Tic Tac Toe Algorithm

import random

# Function to print the current board

def print\_board(board):

print("\n")

for row in range(3):

print(" | ".join(board[row]))

if row < 2:

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

print("\n")

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

return True

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

# Function to check if the board is full

def is\_full(board):

return all([board[i][j] != " " for i in range(3) for j in range(3)])

# Minimax Algorithm to find the best move

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

if check\_win(board, "O"): # Computer wins

return 1

if check\_win(board, "X"): # Player wins

return -1

if is\_full(board): # Draw

return 0

if is\_maximizing: # Maximizing for computer (O)

max\_eval = float('-inf')

for i in range(3):

for j in range(3):

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

board[i][j] = "O"

eval = minimax(board, depth + 1, False, alpha, beta)

board[i][j] = " "

max\_eval = max(max\_eval, eval)

alpha = max(alpha, eval)

if beta <= alpha:

break

return max\_eval

else: # Minimizing for player (X)

min\_eval = float('inf')

for i in range(3):

for j in range(3):

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

board[i][j] = "X"

eval = minimax(board, depth + 1, True, alpha, beta)

board[i][j] = " "

min\_eval = min(min\_eval, eval)

beta = min(beta, eval)

if beta <= alpha:

break

return min\_eval

# Function to get the best move for the computer

def best\_move(board):

best\_val = float('-inf')

move = (-1, -1)

for i in range(3):

for j in range(3):

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

board[i][j] = "O"

move\_val = minimax(board, 0, False, float('-inf'), float('inf'))

board[i][j] = " "

if move\_val > best\_val:

best\_val = move\_val

move = (i, j)

return move

# Function to play the game

def play\_game():

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

print("Welcome to Tic Tac Toe!")

print("You are X, the computer is O.\n")

while True:

print\_board(board)

# Player's move

row, col = map(int, input("Enter your move (row col): ").split())

if board[row][col] != " ":

print("Cell already taken. Try again.")

continue

board[row][col] = "X"

if check\_win(board, "X"):

print\_board(board)

print("Congratulations! You win!")

break

if is\_full(board):

print\_board(board)

print("It's a draw!")

break

# Computer's move

print("Computer's turn...")

move = best\_move(board)

board[move[0]][move[1]] = "O"

if check\_win(board, "O"):

print\_board(board)

print("Computer wins!")

break

if is\_full(board):

print\_board(board)

print("It's a draw!")

break

# Start the game

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

play\_game()

OUTPUT:

Welcome to Tic Tac Toe!

You are X, the computer is O.

| |

---------

| |

---------

| |

Enter your move (row col): 1 1

| |

---------

| X |

---------

| |

Computer's turn...

| |

---------

| X | O

---------

| |

Enter your move (row col): 0 0

X | |

---------

| X | O

---------

| |

Computer's turn...

| O |

---------

| X | O

---------

| |

Enter your move (row col): 2 2

X | |

---------

| X | O

---------

| | X

Computer's turn...

Computer wins!