)$ 。

B - Bitwise Puzzle

题意

- 给定三个非负整数 a 、 b 和 c 。你可以执行以下操作最多 64 次：
 - 1 $a \leftarrow a \cdot 2;$
 - 2 $b \leftarrow \lfloor \frac{b}{2} \rfloor;$
 - 3 $a \leftarrow a \oplus b$, 其中 \oplus 是按位异或;
 - 4 $b \leftarrow b \oplus a。$
- 请在不超过 64 次操作的情况下使 $a = b = c$, 或判断无解。可以证明, 给定约束条件下, 若合法方案存在, 一定存在不超过 64 次操作的方案。
- 多组数据, $1 \leq t \leq 10^4$, $0 \leq a, b, c < 2^{31}$ 。

B - Bitwise Puzzle

题解

- 当且仅当 $a = b = 0$ 且 $c \neq 0$ 时无解。
- 我们假设 c 的位数是最大的，考虑怎么让 a 变成 c 。首先我们可以进行最多一次异或操作，让 a 和 b 的最高位相同；然后我们让 a 不断乘 2，在这个过程中用 $a \leftarrow a \oplus b$ 进行调整（具体地，如果当前 b 最高位对应的 a 的最终位和 c 的最终位不一致，因为 b 的最高位一定是 1，使用一次异或即可）；在 a 和 c 的位数一样后，再让 b 不断除以 2，在这个过程中仍然用 $a \leftarrow a \oplus b$ 进行调整，这样我们就让 a 和 c 相等了。于是我们只需要让 b 除成 0 并进行一次 $b \leftarrow a \oplus b$ 即可。

B - Bitwise Puzzle

题解 (cont'd)

- 这样每一位的代价均为 2，算上第一次的异或操作和最后一次的异或操作后，最多进行 64 次操作。
- 如果 c 的位数不是最大的话，只需要在让 a 和 b 的最高位相同后不断执行 $a \leftarrow a \oplus b$ 和 b 整除 2 即可，每一位的代价仍然为 2。还是最多只需要 64 次操作，问题解决。

K - Kaleidoscope

题意

- 长度为 n 的排列 p_1, \dots, p_n 缺失了一部分。
- 对于排列 p , k 称为 p 的前缀最大位置, 当且仅当 $p_k = \max(p_1, \dots, p_k)$ 。
- 称两个排列 p 和 q 相似, 当且仅当 p, q 长度相同, 且前缀最大位置构成的集合相同。
- 求不同排列 q 的数量, 使得存在填补 p 缺失位置的方法令 p 与 q 相似。
- 答案对 998244353 取模。
- 多组数据, $1 \leq t \leq 1.8 \cdot 10^3$, $1 \leq n \leq 5 \cdot 10^3$, $\sum n \leq 10^4$ 。

K - Kaleidoscope

题解

- 考虑如果要求集合 S 内的元素恰为前缀最大值，计算合法的排列 q 个数：这个问题是经典的，我们只给出最后的结果为

$$(n-1)! [1 \in S] \prod_{u \in S, u > 2} \frac{1}{u-1},$$

K - Kaleidoscope

题解 (cont'd)

- 考虑确定哪些集合 S 是可能出现的。考虑贪心。 S 内最靠后的元素显然应当值为 n 。随后从 $k = (n - 1), \dots, 1$ 依次考虑，如果 S 倒序排序里的下一个元素值可能是 k ，那么就令此元素的值为 k ，并跳到下一个元素；否则 k 必须被放在已经被前缀最大值覆盖的后缀空位中。不合法当且仅当某个值为 k 的元素已经存在，但不在被前缀最大值覆盖的空位中；或者空位数量不足。

K - Kaleidoscope

题解 (cont'd)

- 那么从后向前 dp。设 $f_{i,j}$ 表示考虑了位置 i, \dots, n , 且确定位置 i 一定是前缀最大值位置, 且此时 k 的最大值是 j , 可能的贡献和。具体转移时的细节较多:
 - 我们不妨假定给出的所有 p 的位置是单调递增的; 如果某个给定值的位置一定不可能是前缀最大值所在位置, 可以直接删去它并平移所有大于它的值。
 - 同时, 我们应当预处理 t_1, \dots, t_n 表示位置 i 转移到的下一个前缀最大值的值至少是 t_i , 否则空位数量不足; 令 $P_i = \max(p_1, \dots, p_n)$ 。

