```
#include <stdio.h>

// This function clears the content of the board array by putting a space ' '
// at every position.
void clearboard(char board[6][7])
{
	int row, col;

	for (row = 0; row < 6; row++) {
		for (col = 0; col < 7; col++) {
			board[row][col] = ' ';
		}
	}
}

// This function prints the content of the board array. The content is
// displayed so that the top row has an index of 0 and the bottom row has an
// index of 5. Similarly the left column has an index of 0 and the right column
// has an index of 6.
void printboard(char board[6][7])
{
	int row, col;

	// Clear the entire screen, assuming the height of the screen has 25 rows
	for (row = 0; row < 25; row++) {
		printf("\n");
	}

	// Print the content in a tabular structure
	for (row = 0; row < 6; row++) {
		printf("\t --------------------------- \n");
		printf("\t");
		for (col = 0; col < 7; col++) {
			printf("| %c ", board[row][col]);
		}
		printf("|\n");
	}
	printf("\t --------------------------- \n\n\n\n");
}

// This function returns true if there is at least one empty space available
// in the given column. It returns false otherwise
bool isemptycolumn(char board[6][7], int column)
{
	//
	// Task 1
	//

	//
	// 1. Return false if the column number is invalid
	//
	if (column<0||column>6) {
		return false;
	}
	else if(board[0][column]!=' '){
		return false;
	}
	//
	// 2. Return true if the top slot at the column is empty;
```

```
	//    return false otherwise



	return true;
}

// This function drops a disc in a column for a player. For player 1, the
// disc is shown as 'O' whereas the disc is shown as 'X' for player 2.
void playdisc(char board[6][7], int column, int player)
{
	//
	// Task 2
	//
	int row = 5; // Start from the bottom row

			// Search for the first empty slot in the column
			// from the bottom row to the top row
	while (board[row][column] != ' ') row--;
	if (player == 1)
		board[row][column] = 'O';
	else
		board[row][column] = 'X';
	//
	// From the bottom row, search for an empty slot and put a new disc there
	//

}

// This function returns true if the board is fully occupied. It returns false
// otherwise.
bool isboardfull(char board[6][7])
{
	int col;

	// Check all columns for empty spaces
	for (col = 0; col < 7; col++) {
		if (isemptycolumn(board, col))
			return false; // This will immediately stop the function
	}

	return true;
}

// This function returns true if there are four connected discs in any
// directions. This is done by checking the horizontal direction,
// vertical direction and the two diagonal directions for each disc.
bool isconnected(char board[6][7])
{
	//
	// Task 3
	//

	// Use a for loop to check each disc:
	//
	int row, column;

	//  Go through all the rows
	for (row = 0; row < 6; row++) {
```

```
            // Go through all the columns
            for (column = 0; column < 7; column++) {

                  //     If the disc is a player's disc, check for
                  //
                  if (board[row][column] != ' ') {
                        //         a connected four in the horizontal direction and
return true if found
                        //
                        if (column <= 3)
                              if (board[row][column] == board[row][column + 1] &&
                                    board[row][column] == board[row][column + 2] &&
                                    board[row][column] == board[row][column + 3])
                                    return true;
                        //         a connected four in the vertical direction and
return true if found
                        //
                        if (row <= 2)
                              if (board[row][column] == board[row + 1][column] &&
                                    board[row][column] == board[row + 2][column] &&
                                    board[row][column] == board[row + 3][column])
                                    return true;
                        //         a connected four in the diagonal up direction
and return true if found
                        //
                        if (row >= 3 && column <= 3)
                              if (board[row][column] == board[row - 1][column + 1]
&&
                                    board[row][column] == board[row - 2][column +
2] &&
                                    board[row][column] == board[row - 3][column +
3])
                                    return true;
                        //         a connected four in the diagonal down direction
and return true if found
                        if (row <= 2 && column <= 3)
                              if (board[row][column] == board[row + 1][column + 1]
&&
                                    board[row][column] == board[row + 2][column +
2] &&
                                    board[row][column] == board[row + 3][column +
3])
                                    return true;
                  }
            }
      }


      // Found nothing, return false
      return false;
}

int main()
{
      // This is the game board array, storing the current discs in the board.
      // The value at each position is either:
      // - ' ' means no disc
      // - 'O' means player 1
      // - 'X' means player 2
```

```
char gameboard[6][7];

// This is the current player, which can be 1 or 2.
int currentplayer = 1;

// This is the current disc letter, which can be 'O' or 'X'.
char currentdisc = 'O';

// This is the move, a column number from 1 to 7, of the current player.
int nextmove;

// This indicates whether the game is over, i.e.:
// - the game board is full, or
// - one of the players wins the game
bool gameover = false;

// First, clear the game board, i.e. fill it with spaces
clearboard(gameboard);

// This while loop runs while the game is not over.
// It keeps on asking the next move of the players until the board is full
// or one of the players wins the game
while (!gameover) {
	// Print the current game board
	printboard(gameboard);

	// Ask the player (can be either player 1 or 2) for the next move;
	// The next move has to be a valid column, this is one of the things
	// checked by the isemptycolumn() function.
	do {
		printf("Player %d (%c), please select your column (1 to 7): ",
			currentplayer, currentdisc);
		scanf("%d", &nextmove);
	} while (!isemptycolumn(gameboard, nextmove - 1));

	// Drop the disc in the selected column for the current player
	playdisc(gameboard, nextmove - 1, currentplayer);

	// This if statement checks if the game is finished.
	// If so, the game is over; otherwise, move on to the next player
	if (isboardfull(gameboard) || isconnected(gameboard)) {
		// Set the gameover variable to true and the while loop will stop
		gameover = true;
	}
	else {
		// Move on to the next player
		if (currentplayer == 1) {
			currentplayer = 2;
			currentdisc = 'X';
		}
		else {
			currentplayer = 1;
			currentdisc = 'O';
		}
	}
}

// Print the final game board
printboard(gameboard);
```

```
	// Show the game over message
	if (isconnected(gameboard)) {
		printf("Player %d (%c) won the game!\n\n", currentplayer, currentdisc);
	}
	else {
		printf("No more moves left, the result is a draw!\n\n");
	}
}
```