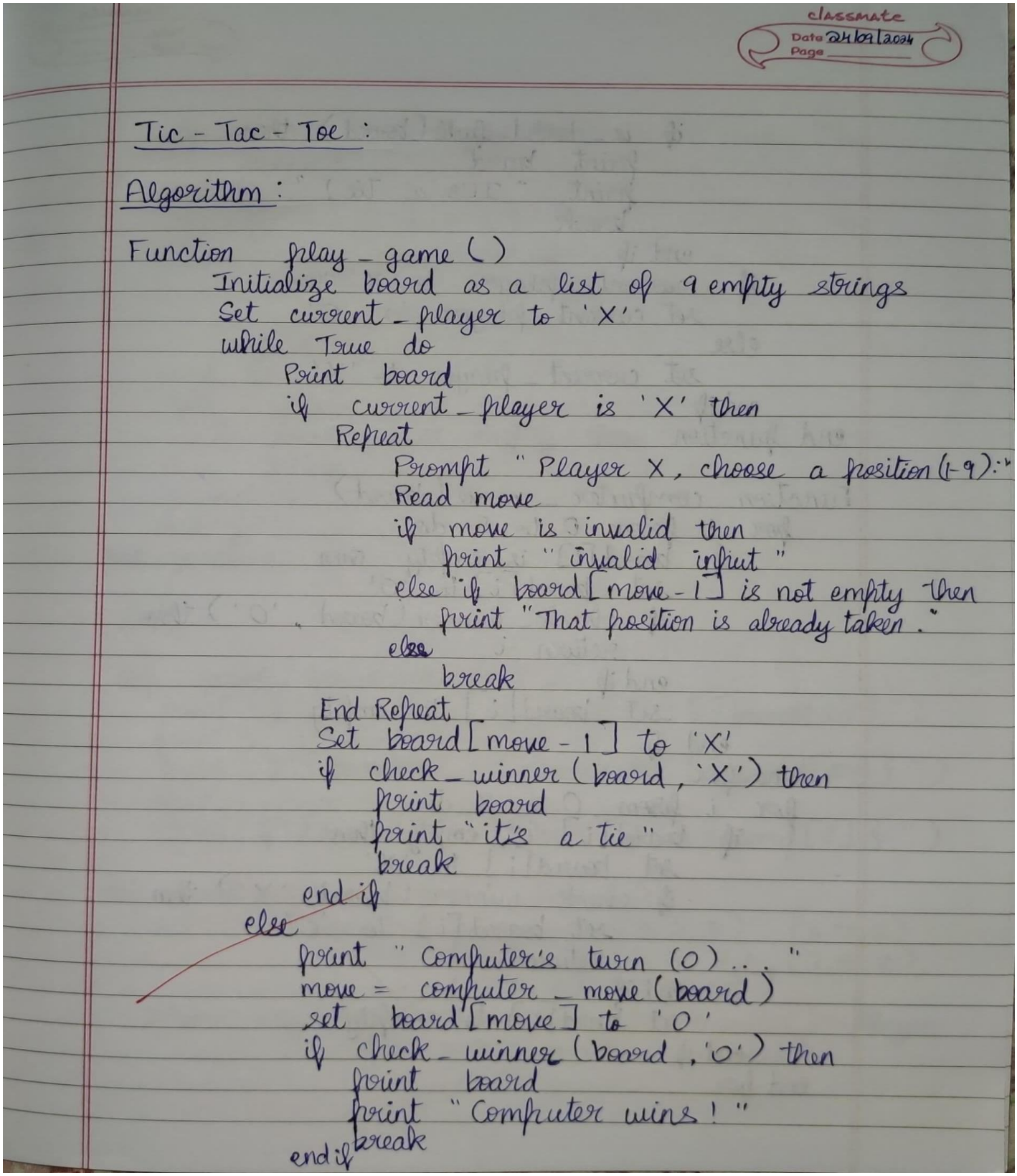
Tic-Tac-Toe (Lab 1: 24-09-2024)

Observation Book:



Tic - Tac - Toe :

