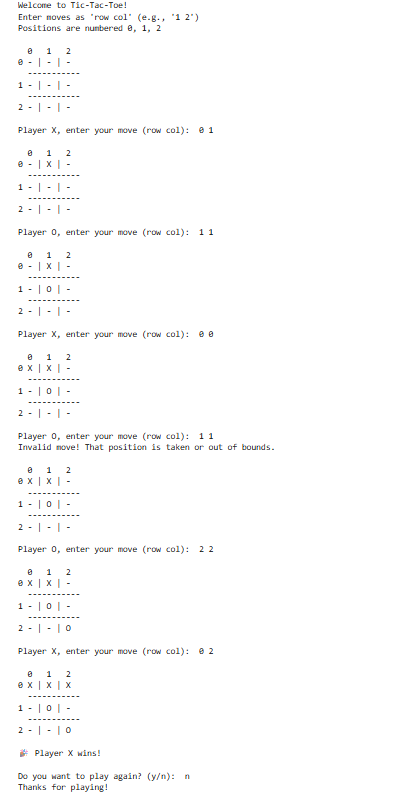
**Week 1  
Tic – tac - toe**

Program:

def print\_board(board):  
    print("\n  0   1   2")  
    for i in range(3):  
        print(f"{i} {board[i][0]} | {board[i][1]} | {board[i][2]}")  
        if i < 2:  
            print("  -----------")  
    print()  
  
def check\_winner(board):  
    # Check rows  
    for row in board:  
        if row[0] == row[1] == row[2] != '-':  
            return row[0]  
     
    # Check columns  
    for col in range(3):  
        if board[0][col] == board[1][col] == board[2][col] != '-':  
            return board[0][col]  
     
    # Check diagonals  
    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\_board\_full(board):  
    for row in board:  
        if '-' in row:  
            return False  
    return True  
  
def is\_valid\_move(board, row, col):  
    return 0 <= row < 3 and 0 <= col < 3 and board[row][col] == '-'  
  
def get\_player\_move(player):  
    while True:  
        try:  
            move = input(f"Player {player}, enter your move (row col): ")  
            row, col = map(int, move.split())  
            return row, col  
        except (ValueError, IndexError):  
            print("Invalid input! Please enter row and column as two numbers (0-2).")  
  
def play\_tic\_tac\_toe():  
    # Initialize empty board  
    board = [['-' for \_ in range(3)] for \_ in range(3)]  
    current\_player = 'X'  
     
    print("Welcome to Tic-Tac-Toe!")  
    print("Enter moves as 'row col' )  
    print("Positions are numbered 0, 1, 2")  
     
    # Main game loop  
    while True:  
        print\_board(board)  
         
        # Get player move  
        row, col = get\_player\_move(current\_player)  
         
        # Check if move is valid  
        if is\_valid\_move(board, row, col):  
            # Make the move  
            board[row][col] = current\_player  
             
            # Check for winner  
            winner = check\_winner(board)  
            if winner:  
                print\_board(board)  
                print(f" Player {winner} wins!")  
                break  
             
            # Check for tie  
            if is\_board\_full(board):  
                print\_board(board)  
                print(" It's a tie!")  
                break  
             
            # Switch players  
            current\_player = 'O' if current\_player == 'X' else 'X'  
        else:  
            print("Invalid move! That position is taken or out of bounds.")  
  
def main():  
    while True:  
        play\_tic\_tac\_toe()  
         
        # Ask to play again  
        play\_again = input("\nDo you want to play again? (y/n): ").lower()  
        if play\_again != 'y':  
            print("Thanks for playing!")  
            break  
  
# Run the game  
if \_\_name\_\_ == "\_\_main\_\_":  
    main()

Output:



**Vacuum cleaner**

import random  
  
# Function to clean a room  
def clean\_room(rooms, position):  
    if rooms[position] == 1:  
        print(f"Room {position+1} is Dirty. Cleaning...")  
        rooms[position] = 0  
        print(f"Room {position+1} is now Clean.")  
    else:  
        print(f"Room {position+1} is already Clean.")  
  
# Function to move to the next room  
def move(position, total\_rooms):  
    position = (position + 1) % total\_rooms  
    print(f"Moving to Room {position+1}")  
    return position  
  
# Function to run the vacuum cleaner  
def run(rooms, steps):  
    position = 0  # start at first room  
    for \_ in range(steps):  
        clean\_room(rooms, position)  
        position = move(position, len(rooms))  
    print("Final Room Status:", rooms)  
  
  
# Initialize 4 rooms randomly (0 = clean, 1 = dirty)  
rooms = [random.choice([0, 1]) for \_ in range(4)]  
print("Initial Room Status:", rooms)  
  
# Run for 8 steps  
run(rooms, 8)

Output:

