

<terminated> TicTacToe [Java Application] R:\Program

Welcome to Tic-Tac-Toe

=====

| |

| |

| |

Player 1 turn 'X'

Which Row would you like?

0

Which Col would you like?

0

X| |

| |

| |

Player 2 turn 'O'

Which Row would you like?

1

Which Col would you like?

0

X| |

O| |

| |

Player 1 turn 'X'

Which Row would you like?

1

Which Col would you like?

1

X| |

O|X|

| |

Player 2 turn 'O'

Which Row would you like?

1

Which Col would you like?

1

Illegal Move

Which Row would you like?

0

Which Col would you like?

2

X| |O

O|X|

| |

Player 1 turn 'X'

Which Row would you like?

2

Which Col would you like?

2

X Player Wins!

X| O

O|X|

| X

<terminated> TicTacToe [Java Application] R:\Program

Welcome to Tic-Tac-Toe

=====

| |

| |

| |

Player 1 turn 'X'

Which Row would you like?

0

Which Col would you like?

0

X| |

| |

| |

Player 2 turn 'O'

Which Row would you like?

1

Which Col would you like?

1

X| |

|O|

| |

Player 1 turn 'X'

Which Row would you like?

0

Which Col would you like?

1

X|X|

|O|

| |

Player 2 turn 'O'

Which Row would you like?

0

Which Col would you like?

2

X|X|O

|O|

| |

Player 1 turn 'X'

Which Row would you like?

2

Which Col would you like?

0

X|X|O

|O|

X| |

Player 2 turn 'O'

Which Row would you like?

0

Which Col would you like?

1

```
Illegal Move
Which Row would you like?
1
Which Col would you like?
0
X|X|O
-----
O|O|
-----
X| |
Player 1 turn 'X'
Which Row would you like?
1
Which Col would you like?
2
X|X|O
-----
O|O|X
-----
X| |
Player 2 turn 'O'
Which Row would you like?
2
Which Col would you like?
2
X|X|O
-----
O|O|X
-----
X| |O
Player 1 turn 'X'
Which Row would you like?
1
Which Col would you like?
2
Illegal Move
Which Row would you like?
2
Which Col would you like?
1
Cat's Game!
X|X|O
-----
O|O|X
-----
X|X|O
```

```

1 import java.util.Scanner;
2
3 public class TicTacToe {
4
5     public static void main(String[] args) {
6         int Turn = 1;
7         boolean PlayerWin = false;
8         PlayerSide activePlayer = PlayerSide.X;
9         Game_Board board = new Game_Board();
10        int inputCol;
11        int inputRow;
12        Scanner input = new Scanner(System.in);
13
14
15        System.out.println("Welcome to Tic-Tac-Toe\n=====");
16
17        // printBoard(CurrentMoves);
18        while(Turn < 10 && !PlayerWin) {
19
20            board.displayBoard();
21
22            if(activePlayer == PlayerSide.X) {
23                System.out.println("Player 1 turn 'X'");
24            }
25            else {
26                System.out.println("Player 2 turn 'O'");
27            }
28
29            System.out.println("Which Row would you like?");
30            inputRow = input.nextInt();
31            input.nextLine();
32
33            System.out.println("Which Col would you like?");
34            inputCol = input.nextInt();
35            input.nextLine();
36
37            while(!board.updateBoard(inputCol, inputRow, activePlayer)) {
38                System.out.println("Illegal Move");
39
40                System.out.println("Which Row would you like?");
41                inputRow = input.nextInt();
42                input.nextLine();
43
44                System.out.println("Which Col would you like?");
45                inputCol = input.nextInt();
46                input.nextLine();
47            }
48
49
50            PlayerWin = checkWin(board.getStateBoard(), activePlayer);
51
52            if(!PlayerWin) {
53
54                Turn ++;
55
56                if(activePlayer == PlayerSide.X) {
57                    activePlayer = PlayerSide.O;
58                }

