```
# Define the board as a list of 9 elements, representing the 3x3 grid
board = [" " for _ in range(9)]

# Function to print the board
def print_board(board):
    print("-------------")
    for i in range(3):
        print(f"| {board[i*3]} | {board[i*3+1]} | {board[i*3+2]} |")
        print("-------------")

# Function to check if the board is full
def is_board_full(board):
    return " " not in board

# Function to check if a player has won
def check_winner(board, player):
    # Check rows, columns, and diagonals
    win_conditions = [
        [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
    ]
    for condition in win_conditions:
        if all(board[i] == player for i in condition):
            return True
    return False
    # Minimax algorithm with Alpha-Beta Pruning
def minimax(board, depth, is_maximizing, alpha, beta):
    if check_winner(board, "X"):
        return -10 + depth
    if check_winner(board, "O"):
        return 10 - depth
    if is_board_full(board):
        return 0

    if is_maximizing:
        best_score = -float('inf')
        for i in range(9):
            if board[i] == " ":
                board[i] = "O"
                score = minimax(board, depth + 1, False, alpha, beta)
                board[i] = " "
                best_score = max(score, best_score)
                alpha = max(alpha, best_score)
                if beta <= alpha:
                    break
        return best_score
    else:
        best_score = float('inf')
        for i in range(9):
            if board[i] == " ":
                board[i] = "X"
                score = minimax(board, depth + 1, True, alpha, beta)
                board[i] = " "
                best_score = min(score, best_score)
                beta = min(beta, best_score)
                if beta <= alpha:
                    break
        return best_score
        # Function to determine the best move for the AI
def ai_move(board):
    best_score = -float('inf')
    best_move = None
    for i in range(9):
        if board[i] == " ":
            board[i] = "O"
            score = minimax(board, 0, False, -float('inf'), float('inf'))
            board[i] = " "
            if score > best_score:
                best_score = score
                best_move = i
    return best_move
    # Main game loop
def play_game():
    print("Welcome to Tic-Tac-Toe!")
    print_board(board)

    while True:
        # Human player's turn
        human_move = int(input("Enter your move (0-8): "))
        if board[human_move] != " ":
            print("Invalid move! Try again.")
            continue
        board[human_move] = "X"
        print_board(board)

        if check_winner(board, "X"):
            print("You win!")
            break
        if is_board_full(board):
            print("It's a tie!")
            break

        # AI's turn
        print("AI is making a move...")
        ai_move_index = ai_move(board)
        board[ai_move_index] = "O"
        print_board(board)

        if check_winner(board, "O"):
            print("AI wins!")
            break
        if is_board_full(board):
            print("It's a tie!")
            break

# Start the game
play_game()
```

```
Welcome to Tic-Tac-Toe!
-------------
|   |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------
Enter your move (0-8): 2
-------------
|   |   | X |
-------------
|   |   |   |
-------------
|   |   |   |
-------------
AI is making a move...
-------------
|   |   | X |
-------------
|   | O |   |
-------------
|   |   |   |
-------------
Enter your move (0-8): 5
-------------
|   |   | X |
-------------
|   | O | X |
-------------
|   |   |   |
-------------
AI is making a move...
-------------
|   |   | X |
-------------
|   | O | X |
-------------
|   |   | O |
-------------
Enter your move (0-8): 0
-------------
| X |   | X |
-------------
|   | O | X |
-------------
```

```
|   |   | O |
-------------
AI is making a move...
-------------
| X | O | X |
-------------
|   | O | X |
-------------
|   |   | O |
-------------
Enter your move (0-8): 7
```