

NAME: SIDDHARTH R

REGISTER NO: 241801269

EXP NO: 3

EXP NAME: IMPLEMENTATION OF MINIMAX ALGORITHM

PROGRAM:

PLAYER\_X = 1

PLAYER\_O = -1

EMPTY = 0

```
def evaluate(board):
```

```
    for row in range(3):
```

```
        if board[row][0] == board[row][1] == board[row][2] != EMPTY:
```

```
            return board[row][0]
```

```
    for col in range(3):
```

```
        if board[0][col] == board[1][col] == board[2][col] != EMPTY:
```

```
            return board[0][col]
```

```
    if board[0][0] == board[1][1] == board[2][2] != EMPTY:
```

```
        return board[0][0]
```

```
    if board[0][2] == board[1][1] == board[2][0] != EMPTY:
```

```
        return board[0][2]
```

```
    return 0
```

```
def isMovesLeft(board):
```

```
    for row in range(3):
```

```
        for col in range(3):
```

```
        if board[row][col] == EMPTY:
            return True
    return False
```

```
def minimax(board, isMax):
```

```
    score = evaluate(board)
```

```
    if score == PLAYER_X: return score
```

```
    if score == PLAYER_O: return score
```

```
    if not isMovesLeft(board): return 0
```

```
    if isMax:
```

```
        best = -float('inf')
```

```
        for row in range(3):
```

```
            for col in range(3):
```

```
                if board[row][col] == EMPTY:
```

```
                    board[row][col] = PLAYER_X
```

```
                    best = max(best, minimax(board, not isMax))
```

```
                    board[row][col] = EMPTY
```

```
        return best
```

```
    else:
```

```
        best = float('inf')
```

```
        for row in range(3):
```

```
            for col in range(3):
```

```
                if board[row][col] == EMPTY:
```

```
        board[row][col] = PLAYER_O

        best = min(best, minimax(board, not isMax))

        board[row][col] = EMPTY

    return best
```

```
def findBestMove(board):
```

```
    bestVal = -float('inf')

    bestMove = (-1, -1)

    for row in range(3):

        for col in range(3):

            if board[row][col] == EMPTY:

                board[row][col] = PLAYER_X

                moveVal = minimax(board, False)

                board[row][col] = EMPTY

                if moveVal > bestVal:

                    bestMove = (row, col)

                    bestVal = moveVal

    return bestMove
```

```
def printBoard(board):
```

```
    for row in board:

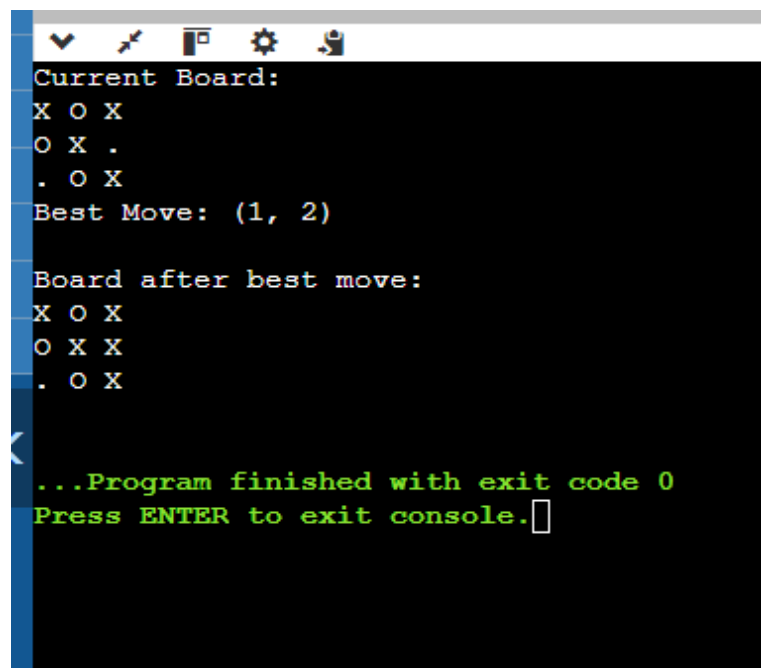
        print(" ".join(["X" if x == PLAYER_X else "O" if x == PLAYER_O else "." for x in row]))
```

```
board = [

    [PLAYER_X, PLAYER_O, PLAYER_X],
```

```
[PLAYER_O, PLAYER_X, EMPTY],  
[EMPTY, PLAYER_O, PLAYER_X]  
]  
  
print("Current Board:")  
  
printBoard(board)  
  
move = findBestMove(board)  
  
print(f"Best Move: {move}")  
  
board[move[0]][move[1]] = PLAYER_X  
  
print("\nBoard after best move:")  
  
printBoard(board)
```

OUTPUT:



```
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  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```