K - Kaleidoscope

题解 (cont'd)

■ 具体转移时的细节较多：

- 考虑具体转移：若 $p_i \neq 0$ ，则此位置已经被确定。对于 (i', j') 转移到 i ，一定不存在 $i < i'' < i'$ 满足 $p_{i''} > j'$ ，这说明每个 (i', j') 转移到的 $p_i \neq 0$ 位置唯一（如果在二维平面上，这形如若干个 L 形），直接依此进行转移即可 $\mathcal{O}(n^2)$ ；
- 若 $p_i = 0$ ，则枚举 $(P_i, n]$ 内的每种未被占用的值 j ，若 $j = n$ 则 $f_{i,j} = 1$ ；否则能转移到 (i, j) 的 j' 要么是 $j + 1$ ，要么是 $(j, n]$ 内最小的未被占用的值。对于这一类转移，合法的 i' 需要满足 $j \geq t_{i'}$ ，而对于每一个固定的 j' ，其能转移到的 j 若存在则唯一，找到最大合法的 i' 并计算即可。



K - Kaleidoscope

题解 (cont'd)

- 时间复杂度 $\mathcal{O}(n^2)$ 。

K - Kaleidoscope

题解 2

- 考虑直接从前向后 dp，和上一个做法同理地，如果位置 i 是前缀最大值，则有 $\times 1$ 的系数，否则有 $\times(i-1)$ 的系数。贪心地考虑，令 $g_{i,j}$ 表示对于前缀 $[1, i]$ ，当前前缀最大集合等价类中，前缀最大值的最小值为 j 的所有贡献系数和；令 r_v 表示最小的 $u \geq v$ 满足值 u 不在 p 中出现过，不存在则为 $n+1$ 。

K - Kaleidoscope

题解 2 (cont'd)

- 那么考虑转移，枚举 $f_{i,j}$,
- 若 $p_{i+1} = 0$ ，
 - 欽定 $i+1$ 是前缀最大值： $f_{i+1,r_{j+1}} \leftarrow^+ f_{i,j}$;
 - 欽定 $i+1$ 不是前缀最大值：若 $\leq j$ 且未出现或出现位置不超过 $i+1$ 的值个数至少是 $i+1$ ，那么 $f_{i+1,j} \leftarrow^+ f_{i,j} \cdot i$; 否则，若 j 未出现，则 $f_{i+1,r_{j+1}} \leftarrow^+ f_{i,j} \cdot i$ (只需要把原先 j 的位置拔高为 r_{j+1} ，在 i 的位置填写 j 即可)。

K - Kaleidoscope

题解 2 (cont'd)

- 考虑转移，枚举 $f_{i,j}$,
- 若 $p_{i+1} \neq 0$ ，
 - 若 $p_{i+1} > j$ ，
 - 钦定 $i+1$ 是前缀最大值： $f_{i+1,p_{i+1}} \leftarrow^+ f_{i,j}$ ；
 - 钦定 $i+1$ 不是前缀最大值：若 j 未出现，则 $f_{i+1,r_{p_{i+1}+1}} \leftarrow^+ f_{i,j} \cdot i$ 。
 - 若 $p_{i+1} < j$ ，一定不是前缀最大值， $f_{i+1,j} \leftarrow^+ f_{i,j} \cdot i$ 。
- 最终的答案是 $f_{n,n}$ ，时间复杂度 $\mathcal{O}(n^2)$ 。

AC.

I - Infinity

题意

- 设 S_n 是所有 n 阶置换构成的集合。对于 $\sigma \in S_n$, 设 $\nu(\sigma)$ 是集合 $\{\mu^{-1}\sigma\mu \mid \mu \in S_n\}$ 中的元素数量。
- 固定 k , 多组询问 n , 请计算

$$\sum_{\sigma \in S_n} \nu(\sigma)^k$$

- 答案对 998244353 取模。
- $1 \leq t \leq 10^3$, $1 \leq k \leq 10^9$, $1 \leq n \leq 2 \cdot 10^5$ 。

