```
import math

# Constants
HUMAN = 'O'
AI = 'X'
EMPTY = ' '

board = [[EMPTY for _ in range(3)] for _ in range(3)]

def print_board(b):
    for row in b:
        print('|'.join(row))
        print('-'*5)

def is_moves_left(b):
    for row in b:
        if EMPTY in row:
            return True
    return False

def evaluate(b):

    for row in b:
        if row.count(AI) == 3:
            return 10
        if row.count(HUMAN) == 3:
            return -10
```

```
    for col in range(3):
        if all(b[row][col] == AI for row in range(3)):
            return 10
        if all(b[row][col] == HUMAN for row in range(3)):
            return -10



    if all(b[i][i] == AI for i in range(3)):
        return 10
    if all(b[i][i] == HUMAN for i in range(3)):
        return -10

    if all(b[i][2 - i] == AI for i in range(3)):
        return 10
    if all(b[i][2 - i] == HUMAN for i in range(3)):
        return -10

    return 0



def minimax(b, depth, is_maximizing):
    score = evaluate(b)

    if score == 10 or score == -10:
        return score

    if not is_moves_left(b):
        return 0
```

```
    if is_maximizing:
        best = -math.inf
        for i in range(3):
            for j in range(3):
                if b[i][j] == EMPTY:
                    b[i][j] = AI
                    value = minimax(b, depth + 1, False)
                    best = max(best, value)
                    b[i][j] = EMPTY
        return best
    else:
        best = math.inf
        for i in range(3):
            for j in range(3):
                if b[i][j] == EMPTY:
                    b[i][j] = HUMAN
                    value = minimax(b, depth + 1, True)
                    best = min(best, value)
                    b[i][j] = EMPTY
        return best


def find_best_move(b):
    best_val = -math.inf
    best_move = (-1, -1)

    for i in range(3):
        for j in range(3):
            if b[i][j] == EMPTY:
                b[i][j] = AI
                move_val = minimax(b, 0, False)
```

```
            b[i][j] = EMPTY

            if move_val > best_val:
                best_val = move_val
                best_move = (i, j)

    return best_move


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

    while is_moves_left(board) and evaluate(board) == 0:

        try:
            row, col = map(int, input("Enter your move (row and column 0-2): ").split())
            if board[row][col] != EMPTY:
                print("Cell already taken! Try again.")
                continue
        except:
            print("Invalid input! Enter two numbers between 0 and 2.")
            continue

        board[row][col] = HUMAN
        print_board(board)

        if evaluate(board) != 0 or not is_moves_left(board):
            break
```

```
            print("AI is making a move...")
            ai_row, ai_col = find_best_move(board)
            board[ai_row][ai_col] = AI
            print_board(board)

        score = evaluate(board)
        if score == 10:
            print("AI wins!")
        elif score == -10:
            print("You win!")
        else:
            print("It's a draw!")


play_game()
```