

**MINI PROJECT : TIC TAC TOE**

NAMES REGISTER NUMBER

ALAN CHRIST.C 123012019004

GURUMOORTHY.R 123012019012

TAMIL SELVAN.K 123012019031

CLASS : II B SEC CSE SPEC WITH AI &ML

COURSE NAME : INTRODUCTION TO ARTIFICIAL INTERLLIGENCE

SUBJECT CODE : XCSHA1

**TIC TAC TOE PROGRAM**

def print\_board(board):

print("\n")

for row in board:

print(" | ".join(row))

print("-" \* 5)

print("\n")

def check\_winner(board, player):

for i in range(3):

if all([cell == player for cell in board[i]]): # check rows

return True

if all([board[j][i] == player for j in range(3)]): # check columns

return True

if board[0][0] == player and board[1][1] == player and board[2][2] == player:

return True

if board[0][2] == player and board[1][1] == player and board[2][0] == player:

return True

return False

def is\_board\_full(board):

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

def player\_turn(board, player):

while True:

try:

move = int(input(f"Player {player}, enter your move (1-9): ")) - 1

row, col = divmod(move, 3)

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

board[row][col] = player

break

else:

print("Cell already occupied, try again.")

except (ValueError, IndexError):

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

def play\_game():

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

current\_player = "X" # Player X starts

print\_board(board)

while True:

player\_turn(board, current\_player)

print\_board(boarf check\_winner(board, current\_player):

print(f"Player {current\_player} wins!")

break

if is\_board\_full(board):

print("It's a tie!")

break

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

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

play\_game()

**Explanation:**

1. **Board Representation**: The board is a 3x3 list of lists, initialized with spaces (" "), representing empty cells.
2. **Printing the Board**: The print\_board() function prints the current state of the board, formatting it to resemble a Tic-Tac-Toe grid.
3. **Player Input**: The player\_turn() function takes care of getting input from the players. It checks that the input is valid and that the chosen cell is empty.
4. **Win Condition**: The check\_winner() function checks if any player has won by checking rows, columns, and diagonals.
5. **Tie Condition**: The is\_board\_full() function checks if the board is full, indicating a tie if there is no winner.
6. **Main Loop**: The game alternates between players X and O until there's a winner or a tie.

**Summary of the Game Flow:**

1. **Player X** makes a move.
2. The board is printed.
3. The program checks if Player X has won or if the board is full.
4. If no one has won and the board isn’t full, it’s Player O’s turn.
5. The process repeats until there’s a winner or a tie.