以下为本场的难度分布，其中加粗的题目是 yummy 自己很喜欢的。

Easy：B, C, D, E

Medium：**F**, G, H

Hard：**A**, **I**, J

Easy

该部分难度低于正式赛题目，主要考察大家的算法基本功和手速。

B. 学历史导致的（字符串模拟）

什么？你不会真的读入一个字符串后，先分离出天干、地支，然后再用 exCRT 算出年份吧？

注意到年份只有 60 个，事先算出每个年份对应的干支纪年法，存到 `map` 里面，使用时直接查找即可。

C. 学数数导致的（枚举计数）

本题难点在于如何不重复计数。

核心思路是枚举第三个数字 p，如果这个 p 是好的（合法且前面没出现过），那么就把这个 p 之后的不同 q 个数加进去。

为了达到这个目的，需要预处理出每个数字 i 首次出现的下标 fir_i 和每个下标 i 之后的不同正整数 q 的个数 uni_i。接下来，用一重循环从 1 扫到 n， 记录截至目前最后一次出现 0 的下标 $last$，然后：

- 若 $fir(a_i) < last$，则 a_i 是一个合法的 p，统计答案。
- 统计好答案后，把 $fir(a_i)$ 设置成 $+\infty$，这样后面碰到相同的 p 是就自动不合法了。

时间复杂度 $O(n + \max a_i)$。实现精细一点，所有测试点总和可以做到 $O(\sum n + \max a_i)$，但是本题没卡多测清空。

D. 学 DP 导致的（简单 DP、分类讨论）

如果 $k \geq 26$，那么答案就是 S 内不同字符个数。

如果 $k < 26$，那么暴力算出 S'，然后计算 S' 的最长上升子序列即可。本题数据范围较小，所以 $O((nk)^2)$ 是可以通过的。

E. 学几何导致的（简单推式子）

如果 k 是奇数则答案为 0。

先讨论 $k = 2$ 的情形，此时有 $\lceil n/2 \rceil$ 条平行 l_0 的直线和 $\lfloor n/2 \rfloor$ 条垂直 l_0 的直线，答案就是二者的积。

对于 $k > 2$ 的情形，可以把所有 l_i 按照 $i \bmod (k/2)$ 分组，此时有 $n \bmod (k/2)$ 组直线有 $\lfloor n/k \rfloor + 1$ 条，另外 $(k/2) - (n \bmod (k/2))$ 组直线有 $\lfloor n/k \rfloor$ 条。显然垂直的直线都在同一个组内，并且每一个组自身都是一个 $k = 2$ 的情形，套用上面的结论即可。

Medium

该部分难度大约是正式赛基础题。

F. 学博弈论导致的（博弈论，猜结论）

假设红宝石 1 元钱，蓝宝石 2 元钱，宝盒 4 元钱，那么问题完全等价于：

> 桌上有 $r + 2b + 4m$ 元钱，Alice, Bob 轮流取钱，每次可以取 $1 \sim 3$ 元，问最后谁胜。

这是经典结论了，若 $(r + 2b + 4m) \bmod 4$ 是 0 则 Alice 胜，否则 Bob 胜。

G. 学计算导致的（高维 DP）

考虑如何计算表达式。把用乘法连接的若干个数称为一项。

从前往后扫描表达式，用 sum, mul, cur 分别表示已经结束的所有项之和、当前项的乘数，以及当前要计算的数字。例如，计算 `31-16+8*243` 时，若当前读取到数字 4，那么此时 $sum = 15, mul = 8, cur = 24$。

接下来对于读取到的每个字符：

- 若为数字 d，只要 $cur \leftarrow (10cur + d)$。
- 若为乘号，则 $mul \leftarrow mul \cdot cur$，$cur \leftarrow 0$。
- 若为加号，则 $sum \leftarrow (sum + mul \cdot cur), mul \leftarrow 1, cur \leftarrow 0$。
- 若为减号，则 $sum \leftarrow (sum + mul \cdot cur), mul \leftarrow -1, cur \leftarrow 0$。

上述运算均在 $\bmod k$ 意义下进行。

然后令 $f(x,y,sum,mul,cur)$ 为"走到 x,y 这个格子时，有 sum,mul,cur 的方案数"，就已经是一个时间复杂度为 $O(nmk^3)$ 的算法，能通过本题了。然而这写起来太丑怎么办？

我们发现把 (sum,mul,cur) 当成状态，并赋予一个 $0\sim k^3-1$ 的编号，那么上述表达式计算流程构成自动机，并且自动机只和 k 有关。

在 DP 前先把这个自动机造好，然后令 $f(x,y,state)$ 为"走到 x,y 这个格子时，状态为 $state$ 的方案数"，转移就不用分别写了。std 码量为 1.1 KB。

H. 学画画导致的（图论）

第 i 行第 j 列的格子被 $\lceil j/2\rceil,2n-i+1,2n+i-\lfloor j/2\rfloor$ 三个刷子经过。如果有格子颜色不是三者之一，那么直接输出 `NO`。

