```
#Assignment no: 5
#Name: Samiksha Bandgar
#Roll_no: 3307
# Tic Tac Toe (Using Minmax Algorithm)

import math
# Game board
board = [[' ', ' ', ' '],
         [' ', ' ', ' '],
         [' ', ' ', ' ']]

# Function to print the game board
def print_board(board):
    print('-------------')
    for row in board:
        print('| ' + ' | '.join(row) + ' |')
        print('-------------')

# Function to check if a player has won
def check_win(board, player):
    for i in range(3):
        if (board[i][0] == player and board[i][1] == player and
board[i][2] == player) or \
           (board[0][i] == player and board[1][i] == player and
board[2][i] == player) or \
           (board[0][0] == player and board[1][1] == player and
board[2][2] == player) or \
           (board[2][0] == player and board[1][1] == player and
board[0][2] == player):
            return True
    return False

# Function to check if the game has ended
def check_game_over(board):
    if check_win(board, 'X'):
        return 'X'
    elif check_win(board, 'O'):
        return 'O'
    elif sum(row.count(' ') for row in board) == 0:
        return 'Tie'
    else:
       return None

# Function to evaluate the board
def evaluate_board(board):
    if check_win(board, 'X'):
        return 1
    elif check_win(board, 'O'):
        return -1
    else:
        return 0

# Minimax algorithm

def minimax(board, depth, is_maximizing_player):
    result = check_game_over(board)
    if result is not None:
        return evaluate_board(board)
    if is_maximizing_player:
        best_score = -math.inf
        for i in range(3):
            for j in range(3):
                if board[i][j] == ' ':
                    board[i][j] = 'X'
                    score = minimax(board, depth + 1, False)
                    board[i][j] = ' '
                    best_score = max(best_score, score)

        return best_score
    else:
        best_score = math.inf
        for i in range(3):
            for j in range(3):
                if board[i][j] == ' ':
                    board[i][j] = 'O'
                    score = minimax(board, depth + 1, True)
                    board[i][j] = ' '
                    best_score = min(best_score, score)
        return best_score

# Function to get the best move using the Minimax algorithm
```

```
    def get_best_move(board):
        best_score = -math.inf
        best_move = None
        for i in range(3):
            for j in range(3):
                if board[i][j] == ' ':
                    board[i][j] = 'X'
                    score = minimax(board, 0, False)
                    board[i][j] = ' '
                    if score > best_score:
                        best_score = score
                        best_move = (i, j)
            return best_move

    # Play the game
    # Play the game
    print_board(board)
    while True:
    # Player's move
        while True:
          row_input = input('Enter row number (1-3): ')
          if row_input.isdigit() and 1 <= int(row_input) <= 3:
              row = int(row_input) - 1
              break
          else:
              print('Invalid input. Please enter a number between 1 and 3.')
        while True:
            col_input = input('Enter column number (1-3): ')
            if col_input.isdigit() and 1 <= int(col_input) <= 3:
                col = int(col_input) - 1
                break
            else:
              print('Invalid input. Please enter a number between 1 and 3.')
        if board[row][col] != ' ':
            print('Invalid move. Please try again.')
            continue
        board[row][col] = 'O'
    # Check if the game is over
        result = check_game_over(board)
        if result is not None:
            print_board(board)
            if result == 'Tie':
                print('The game is a tie!')
            else:
                print('You win!')
            break
    # Computer's move
        row, col = get_best_move(board)
        board[row][col] = 'X'
    # Check if the game is over
        result = check_game_over(board)
        if result is not None:
            print_board(board)
            if result == 'Tie':
                print('The game is a tie!')
            else:
              print('You lose!')
            break
    # Print the updated game board
        print_board(board)
```

```
 -------------
 |   |   |   |
 -------------
 |   |   |   |
 -------------
 |   |   |   |
 -------------
 Enter row number (1-3): 2
 Enter column number (1-3): 1
 -------------
 | X |   |   |
 -------------
 | O |   |   |
 -------------
 |   |   |   |
 -------------
 Enter row number (1-3): 2
 Enter column number (1-3): 3
 -------------
 | X | X |   |
 -------------
 | O |   | O |
 -------------
```

```
|   |   |   |
-------------
Enter row number (1-3): 2
Enter column number (1-3): 2
-------------
| X | X |   |
-------------
| O | O | O |
-------------
|   |   |   |
-------------
You win!
```