```
############################################################
#                                                          #
#                        EXPERIMENT 5                      #
#              Tic Tac Toe Using Min Max Algorithm         #
#                   Nathan Cordeiro 22co09                 #
#                                                          #
############################################################

# -*- coding: utf-8 -*-
#Initial board setup

board = [' ',' ',' ',' ',' ',' ',' ',' ',' ',' ']

player = 1

Win = 1

Draw = -1

Running = 0

Stop = 1

Game = Running


def DrawBoard():

    """Function to draw the Tic-Tac-Toe board."""

    print(" %c | %c | %c " % (board[1], board[2], board[3]))

    print("___|___|___")

    print(" %c | %c | %c " % (board[4], board[5], board[6]))

    print("___|___|___")

    print(" %c | %c | %c " % (board[7], board[8], board[9]))

    print("   |   |   \n")



def CheckPosition(x):
```

```
    """Function to check if a position on the board is free."""
    if board[x] == ' ':
        return True
    else:
        return False


def CheckWin():
    """Function to check if there is a winner or a draw."""
    global Game
    #Horizontal Winning condition
    if(board[1] == board[2] and board[2] == board[3] and board[1]!=' '):
        Game = Win
    elif(board[4] == board[5] and board[5] == board[6] and board[4]!=' '):
        Game = Win
    elif(board[7] == board[8] and board[8] == board[9] and board[7]!=' '):
        Game = Win
    #Vertical Winning condition
    elif(board[1] == board[4] and board[4] == board[7] and board[1]!=' '):
        Game = Win
    elif(board[2] == board[5] and board[5] == board[8] and board[2]!=' '):
        Game = Win
    elif(board[3] == board[6] and board[6] == board[9] and board[3]!=' '):
        Game = Win
    #Diagonal Winning condition
    elif(board[1] == board[5] and board[5] == board[9] and board[5]!=' '):
```

```
    Game = Win
  elif(board[3] == board[5] and board[5] == board[7] and board[5]!=' '):
    Game = Win
  #Check for a draw
  elif(board[1] !=' ' and board[2] !=' ' and board[3] !=' ' and board[4] !=' ' and board[6] !=' ' and
board[7] !=' ' and board[8] !=' ' and board[9] !=' '):
    Game = Draw
  else:
    Game = Running


def Minimax(board, depth, isMaximizing):
  """Minimax algorithm to find the optimal move."""
  score = evaluate(board)
  #If a terminal state is found, return the score
  if score == 10 or score == -10:
    return score
  #If it's a draw, return 0
  if not any([space == ' ' for space in board[1:]]):
    return 0
  if isMaximizing:
    best = -1000
    for i in range(1, 10):
      if board[i] == ' ':
        board[i] = 'X'
        best = max(best, Minimax(board, depth + 1, not isMaximizing))
```

```
            board[i] = ' '
        return best
    else:
        best = 1000
        for i in range(1, 10):
            if board[i] == ' ':
                board[i] = 'O'
                best = min(best, Minimax(board, depth + 1, not isMaximizing))
                board[i] = ' '
        return best

def findBestMove(board):
    """Function to find the best move from the AI player."""
    bestVal = -1000
    bestMove = -1
    for i in range(1, 10):
            if board[i] == ' ':
                board[i] = 'X'
                moveVal = Minimax(board, 0, False)
                board[i] = ' '
                if moveVal > bestVal:
                    bestMove = i
                    bestVal = moveVal
    return bestMove
```

