20241210题解

A

设字符串为s[0],s[1],s[2],s[3]，则直接输出0,s[0],s[1],s[2]。

B

只要判断是否存在i,j,k，使得a[i]==b[j]且b[i]==a[k]即可。

由于n<=100，直接O(n^3)枚举就行。

C

单纯的递归，直接写个递归就可以了。

void digui(x){

if(!x) return;

digui(x-1);

cout << x << “ “;

digui(x-1);

}

D

用队列模拟即可，入队直接在队尾插入{x,c}表示c个x，出队时看队头front的第二个值是否大于当前的c，不大于的话全部弹出并将front.x\*front.c加到答案里，再将c减去front.c，否则答案加上front.x\*(front.c-c)，然后队头的第二个值减c就行，最后输出答案。这样的话保证时间复杂度是正确的。

E

做法很多，有容斥的、DP的、直接枚举结尾的。我们介绍最后一种做法。

对于每个结尾r，我们可以找它左边的，离它最近的一个点l，使得[l,r]内同时包含x和y，且l不能越过r左边的第一个不在[x,y]范围内的数。这个也可以直接在往右扫描的时候直接统计。反正就是扫描的时候记录三个变量r\_x,r\_y,r\_out分别表示三类数字出现的最右边的坐标，然后比较一下就行了。

F

我们考虑给对应数字所在的卡片建边，那么我们会发现，生成的图肯定是若干个连通块，每个连通块都恰好是一个环（或者单独一个点）。而每一条边的两边都至少要取一个点。

所以我们只需要考虑，每一个环有多少种方案，最后根据乘法原理把它们相乘就可以了。不难发现，环的贡献只与环的大小有关。我们设f[i]表示总共有 i 个点的环的方案数。经过列举以后发现：f[1]=1,f[2]=3,f[3]=4,f[4]=7,f[5]=11，不难猜出一个结论：f[i]=f[i−1]+f[i−2]

证明：假设环上总共有n个点，其中有三个点：A点，它两边的点分别是B点和C点，环上剩余的部分用O表示。那么，我们考虑是否选A点。如果选A点，那么B点和C点就可以任取了，所以这个环就变成了一个B+O+C的链，这个链不好计算，我们不妨作一条辅助线——连接BC。作完辅助线以后，这个图的答案就是f[n−1]，但是我们还漏算了一种情况，就是B,C 两个点都不取的情况。我们把这两个点压成一个点P。我们设不取P的答案是k。

如果不选A点，那么B点和C点就都必须选。所以我们可以把两个点还有A压成一个点P。此时新图的答案就是f[n−2]。但是我们多算了一种情况——在新图里不取P的情况。这时，我们发现：这种情况的个数恰好就是上文提到的 k！

所以答案就非常明显了：f[n]=f[n−1]+k+f[n−2]−k=f[n−1]+f[n−2]

那么最后还面临着一个小问题——怎么计算连通块里面点的个数。没错，使用并查集算法求解即可。

G

最小费用最大流板子题。对于交点所在的行列建边，流量为 1，费用为 −c[i]。在二分图上跑最小费用最大流，第 i 次网络的费用就是选 i 个点的分数。最后答案取负即可。

Ex

考虑我们如何判定一个排列是否能成为最终答案。连边i->p[i]，设环数为k，那么最少交换次数为n-k。那么充要条件是，每个环的所有c[i]相同，且n-k<=K且两者奇偶相同。

也就是说现在要求出，每个环所有点的c[i]相同，共形成了k个环的p[i]的方案数。先考虑一个O(n^2)的DP，设dp[i][j]表示考虑前i个共形成j个环的方案数，转移就是讨论i是加入原有环还是新开一个环，设a[i]为c[i]和前面相等的数量，则dp[i][j]=dp[i-1][j-1]+dp[i-1][j]\*a[i]。

考虑优化，发现dp[i][j]会转移到dp[i][j+1]和dp[i+1][j+1]，因此考虑构造多项式g(x)=(a[1]+x)\*(a[2]+x)\*...\*(a[n]+x)，就是求g(x)的前n项。直接乘肯定不行，考虑分治乘，时间复杂度O(n logn^2)。