LAB-1

24/9/24

1) Write an algorithm and program to create a tic-tac-toe game

Step 1 :- Create a 2D array of size 3×3 and initialize all the elements as '-'.

=) Create 3 rows as 3 different lists [[ ],[ ],[ ]]

```
[[-, -, -]
 [-, -, -]
 [-, -, -]]
```

Step 2 :- Use 'random' to select who will be the first player to place the move. First move will be 'X'.

Display the board after each move

```
def print_b(board):
    for row in board:
        print('|'.join(row))
        print('_'*5)
```

Step3 :- Check for winner after each move

=) There are 8 possibilities ~~and~~ (3 rows, 3 columns, 2 diagonals)

```
def check(board):
    for i in range(3):
        if board[i][0] == board[i][1] == board[i][2] != '-':
            return board[i][0]
```

Similarly check for columns & diagonals

Step4 :- Player move:-

The player should enter the row & column index to make a move

```
if board[row][col] == '-':
    board[row][col] = 'X'.
else:
    print("cell already taken! Try again")
```

Step5 :- Computer move :-

→ Check for winning move

```
for i in range(3):
    for j in range(3):
        if board[i][j] == '_':
            board[i][j] = 'O' :
            ~~if check_winner(board) == 'O';~~
                ~~return~~
```

→ If no winning move, pick a random valid move or prevent player from winning

X	X	.
	X	
X	X	✓ C

Step6 :-

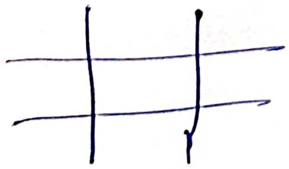
```
if check_winner(board):
    print ("Player X wins")
if is_tie(board):
    print("It's a tie")
```

Program :-

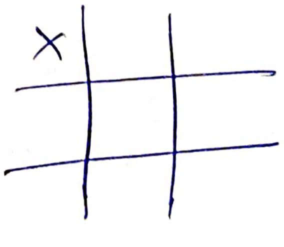
```
import random
def print_board(board):
    for row in board:
        print(" | ".join(row))
        print("-" * 9)

def check_winner(board):
    for i in range(3):
        if board[i][0] == board[i][1] == board[i][2] != " ":
            return board[i][0]
        if board[0][i] == board[1][i] == board[2][i] != " ":
            return board[0][i]

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

def find_winning_move(board, player):
    for i in range(3):
        for j in range(3):
            if board[i][j] == " ":
                board[i][j] = player
                if check_winner(board) == player:
                    board[i][j] = " "
                    return (i, j)
                board[i][j] = " "
    return None
```

```
def get-computer_move (board):
    move = find-winning_move (board, "o")
    if move:
        return move

    move = find-winning_move (board, "x")
    if move:
        return move

    if board [1][1] == " ":
        return (1,1)
    corners = [(0,0), (0,2), (2,0), (2,2)]
    for corner in corners:
        if board[corner[0]][corner[1]] = = " ":
            return corner

    for i in range (3):
        for j in range (3):
            if board [i][j] == " ":
                return (i,j)


def tic_tac_toe ():
    board : [[" " for _ in range(3)] for _in range(3)]
    current_player = "x"
    computer_player = "o"
    print ("Player X goes first.")
    while True :
        ~~print_boaplayer~~
        print_board (board)
        if current_player == "x":
            while True:
                try:
                    row = int (input ("Player X, enter the row: "))
                    col = int (input (" Player X, enter the column: "))
            if board[row][col] == " ":
                break
            else:
                print("Cell is already taken! try again")

        except (ValueError, IndexError):
            print("Invalid input!")

    else:
        print("Computer's turn ....")
        row, col = get_computer_move(board)
        print(f"Computer chooses row{row}, column{col}")

    board[row][col] = current_player

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

    if is_full(board):
        print_board(board)
        print("It's a tie!")
        break

    current_player = computer_player if current_player == "X" else "X"


if _name_ == "_main_":
    tic_tac_toe()
```

Output:- Player X goes first.



Player X enter row : 0
Player X enter col : 0



Computer chooses row 1, column 1

X		
	O	

Player X, enter row: 0

Player X, enter col: 1

X	X	
	O	

Computer chooses row 0, column 2

X	X	O
	O	

Player X enters the row: 1

Player X, enter the col: 0

X	X	O
X	O	

computer chooses row 2, col 0

X	X	O
X	O	
O		

Player 'O' wins!

24/9/20