CAT 2 GitHub Submission – [20 Marks] - DUE 26/11/2024

Instructions for CAT 2:

1. Complete the following practical tasks and ensure your code is well-documented.
2. Create a GitHub repository titled `ICS3203-CAT2-Assembly-<YourName and Admn Number>`.
3. Include a `README.md` file with:
    a. A brief overview of each program's purpose.
    b. Instructions on compiling and running the code.
    c. Insights or challenges encountered in each task.

TASKS:

1. **Control Flow and Conditional Logic (6 Marks)**
    a. Write a program that:
        i. Prompts for a user input number.
        ii. Uses branching logic to classify the number as "POSITIVE," "NEGATIVE," or "ZERO."
        iii. Use both conditional and unconditional jumps in the program to handle these cases effectively.
    b. **Documentation Requirement:** In your comments, explain why you chose specific jump instructions and how each one impacts the program flow.
2. **Array Manipulation with Looping and Reversal (6 Marks)**
    a. Implement a program that:
        i. Accepts an array of integers (e.g., five values) as input from the user.
        ii. Reverses the array in place.
        iii. Outputs the reversed array.
    b. Requirements:
        i. Avoid using additional memory to store the reversed array.
        ii. Use loops to perform the reversal.
    c. **Documentation Requirement:** Comment each step of your reversal process and explain any challenges with handling memory directly.
3. **Modular Program with Subroutines for Factorial Calculation (4 Marks)**
    a. Develop a program that:
        i. Computes the factorial of a number received as input.
        ii. Uses a separate subroutine (function-like code block) to perform the calculation.

iii. Uses the stack to preserve registers, demonstrating an understanding of modular code and register handling.
iv. Places the final result in a general-purpose register.

b. **Documentation Requirement:** Document how registers are managed, particularly how values are preserved and restored in the stack.

4. **Data Monitoring and Control Using Port-Based Simulation (4 Marks)**
   a. Simulate a control program that:
      i. Reads a "sensor value" from a specified memory location or input port (e.g., simulating a water level sensor).
      ii. Based on the input, performs actions such as:
         1. Turning on a "motor" (by setting a bit in a specific memory location).
         2. Triggering an "alarm" if the water level is too high.
         3. Stopping the motor if the water level is moderate.
   b. **Documentation Requirement:** Explain how the program determines which action to take based on the "sensor" input and how memory locations or ports are manipulated to reflect the motor or alarm status.