24/9/24 LAB-1

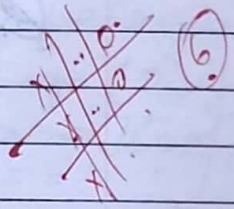
Date __/__/__
Page _____

TIC TAC TOE

Algorithm

Step 1: import numpy, Random library
Step 2: Create a 3 × 3 matrix using 2-dimensional Array
Step 3: initialize the entire matrix as blank using for loop(-)hyphen
Step 4: Display a message for the users to either choose to begin or not
Step 5: Set patterns which is considered as win
Step 6: check the board if it matches any pattern after each more if yes then display win else after all blanks are filled Display tie.

Refer Algorithm



code

```
import random

def initialize_board():
    return [[' ' for _ in range(3)]
            for _ in range(3)]

def display_board(board):
    for row in board:
        print('|'.join(row))
        print('-' * 5)

def check_winner(board):
    for row in board:
        if row[0] == row[1] == row[2] != ' ':
            return row[0]

    for col in range(3):
        if board[0][col] == board[1][col] == board[2][col] != ' ':
            return board[0][col]

    if board[0][0] == board[1][1] == board[2][2] != ' ':
        return board[0][0]

    if board[0][2] == board[1][1] == board[2][0] != ' ':
        return board[0][2]

    return None
```

```
def available_moves(board):
    return [(i,j) for i in range(3)
                  for j in range(3)
                  if board[i][j]==' ']

def check_two_in_a_row(board,
                       player):
    for row in range(3):
        if board[row].count(player)==2
                and board[row].
                count(' ')
                ==1:
            return row, board[row].index(' ')

    for col in range(3):
        if [board[row][col] for row in
                range(3)].count
                (player)==2:
            empty_index=[row for row
                in range(3) if
                board[row][col]
                ==' ']

            if empty_index:
                return empty_index[0], col

    if [board[i][i] for i in range(3)].
                count(player)==2

        empty_index=[i for i in range(3)
                if board[i][i]==' ']
        if empty_index:
            return empty_index[0],
                empty_index
                [0]
```

```
if [board[i][2-i] for i in range(3)].
                    count(player)
                    == 2:
    empty_index = [i for i in range(3)
                    if board[i]
                    [2-i]
                    == ' ']
    if empty_index:
        return empty_index[0], 2-empty
                    index[0]
return None

def make_move(board, player, move):
    board[move[0]][move[1]] = player

def computer_move(board):
    move = check_two_in_a_row(board,
                    'O')
    if move:
        make_move(board, 'O', move)
        return

    move = available_moves(board)
    if moves:
        move = random.choice(moves)
        make_move(board, 'O', move)

def user_move(board):
    while True:
        try:
            row = int(input("Enter
                    row(0-2):
                    "))
```

```
col = int(input("Enter column (0-2): "))
if board[row][col] == ' ':
    make_move(board, 'X', (row, col))
    return
else:
    print("That spot is already taken. Try again!")

except (ValueError, IndexError):
    print("Invalide input. Please Enter number between 0 and 2")
def play_game():
    board = initialize_board()
    players = ['X', 'O']
    current_player = 0

for _ in range(9):
    display_board(board)
    if current_player == 0:
        user_move(board)
    else:
        computer_move(board)

winner = check_winner(board)
if winner:
    display_board(board)
    print(f"Player {winner} wins!")
    return

current_player = 1 - current_player
```

```
display_board(board)
print("Its a draw!")

play_game()
```

output

```
  |   |
--|---|--
  |   |
--|---|--
  |   |
```

Enter row(0-2) : 1
Enter column(0-2) : 2

```
  |   |
--|---|--
  |   | X
--|---|--
  |   |
```

```
  |   |
--|---|--
  |   | X
--|---|--
  | 0 |
```

Enter row(0-2) : 0
Enter column(0-2) : 0

```
X |   |
--|---|--
  |   | X
--|---|--
  | 0 |
```

```
X | 0 |
--|---|--
  |   | X
--|---|--
  | 0 |
```

Date __/__/__
Page _____

Enter row (0-2): 1
Enter column (0-2) : 1

```
 | | 
 |x|x
 |o|
```

```
x|o| 
o|x|x
 |o|
```

Enter row (0-2) : 2
Enter column (0-2) : 2

```
x|o|
o|x|x
 |o|x
```

Player X wins!

24/9

Output

```
 | |
-----
 | |
-----
 | |
-----
Enter row (0-2): 1
Enter column (0-2): 2
 | |
-----
 | |X
-----
 | |
-----
 | |
-----
 | |X
-----
 |O|
-----
Enter row (0-2): 0
Enter column (0-2): 0
X| |
-----
 | |X
-----
 |O|
```

Output

```
 | |X
-----
 |O|
-----
Enter row (0-2): 1
Enter column (0-2): 1
X|O|
-----
 |X|X
-----
 |O|
-----
X|O|
-----
O|X|X
-----
 |O|
-----
Enter row (0-2): 2
Enter column (0-2): 2
X|O|
-----
O|X|X
-----
 |O|X
-----
Player X wins!

=== Code Execution Successful ===
```