6/24/24, 7:31 PM maze.c

E:\maze.c

```
1 #include <stdio.h>
    #include <stdbool.h>
   #include <string.h>
 3
 4
 5
    const char direction[] = "DLRU";
 6
    const int dr[] = \{1, 0, 0, -1\};
 7
    const int dc[] = \{0, -1, 1, 0\};
 8
 9
    const int n = 4; // Define n as the size of the maze
10
11
    bool isValid(int r, int c, int maze[n][n]) {
12
        return r >= 0 && c >= 0 && r < n && c < n && maze[r];
13
    }
14
15
    void solve(int r, int c, int maze[n][n], char currentPath[], char ans[][100], int* count) {
16
        if (r == n - 1 && c == n - 1) {
17
            strcpy(ans[*count], currentPath);
18
            (*count)++;
19
            return;
20
        }
21
22
        maze[r] = 0;
23
24
        for (int i = 0; i < 4; i++) {
25
            int nextr = r + dr[i];
26
            int nextc = c + dc[i];
27
28
            if (isValid(nextr, nextc, maze)) {
29
                currentPath[strlen(currentPath)] = direction[i];
30
                solve(nextr, nextc, maze, currentPath, ans, count);
31
                currentPath[strlen(currentPath) - 1] = '\0';
32
            }
33
        }
34
35
        maze[r] = 1;
36
37
38
    void findPath(int maze[n][n]) {
39
        if (maze[0][0] == 1) {
40
            char currentPath[100];
41
            char ans[100][100];
42
            int count = 0;
43
            currentPath[0] = '\0';
44
            solve(0, 0, maze, currentPath, ans, &count);
45
            if (count == 0)
46
47
                printf("-1");
48
            else {
49
                for (int i = 0; i < count; i++) {</pre>
50
                    printf("%s ", ans[i]);
51
                }
52
            }
53
        }
54
```

```
55
   int main() {
56
57
        int maze[4][4] = \{ \{1, 0, 0, 0\}, \}
58
                        {1, 1, 0, 1},
                        {1, 1, 0, 0},
59
60
                        {0, 1, 1, 1} };
61
        findPath(maze);
62
63
        return 0;
64
65 }
66
```