4/10/24

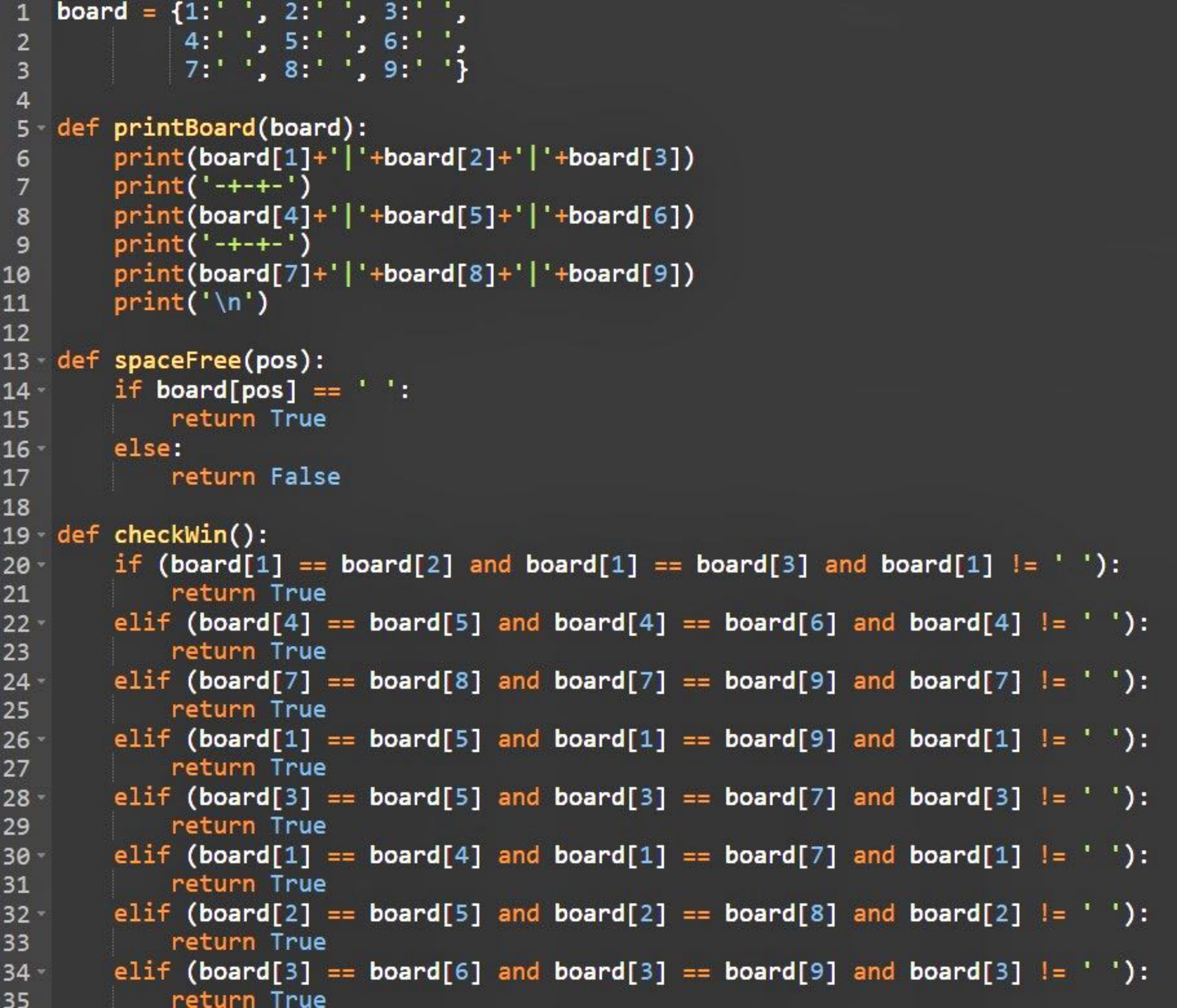
LAB-1

1) Implement a tic tac toe game using python

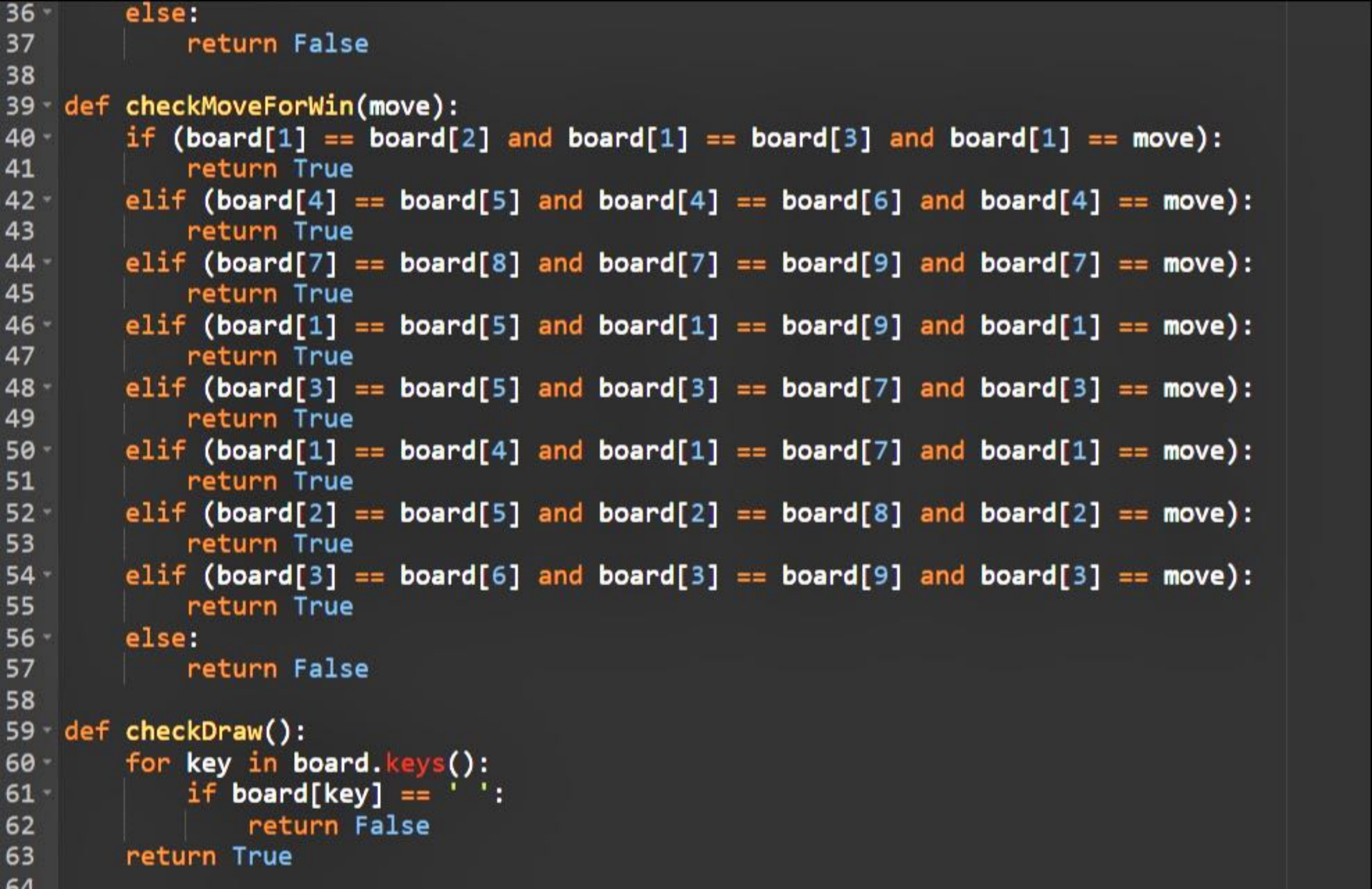
Pseudocode :-

```
function minimax (node, depth, isMaximizingPlayer):
    if node is a terminal state:
        return evaluate (node)

    if isMaximizingPlayer:
        bestValue = - infinity
        for each child in node:
            value = minimax (child, depth + 1, false)
            bestValue = max( bestValue, value)
        return bestValue

    else:
        bestValue = + infinity
        for each child in node:
            value = minimax (child, depth + 1, true)
            bestValue = min (bestValue, value)
        return bestValue
```

