问题 1

(a)

$$f_i = \max_{j<i, a_j \le 2a_i} f_j + 1$$

(b)

使用平衡树，以 a_i 为键值，每个节点维护子树中 f_i 的最大值 mx，故每次只需查询 $O(\log n)$ 个节点，更新也只会更新 $O(\log n)$ 个祖先节点的 mx。

(c)

维护一个数组 $tails$，其中 $tails[j]$ 表示当前长度为 $j+1$ 的子序列的最小末尾元素，容易发现 $tails[j]$ 数组单调增。每次更新时，a_i 可以接在所有 $f_j \le 2a_i$ 的长度为 j 的子序列之后，并更新对应的 $f_{j+1} = \min(f_{j+1}, a_i)$。即，每次更新时会将从开头开始的一个区间对 a_i 取 $\min$，同时可以发现该数组仍单调增。使用线段树即可维护该操作。

问题 2

(a)

$$\begin{aligned}
\sum_{i=1}^{n} ||x_i - y||_2^2 &= \sum_{i=1}^{n}\sum_{j=1}^{d}(x_{i,j} - y_j)^2 \\
&= \sum_{j=1}^{d}\left(\sum_{i=1}^{n}(y_j^2 - 2x_{i,j}y_j + x_{i,j}^2)\right) \\
&= \sum_{j=1}^{d}\left(ny_j^2 - 2\sum_{i=1}^{n}x_{i,j}y_j + \sum_{i=1}^{n}x_{i,j}^2\right)
\end{aligned}$$

括号中是一个二次函数，易得最小值在 $y_j = \frac{\sum_{i=1}^{n} x_{i,j}}{n}$ 处取到。

(b)

$$\begin{aligned}
\sum_{i=1}^{n} ||x_i - y||_1 &= \sum_{i=1}^{n}\sum_{j=1}^{d}|x_{i,j} - y_j| \\
&= \sum_{j=1}^{d}\sum_{i=1}^{n}|x_{i,j} - y_j|
\end{aligned}$$

故对于距离之和，各维独立，可以分别考虑每一维坐标，问题转化为一维情形。

对于一维情况，可视为分段函数，且每段内部可导。每一段都是线性函数，所以最小值在某一点处取到。当 y 在某一段中时，若左边的点的个数大于右边，则 y 往左边移动函数会减小，反之同理。故只需找到所有点的中位数即可。可以使用类似快排的算法达到期望 $O(\log n)$，也可使用 the median-of-medians 方法达到严格 $O(\log n)$。

问题 3

令 f_u 表示 u 在独立集中时，子树中的最大独立集大小，g_u 表示 u 不在独立集中时子树的最大独立集大小

$$f_u = 1 + \sum_{v \in u.child} g_v$$

$$g_u = \sum_{v \in u.child} f_v$$

之后根据每个点 u 中 f_u 和 g_u 的大小递归构造方案，时间复杂度 $O(n)$。

问题 4

根据题意给出的操作方法，抽象来讲将一个数插入到它右边的一个数前，由此可以发现，我们不可能将一个数插入到数组最后（因为这之后没有其他数了），所以如果 $p_n \neq n$，直接输出 -1。

假设已知 $p_n = n$，观察 $n-1$ 这个数如何归位，存在两种情况：

1. **情况1**：如果 $p_{n-1} = n-1$，无需操作即可归位，对答案没有贡献。
2. **情况2**：如果 $p_{n-1} \neq n-1$，必须通过一次操作将其插入到 n 前面，对答案贡献为 1。

接下来考虑 $n-2$ 这个数。由于 $n-1 \sim n$ 已在前面操作中归位，同理分为上述两种情况。
因此，可以按照 $n-1 \sim 1$ **的顺序依次强制归位**，答案即为最小值。

证明

- 一次操作中，除操作数向右移动外，其他数不动或向左移动。
- 若一个数无需操作即可归位，当且仅当比它大的数均在其右侧。否则，后续操作中它会被向左挤，必须操作一次以跨过右侧更小的数。因此它必然对答案有贡献（至少操作一次）。
- 按上述策略，所有需操作的数（满足情况2）仅被操作一次，故该策略为最优解。