```
import math

def print_board(board):
    print("\n".join([" | ".join(row) for row in board]))
    print()

def check_winner(board):

    lines = board + [list(col) for col in zip(*board)]
    lines.append([board[i][i] for i in range(3)])
    lines.append([board[i][2 - i] for i in range(3)])

    for line in lines:
        if line.count(line[0]) == 3 and line[0] != " ":
            return line[0]
    return None

def is_full(board):
    return all(cell != " " for row in board for cell in row)

def minimax(board, depth, is_maximizing, alpha, beta, ai_symbol, human_symbol):
    winner = check_winner(board)
    if winner == ai_symbol:
        return 1
    elif winner == human_symbol:
        return -1
    elif is_full(board):
        return 0

    if is_maximizing:
        max_eval = -math.inf
        for i in range(3):
            for j in range(3):
                if board[i][j] == " ":
                    board[i][j] = ai_symbol
                    eval = minimax(board, depth + 1, False, alpha, beta, ai_symbol, human_symbol)
                    board[i][j] = " "
                    max_eval = max(max_eval, eval)
                    alpha = max(alpha, eval)
                    if beta <= alpha:
                        break
        return max_eval
    else:
        min_eval = math.inf
        for i in range(3):
            for j in range(3):
                if board[i][j] == " ":
                    board[i][j] = human_symbol
                    eval = minimax(board, depth + 1, True, alpha, beta, ai_symbol, human_symbol)
                    board[i][j] = " "
                    min_eval = min(min_eval, eval)
                    beta = min(beta, eval)
                    if beta <= alpha:
                        break
        return min_eval

def best_move(board, ai_symbol, human_symbol):
    best_score = -math.inf
    move = None
    for i in range(3):
        for j in range(3):
            if board[i][j] == " ":
                board[i][j] = ai_symbol
                score = minimax(board, 0, False, -math.inf, math.inf, ai_symbol, human_symbol)
                board[i][j] = " "
                if score > best_score:
                    best_score = score
                    move = (i, j)
    return move

def play_game():
    board = [[" " for _ in range(3)] for _ in range(3)]
    human_symbol = input("Choose your symbol (X/O): ").upper()
    ai_symbol = "O" if human_symbol == "X" else "X"
    turn = "X"

    print_board(board)

    while True:
        if turn == human_symbol:
            try:
```

```
                row = int(input("Enter row (0-2): "))
                col = int(input("Enter col (0-2): "))
                if board[row][col] != " ":
                    print("Cell already taken!")
                    continue
                board[row][col] = human_symbol
            except (ValueError, IndexError):
                print("Invalid input. Please enter numbers from 0 to 2.")
                continue
        else:
            print("AI is making a move...")
            row, col = best_move(board, ai_symbol, human_symbol)
            board[row][col] = ai_symbol

        print_board(board)
        winner = check_winner(board)
        if winner:
            print(f"{winner} wins!")
            break
        if is_full(board):
            print("It's a draw!")
            break

        turn = ai_symbol if turn == human_symbol else human_symbol

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

```
Choose your symbol (X/O): o
  |   |
  |   |
  |   |

AI is making a move...
X |   |
  |   |
  |   |

Enter row (0-2): 2
Enter col (0-2): 2
X |   |
  |   |
  |   | O

AI is making a move...
X |   | X
  |   |
  |   | O

Enter row (0-2): 0
Enter col (0-2): 1
X | O | X
  |   |
  |   | O

AI is making a move...
X | O | X
  |   |
X |   | O

Enter row (0-2): 1
Enter col (0-2): 0
X | O | X
O |   |
X |   | O

AI is making a move...
X | O | X
O | X |
X |   | O

X wins!
```