3) (10 pts) DSN (Backtracking)

For the purposes of this question we define a Top Left Knight's Path to be a sequence of jumps taken by a single knight piece, starting at the top left corner of a R by C board, which visits each square exactly once. The knight may end on any square on the board. Recall that the way a knight jumps is by either moving 1 square left or right, followed by 2 squares up or down, OR by moving 2 squares left or right, followed by 1 square up or down. In this question you'll complete a recursive function that counts the total number of Top Left Knight's Paths for a particular R and C (which will be constants in the code.) Your code should use the constants R and C and should still work if the values of these constants were changed. Complete the recursive function below so that it correctly solves this task. Just fill out the blanks given to you in the recursive function.

```
#include <stdio.h>                          // Grading: 1 pt per slot, must be
#include <stdlib.h>                         // perfectly correct to earn the point.
#define R 6
#define C 6
#define NUMDIR 8
const int DR[] = {-2,-2,-1,-1,1,1,2,2};
const int DC[] = {-1,1,-2,2,-2,2,-1,1};

int main() {
    printf("There were %d Top Left Knight Paths.\n", countTours());
    return 0;
}
int countTours() {
    int used[R][C], i, j;
    for (i=0; i<R; i++)
        for (j=0; j<C; j++)
            used[i][j] = 0;
    return countToursRec(used, 0, 0, 0);
}
int countToursRec(int used[][C], int numMarked, int curR, int curC) {

    if (numMarked == R*C )

        return  1;

    int i, res = 0;
    for (i=0; i<NUMDIR; i++) {

        int nextR = curR + DR[i];
        int nextC = curC + DC[i];

        if (inbounds(nextR, nextC) && !used[nextR][nextC] ) {
            used[nextR][nextC] = 1 ;
            res += countToursRec(used, numMarked + 1, nextR , nextC );
            used[nextR][nextC] = 0 ;
        }
    }
    return res;
}

int inbounds(int nextR, int nextC) {
    return nextR >= 0 && nextR < R && nextC >= 0 && nextC < C;
}
```