```cpp
#include <iostream>

#define M 16
#define N 12


void drawField();
void drawNewPosition(char);


typedef struct {
    char f;
    bool isWall;
} field;


////////////////////////////////////////////////////////////////////////
//                          SETUP
////////////////////////////////////////////////////////////////////////

// Initial Position
int x = 5, y = 5;

// Define a playing area
field playground[N][M];


////////////////////////////////////////////////////////////////////////
//                          GAME ON!
////////////////////////////////////////////////////////////////////////
// Note that this is not optimized: there's no control over what is given
// as input and what the program should do in case of unexpected commands
// though this probably won't cause problems


int main() {
    // Let's get started
    drawField();

    // First move
    char c;
    std::cin >> c;

    // Check the first move and continue to receive new moves until
    // the end of the game (command 'q')
    while (c != 'q') {
        if (c == 'l' || c == 'r' || c == 'u' || c == 'd')
            drawNewPosition(c);
        std::cin >> c;
    }

    return 0;
}
```

```cpp

// DRAW INITIAL STATE OF THE GAME

void drawField() {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++) {
            // In the borders of the field, the following will result true.
            // Note that there's an opening at the top side of the field
            // (at the fourth column).
            playground[i][j].isWall = ((i == 0 && j != 3) || i == (N - 1) ||
                                       (j == 0) || j == (M - 1));

            // Drawing
            if (playground[i][j].isWall)         // Wall
                playground[i][j].f = '*';
            else if (i == x && j == y)           // Initial Position
                playground[i][j].f = 'O';
            else
                playground[i][j].f = ' ';        // Inside field

            // Printing
            std::cout << playground[i][j].f;
        }
        std::cout << "\n";
    }
}


// UPDATE STATE OF THE GAME

void drawNewPosition(char c) {
    // Free old position
    playground[x][y].f = ' ';

    // Check that the player remain into the field
    // after moving
    if (c == 'l' && y < M - 1 && y > 1 && x != 0)
        y--;
    else if (c == 'r' && y < M - 2 && y > 0 && x != 0)
        y++;
    else if (c == 'u' && x < N && (x > 0 && y == 3))
        x--;
    else if (c == 'u' && x < N && (x > 1 && y != 3))
        x--;
    else if (c == 'd' && x < N - 2 && (x >= 0))
        x++;

    // Draw new position
    playground[x][y].f = 'O';

    // Print new state of the game
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++)
            std::cout << playground[i][j].f;
        std::cout << "\n";
    }
}
```