

E:\tic tac toe.py

```
1 # Tic-Tac-Toe Program using
2 # random number in Python
3
4 # importing all necessary libraries
5 import numpy as np
6 import random
7 from time import sleep
8
9 # Creates an empty board
10
11
12 def create_board():
13     return(np.array([[0, 0, 0],
14                     [0, 0, 0],
15                     [0, 0, 0]]))
16
17 # Check for empty places on board
18
19
20 def possibilities(board):
21     l = []
22
23     for i in range(len(board)):
24         for j in range(len(board)):
25
26             if board[i][j] == 0:
27                 l.append((i, j))
28     return(l)
29
30 # Select a random place for the player
31
32
33 def random_place(board, player):
34     selection = possibilities(board)
35     current_loc = random.choice(selection)
36     board[current_loc] = player
37     return(board)
38
39 # Checks whether the player has three
40 # of their marks in a horizontal row
41
42
43 def row_win(board, player):
44     for x in range(len(board)):
45         win = True
46
47         for y in range(len(board)):
48             if board[x, y] != player:
```

```
49         win = False
50         continue
51
52     if win == True:
53         return(win)
54     return(win)
55
56 # Checks whether the player has three
57 # of their marks in a vertical row
58
59
60 def col_win(board, player):
61     for x in range(len(board)):
62         win = True
63
64         for y in range(len(board)):
65             if board[y][x] != player:
66                 win = False
67                 continue
68
69         if win == True:
70             return(win)
71     return(win)
72
73 # Checks whether the player has three
74 # of their marks in a diagonal row
75
76
77 def diag_win(board, player):
78     win = True
79     y = 0
80     for x in range(len(board)):
81         if board[x, x] != player:
82             win = False
83     if win:
84         return win
85     win = True
86     if win:
87         for x in range(len(board)):
88             y = len(board) - 1 - x
89             if board[x, y] != player:
90                 win = False
91     return win
92
93 # Evaluates whether there is
94 # a winner or a tie
95
96
97 def evaluate(board):
98     winner = 0
```

```
99
100     for player in [1, 2]:
101         if (row_win(board, player) or
102             col_win(board, player) or
103             diag_win(board, player)):
104
105             winner = player
106
107     if np.all(board != 0) and winner == 0:
108         winner = -1
109     return winner
110
111 # Main function to start the game
112
113
114 def play_game():
115     board, winner, counter = create_board(), 0, 1
116     print(board)
117     sleep(2)
118
119     while winner == 0:
120         for player in [1, 2]:
121             board = random_place(board, player)
122             print("Board after " + str(counter) + " move")
123             print(board)
124             sleep(2)
125             counter += 1
126             winner = evaluate(board)
127             if winner != 0:
128                 break
129     return(winner)
130
131
132 # Driver Code
133 print("Winner is: " + str(play_game()))
134
```