```c
#include <stdio.h>
#include <stdlib.h>
int sudokuSolv(int** board, int* emptySet, int depth, int size, int emptySetSize){
    // base case - all done
    if(depth == emptySetSize){
    for(int i = 0; i < size; i++)
        for(int j = 0; j < size; j++)
            printf("%d", board[i][j]);
            printf("\n");
        return 1;
    }
    int soln = 0;

    // Find the row and column of the next blank cell
    int row = emptySet[depth] / size;
    int col = emptySet[depth] % size;
    int temp = 1;

    // Can I place i in the next blank position?
    for(int i = 1; i <= size; i++){
        temp = 1;
        // Has i already been used in this row or this column?
        for(int j = 0; j < size; j++){
            temp *= (board[row][j] != i);
            temp *= (board[j][col] != i);
        }
        // Good - i is available for this row and column
        if(temp){
            board[row][col] = i;
            soln += sudokuSolv(board, emptySet, depth + 1, size, emptySetSize);
        }
    }
    board[row][col] = 0; // Reset
    return soln;
}

int main(){
    int n;
    scanf("%d", &n);
    int **board = (int**)calloc(n, sizeof(int*));
    for(int i = 0; i < n; i++)
        board[i] = (int*)calloc(n, sizeof(int));

    int r, c, v;
    int count = 0;
    while(1){
        scanf("%d %d %d", &r, &c, &v);
        if (r == -1)
            break;
        count += 1;
        board[r][c] = v;
    }

    int *emptySet = (int*)(calloc(n * n - count, sizeof(int)));
    int index = 0;
    for(int i = 0; i < n; i++){
        for (int j = 0; j < n; j++){
            if (!board[i][j]){ // Unfilled position
                emptySet[index] = i * n + j;
                index += 1;
            }
        }
    }
    int soln = sudokuSolv(board, emptySet, 0, n, index);
    printf("%d", soln);
    return 0;
}
```