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

public class TicToe {


   public static void main(String[] args) {

       char[][] board = {{' ', ' ', ' '},
               {' ', ' ', ' '},
               {' ', ' ', ' '}};

       PrintBoard(board);

       while (true) {
           //   playerTurn(board);
           if (winners(board)) {
               break;
           }
           PrintBoard(board);
           computerTurn(board);
           if (winners(board))
               break;
           PrintBoard(board);
       }
   }

   private static void PrintBoard(char[][] board) {

       System.out.println(board[0][0] + "|" + board[0][1] + "|" +
board[0][2]);
       System.out.println("-+-+-");
       System.out.println(board[1][0] + "|" + board[1][1] + "|" +
board[1][2]);
       System.out.println("-+-+-");
       System.out.println(board[2][0] + "|" + board[2][1] + "|" +
board[2][2]);

   }

   private static void moves(char[][] board, String position, char symbol) {
       switch (position) {
           case "1":
               board[0][0] = symbol;
               return;
           case "2":
               board[0][1] = symbol;
               return;
           case "3":
               board[0][2] = symbol;
               return;
           case "4":
               board[1][0] = symbol;
```

```
            return;
        case "5":
            board[1][1] = symbol;
            return;
        case "6":
            board[1][2] = symbol;
            return;
        case "7":
            board[2][0] = symbol;
            return;
        case "8":
            board[2][1] = symbol;
            return;
        case "9":
            board[2][2] = symbol;
            return;
        default:
    }
}

public static void computerTurn(char[][] board) {

    Random rand = new Random();
    int computerMove;

    while (true) {
        computerMove = rand.nextInt(9) + 1;
        if (isValidMove(board, Integer.toString(computerMove))) {
            break;
        }
    }

    System.out.println("computer move -> " + computerMove);
    moves(board, Integer.toString(computerMove), '0');

}

public static void playerTurn(char[][] board) {
    Scanner sc = new Scanner(System.in);

    String position;
    while (true) {
        System.out.println("where would you like to move ? from (1 -9");
        position = sc.next();
        if (isValidMove(board, position)) {
            break;
        }
    }

    System.out.println("player move -> " + position);
    moves(board, position, 'x');
    }

    private static boolean isValidMove(char[][] board, String position) {
        switch (position) {
            case "1":
                return (board[0][0] == ' ');
            case "2":
                return (board[0][1] == ' ');
            case "3":
                return (board[0][2] == ' ');
            case "4":
                return (board[1][0] == ' ');
            case "5":
                return (board[1][1] == ' ');
            case "6":
                return (board[1][2] == ' ');
            case "7":
                return (board[2][0] == ' ');
            case "8":
                return (board[2][1] == ' ');
            case "9":
                return (board[2][2] == ' ');
            default: {
                System.out.println("wrong move");
            }
        }
        return false;
    }

    private static boolean requirementsOfWin(char[][] board, char symbol) {

        if ((board[0][0] == symbol && board[0][1] == symbol && board[0][2] ==
symbol) ||
                (board[1][0] == symbol && board[1][1] == symbol && board[1][2]
== symbol) ||
                (board[2][0] == symbol && board[2][1] == symbol && board[2][2]
== symbol) ||

                (board[0][0] == symbol && board[1][0] == symbol && board[2][0]
== symbol) ||
                (board[0][1] == symbol && board[1][1] == symbol && board[2][1]
== symbol) ||
                (board[0][2] == symbol && board[1][2] == symbol && board[2][2]
== symbol) ||

                (board[0][0] == symbol && board[1][1] == symbol && board[2][2]
== symbol) ||
                (board[0][2] == symbol && board[1][1] == symbol && board[2][0]
== symbol)) {
            return true;

        }
        return false;
```

```java
    }

    static boolean winners(char[][] board) {

        if (requirementsOfWin(board, 'x')) {
            PrintBoard(board);
            System.out.println("player win");
            return true;
        }
        if (requirementsOfWin(board, '0')) {
            PrintBoard(board);
            System.out.println("computer win");
            return true;
        }

        for (int i = 0; i < board.length; i++) {
            for (int j = 0; j < board[i].length; j++) {
                if (board[i][j] == ' ') {
                    return false;
                }
            }
        }
        PrintBoard(board);
        System.out.println("Game is tie");

        return true;
    }
}
```