I - Infinity

题解

- 首先在置换群 S_n 上定义等价关系：若存在 $\mu \in S_n$ 使得 $a\mu = \mu b$ ，则 $a \sim b$ 。一般而言，这个等价关系也被称为「共轭」。那么 $\nu(\sigma)$ 计算的就是 σ 所在的等价类的大小。
- 为解决原问题，我们有如下定理：
 - 若 $a, b \in S_n$ ，那么 $a \sim b$ 当且仅当 a 与 b 有着相同的轮换结构。
- 其中「相同的轮换结构」是指，对于任意正整数 l ， a 与 b 中长度为 l 的轮换数量相同。

I - Infinity

题解 (cont'd)

- 直接应用此定理，我们设出 σ 的轮换结构（长度为 i 的轮换有 c_i 个）就能得到：

$$\nu(\sigma) = n! \prod_{i=1}^n \frac{(i-1)!^{c_i}}{i!^{c_i} c_i!} = n! \prod_{i=1}^n \frac{1}{i^{c_i} c_i!}.$$

(相同长度的轮换间无序，所以要除以 $c_i!$)

AC.

I - Infinity

题解 (cont'd)

- 然后枚举 c_i 就有

$$\begin{aligned}\sum_{\sigma \in S_n} \nu(\sigma)^k &= \sum_{\sum_i i c_i = n} \left(n! \prod_{i=1}^n \frac{1}{i^{c_i} c_i!} \right)^{k+1} \\ &= (n!)^{k+1} [z^n] \prod_{i=1}^n \left(\sum_{j=0}^{\infty} \frac{z^{ij}}{(i^j j!)^{k+1}} \right).\end{aligned}$$

AC.

I - Infinity

题解 (cont'd)

- 这个幂级数的系数也是有经典方法计算的，设

$$F(z) = \sum_{j=0}^{\infty} \frac{z^j}{(j!)^{k+1}},$$

则答案为

$$(n!)^{k+1} [z^n] \prod_{i=1}^{\infty} F\left(\frac{z^i}{i^{k+1}}\right).$$

AC.

I - Infinity

题解 (cont'd)

- 设 $G(z) = \ln F(z)$, 就只需要求出

$$\sum_{i=1}^{\infty} G\left(\frac{z^i}{i^{k+1}}\right) = \sum_{i=1}^{\infty} \sum_{j=0}^{\infty} g_j \frac{z^{ij}}{i^{(k+1)j}}.$$

- 这是容易以 $\Theta(n \log n)$ 的时间复杂度算出 n 以内每一项系数的, 最后做一下多项式 \exp 即可。

I - Infinity

题解 (cont'd)

■ 定理证明:

- 我们知道一个置换可以分解为若干**不相交轮换**的乘积（即分别进行这些轮换，等价于进行原置换）: $\sigma = l_1 \times \cdots \times l_m$ 。
- 现在有了轮换分解，来看 $\mu^{-1}\sigma\mu$ 能得到什么：

$$\mu^{-1}\sigma\mu = (\mu^{-1}l_1\mu) \times \cdots \times (\mu^{-1}l_m\mu)$$

I - Infinity

题解 (cont'd)

■ 定理证明：

- 容易证明 $\mu^{-1}l_i\mu$ 仍然是一个轮换：设 l_i 轮换的长度为 c ，那么显然有且仅有 c 个元素经过 $\mu^{-1}l_i\mu$ 的映射后不为自身；且从中任选一个元素，经过 c 次映射后会遍历其它 $c - 1$ 个后回到自身。
- 根据这个证明过程，也可以得到推论： $\mu^{-1}l_i\mu (1 \leq i \leq m)$ 是互不相交的轮换。因为 l_i 之间互不相交，而不被 $\mu^{-1}l_i\mu$ 映射到自身的元素的集合，关于 l_i 是不同的。如此也就得到 $\mu^{-1}\sigma\mu$ 和 σ 有着相同的轮换结构。

C - Capella

题意

- 称仅包含 $a \sim z$ 的字符串 S 与 Capella 类似，当且仅当：
 - 在 S 中出现奇数次的字母有奇数个；
 - 在 S 中出现非零偶数次的字母有偶数个；
- 给定字符串 S ，有 q 次操作，第 i 次操作将第 p_i 个位置上的字符 S_{p_i} 修 改为给定字符 c_i 。
- 请在所有操作执行前和每次操作执行后，求出 S 所有连续子串中最长的 与 Capella 类似的子串的长度。
- $1 \leq n \leq 2 \cdot 10^5$, $1 \leq q \leq 2 \cdot 10^5$ 。

