A - 签到题

做法 1

$k \geq 3$ 时做法是容易的：令 $p = n^{\frac{1}{k}}$ ，枚举 $1 \leq a \leq p$ 并判断 a 是否合法即可。复杂度可以被看做 $O(n^{\frac{1}{3}})$ 。$k = 1$ 时 $a = n$。

$k = 2$ 时无法直接枚举，怎么办呢。我们令 n 质因数分解的结果是 $\prod p_i^{c_i}$ ，那 a 就等于 $\prod p_i^{\lfloor \frac{c_i}{2} \rfloor}$ 。我们先找出所有 p_i 满足 $p_i \leq n^{\frac{1}{3}}$ ，这可以通过枚举实现。把它们从 n 除掉以后，看剩下的 n 会是什么形式。此时只有四种可能：$1, p, pq, p^2$ 。可以发现只有 p^2 会让 a 的值产生变化，于是我们只需要判断 n 是否是完全平方数即可，如果是的话就将 a 乘上 $\sqrt{n}$ 。总复杂度 $O(n^{\frac{1}{3}})$ 。

做法 2

利用 Pollard-rho 进行质因数分解即可。复杂度 $O(n^{\frac{1}{4}})$ 。

B - 结论题

我们先将操作进行转化：同时交换颜色和点权，并将颜色进行翻转。也就是说此时"颜色"是跟随着"点权"的。同时可以发现操作是可逆的，于是我们只需找到看两个状态的某些"特性"是否相同即可。

对图的每个联通块分别考虑。分两种情况讨论：

1. 二分图。把图分成左右两部之后，可以发现若一个"点权" 仍然在初始的部分，则颜色不变；否则会发生改变。于是对于一个状态，我们将"颜色" 重定义成 "在左部时的颜色" ，可以发现此时每次操作等同于将 $(点权, 颜色)$ 的二元组进行交换。由于图是联通的，所以二元组可以任意排列，我们只需判断初始状态与目标状态构成的二元组集合是否相同即可。

2. 不是二分图。此时我们要求：初始点权与目标点权的集合是相同的，且初始蓝色点的个数与目标蓝色点的个数奇偶性相同。不难发现这是必要的，下面我们证明它是充分的：首先这个图一定存在一个子图是联通二分图（其生成树）。我们尝试构造以下操作：对于奇环的相邻两个点 u, v，在点权不变的情况下将这两点的点权进行翻转。

   如果能做到这一点那证明就完成了：考虑 $(点权, 在左部时的颜色)$ 构成的二元组，我们可以把任意两个二元组通过交换换到 u, v 并进行操作，由于"颜色"错误的二元组恰有偶数个，所以两两将其修正即可。

   构造如下：令奇环上的点依次为 $1, 2, 3, \ldots, n$ ，我们依次操作 $(1, 2), (2, 3), \ldots, (n-1, n), (n, 1)$ ，再操作 $(n-1, n), (n-2, n-1), \ldots, (2, 3)$ 即可。可以发现此时点权不变；对于一开始在 $3, 4, \ldots, n$ 的点权，它们参与了偶数次交换，而一开始在 $1, 2$ 的点权参与了奇数次，就达成了点权不变，$1, 2$ 颜色翻转的效果。

C - 简单题

观察组合意义：对 $n+1$ 个球染色，使得中间有一个绿球，左边有 a 个红球，右边有 b 个白球，且左边每个白球都会带来 x 的系数，右边每个白球带来 y 的系数。

设 $F(n,a,b)$ 是此时的答案，那如果 $a>0$ ，枚举最左边的球颜色是红/白即可得到 $F(n,a,b)=F(n-1,a-1,b)+xF(n-1,a,b)$ ；否则枚举是绿/白即可得到

$F(n,a,b)=\binom{n}{b}y^{n-b}+xF(n-1,a,b)$ 。

同理尝试枚举最右边的球颜色，我们也有 $b>0$ 时 $F(n,a,b)=F(n-1,a,b-1)+yF(n-1,a,b)$,否则为 $\binom{n}{a}x^{n-a}+yF(n-1,a,b)$ 。

对比两式，发现直接相减，我们即可由 $F(n-1,a,b-1),F(n-1,a-1,b)$ 推到 $F(n-1,a,b)$ 了，这个题就做完了。

另一种做法是注意到 $F(n,a,b)=[z^{n-a-b}](\frac{1}{1-xz})^{a+1}(\frac{1}{1-yz})^{b+1}$ 。将其乘上 $(1-xz)+xz$ 或 $(1-yz)+yz$ ，即可得到和上面类似的递推式。

D - 套路题

对 L_i,R_i 进行离散化，这样数轴被划分成 $O(n)$ 段，每一段的花盆只会同时种上花。

对每个二元组维护权值为其内部的空花盆个数。

每次找到权值最小的二元组 p ，利用并查集找到 $[L_p,R_p)$ 内有哪些段还没有种花，种上花的同时对包含这一段的二元组的权值进行更新。

可以发现我们难以即时维护所有二元组的权值。

但由于每次只需要找权值最小的二元组，考虑对于两个二元组 i,j ，如果 $L_i\le L_j\le R_j\le R_i$ ，且若 $L_i=L_j,R_i=R_j$ 则 $j<i$ 时，j 一定比 i 优。 换个角度，我们对所有二元组以 R_i 为第一关键字，$-L_i$ 为第二关键字，i 为第三关键字排序，那若一个二元组权值最小，则它前面不存在 L 比它大的。我们称这样的二元组是合法的。

后面为了方便描述，我们按照排序结果对二元组进行重编号，但注意找 p 时还是得以原编号为第二关键字。

现在只即时维护合法二元组的权值，可以发现它们 L,R 都是单调递增的，用一棵线段树维护这些合法二元组，一次修改影响的范围就是一段区间，我们支持单点修，区间加，全局 min 即可找到每次操作的 p 。（不合法的二元组权值当成 $+\infty$ ，合法之后进行单点修改）。还需要对每个节点维护 L 最大值（不合法的二元组 L 当成 $-\infty$），以通过线段树二分找到每次要修改的区间。

另一件事情是每次操作完 p 之后要找到新的合法二元组。先在前面的线段树中删去 p 的影响，并查询 $\leq p$ 的部分中 L 最大值 t 。令 $> p$ 的最小合法二元组是 q (可以用 set 找到)， $x = p$ ，我们每次找到 $[x, n]$ 中第一个满足 $L > t$ 的不合法二元组 y ，如果 $y \geq q$ 就结束过程，否则 y 成为新的合法二元组，令 $x := y + 1$ 继续找下去。

用另一棵线段树维护当前不合法的二元组，每个节点维护 L 最大值，每次线段树二分即可实现上述过程。找到新的合法二元组时利用 BIT 计算它现在的权值，再对第一棵线段树进行修改。

复杂度 $O(n \log n)$ ，常数较大。