Algorithm :

```
Function play_game()
    Initialize board as a list of 9 empty strings
    Set current_player to 'X'
    while True do
        Print board
        if current_player is 'X' then
            Repeat
                Prompt "Player X, choose a position (1-9):"
                Read move
                if move is invalid then
                    print "invalid input"
                else if board[move-1] is not empty then
                    print "That position is already taken."
                else
                    break
            End Repeat
            Set board[move-1] to 'X'
            if check_winner(board, 'X') then
                print board
                print "it's a tie"
                break
            end if
        else
            print "Computer's turn (O)..."
            move = computer_move(board)
            set board[move] to 'O'
            if check_winner(board, 'O') then
                print board
                print "Computer wins!"
            end if break
```

```
            if is_board_full(board) then
                print board
                print "It's a tie!"
                break
            end if
        if current_player is 'X' then
            set current_player to 'O'
        else
            set current_player to 'X'
        end if
end function

Function computer_move(board)
    for i from 0 to 8 do
        if board[i] is empty then
            set board[i] to 'O'
            if check_winner(board, 'O') then
                return i
            end if
            set board[i] to empty
        end if
    end for
    for i from 0 to 8 do
        if board[i] is empty then
            set board[i] to 'X'
            if check_winner(board, 'X') then
                set board[i] to 'O'
                return i
            end if
            set board[i] to empty
        end if
    end for
```

```
    if board[4] is empty then
        return 4.
    end if
    corners = [0, 2, 6, 8]
    for corner in corners do
        if board[corner] is empty then
            return corner
        end if
    end for
    for i from 0 to 8 do
        if board[i] is empty then
            return i
        end if
    end for
end function
```

Code :

```
def print_board(board):
    print(f" {board[0]} | {board[1]} | {board[2]} ")
    print("---+---+---")
    print(f" {board[3]} | {board[4]} | {board[5]} ")
    print("---+---+---")
    print(f" {board[6]} | {board[7]} | {board[8]} ")

def check_winner(board, player):
    win_conditions = [(0,1,2), (3,4,5), (6,7,8),
                      (0,3,6), (1,4,7), (2,5,8),
                      (0,4,8), (2,4,6)]
    return any(board[a] == board[b] == board[c] == player
               for a,b,c in win_conditions)
```

```
def is_board_full(board)
    return all(cell in ('X', 'O') for cell in board)

def computer_move(board):
    for i in range(9):
        if board[i] == ' ':
            board[i] = 'O'
            if check_winner(board, 'O'):
                return i
            board[i] = ' '
    for i in range(9):
        if board[i] == ' ':
            board[i] = 'X'
            if check_winner(board, 'X'):
                board[i] = 'O'
                return i
            board[i] = ' '
    if board[4] == ' ':
        return 4
    corners = [0, 2, 6, 8]
    for corner in corners:
        if board[corner] == ' ':
            return corner
    for i in range(9):
        if board[i] == ' ':
            return i

def play_game():
    board = [' ' for _ in range(9)]
    current_player = 'X'
    while True:
        print_board(board)
        if current_player == 'X':
```

