

# **ARTIFICIAL INTELLIGENCE LAB**

## **ASSIGNMENT – 3**

### **TIC TAC TOE**

**NAME : PRATHAPANI SATWIK**

**REG.NO. : 20BCD7160**

#### **CODE :**

```
class Main
{
    static class Move
    {
        int row, col;
    };
    static char player = 'x', opponent = 'o';
    static Boolean isMovesLeft(char board[][])
    {
        for (int i = 0; i < 3; i++)
            for (int j = 0; j < 3; j++)
                if (board[i][j] == '_')
                    return true;
        return false;
    }
    static int evaluate(char b[][])
    {
        for (int row = 0; row < 3; row++)
        {
```

```
    if (b[row][0] == b[row][1] &&
        b[row][1] == b[row][2])
    {
        if (b[row][0] == player)
            return +10;
        else if (b[row][0] == opponent)
            return -10;
    }
}

for (int col = 0; col < 3; col++)
{
    if (b[0][col] == b[1][col] &&
        b[1][col] == b[2][col])
    {
        if (b[0][col] == player)
            return +10;

        else if (b[0][col] == opponent)
            return -10;
    }
}

if (b[0][0] == b[1][1] && b[1][1] == b[2][2])
{
    if (b[0][0] == player)
        return +10;
```

```

        else if (b[0][0] == opponent)
            return -10;
    }
    if (b[0][2] == b[1][1] && b[1][1] == b[2][0])
    {
        if (b[0][2] == player)
            return +10;
        else if (b[0][2] == opponent)
            return -10;
    }
    return 0;
}

static int minimax(char board[][[]],
                    int depth, Boolean isMax)
{
    int score = evaluate(board);
    if (score == 10)
        return score;
    if (score == -10)
        return score;
    if (isMovesLeft(board) == false)
        return 0;
    if (isMax)
    {
        int best = -1000;

```

```
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            if (board[i][j]=='_')
            {
                board[i][j] = player;
                best = Math.max(best, minimax(board,
                    depth + 1, !isMax));
                board[i][j] = '_';
            }
        }
    }
    return best;
}
else
{
    int best = 1000;
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            if (board[i][j] == '_')
            {
                board[i][j] = opponent;
```

```

        best = Math.min(best, minimax(board,
                                     depth + 1, !isMax));
        board[i][j] = '_';
    }
}
}
return best;
}
}

static Move findBestMove(char board[][])
{
    int bestVal = -1000;
    Move bestMove = new Move();
    bestMove.row = -1;
    bestMove.col = -1;
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            if (board[i][j] == '_')
            {
                board[i][j] = player;
                int moveVal = minimax(board, 0, false);
                board[i][j] = '_';
                if (moveVal > bestVal)

```



Main.java

```
1
2 class Main
3 {
4     static class Move
5     {
6         int row, col;
7     };
8
9     static char player = 'x', opponent = 'o';
10    static Boolean isMovesLeft(char board[][])
11    {
12        for (int i = 0; i < 3; i++)
13            for (int j = 0; j < 3; j++)
14                if (board[i][j] == '_')
15                    return true;
16        return false;
17    }
18    static int evaluate(char b[][])
19    {
20        for (int row = 0; row < 3; row++)
21        {
22            if (b[row][0] == b[row][1] &&
23                b[row][1] == b[row][2])
24            {
25                if (b[row][0] == player)
26                    return +10;
27                else if (b[row][0] == opponent)
28                    return -10;
29            }
30        }
31        for (int col = 0; col < 3; col++)
32        {
33            if (b[0][col] == b[1][col] &&
```

```

34         b[1][col] == b[2][col])
35     {
36         if (b[0][col] == player)
37             return +10;
38
39         else if (b[0][col] == opponent)
40             return -10;
41     }
42 }
43 if (b[0][0] == b[1][1] && b[1][1] == b[2][2])
44 {
45     if (b[0][0] == player)
46         return +10;
47     else if (b[0][0] == opponent)
48         return -10;
49 }
50
51 if (b[0][2] == b[1][1] && b[1][1] == b[2][0])
52 {
53     if (b[0][2] == player)
54         return +10;
55     else if (b[0][2] == opponent)
56         return -10;
57 }
58 return 0;
59 }
60 static int minimax(char board[][],
61                    int depth, Boolean isMax)
62 {
63     int score = evaluate(board);
64     if (score == 10)
65         return score;

```





```

99         depth + 1, !isMax));
100         board[i][j] = '_';
101     }
102 }
103 }
104 return best;
105 }
106 }
107 static Move findBestMove(char board[][])
108 {
109     int bestVal = -1000;
110     Move bestMove = new Move();
111     bestMove.row = -1;
112     bestMove.col = -1;
113     for (int i = 0; i < 3; i++)
114     {
115         for (int j = 0; j < 3; j++)
116         {
117             if (board[i][j] == '_')
118             {
119                 board[i][j] = player;
120                 int moveVal = minimax(board, 0, false);
121                 board[i][j] = '_';
122                 if (moveVal > bestVal)
123                 {
124                     bestMove.row = i;
125                     bestMove.col = j;
126                     bestVal = moveVal;
127                 }
128             }
129         }
130     }
131 }

```

```

32 ▾    System.out.printf("The value of the best Move " +
33      |   |   |   |   |   |   |   |   "is : %d\n\n", bestVal);
34
35      return bestMove;
36  }
37  public static void main(String[] args)
38  {
39      char board[][] = {{ 'x', 'o', 'x' },
40      |   |   |   |   |   |   |   |   { 'o', 'o', 'x' },
41      |   |   |   |   |   |   |   |   { ' _', ' _', ' _' }};
42
43      Move bestMove = findBestMove(board);
44      System.out.printf("The Optimal Move is :\n");
45 ▾    System.out.printf("ROW: %d COL: %d\n\n",
46      |   |   |   |   |   |   |   |   bestMove.row, bestMove.col );
47  }
48
49  }

```

## OUTPUT :

### Output

```
java -cp /tmp/5LfMfDAiwa Main
```

```
The value of the best Move is : 10
```

```
The Optimal Move is :
```

```
ROW: 2 COL: 2
```