OOPS VERSION 1 ASSIGNMENT.

TIC-TAC-TOE GAME PROJECT DOCUMENTATION

Submitted By: ***Shivani & Puja Kumari***

Contribution:

Shivani: Done logic part.

Puja Kumari: Frontend has been done by her.

Working Stage of the project: Completed the logic portion and working on frontend.

---

1. Introduction

This project is a simple implementation of the classic **Tic-Tac-Toe** game using programming concepts such as loops, conditions, functions, and arrays/lists. Tic-Tac-Toe is a two-player game where players take turns marking spaces in a 3×3 grid. The goal is to place three marks in a horizontal, vertical, or diagonal row.

This project demonstrates fundamental logical thinking, modular programming, and clean code principles. It also helps in understanding how interactive console-based games work.

---

2. Objectives

- To design a fully functional Tic-Tac-Toe game.
- To practice modular programming by dividing the game into meaningful functions.
- To implement validation checks and game-play rules.
- To allow two players to interactively play on the same system.
- To display the game board after each move and declare the result.

---

3. Approach

1. Initialize an empty 3×3 board.
2. Display the board in a user-friendly format.
3. Allow players to input their moves alternately.
4. Validate each move to prevent overwriting a filled space.
5. Update the board and check for a win or draw after every move.
6. Continue until a result is reached.
7. Optionally restart the game.

The game is constructed using multiple functions to ensure readability, modularity, and easier debugging.

---

4. Functions Included in the Game

Function Name	Purpose	Description / Function Content
displayBoard(board)	To display the 3×3 game board.	Prints the current state of the board in a formatted layout. Shows positions row-wise and updates after each move.
playerInput(board, player)	To take the current player's move.	Accepts a position (1–9) from the player. Validates the input, checks whether the space is empty, and updates the board with the player's symbol (X or O). Displays an error for invalid moves.
checkWin(board, player)	To check if a player has won.	Evaluates all possible winning combinations: 3 rows, 3 columns, and 2 diagonals. Returns TRUE if the player's symbol forms any winning pattern.
checkDraw(board)	To determine if the game is a draw.	Checks whether all cells are filled and no winning condition is met. Returns TRUE if the game is tied.
switchPlayer(player)	To alternate between players.	Switches the current player: if player = X → O, and if player = O → X.
resetBoard()	To initialize a fresh game board.	Creates a new empty 3×3 board (list/array) with blank values, used when players choose to restart the game.
main()	To control and run the entire game.	Contains the main game loop: initializes board, manages turns, calls display & validation functions, checks result after each move, declares the winner/draw, and asks for replay.

---

5. Sample Flow of the Game

1. Display empty board.
2. Player X starts and chooses a position.
3. Board is updated.
4. Player O plays next.
5. Repeat until win or draw.
6. Display the winner or draw message.
7. Prompt for replay.