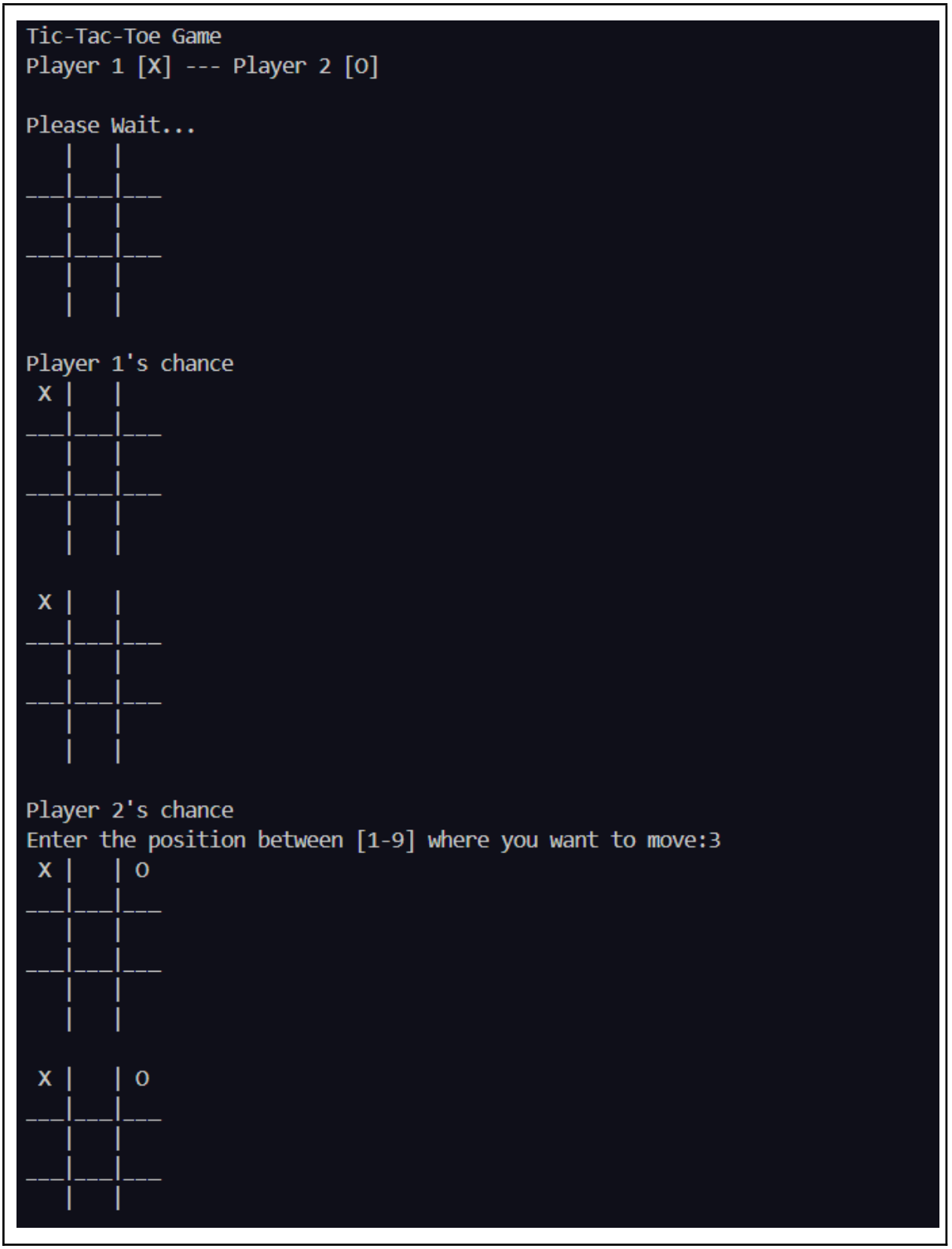
```
def evaluate(board):
    """Function to evaluate the board state and return a score."""
    #Horizontal Winning condition
    if(board[1] == board[2] and board[2] == board[3]):
        if board[1] == 'X':
            return 10
        elif board[1] == 'O':
            return -10
    if(board[4] == board[5] and board[5] == board[6]):
        if board[4] == 'X':
            return 10
        elif board[1] == 'O':
            return -10
    if(board[7] == board[8] and board[8] == board[9]):
        if board[1] == 'X':
            return 10
        elif board[1] == 'O':
            return -10
    #Vertical Winning condition
    if(board[1] == board[4] and board[4] == board[7]):
        if board[1] == 'X':
            return 10
        elif board[1] == 'O':
            return -10
```

