ti

原题

首先注意到我们可以在操作的任意时刻进行任意次操作 3，因此我们可以先把序列视作一个首尾相接的环，每次操作 2 就是随意旋转之后确定当前序列，再进行操作。

考虑观察一下操作 2 实际上在干什么。发现其实就是在环上固定一个位置不要动，其他位置视为一个环，随意旋转。进一步的，因为我们操作的是一个环，因此我们可以将其视为将一个位置插入进任意其它位置。

然后这道题就差不多结束了。最后可行方案就是枚举这个环一个起点，在以这个起点开头的序列中找到最长上升子序列，这些位置不动，其它位置每个位置进行一次操作 2 和若干次操作 3 放到对的位置。结果就是 n 减去最长上升子序列的长度。最终答案就是每个起点的结果取最小值。

需要枚举起点以及 $O(n\log n)$ 计算最长上升子序列，总复杂度 $O(n^2\log n)$。

tii

原题，原题题解（需要下载）（英文）

令最后选出来的区间的区间为 $[l+1,r]$，1 的占比为 S。同时令 s_i 表示在为原序列做一遍前缀和之后的结果。我们先钦定 $S\le p$，$>$ 的情况可以零一反转再做一遍。

那么观察到，我们相当于要让下面这个式子在大于等于 0 的情况下最小：

$$\begin{aligned}p-S&=p-\frac{s_r-s_l}{r-l}\\&=\frac{pr-pl-(s_r-s_l)}{r-l}\\&=\frac{(pr-s_r)-(p_l-s_l)}{r-l}\end{aligned}$$

注意到最后这个形式比较好看。令 $b_i=(i,pi-s_i)$，即 b_i 为一个点。那么发现这个式子其实就是两个点之间斜率的形式。因此我们的问题变成了：给出 n 个横坐标互不相同的点，问任意两个点之间的非负最小斜率，并输出取到这一斜率的解中左边的点的横坐标最小的那组解的左边的点的横坐标。

然后再继续观察到，这样的两个点的 y 坐标一定相邻。具体的，假设最优的两个点分别是 x,z，它们 y 坐标之间夹着任意一点 t。那么简单画一下图就可以发现，点对 $(x,t),(t,z)$ 之间一定至少有一对不劣于 (x,z)。因此得证。

然后就做完了。复杂度 $O(n\log n)$，瓶颈在排序。

tiii

原题，原题题解（需要下载）（英文）

感觉这个数据范围很 DP，考虑往这方面想。

尝试刻画某一个时刻的状态。先只考虑正方向没被捉住的鸡的集合 S。我们发现：假如钦定捉鸡人现在在原点，那么 S 中速度最大的那只鸡 x 以及当前时刻 now 就足以刻画状态。尝试对剩下的鸡分类讨论：

- 速度比 x 大：已经被捉住了。
- 速度比 x 小且在当前时刻位置比 x 小：我们可以不管这种鸡，因为如果 x 被捉住，那么这只鸡也一定会被捉住。
- 速度比 x 小且在当前时刻位置比 x 大：一定没有被捉住。

因此设 $f_{i,j}$ 表示捉鸡人在原点，负方向速度最大的没被捉住的鸡为 i，正方向速度最大的没被捉住的鸡为 j，达成这个状态所需要的最小时间。转移可以简单 $O(n)$ 转移。

然后就做完了，复杂度 $O(n^3)$。

tiv

原题，原题题解

令一个询问 (u,v,d) 的答案为 $f(u,v,d)$。令 u 和 v 的 lca 为 c，那么通过一些差分，我们有：

$$f(u,v,d)=f(1,u,d)+f(1,v,d)-2\times f(1,c,d)+f(c,c,d)$$

具体证明可以手玩一下。然后对于 $f(u,u,d)$，离线点分治或者点分树即可 $O(n\log n)$ 解决。那么现在我们只需要查询 $f(1,u,d)$ 即可。

考虑轻重链剖分。那么有经典结论：每个点的轻儿子子树大小之和是 $O(n\log n)$ 的。同时对于 $1\to u$ 的链，其只会包含 $O(\log n)$ 条轻边。令这些轻边分别为 $(v_i,u_i),i\le k$（v_i 深度更浅）。

定义 $h(u,d)$ 表示 u **子树**中距离 u 小于等于 d 的点数，$g(u,d)$ 表示 u **轻子树**中距离 u 小于等于 d 的点数。令 $1\to u$ 的点集为 S。

那么再通过一些差分，我们有：

$$\begin{aligned}f(1,u,d)=&+|S|\\&+\sum_{t\in S}g(t,d)\\&-\sum_{i\le k}h(u_i,d-1)\\&+\sum_{i\le k}h(\mathrm{hson}(v_i),d-1)\\&+h(\mathrm{hson}(u),d-1)\end{aligned}$$

第一行表示这条链自身的贡献，相当于深度加一。

第二行这条链的点的轻儿子的贡献。

第三行表示第一行多算了轻边连接部分中 u_i 子树的贡献，需要减去。

第四行表示少算了轻边连接部分中 v_i 的重儿子的贡献，需要加上。

第五行是这条链最底下的重链的贡献，需要特殊计算。

因此我们把一个 $f(1,u,d)$ 的查询拆成了 $O(\log n)$ 个 h 和 g 的查询。考虑如何求出 $g(u,d)$。跑一遍 dfs，每到一个点就把其轻子树的所有点以深度为下标放到一个数据结构上，那么在数据结构上进行区间查询即可 $O(\log n)$ 获得一个 $g(u,d)$。具体

至于 $h(u,d)$ 则可以通过启发式搜索或者启发式合并或者线段树合并以类似的方式解决。实际上和 g 的求法类似，都可以用启发式搜索套树状数组解决。

因此最后我们即可在 $O(n\log^2 n)$ 的时间复杂度内解决本题。空间复杂度则视实现为 $O(n\log n)$ 左右。如你所见，需要若干数据结构，有一定的代码复杂度，同时可能需要注意一下常数。