```

```

59         else {
60             activePlayer = PlayerSide.X;
61         }
62     }
63
64 }
65
66 if(PlayerWin) {
67     System.out.println(activePlayer.getSide() + " Player Wins!");
68 }
69 else {
70     System.out.println("Cat's Game!");
71 }
72
73 board.displayBoard();
74
75
76 input.close();
77 }
78
79 public static boolean checkWin(PlayerSide CurrentMovesf[ ][ ], PlayerSide CurrentPlayer) {
80     //col 1
81     if(AreEqual(CurrentPlayer, CurrentMovesf[0][0], CurrentMovesf[1][0], CurrentMovesf[2][0])) {
82         return true;
83     }
84     //col 2
85     if(AreEqual(CurrentPlayer, CurrentMovesf[0][1], CurrentMovesf[1][1], CurrentMovesf[2][1])) {
86         return true;
87     }
88     //col 3
89     if(AreEqual(CurrentPlayer, CurrentMovesf[0][2], CurrentMovesf[1][2], CurrentMovesf[2][2])) {
90         return true;
91     }
92     //row 1
93     if(AreEqual(CurrentPlayer, CurrentMovesf[0][0], CurrentMovesf[0][1], CurrentMovesf[0][2])) {
94         return true;
95     }
96     //row 2
97     if(AreEqual(CurrentPlayer, CurrentMovesf[1][0], CurrentMovesf[1][1], CurrentMovesf[1][2])) {
98         return true;
99     }
100    //row 3
101    if(AreEqual(CurrentPlayer, CurrentMovesf[2][0], CurrentMovesf[2][1], CurrentMovesf[2][2])) {
102        return true;
103    }
104    //diag 1
105    if(AreEqual(CurrentPlayer, CurrentMovesf[0][0], CurrentMovesf[1][1], CurrentMovesf[2][2])) {
106        return true;
107    }
108    //diag 2
109    if(AreEqual(CurrentPlayer, CurrentMovesf[0][2], CurrentMovesf[1][1], CurrentMovesf[2][0])) {
110        return true;
111    }
112    return false;
113 }
114
115 public static boolean AreEqual(PlayerSide x1, PlayerSide x2, PlayerSide x3, PlayerSide x4) {
116     if(x1 == x2 && x3 == x4 && x1 == x3) {
117
118         return true;
119     }
120     return false;
121 }
122 }

```

```

1
2 public enum PlayerSide {
3     X('X'), O('O'), nan(' ');
4
5     private char side;
6
7     private PlayerSide(char sidef) {
8         side = sidef;
9     }
10
11     public char getSide() {
12         return side;
13     }
14 }
15 }
16
1
2 public class Game_Board {
3     PlayerSide currentBoard[][];
4
5     Game_Board(){
6
7         currentBoard = new PlayerSide[3][3];
8
9         for(int i = 0; i < 3; i++) {
10             for(int j = 0; j < 3; j++) {
11                 currentBoard[i][j] = PlayerSide.nan;
12             }
13         }
14     }
15
16     boolean updateBoard(int col, int row, PlayerSide activePlayer) {
17         if(currentBoard[col][row] == PlayerSide.nan) {
18             currentBoard[col][row] = activePlayer;
19             return true;
20         }
21         else {
22             return false;
23         }
24     }
25
26     void displayBoard() {
27         System.out.println(currentBoard[0][0].getSide() + "|" + currentBoard[1][0].getSide() + "|"
28             + currentBoard[2][0].getSide());
29         System.out.println("-----");
30         System.out.println(currentBoard[0][1].getSide() + "|" + currentBoard[1][1].getSide() + "|"
31             + currentBoard[2][1].getSide());
32         System.out.println("-----");
33         System.out.println(currentBoard[0][2].getSide() + "|" + currentBoard[1][2].getSide() + "|"
34             + currentBoard[2][2].getSide());
35     }
36
37     PlayerSide[][] getStateBoard(){
38         return currentBoard;
39     }
40 }
41
42

```