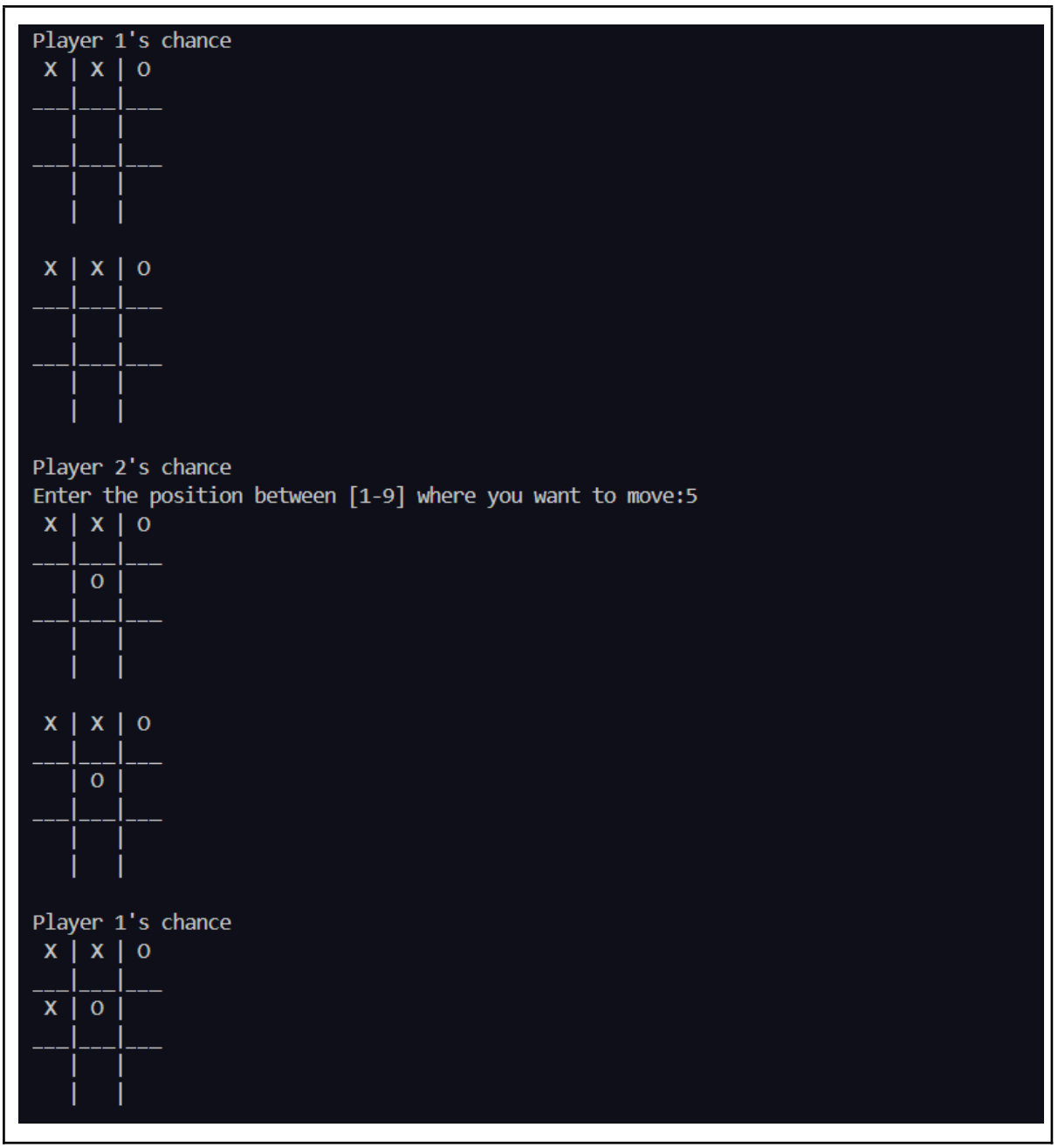
```
    if(board[2] == board[5] and board[5] == board[8]):
        if board[2] == 'X':
            return 10
        elif board[2] == 'O':
            return -10
    if(board[3] == board[6] and board[6] == board[9]):
        if board[3] == 'X':
            return 10
        elif board[3] == 'O':
            return -10
    #Diagonal Winning condition
    if(board[1] == board[5] and board[5] == board[9]):
        if board[1] == 'X':
            return 10
        elif board[1] == 'O':
            return -10
    if(board[3] == board[5] and board[5] == board[7]):
        if board[3] == 'X':
            return 10
        elif board[3] == 'O':
            return -10
    return 0
print("Tic-Tac-Toe Game")
print("Player 1 [X] --- Player 2 [O]\n")
print("Please Wait...")
```

