University of Dhaka
CSE 3113: Microprocessor and Assembly Language Lab
Lab Exam
Date: 27.05.2025

1. Write an assembly language program to print numbers 0 to 15 in Hex.

2. Write an assembly language code to print all numbers from 0x98 to 0xA5

3. Write an assembly language program to compare two values stored in memory and store the larger one at a new location.

4. Write an assembly language code that sums 5 integers stored in memory and stores the result in another memory location.

5. You are given a 32-bit value. Write an assembly code to **toggle** every even bit (bit positions 0, 2, 4, ...) and store the result. Input value will be in the memory address 0x20000000 and store the result in the memory 0x 20000004.

6. Write an assembly code to compare two signed 32-bit integers and store the result in memory.

- 1 if A > B
- 0 if A == B
- -1 if A < B

7. Write an assembly language code to perform circular bit rotation of a 32-bit integer by n positions to the left. Store the result in memory.
   Sample Input:
   Value at 0x20000000: 0x12345678
   Rotate left by n = 8 bits (value stored at 0x20000004)
   Sample Output:
   Result at 0x20000008: 0x34567812

8. Write an assembly language code to count how many bits are set (1s) in a 32-bit integer and store the result in the memory.

   Sample Input: 0xF0F0F0F0
   Sample output: 16 (since 16 bits are 1)

9. Write an assembly code to count the number of 1s in an 8-bit value which is stored in memory. If it's **even**, store 0 (even parity) and If it's **odd**, store 1 (odd parity).

10. Write an assembly code to implement emergency braking logic using sensor inputs and conditional branching.

    I**nputs:**

- SPEED (0x20000000): 32-bit
- DISTANCE (0x20000004): 32-bit
- BRAKE_PRESSURE (0x20000008): 32-bit

Output:

- MODE (0x2000000C): 0=Idle, 1=Normal Brake, 2=Warning, 3=Emergency Brake

Conditions:

- SPEED > 80 and DISTANCE < 100 → MODE = 3
- DISTANCE < 100 → MODE = 2
- BRAKE_PRESSURE > 80 → MODE = 1
- Else → MODE = 0

11. Write an assembly code to implement a control system mode based on battery level and system load using conditional execution.

Inputs:

- BATTERY_LEVEL (0x20000000): 0–100
- LOAD_STATUS (0x20000004): 0 = Light, 1 = Heavy

Output:

- MODE (0x20000008): 1=Low-power, 2=Normal, 3=High-performance

Conditions:

- Battery < 20 and Load = Heavy → Low-power
- Battery > 80 → High-performance
- Else → Normal