Codeforces Round #732 (Div. 1)

比赛链接

C. AquaMoon and Permutations

题意

给定 $2n$ 个 $1\sim n$ 的排列 $p_1,p_2\cdots p_{2n}$，其中有 $p_1,p_2\cdots p_n$ 构成拉丁方。

且对于 $k=1\sim n$，p_k 与 p_{k+n} 至少有一个元素相同，保证 $p_1,p_2\cdots p_{2n}$ 中不存在两个完全相同的排列。

输入给定打乱顺序后的 $2n$ 个排列，问有多少个 n 子集可以构成拉丁方，同时输出任意一种合法方案。

题解

问题可以转化为有 $n\times n$ 个鸽巢，第 (i,j) 个巢表示排列的第 i 个位置为 j 的情况。

每个排列 P 可以视为 n 个物品 $(1,P_1),(2,P_2)\cdots (n,P_n)$，于是问题等价于选择 n 个排列使得每个巢恰好有一个物品。

接下来按下述流程操作 n 次来构造合法方案，同时统计答案个数：

1、将剩余的所有排列的所有物品放入鸽巢。

2、如果某个巢中只有一个物品，则选定该物品对应的排列 P，同时删去与该排列在相同位置有相同数值的其他排列。

执行完删去操作后，已经不存在 $(1,P_1),(2,P_2)\cdots (n,P_n)$ 这 n 中物品，于是至少可以删去 $(1,P_1),(2,P_2)\cdots (n,P_n)$ 这 n 个鸽巢。

由于原来的 $p_1,p_2\cdots p_n$ 正好对应 n^2 个鸽巢，且它们之间不会相互删除，所以上述操作恰好删去 n 个鸽巢。

3、如果不存在这样的巢，假设已经选定了 k 个排列，于是鸽巢还剩下 $n(n-k)$ 个。

每个鸽巢至少有两个物品，所以至少有 $2n(n-k)$ 个物品，对应至少有 $2(n-k)$ 个排列。

实际上，由于原来的 p_i 和 p_{i+n} 一定会相互删除，所以选定 k 个排列则至少额外删除了 k 个排列，于是至多有 $2(n-k)$ 个排列。

综上所述，剩下的排列恰好有 $2(n-k)$ 个，且每个鸽巢恰有两个物品。

$2(n-k)$ 个排列对应原来 $p_1,p_2\cdots p_n$ 中的 $n-k$ 个和 $p_{n+1},p_{n+2}\cdots p_{2n}$ 中的 $n-k$ 个。

而原来 $p_1,p_2\cdots p_n$ 中的 $n-k$ 个排列是拉丁方的一部分，所以恰好在 $n(n-k)$ 个鸽巢中各放一

个物品。

由于每个鸽巢有恰两个物品，于是$p_{n+1},p_{n+2}\cdots p_{2n}$ 中的 $n-k$ 个排列也恰好在 $n(n-k)$ 个鸽巢中各放一个物品。

于是从这两组中任选一个最后都能组成拉丁方，方案数乘以 2。

每轮操作时间复杂度 $O(n^2)$，总时间复杂度 $O(n^3)$。

```
const int MAXN=505,mod=998244353;
int a[MAXN<<1][MAXN],b[MAXN],c[MAXN];
bool vis[MAXN<<1];
int main()
{
    int T=read_int();
    while(T--){
        int n=read_int();
        _for(i,0,n<<1){
            _for(j,0,n)
            a[i][j]=read_int()-1;
            vis[i]=false;
        }
        int ans=1;
        _for(k,0,n){
            int pos=-1;
            _for(i,0,n){
                _for(j,0,n)
                c[j]=0;
                _for(j,0,n<<1){
                    if(vis[j])continue;
                    c[a[j][i]]++;
                }
                _for(j,0,n<<1){
                    if(vis[j])continue;
                    if(c[a[j][i]]==1){
                        pos=j;
                        break;
                    }
                }
                if(pos!=-1)
                break;
            }
            if(pos==-1){
                ans=(ans<<1)%mod;
                _for(i,0,n<<1){
                    if(!vis[i]){
                        pos=i;
                        break;
                    }
                }
```

```
            }
            b[k]=pos;
            vis[pos]=true;
            _for(i,0,n<<1){
                if(vis[i])continue;
                _for(j,0,n){
                    if(a[i][j]==a[pos][j]){
                        vis[i]=true;
                        break;
                    }
                }
            }
        }
        enter(ans);
        _for(i,0,n)
        space(b[i]+1);
        puts("");
    }
    return 0;
}
```