bool check\_using\_dp(int adj[][MAXN], int n){

bool dp[MAXN][1<<MAXN]={0};

for(int i=0; i<n; i++)

dp[i][1<<i]=true;

for(int i=0; i<(1<<n); i++){

for(int j=0; j<n; j++)

if(i&(1<<j)){

for(int k=0; k<n; k++)

if(i&(1<<k) && adj[k][j] && k!=j && dp[k][i^(1<<j)]){

dp[j][i]=true;

break;

}

}

}

for(int i=0; i<n; i++)

if(dp[i][(1<<n)-1])

return true;

return false;

}

#define NOT\_IN\_STACK 0  
#define IN\_STACK 1

bool dfs(int v, bool adj[][MAXN], int label[MAXN], int instack\_count, int n){

if(instack\_count == n)

return true;

for(int i=0; i<n; i++)

if(adj[v][i] && label[i] == NOT\_IN\_STACK){

label[i]=IN\_STACK;

if(dfs(i, adj, label, instack\_count+1, n))

return true;

label[i]=NOT\_IN\_STACK;

}

return false;

}

bool check\_using\_dfs(bool adj[][MAXN], int n){

int label[MAXN];

for(int i=0; i<n; i++)

label[i]=NOT\_IN\_STACK;

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

label[i]=IN\_STACK;

if(dfs(i, adj, label, 1, n))

return true;

label[i]=NOT\_IN\_STACK;

}

return false;

}