```java
/**
 * @author
 * CIS 36b
 * Activity 5.1
 */

import java.util.Random;
import java.util.Scanner;

public class TicTacToe2d {
    /**
     * Initializes the board by assigning all array elements the value of '-'
     * @param board the array representing the tic-tac-toe board
     */
    public static void initializeBoard(String board[][]) {

        for (int i = 0; i < board.length; i++){
             for (int j = 0; j < board[i].length; j++){
                   board[i][j] = "-";//Fill in missing method body here
             }
        }

    }

    /**
     * Prints the board to the console in the form of a grid,
     * including column and row numbers
     * @param board the array representing the tic-tac-toe board
     */
    public static void printBoard(String board[][]) {
        System.out.print("\nTic-Tac-Toe:\n ");
        for (int i = 1; i <= 3; i++) {
            System.out.print(" " + i);
        }
        System.out.println();

        for (int i = 0; i < board.length; i++) {
            System.out.print((i + 1) + " ");

            for (int j = 0; j < board[i].length; j++ ){

                  System.out.print(board[i][j] + " ");
            }
            //fill in for loop here

            System.out.println();

        }
    }


/**
* Determines whether a particular position
* on the board has already been taken.
* @param board the array representing the game board
* @param row the row to check
* @param col the column to check
* @return whether that position has already been taken
*/

    public static boolean alreadyTaken(String board[][], int row, int col) {
        return (!board[row - 1][col - 1].equals("-"));
    }
```

```java
/**
* Places an X or O into the correct position on the board.
* Called when either the player or computer makes its move.
* @param board the array representing the tic-tac-toe board
* @param row the row in the array at which to place the X or O
* @param col the column in the array at which to place the X or O
* @param character either X or O
*/

    public static void makePlacement(String board[][], int row, int col, String
character) {

        board[row-1][col-1] = character;//fill in missing line here

    }

    /**
     * Gives a random position on the board
     * Used for generating moves by the computer
     * @return a random row or column
     */
    public static int randomPosition() {
        final int SIZE = 3; //3 X 3 array
        Random r = new Random(System.currentTimeMillis());
        return r.nextInt(SIZE) + 1;

    }

    /**
     * Determines whether three Strings are all Xs or all Os
     * Used as a helper method to the gameOver method
     * @param a the first String to compare, either X, O, or -
     * @param b the second String to compare, either X, O, or -
     * @param c the third String to compare, either X, O or -
     * @return whether the Strings are all Xs or all Os
     */
    public static boolean threeInRow(String a, String b, String c) {
        if (a.equals(b) && b.equals(c) && ! a.equals("-")) {
            return true;
        }
        return false;
    }

    /**
     * Determines whether the game is over
     * due to one player winning, using
     * a series of if statements.
     * Calls the threeInRow method for each
     * possible row on the board.
     * @param board the tic-tac-toe game board
     * @return whether the game is over
     */
    public static boolean gameOverWinner(String board[][]) {
        boolean winner = false;

        //Check if winning across

        if (threeInRow(board[0][0], board[0][1], board[0][2])) {
            winner = true;
        } else if (threeInRow(board[1][0], board[1][1], board[1][2])) {
            winner = true;
        } else if (threeInRow(board[2][0], board[2][1], board[2][2])) {
            winner = true;
        //Fill in the missing 3 else if statements for winning going down

        //Check if winning diagonally
        }else if (threeInRow(board[0][0], board[1][0], board[2][0])) {
                winner = true;
            } else if (threeInRow(board[0][1], board[1][1], board[2][1])) {
                winner = true;
            } else if (threeInRow(board[0][2], board[1][2], board[2][2])) {
                winner = true;

        } else if(threeInRow(board[0][0], board[1][1], board[2][2])) {
            winner = true;
        } else if(threeInRow(board[0][2], board[1][1], board[2][0])) {
            winner = true;
        } else {
            winner = false;
        }
        return winner;
    }

    /**
     * Determines whether the game is over
     * due to a draw.
     * Compares numMoves to the length.
     * @param board the tic-tac-toe game board
     * @return whether the game is over
     */
    public static boolean gameOverDraw(String board[][]) {
        for (int i = 0; i < board.length; i++) {
            for (int j = 0; j < board[i].length; j++) {
                if (board[i][j].equals("-")) {
                    return false;
                }
            }
        }
        return true;
    }


    public static void main(String[] args) {
        String board[][] = new String[3][3];
        String player = " ";
        String computer = "X";
        int row;
        int col;
        int numMoves = 0;

        System.out.println("Welcome to Tic-Tac-Toe!");
        Scanner input = new Scanner(System.in);
        System.out.print("\nWould you like to play as X or O: ");
        player = input.next().toUpperCase();
        if (player.equals("X")) {
            computer = "O";
        }

       initializeBoard(board);
       printBoard(board);

       while(!gameOverWinner(board) && !gameOverDraw(board)) {
            System.out.print("\nPlease enter your move:\nRow: ");
            row = input.nextInt();
            System.out.print("Column: ");
            col = input.nextInt();
```

```
            while (alreadyTaken(board, row, col)) {
                System.out.print("\nThat spot is already taken!"
                        + "\n\nPlease enter your move: \nRow:");
                row = input.nextInt();
                System.out.print("Column: ");
                col = input.nextInt();

            }
            makePlacement(board, row, col, player);
            numMoves++;
            printBoard(board);

            if(gameOverWinner(board) || gameOverDraw(board)) {
                break;
            }

            row = randomPosition();
            col = randomPosition();

            while (alreadyTaken(board, row, col)) {
                row = randomPosition();
                col = randomPosition();
            }

            makePlacement(board, row, col, computer);
            numMoves++;
            System.out.println("\nCounter move!");

            printBoard(board);
        }
        if(gameOverWinner(board)) {
            if (numMoves % 2 == 0) {
                System.out.println("\n" + computer + " wins!");
            } else {
                System.out.println("\n" + player + " wins!");
            }
        } else {
            System.out.println("\nIt's a tie!");
        }
        System.out.println("\n***Game Over***");
        //input.close();

    }
}
```