T1

30 分

显然每个位置最多操作一次，于是暴力枚举即可，时间复杂度 $\mathcal{O}(2^n)$。

60 分

无脑 dp 即可，显然在最优解（假设最初所有数的 $\gcd \neq 1$ ）中至少有一个位置是操作了的，于是枚举该位置然后在值域上 dp 即可。

正解

多想一想可以发现答案一定不超过 3。

因为 $\gcd(n-1,n)=1$，所以最坏情况就是把位置 $n-1$ 和 n 都操作一遍。

那么就好做了，先检查能不能不操作，再检查能否只操作位置 $n-1$ 或位置 n，都不行就输出 3。

T2

30 分

各种暴力搜索都行。

链

发现每次操作相当于是在染一个后缀。

于是考虑从左往右染，如果 $c_i=c_{i-1}$ 说明 i 不需要再染，否则就需要重新染。

菊花

注意到先染节点 1 一定最优，然后一次检查其他点与节点 1 颜色是否相同，不同就重新染。

正解

根据链和菊花的解法受到启发，发现对于树的情况从上往下染即可，即 $c_u=c_{p_u}$ 说明不需要再染，否则就重新染。

T3

20 分

暴力枚举三元组即可。

40 分

注意到如果确定 i,j，那么 b 就确定了，就可以确定 a_k 的值，于是只需要枚举 i,j 即可。

时间复杂度 $\mathcal{O}(n^2)$。

60 分

注意到 n 其实并不重要，相同的数是可以放到一起的。

80 分

考虑枚举 a_j 的值，那么 a_i 一定是其因子，于是根号枚举即可，时间复杂度 $\mathcal{O}(n\sqrt{V})$，V 是值域。

正解

考虑根号分治，对于 $a_i \le V^{\frac{2}{3}}$ 的时候根号暴力枚举，否则可以注意到 $b \le \lfloor \frac{V}{a_i} \rfloor \le V^{\frac{1}{3}}$，此时枚举 b 即可。

综上时间复杂度 $\mathcal{O}(nV^{\frac{1}{3}})$。

T4

20 分

暴力搜索即可。

40 分

先考虑如何计算使得一个序列升序的最小代价。

注意到在最优解中一定不会存在操作的两个区间有交集，这是显然的，否则对两者的并集排序更优。

知道这个就好做了，首先肯定就有一个指数级的暴力了。

60 分

注意对区间 $[l,l]$ 操作的代价是 0。

不如从最简单的情况开始，最开始操作区间 $[1,n]$ 一步到位，但是这样子很浪费。

例如序列 $1,3,2,5,4$，可以只对 $[1,3],[4,5]$ 操作，也就是说最优解是在最初状态为 $[1,n]$ 的情况下不断对区间进行拆分得到的，区间拆的越多，代价和就越小。

那么一个序列升序的最小代价就十分好计算了，直接对分割点进行计算即可：

$$n-1-\sum_{i=1}^{n-1}[pre_i < suf_{i+1}]$$

其中 pre_i 表示前缀最大值，suf_i 表示后缀最小值。

对 A 的每一个子区间暴力算上面这个东西即可做到 $\mathcal{O}(n^3)$ 的。

80 分

不如直接考虑枚举分割点在哪里，然后重新计算出 pre, suf 算一下上面这个式子就可以了。

时间复杂度 $\mathcal{O}(n^2)$。

正解

注意到问题本质转化为了有多少个三元组 $(i,j,k)(i \le j < k)$ 满足区间 $[i,j]$ 的最大值小于 $[j+1,k]$ 的最小值。

不如考虑枚举区间 $[i,j]$ 的最大值 x，可以通过 set 找到前一个比 x 大的数的位置 a，后一个比 x 大的数的位置 c，c 后面第一个比 x 小的数的位置 d。

那么 i 可以在 $(a,b]$ 中选，j 一定是 $c-1$，k 可以在 $[c,d)$ 中选，贡献为 $(b-a)\times(d-c)$。

时间复杂度 $\mathcal{O}(n\log n)$。