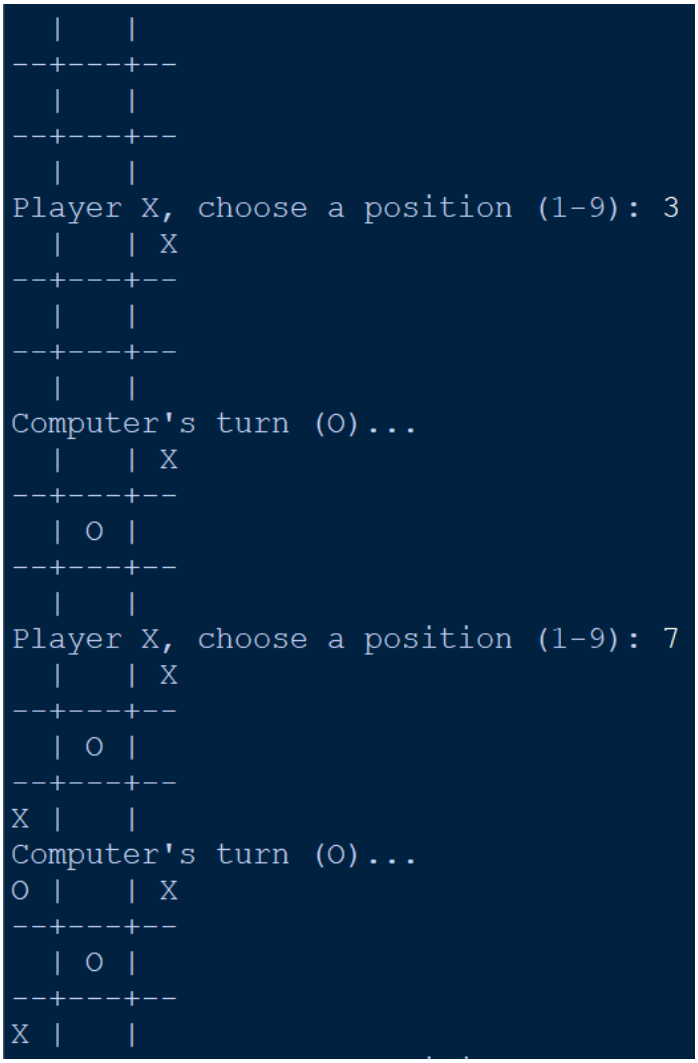
```
            try:
                move = int(input("Player X, choose a position
                                  (1-9): ")) - 1
            except ValueError:
                print("Invalid input. Try again")
                continue
            if move < 0 or move > 8:
                print("Invalid move. Try again")
                continue
        else:
            print("Computer's turn (O)...")
            move = computer_move(board)
        board[move] = current_player
        if check_winner(board, current_player):
            print_board(board)
            print("{} {} player {} wins!")
            break
        if is_board_full(board):
            print_board(board)
            print("It's a tie!")
            break
        current_player = 'O' if current_player == 'X'
                         else 'X'

play_game()
```

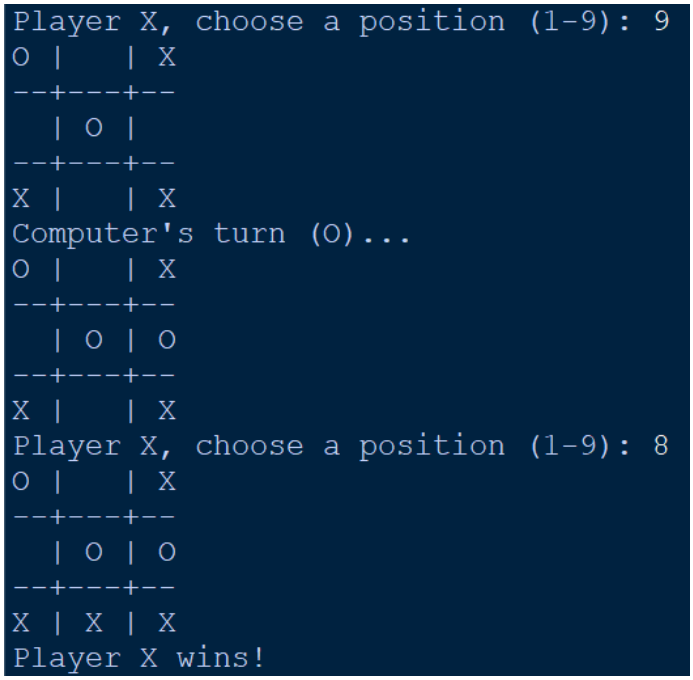
Output:

```
     |     |
  - - + - - - + - -
     |     |
  - - + - - - + - -
     |     |
```