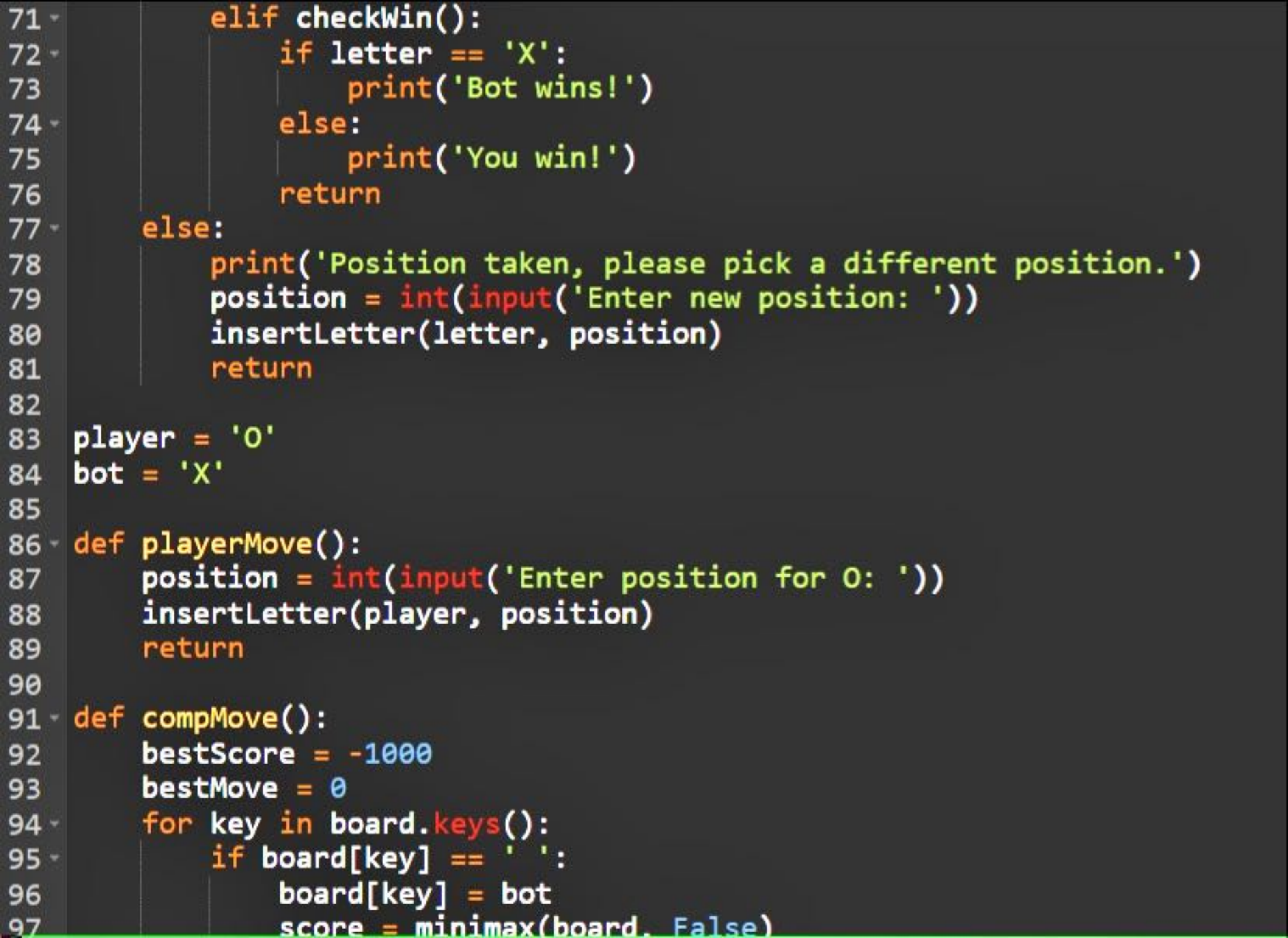
4/10/2024



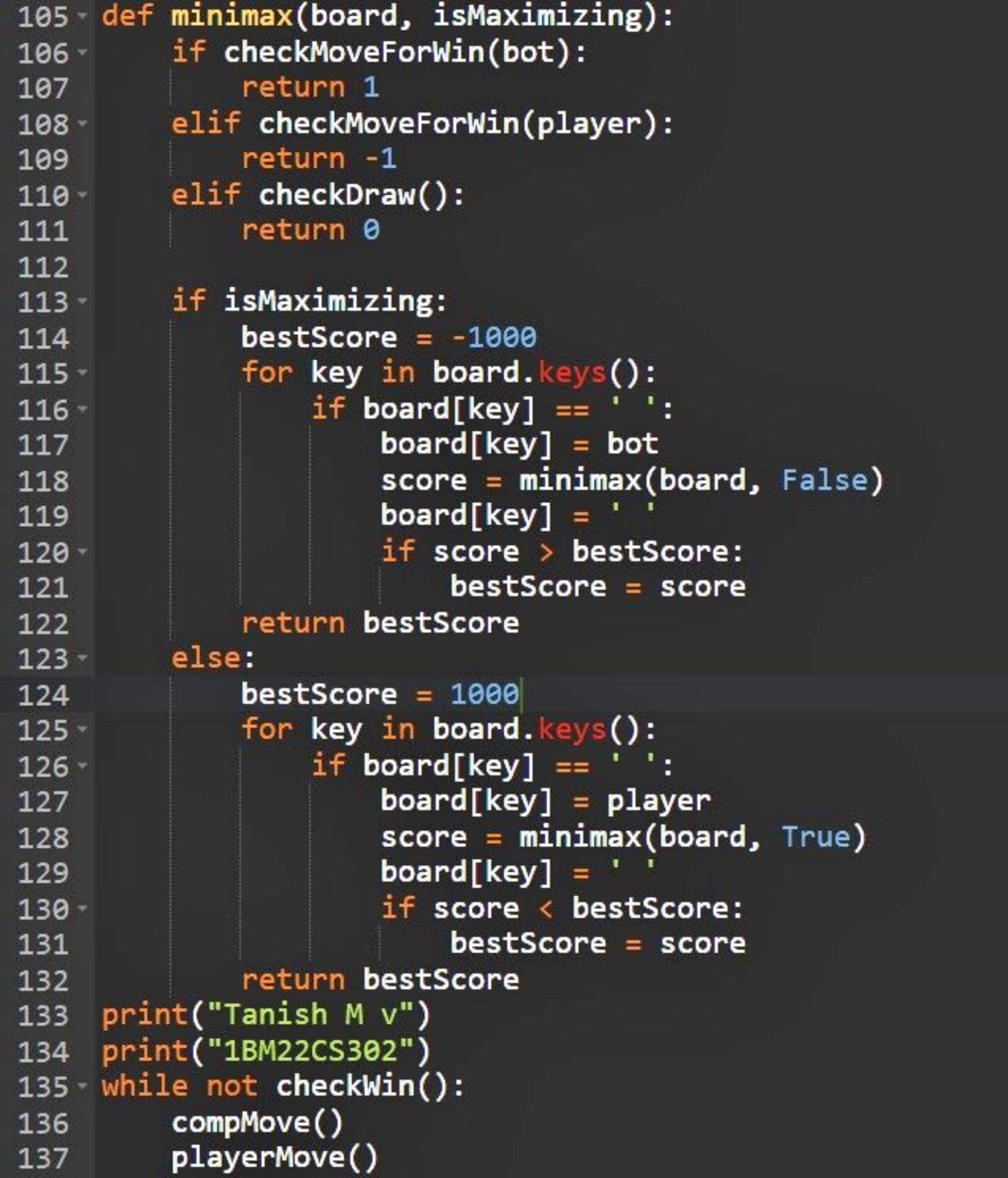
```python
board = {1:' ', 2:' ', 3:' ',
         4:' ', 5:' ', 6:' ',
         7:' ', 8:' ', 9:' '}

def printBoard(board):
    print(board[1]+'|'+board[2]+'|'+board[3])
    print('-+-+-')
    print(board[4]+'|'+board[5]+'|'+board[6])
    print('-+-+-')
    print(board[7]+'|'+board[8]+'|'+board[9])
    print('\n')

def spaceFree(pos):
    if board[pos] == ' ':
        return True
    else:
        return False

def checkWin():
    if (board[1] == board[2] and board[1] == board[3] and board[1] != ' '):
        return True
    elif (board[4] == board[5] and board[4] == board[6] and board[4] != ' '):
        return True
