```
import random

# Represent the board as a list of 9 elements
board = [' ' for _ in range(9)]

# Winning combinations
WINNING_COMBOS = [
    [0, 1, 2], [3, 4, 5], [6, 7, 8],  # Rows
    [0, 3, 6], [1, 4, 7], [2, 5, 8],  # Columns
    [0, 4, 8], [2, 4, 6]  # Diagonals
]

def print_board():
    for i in range(0, 9, 3):
        print(f' {board[i]} | {board[i+1]} | {board[i+2]} ')
        if i < 6:
            print('-----------')

def is_winner(player):
    for combo in WINNING_COMBOS:
        if all(board[i] == player for i in combo):
            return True
    return False

def is_board_full():
    return ' ' not in board

def get_empty_cells():
    return [i for i, cell in enumerate(board) if cell == ' ']

def minimax(is_maximizing):
    if is_winner('O'):
        return 1
    if is_winner('X'):
        return -1
    if is_board_full():
        return 0

    if is_maximizing:
        best_score = float('-inf')
        for move in get_empty_cells():
            board[move] = 'O'
            score = minimax(False)
            board[move] = ' '
            best_score = max(score, best_score)
        return best_score
    else:
        best_score = float('inf')
        for move in get_empty_cells():
            board[move] = 'X'
            score = minimax(True)
            board[move] = ' '
            best_score = min(score, best_score)
        return best_score

def get_best_move():
    best_score = float('-inf')
    best_move = None
    for move in get_empty_cells():
        board[move] = 'O'
        score = minimax(False)
        board[move] = ' '
        if score > best_score:
            best_score = score
            best_move = move
    return best_move

def play_game():
    print("Welcome to Tic-Tac-Toe! You are X, and the AI is O.")
    print_board()

    while True:
        # Human's turn
        while True:
            try:
                move = int(input("Enter your move (0-8): "))
                if move not in get_empty_cells():
                    raise ValueError
                break
            except ValueError:
                print("Invalid move. Try again.")
```

```
            board[move] = 'X'
            print_board()

            if is_winner('X'):
                print("You win! Congratulations!")
                break
            if is_board_full():
                print("It's a tie!")
                break

            # AI's turn
            print("AI is thinking...")
            ai_move = get_best_move()
            board[ai_move] = 'O'
            print_board()

            if is_winner('O'):
                print("AI wins! Better luck next time.")
                break
            if is_board_full():
                print("It's a tie!")
                break

if __name__ == "__main__":
    play_game()
```

```
Welcome to Tic-Tac-Toe! You are X, and the AI is O.
   |   |
-----------
   |   |
-----------
   |   |
Enter your move (0-8): 4
   |   |
-----------
   | X |
-----------
   |   |
AI is thinking...
 O |   |
-----------
   | X |
-----------
   |   |
Enter your move (0-8): 6
 O |   |
-----------
   | X |
-----------
 X |   |
AI is thinking...
 O |   | O
-----------
   | X |
-----------
 X |   |
Enter your move (0-8): 4
Invalid move. Try again.
Enter your move (0-8): 7
 O |   | O
-----------
   | X |
-----------
 X | X |
AI is thinking...
 O | O | O
-----------
   | X |
-----------
 X | X |
AI wins! Better luck next time.
```