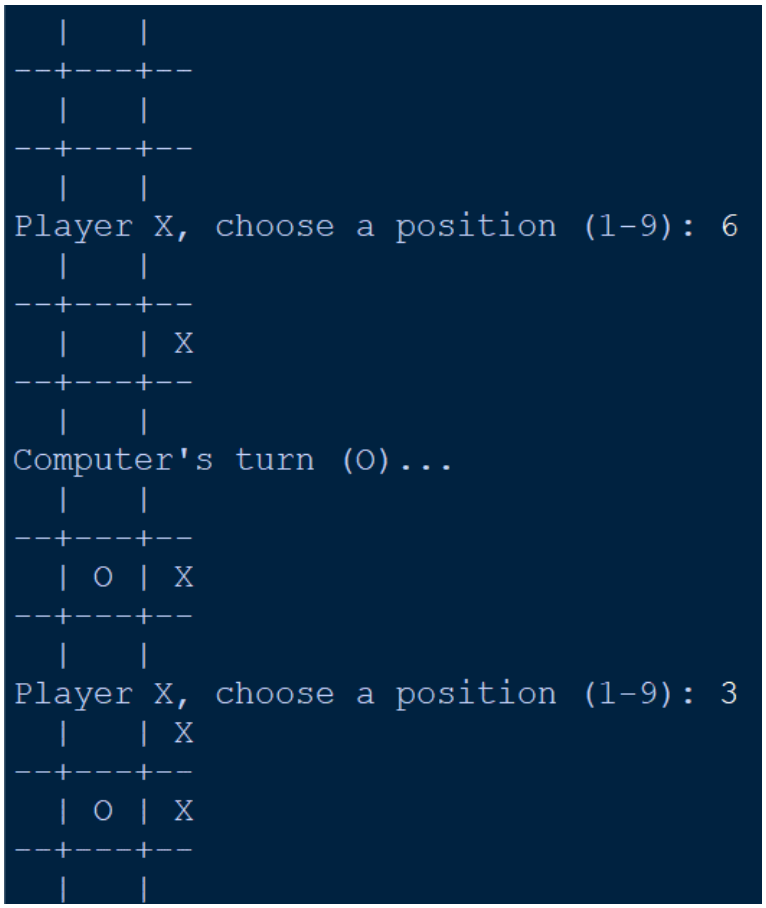
Output Screenshots:

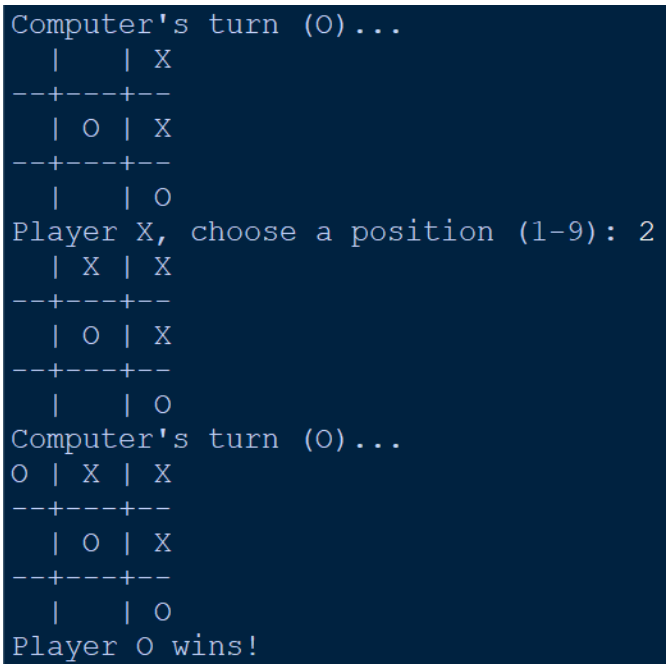
Player X winning:





Player O winning (Computer winning):





Tie:

```
  |   |
--+---+--
  |   |
--+---+--
  |   |
Player X, choose a position (1-9): 5
  |   |
--+---+--
  | X |
--+---+--
  |   |
Computer's turn (O)...
O |   |
--+---+--
  | X |
--+---+--
  |   |
Player X, choose a position (1-9): 7
O |   |
--+---+--
  | X |
--+---+--
X |   |
Computer's turn (O)...
O |   | O
--+---+--
  | X |
--+---+--
X |   |
```

```
Player X, choose a position (1-9): 4
O | X | O
--+---+--
X | X |
--+---+--
X | O |
Computer's turn (O)...
O | X | O
--+---+--
X | X | O
--+---+--
X | O |
Player X, choose a position (1-9): 9
O | X | O
--+---+--
X | X | O
--+---+--
X | O | X
It's a tie!
```