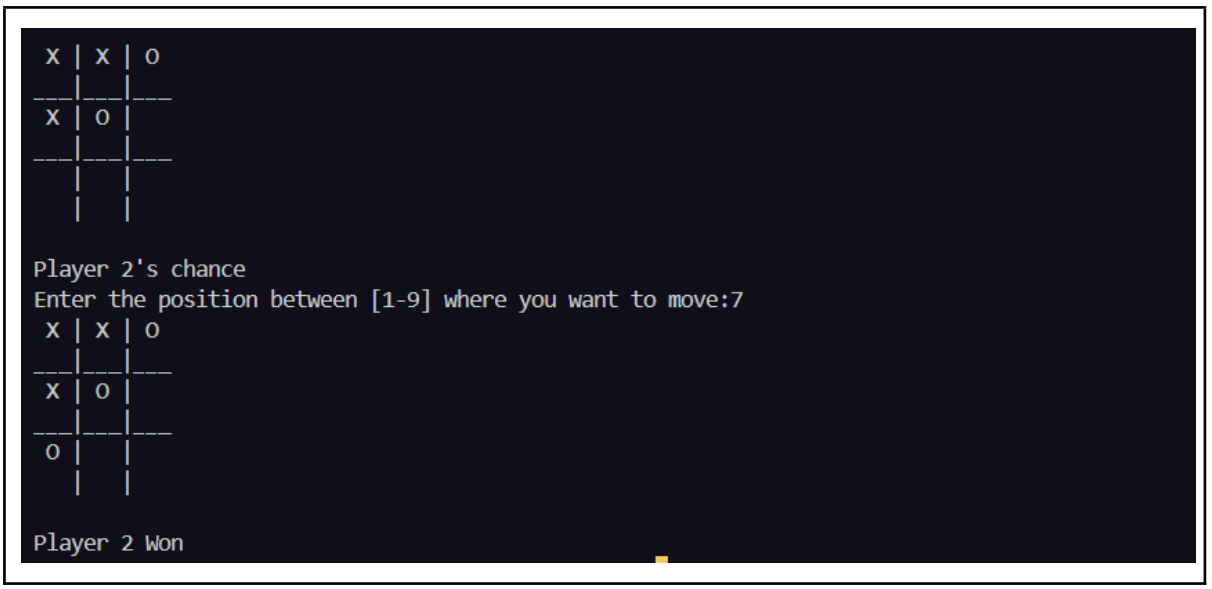
```
#Main game loop

while Game == Running:

    DrawBoard()

    if player % 2 != 0:

        print("Player 1's chance")

        Mark = 'X'

        choice = findBestMove(board)

    else:

        print("Player 2's chance")

        Mark = 'O'

        choice = int(input("Enter the position between [1-9] where you want to move:"))


    if CheckPosition(choice):

        board[choice] = Mark

        player += 1

        CheckWin()


    DrawBoard()

    if Game == Draw:

        print("Game Draw")

    elif Game == Win:

        player -= 1

        if player % 2 != 0:

            print("Player 1 Won")
```

```
    else:
        print("Player 2 Won")
```

OUTPUT:


Tic-Tac-Toe Game
Player 1 [X] --- Player 2 [O]
Please Wait...
Player 1's chance
Player 2's chance
Enter the position between [1-9] where you want to move:3


Player 1's chance
X | X | O
Player 2's chance
Enter the position between [1-9] where you want to move:5
Player 1's chance


X | X | O
X | O |
Player 2's chance
Enter the position between [1-9] where you want to move:7
X | X | O
X | O |
O |
Player 2 Won