import os #Here we import the os module, which provides functions for interacting with the operating system.

def print\_board(board): #print\_board function is defined with board as arguement and display the board in a user-freindly format

    """Prints the current state of the Tic Tac Toe board."""

    print(board[0] + " | " + board[1] + " | " + board[2])

    print("--+---+--")

    print(board[3] + " | " + board[4] + " | " + board[5])

    print("--+---+--")

    print(board[6] + " | " + board[7] + " | " + board[8])

def check\_win(board):#This function defined a list of tuples, where each tuple represents a wining input

    """Checks if there is a winner."""

    wins = [(0, 1, 2), (3, 4, 5), (6, 7, 8), (0, 3, 6),

            (1, 4, 7), (2, 5, 8), (0, 4, 8), (2, 4, 6)] #list of tuples for wining combination

    for win in wins:#This loop iterates through each winning combination

        if board[win[0]] == board[win[1]] == board[win[2]] and board[win[0]] != "-":#Check if all three positions in current wining combination have same player's and are not empty

            return board[win[0]] #If a winner is found, the function returns winning player's mark

    return None

def play\_game(): #This defines the main function to run the game

    """Main function to play the Tic Tac Toe game."""

    board = ["-", "-", "-", "-", "-", "-", "-", "-", "-"] #Initializes the board with empty spaces represented by -

    current\_player = "X" #Sets the starting player to X

    game\_over = False #Initializes the game state to not over.

    while not game\_over: #This loop continues as long as the game is not over

        print\_board(board)

        move = input("Player " + current\_player + ", enter your move (1-9): ")

        try:

            move = int(move) - 1

            if 0 <= move <= 8 and board[move] == "-":

                board[move] = current\_player

                winner = check\_win(board)

                if winner:

                    print\_board(board)

                    print("Player " + winner + " wins!")

                    game\_over = True

                elif "-" not in board:

                    print\_board(board)

                    print("It's a tie!")

                    game\_over = True

                else:

                    current\_player = "O" if current\_player == "X" else "X"

            else:

                print("Invalid move. Please try again.")

        except ValueError:

            print("Invalid input. Please enter a number between 1 and 9.")

    save\_game\_prompt = input("Do you want to save the game? (y/n): ")

    if save\_game\_prompt.lower() == "y":

        save\_game(board, current\_player)

def save\_game(board, current\_player):

    """Saves the current game state to a file."""

    try:

        with open("game\_state.txt", "w") as file:

            file.write(",".join(board) + "\n")

            file.write("Player Turn: " + current\_player)

        print("Game state saved!")

    except IOError:

        print("Error saving game state.")

def load\_game():

    """Loads a saved game state from a file."""

    try:

        with open("game\_state.txt", "r") as file:

            board = file.readline().strip().split(",")

            player\_turn\_line = file.readline().strip()

            # Check if the line contains "Player Turn:" before splitting

            if ": " in player\_turn\_line:

                current\_player = player\_turn\_line.split(": ")[1]

            else:

                current\_player = "X"  # Default to player X if not found

        return board, current\_player

    except FileNotFoundError:

        print("No saved game found.")

        return None, None

    except IOError:

        print("Error loading game state.")

        return None, None

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

    print("Welcome to Tic Tac Toe!")

    load\_saved\_game = input("Do you want to load a saved game? (y/n): ")

    if load\_saved\_game.lower() == "y":

        board, current\_player = load\_game()

        if board and current\_player:

            play\_game()  # Resume the game with loaded state

    else:

        play\_game()  # Start a new game