**872 – Ordering**

public class Main {

static int casos ;

static int cont;

static String[] letras;

static String[] reglas;

static Hashtable<String , Integer> ids;

static Hashtable<Integer , String> idss;

static LinkedList<String> res;

static int map[][];

static int path[];

static int vis[];

static int carcs[];

static int size;

public static void main(String[] args) throws Exception{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

casos = Integer.parseInt(br.readLine().trim());

for(int k = 0 ; k < casos; k++){

br.readLine();

ids = new Hashtable<String , Integer>();

idss = new Hashtable<Integer , String>();

letras = br.readLine().split(" ");

cont = 0 ;

for(String e : letras ){

ids.put(e, cont);

idss.put(cont , e);

cont++;

}

map = new int[cont][cont];

carcs = new int[cont];

for(int i = 0 ; i < cont ; i++){

carcs[i] = 0;

for(int j = 0 ; j < cont ; j++){

map[i][j] = Integer.MAX\_VALUE;

}

}

reglas = br.readLine().split(" ");

for(String regla : reglas){

int id1 = ids.get(regla.charAt(0) + "");

int id2 = ids.get(regla.charAt(2) + "");

map[id1][id2] = 1;

carcs[id2]++;

}

if(k > 0)System.out.println();

solve();

}

}

private static void solve() {

vis = new int[cont];

for(int i = 0 ; i < cont ; i++) vis[i] = 0;

LinkedList<Integer> opciones =new LinkedList<Integer>();

size = 0;

for(int i = 0 ; i < cont ; i++){

if(carcs[i] == 0){

opciones.add(i);

size++;

}

}

path = new int[cont];

res = new LinkedList<String>();

size = opciones.size();

for(int i = 0 ; i < size ;i++){

int v = opciones.removeLast();

path[0] = v;

backTrack( v , 1);

}

if(res.size() == 0)System.out.println("NO");

else{

Collections.sort(res);

for(String s: res){

System.out.println(s);

}

}

}

private static void backTrack(int nodo , int cv ) {

int sz;

if(cv == cont){

String r = "";

for(int i = 0 ; i < cont ; i++)r += idss.get(path[i]);

res.add(r);

}else{

vis[nodo] = 1;

// eliminar arcos nodo

LinkedList<Integer> arcRemoved = new LinkedList<Integer>();

LinkedList<Integer> candidatos = new LinkedList<Integer>();

for(int v = 0 ; v < cont ; v++){

if(map[nodo][v] == 1){

map[nodo][v] = 0;//remover el arco

arcRemoved.add(v);

carcs[v]--;

}

}

for(int v = 0 ; v < cont ; v++){

if(carcs[v] == 0){

candidatos.add(v);

}

}

sz = candidatos.size() ;

for(int v = 0 ; v < sz; v++){

int vv = candidatos.removeLast();

path[cv] = vv;

if(vv != nodo && vis[vv] != 1 ){

backTrack(vv , cv + 1);

}

}

sz = arcRemoved.size();

for(int i = 0 ; i < sz ; i++){

int v = arcRemoved.removeLast();

map[nodo][v] = 1;

carcs[v]++;

}

vis[nodo] = 0;

} }

}