    elif (board[7] == board[8] and board[7] == board[9] and board[7] != ' '):
        return True
    elif (board[1] == board[5] and board[1] == board[9] and board[1] != ' '):
        return True
    elif (board[3] == board[5] and board[3] == board[7] and board[3] != ' '):
        return True
    elif (board[1] == board[4] and board[1] == board[7] and board[1] != ' '):
        return True
    elif (board[2] == board[5] and board[2] == board[8] and board[2] != ' '):
        return True
    elif (board[3] == board[6] and board[3] == board[9] and board[3] != ' '):
        return True
```



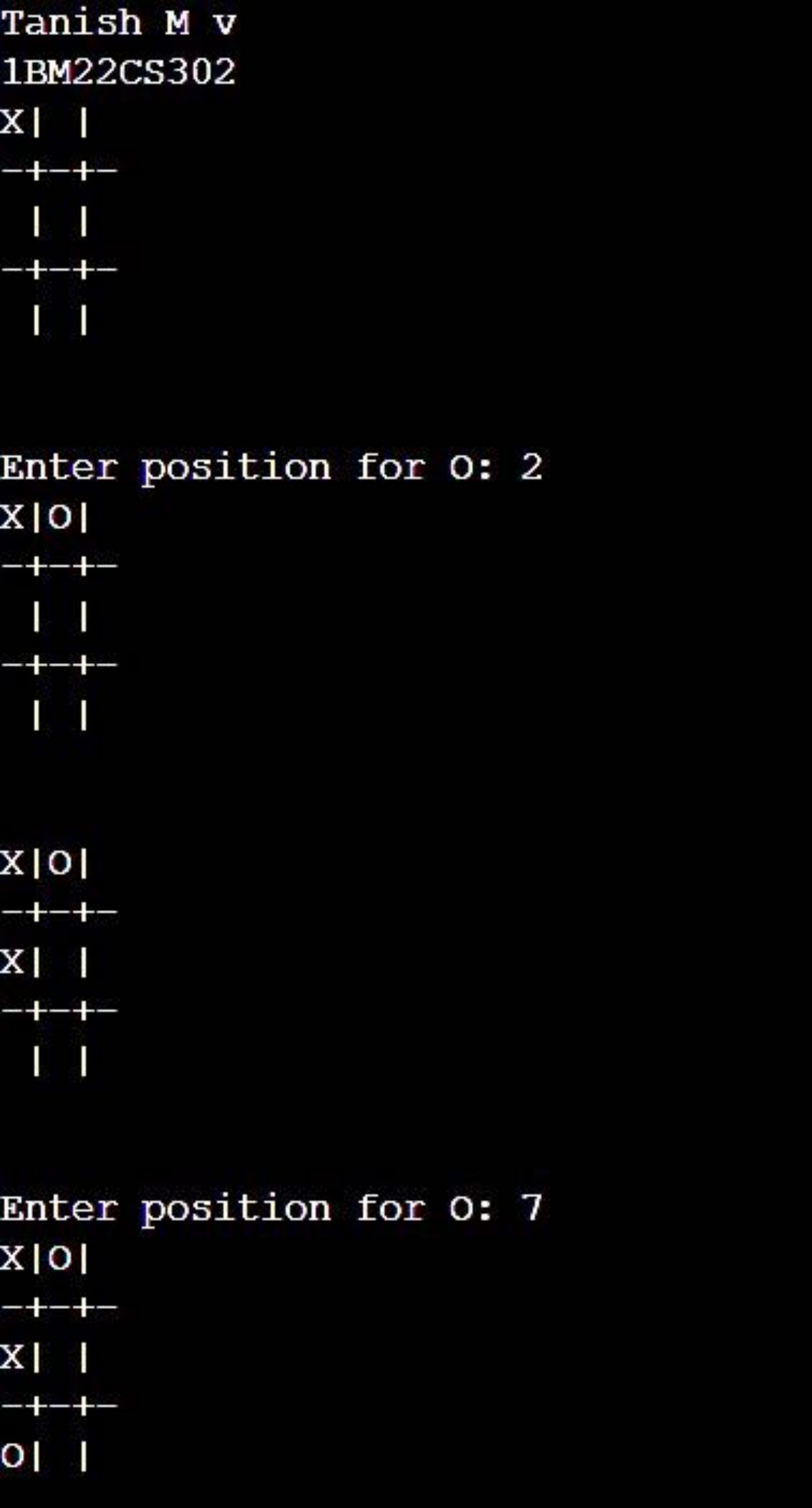
```
    else:
        return False

def checkMoveForWin(move):
    if (board[1] == board[2] and board[1] == board[3] and board[1] == move):
