week 01 → 24/09/2024

Date __/__/__
Page _____

tic tok toe algorithm ( ) {

// input :- value of row & column
// output :- win / lose / draw

S1 :-
initialize board as 3x3 3D array with Empty value

```
int [ ] [ ] board = new int[3][3];
for(int i=0; i<3; i++) {
    for(int j=0; j<3; j++) {
        board[i][j] = " ";
    }
}
```

S2 :-
Set user player = "X" & AI = "O";

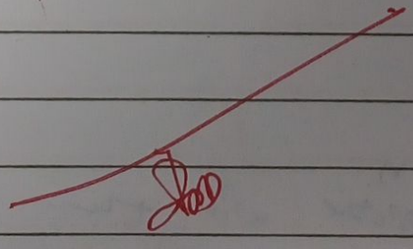
S3 :-
while Game not over. // S4
if player turn { Display Board. ask player for input.
if board [row] [column] is empty
board[row] [column] = 'X'
else
ask for another input
}
else {
// AI turn
if AI suppose win turn
if board [row] [column] is empty
board [row] [column] = 'O'
// first move is AI

Date __/__/__
Page _____

else if Player is suppose to win
block the user/Player.
else.
place random of row, colm
if board [row] [colm] is empty
board [row] [column] = 'O'

GUI:-
// Game over condition

if board [row] [1,2,3] = 'X'
X win;
else if board [1,2,3] [colm] = 'X'
Player win;
else if diagonal = 'X'
Player win;
else if board [row] = 'O'
AI win;
else if board [colm] = 'O'
AI win;
else if diagonal. board [ ] = 'O'
AI win.
else
draw.



Date __/__/__
Page _____

```
import random

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

def check_winner(board):
    #check rows, colums, and diagonals
    lines = []
    #Add rows
    lines.extend(board)
    #Add colums
    lines.extend([[board[i][j] for i in range(3)] for
                  j in range(3)])
    #Add diagonals
    lines.append([board[i][i] for i in range(3)])
    lines.append([board[i][2-i] for i in range(3)])

    for line in lines:
        if line[0] == line[1] == line[2] != '':
            return line[0]
    return None

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

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

Date __/__/__
Page _____

```
def player_move(board):
    while True:
        try:
            row = int(input("Enter row(0-2): "))
            col = int(input("Enter column(0-2): "))
            if board[row][col] == ' ':
                board[row][col] = 'x'
                break
            else:
                print("position already taken.
                        try again.")
        except(valueError, indexError):
            print("Invalid input. plz enter b/w 0 & 2")

def ai_move(board):
    for row, col in get_empty_positions(board):
        board[row][col] = 'o'
        if check_winner(board) == 'o':
            board[row][col] = 'o'
            return
        board[row][col] = ' '

    # block player moves
    for row, col in get_empty_positions(board):
        board[row][col] = 'x'
        if check_winner(board) == 'x':
            board[row][col] = 'o'
            retur
        board[row][col] = ' '

    # chose a random move
    row, col = random.choice(get_empty_pos
                .positions(board))
    board[row][col] = 'o'
```

Date __/__/__
Page _____

```
def main():
    board = [[' ' for _ in range(3)] for _ in
                                    range(3)]
    current_player = 'X'

    while True:
        print_board(board)

        if current_player == 'X':
            player_move(board)
        else:
            ai_move(board)

        winner = check_winner(board)
        if winner:
            print_board(board)
            print(f"{winner} wins!")
            break
        elif is_board_full(board):
            print_board(board)
            print("It's a draw!")
            break

        current_player = 'O' if current_player == 'X'
                                else 'X'

if __name__ == "__main__":
    main()
```

Date __/__/__
Page

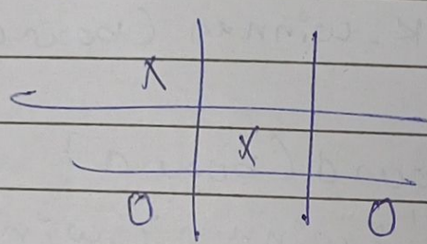
Sample out put :-

```
  |   |
  |   |
  |   |
```

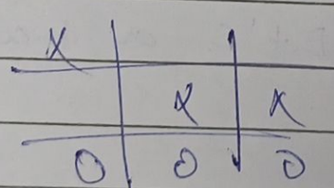
Enter row (0-2): 1
Enter colum (0-2): 1

```
  |   |
  | X |
  |   |
```

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



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



O win

from 24/9/20