भारतीय सूचना प्रौद्योगिकी संस्थान गुवाहाटी
Indian Institute of Information Technology Guwahati

CS236 - Artificial Intelligence Lab
Assignment - 10

Problem Statement

Write a Python program to solve the **Tic-Tac-Toe** game using the **minimax algorithm**. The program should allow a human player to play against an AI that uses the minimax algorithm to make optimal moves. Use a basic text editor such as Notepad, which does not support code highlighting or auto-complete, to write your code. The program must:

1. Accept user input to choose whether to play as ‘X’ or ‘O’.
2. Implement the minimax algorithm to determine the AI’s moves, ensuring optimal play.
3. Display the game board after each move in a user-friendly format.
4. Handle user input for moves, including validation for invalid or occupied cells.
5. Detect when the game ends and announce the winner or declare a draw.
6. Include a function `get_best_move(board, player)` that returns the best move for a given player on a given board state, which will be tested separately.

Algorithm Specifications

1. **Game Representation:**
   i. Represent the Tic-Tac-Toe board as a 3×3 grid using a list of lists (or equivalent structure), with cells containing ‘X’, ‘O’, or ‘ ’ (space for empty).
   ii. Use 0-based indexing internally, but accept 1-based indexing (rows and columns numbered 1 to 3) for user input.
2. **Minimax Algorithm:**
   i. Implement the minimax algorithm to evaluate all possible game outcomes recursively.
   ii. Assign scores: +1 for a win by ‘X’, -1 for a win by ‘O’, and 0 for a draw.
   iii. The AI maximizes its score if playing as ‘X’ (maximizing player) or minimizes it if playing as ‘O’ (minimizing player), assuming the opponent plays optimally.
3. **Game Flow:**

   i. Prompt the user to choose their symbol (‘X’ or ‘O’). If ‘X’, the user plays first; if ‘O’, the AI plays first as ‘X’.
   ii. Alternate between human and AI moves until the game ends.
   iii. After each move, display the updated board and check for a win or draw.

4. **User Input:**

   i. Prompt the user to enter their move as ”row column” (e.g., ”1 2” for row 1, column 2).
   ii. Validate input to ensure the move is within bounds (1 to 3) and the cell is empty.

5. **Best Move Function:**

   i. Implement `get_best_move(board, player)` that returns the best move as a tuple (row, col) using the minimax algorithm.
   ii. Assume the board is not in a terminal state when this function is called.

Test Cases for `get_best_move`

Ensure that `get_best_move(board, player)` returns the correct move for the following test cases:

1. **Test Case 1:**

```
Board:
X | X |
---------
O | O |
---------
  |   |
Player: X
Expected move: (0,2) [to win the game]
```

2. **Test Case 2:**

```
Board:
X | O |
---------
X | O |
---------
  |   | O
Player: X
Expected move: (2,0) [to win the game]
```

3. **Test Case 3:**

```
Board:
X | X |
---------
  | O |
---------
  |   | O
Player: O
Expected move: (0,2) [to block X's win]
```

Note: Coordinates are 0-based internally, but test cases reflect the board state visually. There may be multiple optimal moves; any correct one is acceptable.

Sample Interaction

Below is a sample interaction of the program:

```
Welcome to Tic-Tac-Toe!
Do you want to play as X or O? X
You are X, I am O.
Current board:
  |   |
---------
  |   |
---------
  |   |
Enter your move (row column): 1 1
You played at (1,1)
Current board:
X |   |
---------
  |   |
---------
  |   |
I play at (2,2)
Current board:
X |   |
---------
  | O |
---------
  |   |
Enter your move (row column): 3 3
You played at (3,3)
Current board:
X |   |
---------
  | O |
---------
```

```
  |   | X
I play at (1,2)
Current board:
X | O |
---------
  | O |
---------
  |   | X
Enter your move (row column): 2 1
You played at (2,1)
Current board:
X | O |
---------
X | O |
---------
  |   | X
I play at (3,1)
Current board:
X | O |
---------
X | O |
---------
O |   | X
Enter your move (row column): 1 3
You played at (1,3)
Current board:
X | O | X
---------
X | O |
---------
O |   | X
I play at (2,3)
Current board:
X | O | X
---------
X | O | O
---------
O |   | X
Enter your move (row column): 3 2
You played at (3,2)
Current board:
X | O | X
---------
X | O | O
---------
O | X | X
It's a draw!
```