**Name:** Aqsa Gillani  
**Roll No:** 064

Title: Explanation of Tic-Tac-Toe Minimax Algorithm in Python

1. **Board Initialization:**
   * A list board of 9 spaces represents the Tic-Tac-Toe board.
   * Each index (0-8) corresponds to a cell on the 3x3 grid.
2. **Printing the Board:**
   * print\_board(board) prints the current state of the board in 3 rows.
   * Helps visualize moves made by the player and AI.
3. **Checking for a Winner:**
   * is\_winner(board, player) checks if the given player ('X' for human, 'O' for AI) has won.
   * Uses predefined winning combinations (rows, columns, diagonals).
   * Returns True if any winning condition is met.
4. **Checking if Board is Full:**
   * is\_board\_full(board) returns True if no empty spaces remain, indicating a tie.
5. **Minimax Algorithm:**
   * minimax(board, depth, is\_maximizing) is a recursive function that calculates the best possible score for AI or human.
   * **Base Cases:**
     + If AI has won, returns 1.
     + If Human has won, returns -1.
     + If the board is full (tie), returns 0.
   * **Recursion:**
     + If is\_maximizing is True, AI tries to maximize score.
     + If is\_maximizing is False, Human tries to minimize score.
     + It simulates all possible moves, recursively calls itself, and updates best\_score accordingly.
6. **AI Move Function:**
   * best\_move(board) finds the optimal move for AI.
   * Iterates over all empty cells, calls minimax for each move, and selects the cell with the highest score.
7. **Main Game Loop:**
   * play\_game() handles the game flow:
     + Prints the board.
     + Asks human player to enter a move.
     + Updates board and checks for winner or tie.
     + Calls best\_move() for AI's turn.
     + Continues until there is a winner or tie.
8. **Game Outcome:**
   * Prints the final board and announces whether the human wins, AI wins, or the game is a tie.

**Summary:**

* The code implements a simple Tic-Tac-Toe game where a human plays against an AI.
* The AI uses the Minimax algorithm to make optimal decisions.
* The recursive Minimax ensures the AI never loses and plays strategically to win or tie.