冒泡排序

签到题啊，本来放的是一个 dp 题，但是这样的话整场比赛就很毒，所以放个简单题给大家送个温暖。

仔细观察这个写错的冒泡排序，本质上就是对每个 $\bmod k$ 下的同余类的位置进行了排序，所以模拟一下就好了，复杂度线性对数。

染色

说句闲话，这个题是两年前从 CP-ideas 看到的一个 idea，然后跟同学讨论了一下发现可做，然后就出出来了。

可能比较有个人差的题目?

正向去做是不太好做的，很难保证生成的序列本质不同。

不如直接去考虑最终这个序列长什么样子，即现在有一个序列 b，如何判断 a 能否通过一系列操作生成 b。

这里直接给出结论:先只保留每一个颜色段中一个数,生成新的序列 c(即如果 $b=[1,1,1,2,2,1,1,1]$，认为 $c=[1,2,1]$)，那么 a 能得到 b 当且仅当 c 是 a 的子序列。这个结论如果要形式化的证不是很好说得清楚，建议自己手模几个例子就能明白。

那么现在就转化成了一个**本质不同**子序列计数问题，但是跟我们所熟知的本质不同子序列计数有下面几点不同:

- 子序列的长度是重要的!
- 子序列的相邻两项不能相同!

设 f_i 表示满足上述要求长度为 i 的本质不同子序列个数，那么答案就是 $\sum\binom{n-1}{i-1}f_i$。

考虑如何求解 f_i，首先可以考虑子序列自动机，但是这样子做复杂度跟字符集大小有关，用上 bitset 优化是 $\mathcal{O}(\frac{n^2m}{w})$，可以通过 76 分。

实际上，本质不同子序列还有另外一种求解方法，设 dp_c 表示当前以字符 c 结尾的本质不同子序列个数，转移是简单的，当新加入一个字符 x 时，$dp_x=1+\sum_{i\neq x}dp_i$。优化是 easy 的，时间复杂度 $\mathcal{O}(\frac{n^2}{w})$。

图

对题做的多人来说应该算是套路题吧。

显然的菠萝题（boruvka)，难点在于在有属于不同连通块的限制下，如何快速找到距离每个点最近的点。

首先这个边权就不是很好求，考虑转化，利用一个建虚点的 trick，给 T_2 的每个点下面都挂一个虚点，边权为 $d_1(i,x)$。那么问题变成了 j 在 T_2 上到其他点的最远距离。这个可以通过换根 dp 在 $\mathcal{O}(n^2)$ 做到，用个 Kruscal 做到 $\mathcal{O}(n^2\log n)$，可以得到 25 pts。

如何进一步优化呢? 最远距离，这启发我们往直径的方向思考，事实上，一个经典的结论是两个点集的直径是可合并的。但是注意到这个虚点的边权是在变化的，直接做不太行。仍然考虑换根，先把 T_1 用 dfn 序拍到序列上，用线段树维护 $[l,r]$ 区间组成的点集在 T_2 上的直径，那么每次 i 的改变导致的边权的变化变成了一个区间加的修改，这是很好实现的。

至此，如果用 $\mathcal{O}(n\log n)\sim\mathcal{O}(1)$ 的 LCA，可以 $\mathcal{O}(1)$ 求出一个 $f(i,j)$。

设 u_i, v_i 表示对于 T_1 的点 i，此时 T_2 上直径的两个端点，那么 $f(i,j) = \max(d_2(u_i,j), d_2(v_i,j))$。此时要找到一个 j 使得 $f(i,j)$ 最小，一个想法是直接拉出来这个直径的中点，可是这错的离谱，因为有属于不同连通块的限制。考虑到求的是一个 $\min(\max)$ 状物，不如先把里面的 $\max$ 去掉，即先把 j 分类，这是容易做到的，直接把 u_i 到 v_i 这条链拉出来，对于链上的每个点的子树分类讨论即可，形式化的讲，不如设链上的一个点为 u，其子树里的一个点为 v，认为 $f(i,v) = d_2(u,v) + \max(d_2(u_i,u), d_2(v_i,u))$，可是万一 v 也属于链上其他点的子树里的点怎么办，此时求出的 $f(i,v)$ 可能是错误的，怎么办？其实这并没有什么影响，因为求出的错误 $f(i,v)$ 只会更大不会变小，又因为最外层求的是 $\min$，所以该做法仍然是正确的。现在只需要对链上的点进行分类了，这个通过简单的树上倍增即可做到。

最后使用一个树链剖分 + ST 表查询即可，注意有属于不同连通块的限制，需要维护一个次小值。以及 u_i, v_i 的 LCA 外面还有点，需要一个换根 dp。

时间复杂度 $\mathcal{O}(n \log^2 n)$，常数巨大。

山峦

本来放的是一个很难的计数，但是由于某些原因换了一个简单的计数。

不算难的 T4，想一想就能做出来。

直接做确实不太行，突破点在于 c_1 很小，发现所有可能的 h 的第一行状态全部搜出来，只有 48620 种。

于是考虑状压 dp，设 $f_{i,S,j}$ 表示到了第 i 行，当前状态为 S，当前 h 的总和为 j 的方案数。

转移的话非常简单，枚举当前行的状态 S，上一行的状态 T，看能不能转移。可惜直接来的话最坏枚举次数达到了 48620^2。

注意到能转移的条件本质上是一个高维的包含关系，使用高维后缀和优化即可。

有一些细节需要注意：由于一个合法的状态满足数单调不增，按照正常的高维后缀和写法可能会出现一些问题，例如二维情况下 $(0,0)$ 与 $(1,1)$，按理来说是 $(1,1)$ 先转移到 $(0,1)$ 最后转移到 $(0,0)$。但是 $(0,1)$ 这个状态是不合法的，可能不存在导致贡献统计错误，于是需要进行修正。当高维前缀和进行到第 i 维时，如果是从不合法的状态转移而来，需要将该状态进行修正，即如果前面存在比第 i 个数小的数，滚一遍后缀 $\max$ 即可。例如 $(0,1)$ 就直接修正成 $(1,1)$。