把每个刷子看作一个结点建图。对于每个已知颜色的格子，把所有"颜色不等于刷子颜色"的刷子向"颜色等于刷子颜色"的刷子连边。如果最终得到一个 DAG，那么答案是 `Yes`，否则是 `NO`。

Hard

该部分难度大约是正式赛的铜牌题~银牌题。

本来 A 才是压轴题，但是由于 HDOJ 出不了 Special Judge 题，所以那道题废了，换了个简单题上去。

A. 学位运算导致的（Ad-hoc）

称"a 包含于 b"当且仅当 $a\&b=a$。

令 a_i 为"让第 i 个二进制位是 1 的最小优雅数字"，具体地，初始时 a_i 全是 `-1ull`，然后每次加入一个数 x，就把 x 中所有 1 对应的 a_i 都按位与一下 x。

如果要计算"包含 x 的最小优雅数"，那就是"包含 x 中每个二进制位的最小优雅数"，把 x 中是 1 的二进制位对应的 a_i 全部或起来即可。

接下来处理询问。首先计算包含 x 的最小优雅数。接下来枚举 y,x 的二进制位中，最早出现差异的位置（从而知道要包含哪些二进制位），使用上述做法求值即可。时间复杂度 $O((n+q)w)$，其中 $w=64$。

I. 学高考第 19 题导致的（单调栈、树、时间复杂度分析）

不失一般性，令 $a_0=+\infty$。首先我们弄明白什么样的 p 是合法的。

我们规定 $pa(i)$ 为最大的 $j < i$ 使得 $j \prec_a i$。不难发现这其实就是单调栈的写法，然后全体 $(pa(i), i)$ 构成一棵树，我称之为**单调树**。在这个定义下，$j \prec_a i$ 当且仅当在单调树上 j 是 i 的祖先。

因此，p 合法当且仅当变换后，树的形态没有发生改变，只能修改子树之间的顺序。并且，两个子树能交换当且仅当它们的根相同（否则较小的子树会向较大子树连边而非原来的根）。题目要最大化树的前序遍历。

使用链表来维护所有子树的前序遍历。由于大树排序时需要用到小树的排序结果，所以我们从 n 到 1 逐个确定结点的孩子顺序。

对于两棵子树 x, y，首先如果 $a_x \neq a_y$，那么小的子树排前面；否则按照比较 x 子树的前序遍历和 y 子树的前序遍历，字典序大的排前面；特别地，我们认为链表的结尾是 $+\infty$，也就是"若 x 是 y 的前缀，则 x 排前面"。

这样，$cmp(x, y)$ 的复杂度就是 $O(\min(size_x, size_y))$ 了，直接排序[1] 即可。类似启发式合并，我们可以证明这个过程复杂度是 $O(n \log n)$。

[1]：如果是堆排序，那么复杂度会好证明一点，但是我还不会严格证明 `std::sort` 的对应复杂度（而只会感性理解），虽然直接用是可以过的。如果有选手可以完成复杂度证明，欢迎和出题人交流。

J. 学排列导致的（线段树）

本来这里是送分题的，但是刘老师说那个题太简单了，要求换题。我对照了一下第一场的难度和风格，发现第一场的代码量很大，因此临时加了一个直白的数据结构题负责一下码量。

本题有大量置换，因此可以事先写一个结构体 `perm`，重载 `u*v` 为 $u \circ v$；然后实现 $\circ$ 的逆函数 $\cdot^{-1}$，满足 $u \circ u^{-1} = (1, \ldots, k)$。

注意到 $\circ$ 有结合律，因此直接用线段树维护，每个结点 sgt_{rt}（假定对应 $[l, r]$）维护 $p_l \circ \ldots \circ p_r$，同时维护一个 lazy-tag 表示最近一次整体修改。

考虑如何计算整体修改之后的区间 $\circ$ 结果。注意到本质不同的修改只有 k 种，所以我们直接用 $prod_{z,i}$ 表示将 $p_1, \ldots, p_i$ 都修改成 z 开头后的前缀 $\circ$，类似前缀和。注意此时计算 $p_l \circ \ldots \circ p_r$ 要用 $(prod_{z,l-1})^{-1} \circ prod_{z,r}$，**顺序不可以反**。

查询时，如果当前区间有 lazy-tag 那么直接用 $prod$ 数组算出答案即可，否则分治查询。

预处理 $O(nk^2)$，单次查询 $O(k \log n)$，空间复杂度 $O(nk^2)$。std 用时 843 ms，空间 208 MiB，所以本题时空限制尽管比较紧，但确实是 std 两倍以上，符合出题规范。

由于技术限制（杭电空间限制无法超过 512 MiB），本题需要节省空间。一些实现上的细节：

- 排列中的数因为不超过 30，可以使用 `char` 数组存储。
- 一位验题人口胡的做法是在线段树结点处存储区间内所有 lazy-tag 对应的总和，时空复杂度相同，但是常数是两倍，理论也可通过。如果你使用本做法被卡空间，可以尝试换成前缀和。