PROGRAM-14

TIC TAC TOE GAME PROBLEM

AIM:-

To write and execute the python program for the Tic Tac Toe game program.

PROCEDURE:-

- **print_board function:**
  - Input: board - a 3x3 matrix representing the Tic-Tac-Toe board.
  - Algorithm:
    - Iterate through each row in the board.
    - Print each row, joining elements with " | " to represent the vertical separators.
- **check_winner function**:
  - Input:
    - board - the current state of the Tic-Tac-Toe board.
    - player - the player ('X' or 'O') to check for winning condition.
  - Algorithm:
    - Check diagonals for three consecutive occurrences of the player's symbol.
    - If any winning condition is met, return True, indicating that the player has won. Otherwise, return False.
- **is_board_full function**:
  - Input: board - the current state of the Tic-Tac-Toe board.
  - Algorithm:
    - Check if all cells on the board are filled (i.e., no empty cells).
- **play_tic_tac_toe function**:
  - Initializes the Tic-Tac-Toe board with empty cells.
  - Sets up the players ('X' and 'O') and selects the starting player.
  - Iterates through player turns until the game ends.
    - Prints the current state of the board.
    - Prompts the current player to input their move (row and column).
    - Checks if the chosen cell is empty. If not, prompts the player to choose another cell.
    - Updates the board with the player's move.

- Checks if the current player has won. If so, prints the board and the winning message.
- Checks if the board is full. If so, prints the board and the tie message.
- Alternates the current player for the next turn.

CODING:-

```
def print_board(board):

    for row in board:

        print(" | ".join(row))

        print("-" * 5)

def check_winner(board, player):

    for i in range(3):

        if all(board[i][j] == player for j in range(3)) 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

def is_board_full(board):

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

def play_tic_tac_toe():
```

```
    board = [[' ']*3 for _ in range(3)]
    players = ['X', 'O']
    current_player = players[0]
    while True:
        print_board(board)
        print(f"Player {current_player}'s turn")
        row, col = map(int, input("Enter row and column (0, 1, or 2) separated by space: ").split())
        if board[row][col] == ' ':
            board[row][col] = current_player
            if check_winner(board, current_player):
                print_board(board)
                print(f"Player {current_player} wins!")
                break
            if is_board_full(board):
                print_board(board)
                print("It's a tie!")
                break
            current_player = players[1] if current_player == players[0] else players[0]
        else:
            print("Cell already taken. Try again.")
play_tic_tac_toe()
```

OUTPUT:-

```
  | X |
-----
  | X |
-----
O |   |
-----
Player O's turn
Enter row and column (0, 1, or 2) separated by space: 1 2
  | X |
-----
  | X | O
-----
O |   |
-----
Player X's turn
Enter row and column (0, 1, or 2) separated by space: 2 1
  | X |
-----
  | X | O
-----
O | X |
-----
Player X wins!
```

RESULT:-

Hence the program has been successfully executed and verified.