

main.py



Share

Run

```
1  PLAYER_X = 1
2  PLAYER_O = -1
3  EMPTY = 0
4
5  def evaluate(board):
6      for row in range(3):
7          if board[row][0] == board[row][1] == board[row][2] != EMPTY:
8              return board[row][0]
9      for col in range(3):
10         if board[0][col] == board[1][col] == board[2][col] != EMPTY:
11             return board[0][col]
12     if board[0][0] == board[1][1] == board[2][2] != EMPTY:
13         return board[0][0]
14     if board[0][2] == board[1][1] == board[2][0] != EMPTY:
15         return board[0][2]
16     return 0
17
18  def isMovesLeft(board):
19      for row in range(3):
20          for col in range(3):
21              if board[row][col] == EMPTY:
22                  return True
23     return False
24
```

main.py



Share

Run

```
25 def minimax(board, isMax):
26     score = evaluate(board)
27     if score == PLAYER_X:
28         return score
29     if score == PLAYER_O:
30         return score
31     if not isMovesLeft(board):
32         return 0
33
34     if isMax:
35         best = -float('inf')
36         for row in range(3):
37             for col in range(3):
38                 if board[row][col] == EMPTY:
39                     board[row][col] = PLAYER_X
40                     best = max(best, minimax(board, not isMax))
41                     board[row][col] = EMPTY
42         return best
43     else:
44         best = float('inf')
45         for row in range(3):
46             for col in range(3):
47                 if board[row][col] == EMPTY:
48                     board[row][col] = PLAYER_O
49                     best = min(best, minimax(board, not isMax))
```

main.py



Run

```
49         best = min(best, minimax(board, not isMax))
50         board[row][col] = EMPTY
51     return best
52
53 def findBestMove(board):
54     bestVal = -float('inf')
55     bestMove = (-1, -1)
56     for row in range(3):
57         for col in range(3):
58             if board[row][col] == EMPTY:
59                 board[row][col] = PLAYER_X
60                 moveVal = minimax(board, False)
61                 board[row][col] = EMPTY
62                 if moveVal > bestVal:
63                     bestMove = (row, col)
64                     bestVal = moveVal
65     return bestMove
66
67 def printBoard(board):
68     for row in board:
69         print(" ".join(["X" if x == PLAYER_X else "O" if x ==
70                         PLAYER_O else "." for x in row]))
71
72 board = [
73     [PLAYER_X, PLAYER_O, PLAYER_X],
74     [PLAYER_O, PLAYER_X, EMPTY],
75     [EMPTY, EMPTY, EMPTY]]
```

```
70
71 board = [
72     [PLAYER_X, PLAYER_O, PLAYER_X],
73     [PLAYER_O, PLAYER_X, EMPTY],
74     [EMPTY, PLAYER_O, PLAYER_X]
75 ]
76
77 print("Current Board:")
78 printBoard(board)
79 move = findBestMove(board)
80 print(f"Best Move: {move}")
81 board[move[0]][move[1]] = PLAYER_X
82 print("\nBoard after best move:")
83 printBoard(board)
```

Output

Clear

Current Board:

X O X

O X .

. O X

Best Move: (1, 2)

Board after best move:

X O X

O X X

. O X

=== Code Execution Successful ===