```
board = {1: ' ', 2: ' ', 3: ' ',
         4: ' ', 5: ' ', 6: ' ',
         7: ' ', 8: ' ', 9: ' '}
player = 'O'
computer = 'X'

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 spaceIsFree(position):
    if board[position] == ' ':
        return True
    return False

def insertLetter(letter, position):
    if spaceIsFree(position):
        board[position] = letter
        printBoard(board)
        if checkDraw():
            print("Draw!")
            exit()
        if checkWin():
            if letter == 'X':
                print("Bot wins!")
                exit()
            else:
                print("Player wins!")
                exit()
        return
    else:
        print("Invalid position")
        position = int(input("Please enter a new position: "))
        insertLetter(letter, position)
        return

def checkWin():
```

```
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[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
    elif (board[1] == board[5] and board[1] == board[9] and board[1] != ' '):
        return True
    elif (board[7] == board[5] and board[7] == board[3] and board[7] != ' '):
        return True
    else:
        return False

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

def checkDraw():
```

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

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

def compMove():
    bestScore = -800
    bestMove = 0
    for key in board.keys():
        if board[key] == ' ':
            board[key] = computer
            score = minimax(board, False)
            board[key] = ' '
            if score > bestScore:
                bestScore = score
                bestMove = key
    insertLetter(computer, bestMove)
    return
def minimax(board, isMaximizing):
    if checkWhichMarkWon(computer):
        return 1
    elif checkWhichMarkWon(player):
        return -1
    elif checkDraw():
        return 0

    if isMaximizing:
        bestScore = -800
        for key in board.keys():
            if board[key] == ' ':
                board[key] = computer
                score = minimax(board, False)
                board[key] = ' '
                if score > bestScore:
```

```
        return -1
    elif checkDraw():
        return 0

    if isMaximizing:
        bestScore = -800
        for key in board.keys():
            if board[key] == ' ':
                board[key] = computer
                score = minimax(board, False)
                board[key] = ' '
                if score > bestScore:
                    bestScore = score
        return bestScore
    else:
        bestScore = 800
        for key in board.keys():
            if board[key] == ' ':
                board[key] = player
                score = minimax(board, True)
                board[key] = ' '
                if score < bestScore:
                    bestScore = score
        return bestScore


while not checkWin():
    compMove()
    playerMove()
```

IDLE Shell 3.12.2

File Edit Shell Debug Options Window Help

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\NILA\AppData\Local\Programs\Python\Python312\tic toe.py
X| |
-+-+-
 | |
-+-+-
 | |


Enter a position for 'O': 4
X| |
-+-+-
O| |
-+-+-
 | |


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


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


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