

```
1  #include <iostream>
2
3  #define M 16
4  #define N 12
5
6
7  void drawField();
8  void drawNewPosition(char);
9
10
11 typedef struct {
12     char f;
13     bool isWall;
14 } field;
15
16
17 ///////////////////////////////////////////////////
18 //                                SETUP
19 ///////////////////////////////////////////////////
20
21 // Initial Position
22 int x = 5, y = 5;
23
24 // Define a playing area
25 field playground[N][M];
26
27
28 ///////////////////////////////////////////////////
29 //                                GAME ON!
30 ///////////////////////////////////////////////////
31 // Note that this is not optimized: there's no control over what is given
32 // as input and what the program should do in case of unexpected commands
33 // though this probably won't cause problems
34
35
36 int main() {
37     // Let's get started
38     drawField();
39
40     // First move
41     char c;
42     std::cin >> c;
43
44     // Check the first move and continue to receive new moves until
45     // the end of the game (command 'q')
46     while (c != 'q') {
47         if (c == 'l' || c == 'r' || c == 'u' || c == 'd')
48             drawNewPosition(c);
49         std::cin >> c;
50     }
51
52     return 0;
53 }
54
55
56
57
58
```

```
59
60 // DRAW INITIAL STATE OF THE GAME
61
62 void drawField() {
63     for (int i = 0; i < N; i++) {
64         for (int j = 0; j < M; j++) {
65             // In the borders of the field, the following will result true.
66             // Note that there's an opening at the top side of the field
67             // (at the fourth column).
68             playground[i][j].isWall = ((i == 0 && j != 3) || i == (N - 1) ||
69                                         (j == 0) || j == (M - 1));
70
71             // Drawing
72             if (playground[i][j].isWall) // Wall
73                 playground[i][j].f = '*';
74             else if (i == x && j == y) // Initial Position
75                 playground[i][j].f = 'O';
76             else
77                 playground[i][j].f = ' '; // Inside field
78
79             // Printing
80             std::cout << playground[i][j].f;
81         }
82         std::cout << "\n";
83     }
84 }
85
86
87 // UPDATE STATE OF THE GAME
88
89 void drawNewPosition(char c) {
90     // Free old position
91     playground[x][y].f = ' ';
92
93     // Check that the player remain into the field
94     // after moving
95     if (c == 'l' && y < M - 1 && y > 1 && x != 0)
96         y--;
97     else if (c == 'r' && y < M - 2 && y > 0 && x != 0)
98         y++;
99     else if (c == 'u' && x < N && (x > 0 && y == 3))
100         x--;
101     else if (c == 'u' && x < N && (x > 1 && y != 3))
102         x--;
103     else if (c == 'd' && x < N - 2 && (x >= 0))
104         x++;
105
106     // Draw new position
107     playground[x][y].f = 'O';
108
109     // Print new state of the game
110     for (int i = 0; i < N; i++) {
111         for (int j = 0; j < M; j++)
112             std::cout << playground[i][j].f;
113         std::cout << "\n";
114     }
115 }
```