        return True
    elif (board[4] == board[5] and board[4] == board[6] and board[4] == move):
        return True
    elif (board[7] == board[8] and board[7] == board[9] and board[7] == move):
        return True
    elif (board[1] == board[5] and board[1] == board[9] and board[1] == move):
        return True
    elif (board[3] == board[5] and board[3] == board[7] and board[3] == move):
        return True
    elif (board[1] == board[4] and board[1] == board[7] and board[1] == move):
        return True
    elif (board[2] == board[5] and board[2] == board[8] and board[2] == move):
        return True
    elif (board[3] == board[6] and board[3] == board[9] and board[3] == move):
        return True
    else:
        return False

def checkDraw():
    for key in board.keys():
        if board[key] == ' ':
            return False
    return True
```



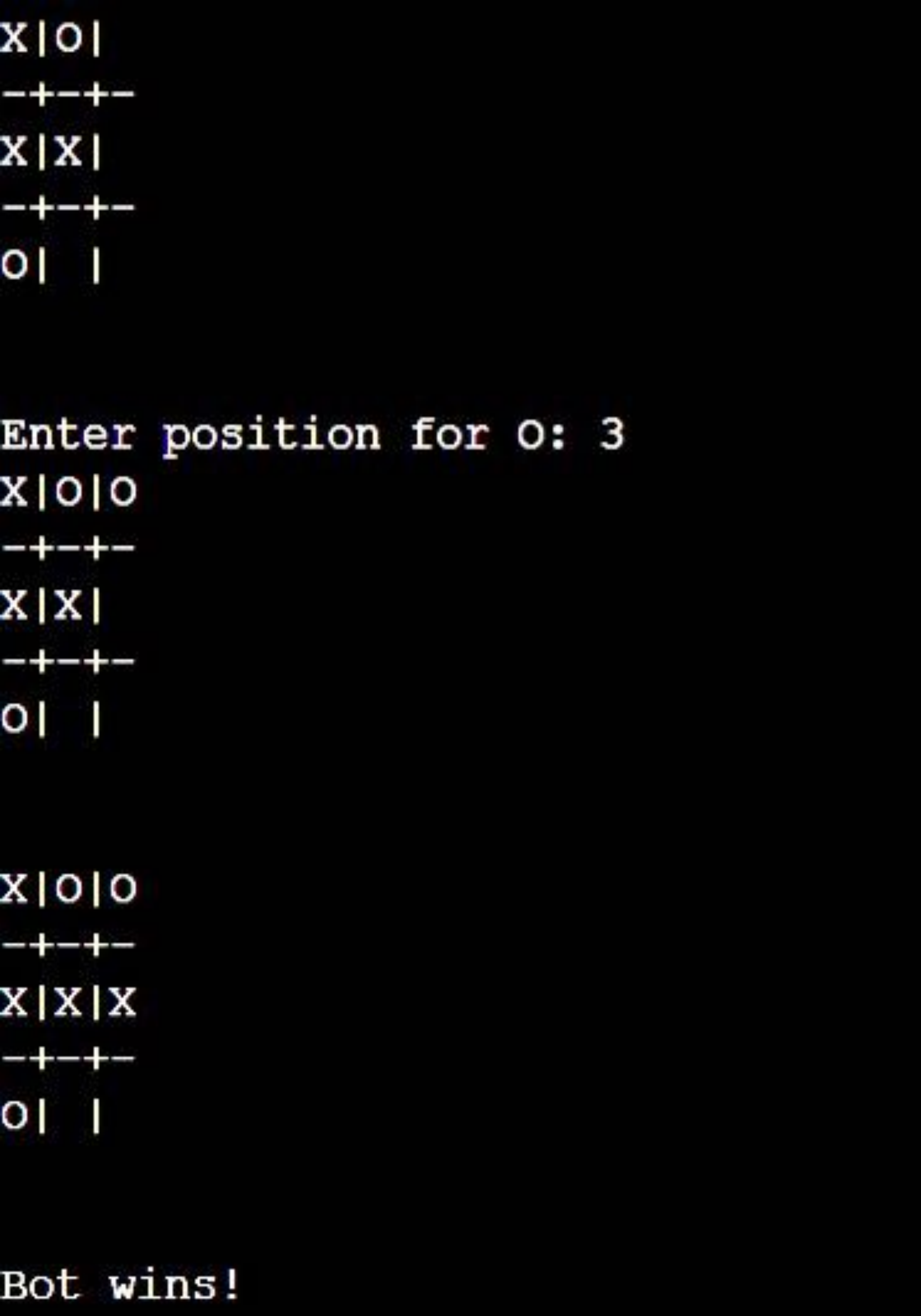
```
        elif checkWin():
            if letter == 'X':
                print('Bot wins!')
            else:
                print('You win!')
            return
    else:
        print('Position taken, please pick a different position.')
        position = int(input('Enter new position: '))
        insertLetter(letter, position)
        return

player = 'O'
bot = 'X'

def playerMove():
    position = int(input('Enter position for O: '))
    insertLetter(player, position)
    return

