Solution for NMHC Round

by **arrow_king**

A

p 是质数

直接前缀积 + 线性逆元即可，复杂度 $O(n)$。

满分

这道题的特殊之处在于区间长度 m 是个定值，因此考虑给原序列分块，块长为 m，这样所有长度为 m 的区间内至少会出现一次下标 $m, 2m, 3m, \ldots$。称这些点为关键点，则容易发现关键点到区间端点的距离都不超过 m。

因此我们预处理所有关键点向左向右 m 格的前缀积，查询时找到关键点把两段区间拼起来就行了。

由于只有 $\left\lfloor \dfrac{n}{m} \right\rfloor$ 个关键点，因此总复杂度为 $O(n)$。

B

回到路过的点

暂且还不能知道给你这个 pos 有啥用，除了判定你走到了一个你走过的点。因此我们从这里下手。

如果 $m = n$，你就直接走一圈回到原点，看看走了多少步就行了，而现在 m 比较小，但是大约是 $O(\sqrt{n})$ 量级，因此我们考虑根号跳。

类似 BSGS 算法，设 $B = \sqrt{n}$，我们先走 B 次，每次走一格，记录下来走过的格子编号（如果这 B 步里走回原点就直接回答）。接下来我们再走 B 步，这次每走一步跳 B 格。因为我们前面走了 B 格，所以这样跳一定能回到前面 B 格中的一个，此时根据它的位置求答案即可。

总花费是 $2\sqrt{n} = 2000$，可以通过。

优化：限制 n 取值区间

想继续优化，你发现只有两条路可走，一个是扩大 k 范围，一个是缩小 n 的范围。但是 k 这条路暂时想不到有什么方法，因此我们缩小 n 的范围。

如果我们知道 n 是一个属于 $[n_0, n_0 + D]$ 区间内的数，那么我们就可以通过最多 $2\sqrt{D} + 1$ 次操作来找到 n 的值：我们先走 $\sqrt{D}$ 步，再跳 $(n_0 - \sqrt{D})$ 格，最后再以 $\sqrt{D}$ 的步长向前跳。那么我们的任务是在比较少的步数内找到一个足够大的 n_0 使得 D 不会很大。怎么找呢？我们用随机化。

我们先随机跳 k 步，记录下路过格子的最大编号并把它当做 n_0。如果我们钦定 $D = \dfrac{1}{4}(1000 - k)^2$，我们就会面对一个失误率，其值是我们随机 k 次都选到 $[1, n - D - 1]$ 的范围里，也就是

$$\left(\frac{n - D - 1}{n}\right)^k$$

在 $k = 300$ 时上式的值约为 9.4×10^{-18}，因此可以通过。

C

判断有无解

如果有边权，我们可以令 f_u 为 u 到根节点路径的异或和，那么 u 到 v 的路径异或和就是 $f_u \oplus f_v$。

我们建立图 G，若有约束 (u_i, v_i, w_i) 就在 G 上建立一条从 u_i 连接到 v_i、边权为 w_i 的无向边。这样子会把 G 分成若干个联通块 $G_1, G_2, \ldots, G_k$。考虑一个联通块是否能有合法方案，其实就是判断其中的限制是否矛盾。我们可以分开每一位考虑，这样等价于将联通块黑白染色，若边权是 0 那么两个节点同色，边权是 1 两个节点反色。只需要一遍 dfs 就可以求出一组合法的解。

然而我们发现其实不用拆位，因为你直接钦定起始点的权值是 0 然后直接做异或也能达到同样的效果。所以我们不仅判断了联通块的有解性，还在有解时给出了一组特解。

优化特解

此时如果没有那个最小化 $a_1 \oplus a_2 \oplus \cdots \oplus a_{n-1}$ 的限制已经做完了，但是我们还得想一下这个限制如何满足。考虑到我们给一个联通块内整体异或上 v 不会改变该联通块的合法性，因此若记当前找到的特解异或总和是 W，我们只需要找到一个能对答案产生影响的联通块，并把所有点权都异或上 W，整体的异或和就是 0。

