subtask1

模拟，时间复杂度$O(n^2)$。

subtask2

线段树维护，信息是每种数字是否在当前区间里出现过，标记要记录对于每个i，区间里所有数字i变成了数字p_i。总时间复杂度$O(nm\log n)$，其中$m=\max a_i$

subtask3

对时间分块，把数字分为会被修改的$O(B)$个特殊数字，和剩余剩余普通数字。对于普通数字可以转化为一个静态问题。静态情况下是一个经典问题，可以转化为二位数点。区别是时间分块之后修改数量为$O(n)$个，查询为$O(B)$个，因此数据结构不要树状数组而是分块做根号平衡。对于特殊的数字，我们维护$a_{i,j}$表示前i个块里有多少个数字j。修改的时候只需要重新计算$a_{x,*}$和$a_{y,*}$即可。当一轮重构结束时，我们需要求出操作后的a序列，因此还要对每个块维护$p_{i,j}$表示第i个块里数字j变成了数字什么。令$B=\sqrt{n}$，时间复杂度$O(n\sqrt{n})$。

subtask4

对于每个数字，我们开一个平衡树维护其位置。每当修改的时候，我们取出来两个平衡树对应区间，并尝试合并他们。但问题在于这两个平衡树的值域有交。于是考虑一下方法来合并两个平衡树

1. 令x为T_1中最小的元素，y为T_2中最小的元素，不妨设$x<y$。
2. 把T_1中小于x的元素分裂出来记为T。
3. 把T_3变成T_3合并T，并重复第一步。

通过类似于归并的方式，我们把T_1和T_2合并到T_3了，不难这个过程的时间复杂度为$O(n\log n)$，其中m为合并后的序列里，来源相同的连续段数量。

令势能函数$\Phi(T)=\sum\ln(x_i-x_{i-1})$，其中$x_i$为平衡树$T$里的元素。考虑经过上述合并后，势能会有什么样的变化。我们直接考虑势能变化最小的情况，即合并后的来源为$12121\cdots$交替状物，设合并后的相邻间距为d_i，则势能变化量为：

$$\sum_{i=2}^{m}\ln(d_i+d_{i-1})-\sum_{i=1}^{m}\ln(d_i)$$

由于log的凸性，$\log(\frac{a+b}{2})\geq\frac{1}{2}(\log a+\log b)$，因此：

$$\begin{aligned}\Delta\Phi(T)&\geq\sum_{i=2}^{m}\ln 2+\ln(\frac{d_i+d_{i-1}}{2})-\sum_{i=1}^{m}\ln(d_i)\\&=(\ln 2)m-\frac{1}{2}(\ln d_1+\ln d_m)\\&=\Theta(m)-O(\log n)\end{aligned}$$

初始势能最多为$O(n\log n)$，每次操作最多增加$O(\log n)$，因此势能总量为$O(n\log n)$。我们用$O(m\log n)$的时间使得势能减少了$\Theta(m)$，因此平衡树的总时间复杂度为$O(n\log^2 n)$。

为了计算答案，我们要维护所有的$(i, next_i)$二元组，$next_i$表示下一个与a_i相同的位置。注意到每次操作，只有$O(m)$个二元组会发生变化，因为连续的一段内部不会有变化。根据前面的势能分析，我们知道了所有的$(i, next_i)$二元组变化量为$O(n\log n)$的。因此我们可以提取出$O(n\log n)$个插入删除点的修改操作，和$O(n)$个查询二维数点的查询操作。直接离线用cdq分治实现是$O(n\log^3 n)$的，但实际上由于修改和查询数量级不对等，我们可以改用$\log n$叉树状数组，在$O(\frac{\log n}{\log\log n})$时间内修改和$O(\frac{\log^2 n}{\log\log n})$时间里查询，最终得到时间复杂度为$O(n\frac{\log^3 n}{\log\log n})$。