def compMove():
    bestScore = -1000
    bestMove = 0
    for key in board.keys():
        if board[key] == ' ':
            board[key] = bot
            score = minimax(board, False)
```



```python
def minimax(board, isMaximizing):
    if checkMoveForWin(bot):
        return 1
    elif checkMoveForWin(player):
        return -1
    elif checkDraw():
        return 0

    if isMaximizing:
        bestScore = -1000
        for key in board.keys():
            if board[key] == ' ':
                board[key] = bot
                score = minimax(board, False)
                board[key] = ' '
                if score > bestScore:
                    bestScore = score
        return bestScore
    else:
        bestScore = 1000
        for key in board.keys():
            if board[key] == ' ':
                board[key] = player
                score = minimax(board, True)
                board[key] = ' '
                if score < bestScore:
                    bestScore = score
        return bestScore
print("Tanish M v")
print("1BM22CS302")
while not checkWin():
    compMove()
    playerMove()
```



```
Tanish M v
1BM22CS302
X| |
-+-+-
 | |
-+-+-
 | |


Enter position for O: 2
X|O|
-+-+-
 | |
-+-+-
 | |


X|O|
-+-+-
X| |
-+-+-
 | |


Enter position for O: 7
X|O|
-+-+-
X| |
-+-+-
O| |
```



```
X|O|
-+-+-
X|X|
-+-+-
O| |


Enter position for O: 3
X|O|O
-+-+-
X|X|
-+-+-
O| |


X|O|O
-+-+-
X|X|X
-+-+-
O| |


Bot wins!
```