C - Capella

题解

- 首先， S 是类似 Capella 的字符串，等价于：
 - S 长度 $|S|$ 为奇数；
 - 且 S 的字符集大小 $|\Sigma(S)|$ 为奇数。
- 设 * 表示任意出现多次的字符，? 表示任意出现一次或多次的字符，abc 等表示出现一次的字符，ABC 等表示出现一次或多次的字符。

C - Capella

题解 (cont'd)

- 对于一个字符串 S , 若 $|\Sigma(S)|$ 为奇数:
 - 若 $|S|$ 为奇数, 则取 S 。
 - 若 $|S|$ 为偶数:
 - 若 S 首尾存在字符出现多次, 即 S 为 $*?..?*wx..yz$ 或 $ab..cd*?..?*$ 的形式, 可取对应的 $?..?*wx..yz$ 或 $ab..cd*?..?$, 字符集大小不变仍为奇数, 并且长度也为奇数。
 - 若不然, 设 S 为 $ab..cd*?..?*wx..yz$ 的形式。删去前缀的 $ab..cd$ 或是后缀的 $wx..yz$ 会同时改变 $|S|$ 和 $|\Sigma(S)|$ 的奇偶性, 因此答案只可能是 $ab..cd*?..?$ 或 $?..?*wx..yz$ 的子串。

C - Capella

题解 (cont'd)

- 若 $|\Sigma(S)|$ 为奇数:

- 若 $|S|$ 为偶数:

- 对于 $ab..cd*?..?$ 或 $?..?*wx..yz$, 若其长度为奇数, 则其字符集大小也为奇数, 可取对应的 $ab..cd*?..?$ 或 $?..?*wx..yz$ 。

- 若其长度为偶数, 则其字符集大小也为偶数, 可取 $b..cd*?..?$ 和 $?..?*wx..y$ 。

- 该部分时间复杂度为 $\mathcal{O}(|\Sigma|)$ 。

C - Capella

题解 (cont'd)

- 若 $|\Sigma(S)|$ 为偶数:

- $\Sigma(S)$ 中的某一字符将 S 划分为若干子串。例如设 S 为
 $?..?A??..?A??..??$, 则 A 将 S 划分为 $?..?$ 、 $??.?$ 、 $??.??$ 。
- 对于 $\Sigma(S)$ 中字符划分得到的所有子串, 设其中最长的子串为
 $?..?A[?..?]A?..?$, 则 $?..?$ 的字符集大小为奇数 $|\Sigma(S)| - 1$ 。
 - 若不然, 则存在 $?..?$ 不包含的 A 以外的字符, 其划分得到的子串将包含 $A?..?A$, 长度比 $?..?$ 更长, 与 $?..?$ 最长矛盾。
 - 最长子串可以使用线段树维护。

C - Capella

题解 (cont'd)

- 若 $|\Sigma(S)|$ 为偶数:
 - 若 $?..?$ 的长度为奇数, 则直接取 $?..?。$
 - 若 $?..?$ 的长度为偶数:
 - 若两侧之一的字符 $?$ 在 $?..?$ 中出现多次, 则 $?..$ 或 $..?$ 长度为奇数, 且字符集大小仍为奇数 $|\Sigma(S)| - 1$, 直接取 $?..$ 或 $..?。$
 - 否则两侧的字符在 $?..?$ 中仅出现一次, 不妨设为 $B..C$, 则 $AB..$ 和 $..CA$ 与 $B..C$ 长度相同且字符集大小相同, 因此可以左右移动寻找一侧字符出现多次的情形。

C - Capella

题解 (cont'd)

- 若 $|\Sigma(S)|$ 为偶数:
 - 若 ?..? 的长度为偶数:
 - 由于 $B..C$ 长度为偶数且字符集大小为奇数，其中必然存在某个字符出现多次，并且这个字符与 $B..C$ 两侧的距离至多为 $|\Sigma|$ ，也即移动至多 $|\Sigma|$ 次必然能找到这个字符。
 - 若移动到 S 最左最右两侧，仍未找到一侧字符出现多次的情形，则说明移动扫过的字符均是在 $B..C$ 的长度下出现单次的字符。又由于 $B..C$ 是划分子串中最长的，因此 S 形如 YZCABIJKL..WXYZCABIJ。