考虑在原树上边权如何从我们定义的点权上还原：连接 (u, v) 的边权是 $f_u \oplus f_v$。因此 f_u 在最后的异或式子中会被计算 deg_u 次，当且仅当 deg_u 是奇数时才会有影响，我们称这样的点是好的。由于你无法对 G 中一个联通块的某个点单独异或，而只能对整个联通块一起异或，你就不得不考虑联通块中好点的数量：当且仅当一个联通块内好点的数量是奇数，这个联通块才会有影响。因此我们只需要找到这样一个联通块并记下它，求出特解的答案 W 后对那个联通块的每一个 f 都异或上 W 就可以使得最后的答案为 0。

时间复杂度 $O(n)$。假设你写了拆位就是 $O(n \log V)$，也许能卡过去。

D

二分答案

我们发现如果直接做，无论如何也绕不开计算 $(n-1)^2$ 次中位数的问题，因此我们考虑转化视角，判断一个答案是否大了。

接下来我们用经典的套路，因为我们只关心答案和 mid 的大小关系，所以我们让最底层满足 $a_i \geq mid$ 的点是黑的，$a_i < mid$ 的点是白的。我们希望这样能够快速维护第一层的点是黑还是白。

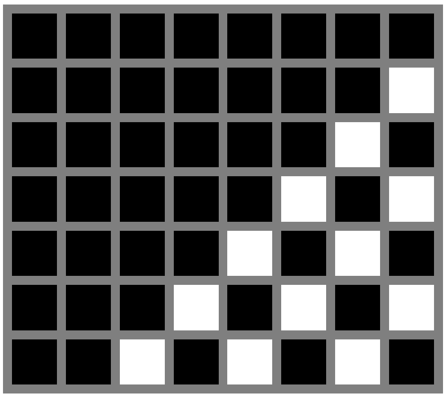
同色格具有延伸性

我们首先发现：求中位数的问题被转化成了比数量问题：三个格子，如果黑的点比白的多那么中位数是黑色；反之是白色。这个性质不难被证明。

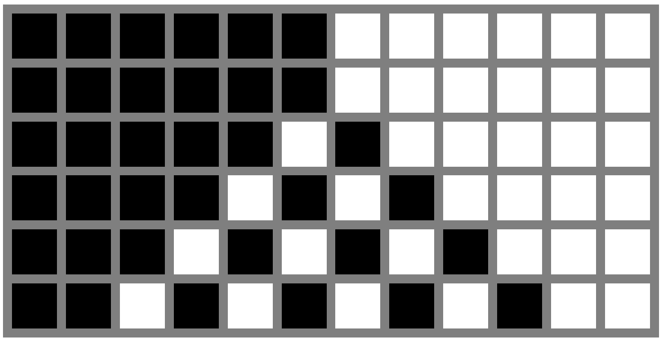
假设底层有两个相邻的黑点 p_1, p_2，根据上面的性质，我们不难发现：p_1 的上面还是黑色，p_2 的上面也还是黑色；如果是相邻的白色性质是相似的。也就是说相邻的同色方块可以向上延伸。

那么黑白交替的情况又如何呢？不难发现，黑色格子上面会变成白色的，因为它两边都是白色；白色格子亦然。

现在我们发现一个惊人的事实：如果两个相邻的黑点 p_1, p_2 右边是黑白交替的，那么它会向右延伸，直到到达区域的结尾并碰上另一个同色组。（figure 4-1）



而两个相邻的同色组的相遇会更加有趣：由于它们是相邻的，因此中间一定是黑白交替的，所以它们会各自延伸并最终碰在一起，然后不再扩张。从下图中可以看出，在到达一定高度后**每个点的颜色仅取决于离它最近的同色组的颜色。**（figure 4-2）



而我们的金字塔一定满足这个“一定高度”的要求，因为金字塔斜率为 1，正好够从一段延伸的颜色块。

因此我们可以统计底端的颜色，如果相邻的颜色相同就统计一下，如果它是离中心点 n 最近的颜色块，我们就确定了 n 的颜色。

一些小问题

第一个问题：会不会出现距离相等的同色块？考虑到同色块内部一定是黑白交替的，因此距离必为偶数。

第二个是没有相邻颜色块的情况，而这个很简单，画一下图就可以发现第一层的颜色取决于底层第一个点的颜色。

第三个是关于二分的。我们的二分实际上找到的是满足条件的最小值，但是这个数**有可能没有出现在我们初始的数集内**，因此要先离散化后再二分。

后话

这个题有 $O(n)$ 的做法，大家可以去题解区看一下。