from collections import deque

class TicTacToe:

def \_\_init\_\_(self):

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

def display\_board(self):

for row in self.board:

print("|".join(row))

print("-----")

def is\_winner(self, player):

for i in range(3):

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

return True

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

return True

return False

def is\_full(self):

return all(self.board[i][j] != ' ' for i in range(3) for j in range(3))

def get\_empty\_cells(self):

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

def make\_move(self, player, row, col):

if self.board[row][col] == ' ':

self.board[row][col] = player

return True

return False

def tic\_tac\_toe\_bfs():

initial\_state = TicTacToe()

queue = deque([(initial\_state, [])])

visited = set()

while queue:

current\_state, moves = queue.popleft()

if current\_state.is\_winner('X'):

return moves

if current\_state.is\_winner('O'):

continue

if current\_state.is\_full():

continue

if current\_state in visited:

continue

visited.add(current\_state)

for i, j in current\_state.get\_empty\_cells():

new\_state = TicTacToe()

new\_state.board = [row.copy() for row in current\_state.board]

new\_state.make\_move('X', i, j)

queue.append((new\_state, moves + [(i, j)]))

return None

def print\_instructions():

print("Tic-Tac-Toe Game")

print("Enter your move in the format 'row,col'. For example, '0,0' for the top-left corner.")

print("You are 'O', and the computer is 'X'. Let's begin!\n")

def main():

print\_instructions()

game = TicTacToe()

while True:

game.display\_board()

player\_move = input("Your move: ")

row, col = map(int, player\_move.split(","))

if game.make\_move('O', row, col):

if game.is\_winner('O'):

game.display\_board()

print("Congratulations! You win!")

break

elif game.is\_full():

game.display\_board()

print("It's a draw!")

break

computer\_moves = tic\_tac\_toe\_bfs()

if computer\_moves:

for move in computer\_moves:

row, col = move

if game.make\_move('X', row, col):

break

else:

print("Invalid move by the computer. Ending the game.")

break

if game.is\_winner('X'):

game.display\_board()

print("Sorry, you lose. Better luck next time!")

break

elif game.is\_full():

game.display\_board()

print("It's a draw!")

break

else:

print("No solution found. It's a draw!")

break

print("Game over.")

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

main()

00

11

22