C - Capella

题解 (cont'd)

- 若 $|\Sigma(S)|$ 为偶数：
 - 若 ?...? 的长度为偶数：
 - 对于 S 的与 $B..C$ 等长的子串，删去左右两侧仅出现一次的字符会同时改变长度与字符集大小的奇偶性，因此必定删去 .. 中的部分字符。
 - 由于 $YZCABIJKL..W$ 和 $L..WXYZCABIJ$ 与 $BIJKL..WXYZC$ 字符集大小相等，均为奇数，并且删去 .. 中的字符时，两侧被删除的字符数量最小。将这两个子串作为 S ，以 $|\Sigma(S)|$ 为奇数的情形求解，并取最大值作为答案即可。
 - 该部分时间复杂度为 $\mathcal{O}(|\Sigma| + \log n)$ 。

AC.

C - Capella

题解 (cont'd)

- 将 S 中的相同字符串成一条链表，即可快速维护当前子串字符集大小。
- 总时间复杂度为 $\mathcal{O}((n + q)(|\Sigma| + \log n))$ 。

G - Gellyfish and Priority Queue

题意

- 有一个小根堆 Q , 初始为空。有 m 次操作。每次操作属于以下两种类型之一:
 - 插入: 给定 l, r 。从 $[l, r]$ 中均匀随机地生成一个整数 x , 将其插入到 Q 中。
 - 弹出: 删除 Q 中当前的最小元素。保证 Q 此时不是空的。
- 在所有操作完成后, 计算 Q 中所有剩余元素的乘积的期望值。答案对 998244353 取模。
- 多组数据, $1 \leq t \leq 200$, $1 \leq m \leq 500$, $1 \leq v \leq 500$, $\sum m \leq 2000$ 。

G - Gellyfish and Priority Queue

题解

- 为了避免两个元素相等时出现重复计数问题，我们钦定若两个元素相等，则更早加入小根堆的元素更小。**考虑直接在加入时钦定元素 x 是否会被保留在最终的优先队列中。**显然，最终被保留的元素任意时刻都构成排序后的某个后缀。基于此，设计 $dp: f_{i,k,l,r}$ 表示考虑到前 i 轮，小根堆里恰好前 k 小的元素会被弹出，被弹出的最大元素为 l （特别地，若 $k = 0$ ，则令 $l = 0$ ），被保留的最小元素为 r ，所有方案的未被弹出元素乘积的总和。

G - Gellyfish and Priority Queue

题解 (cont'd)

- 转移：对于操作 1 L R，枚举 $L \leq x \leq R$ ，则对于 $f_{i,k,l,r}$ ：若 $x < l$ ，转移至 $f_{i+1,k+1,l,r}$ ；若 $x \geq r$ ，乘上 x 转移至 $f_{i+1,k,l,r}$ ；否则枚举 x 是否被保留，若被保留，乘上 x 转移到 $f_{i+1,k,l,x}$ ，否则转移到 $f_{i+1,k+1,x,r}$ 。对于操作 2，将 $k = 0$ 的所有状态抛弃，其余所有状态的 k 减少 1 即可。直接实现的时间复杂度为 $\mathcal{O}(\sum m^2 v^3)$ ，使用前缀和优化即可做到 $\mathcal{O}(\sum m^2 v^2)$ 。

G - Gellyfish and Priority Queue

题解 (cont'd)

- 考虑继续优化，我们发现 $[l, r]$ 两维是很浪费的，我们考虑将其改写为一个维度。具体地，在每次 $k = 0$ 时，考虑直接钦定一个值 r' ，满足 r' 在过程中被加入小根堆内，且在下一次 $k = 0$ 时，最终值小于 r' 的或等于 r' 的某个严格前缀最终被删去。这可以用 $(r', 0/1)$ 维度描述。转移是类似的，通过前缀和优化可以做到 $\mathcal{O}(\sum m^2 v)$ 。

THANKS!

AC.NOWCODER.COM