```cpp
//
//  There are N men and N women, both numbered 1,2,…,N.
//  For each i,j (1≤i,j≤N), the compatibility of Man i and Woman j is given as an integer a[i][j]
//  If a[i][j] = 1, Man i and Woman j are compatible; otherwise not.
//  Taro is trying to make N pairs, each consisting of a man and a woman who are compatible.
//  Here, each man and each woman must belong to exactly one pair.
//  Find the number of ways in which Taro can make N pairs, modulo 10^9 + 7
//
//      Time Complexity: O(2^N * N^2)
//

#include <bits/stdc++.h>
#define ll long long

using namespace std;

const int MOD = 1e9 + 7;
const int maxSetCount = (1 << 21) + 1;

bool match[25][25];
int dp[maxSetCount][21];
int n;

int main() {
    cin >> n;

    // memset the dp
    for (int i = 0; i < maxSetCount; i++) {
        for (int j = 0; j < n; j++) {
            dp[i][j] = 0;
        }
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> match[i][j];
        }
    }

    int full_set = (1 << n) - 1;

    // base case
    for (int i = 0; i < n; i++) {
        dp[0][i] = 1;
    }

    for (int set = 0; set < (1 << n); set++) {
        int count = __builtin_popcount(set);
        for (int last = 0; last < n; last++) {
            if (((1 << last) & set) == 0 && match[last][count]) {
                int next_set = set ^ (1 << last);

                for (int i = 0; i < n; i++) {
                    dp[next_set][i] = (dp[next_set][i] + dp[set][last]) % MOD;
                }
            }
        }
    }

    cout << dp[full_set][0] << endl;
    return 0;
}
```