TASK-2

def print\_board(board):

print(" 1 2 3")

for i, row in enumerate(board, 1):

print(f"{i} {' | '.join(row)}")

if i < 3:

print(" ---------")

def is\_winner(board, player):

# Check rows and columns for a win

for i in range(3):

if all(board[i][j] == player for j in range(3)) or \

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

return True

# Check diagonals for a win

if all(board[i][i] == player for i in range(3)) or \

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

return True

return False

def is\_full(board):

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

def get\_empty\_cells(board):

return [(i, j) for i in range(3) for j in range(3) if board[i][j] == ' ']

def minimax(board, depth, is\_maximizing):

if is\_winner(board, 'O'):

return 1

if is\_winner(board, 'X'):

return -1

if is\_full(board):

return 0

if is\_maximizing:

best\_score = float('-inf')

for row, col in get\_empty\_cells(board):

board[row][col] = 'O'

score = minimax(board, depth + 1, False)

board[row][col] = ' '

best\_score = max(score, best\_score)

return best\_score

else:

best\_score = float('inf')

for row, col in get\_empty\_cells(board):

board[row][col] = 'X'

score = minimax(board, depth + 1, True)

board[row][col] = ' '

best\_score = min(score, best\_score)

return best\_score

def get\_best\_move(board):

best\_score = float('-inf')

best\_move = None

for row, col in get\_empty\_cells(board):

board[row][col] = 'O'

score = minimax(board, 0, False)

board[row][col] = ' '

if score > best\_score:

best\_score = score

best\_move = (row, col)

return best\_move

def play\_game():

board = [[' ' for \_ in range(3)] for \_ in range(3)]

print("Welcome to Tic-Tac-Toe!")

print("You are X, and the AI is O.")

print("Enter rows and columns as 1, 2, or 3.")

print\_board(board)

while True:

# Human player's turn

while True:

try:

row = int(input("Enter the row (1-3): ")) - 1

col = int(input("Enter the column (1-3): ")) - 1

if 0 <= row < 3 and 0 <= col < 3 and board[row][col] == ' ':

break

else:

print("Invalid move. Try again.")

except ValueError:

print("Invalid input. Please enter numbers.")

board[row][col] = 'X'

print\_board(board)

if is\_winner(board, 'X'):

print("Congratulations! You win!")

return

if is\_full(board):

print("It's a tie!")

return

# AI player's turn

print("AI is making a move...")

ai\_row, ai\_col = get\_best\_move(board)

board[ai\_row][ai\_col] = 'O'

print\_board(board)

if is\_winner(board, 'O'):

print("AI wins! Better luck next time.")

return

if is\_full(board):

print("It's a tie!")

return

if \_\_name\_\_ == "\_\_main\_\_":

play\_game()