import math

# Function to print the board def print\_board(board):

for row in board: print("|".join(row))

print("-" \* 5)

# Function to check if a player has won def check\_win(board, player):

# Check rows, columns and diagonals for i in range(3):

if all(board[i][j] == player for j in range(3)): return True

if all(board[j][i] == player for j in range(3)): return True

if all(board[i][i] == player for i in range(3)): return True

if all(board[i][2 - i] == player for i in range(3)): return True

return False

# Function to check for draw def is\_draw(board):

return all(cell != ' ' for row in board for cell in row)

# Evaluation function def evaluate(board):

if check\_win(board, 'X'): return +1

elif check\_win(board, 'O'): return -1

else:

return 0

# Minimax function

def minimax(board, depth, isMax): score = evaluate(board)

# Terminal states if score == 1: return score

if score == -1: return score

if is\_draw(board):

return 0

if isMax:

best = -math.inf for i in range(3):

for j in range(3):

if board[i][j] == ' ':

board[i][j] = 'X'

best = max(best, minimax(board, depth + 1, False)) board[i][j] = ' '

return best else:

best = math.inf for i in range(3):

for j in range(3):

if board[i][j] == ' ':

board[i][j] = 'O'

best = min(best, minimax(board, depth + 1, True)) board[i][j] = ' '

return best

# Function to find the best move for the AI def find\_best\_move(board):

best\_val = -math.inf best\_move = (-1, -1)

for i in range(3):

for j in range(3):

if board[i][j] == ' ':

board[i][j] = 'X'

move\_val = minimax(board, 0, False) board[i][j] = ' '

if move\_val > best\_val: best\_move = (i, j) best\_val = move\_val

return best\_move

# Main Function def main():

board = [ ['X', 'O', 'X'],

['O', 'O', ' '],

[' ', 'X', ' ']

]

print("Current Board:") print\_board(board)

best\_move = find\_best\_move(board)

print(f"\nThe Optimal Move is : Row = {best\_move[0]}, Column = {best\_move[1]}")

board[best\_move[0]][best\_move[1]] = 'X' print("\nBoard After AI Move:") print\_board(board)

if name == " main ": main()