Total Marks	5
Starting Date	Friday, May 09, 2025
Closing Date	Monday, May 12, 2025
Status	Open
Question Title	Graded Discussion Board
Question Description	**GDB Scenario:** Suppose organization XYZ is developing an assembly program for a microcontroller that reads data from a 10×10 sensor grid (where data is stored in a table or matrix form). Each sensor returns a 16-bit (2 bytes) reading. The sensor readings are stored in row-wise order in memory. The program needs to: • Loop the entire grid. • Compare each sensor reading to a threshold value (e.g., 600). • Count how many sensor readings exceed this threshold. **Question:** Now, between the following two addressing modes, which one would you recommend for XYZ's assembly program to remain simple, readable, and efficient? Support your answer with valid reasons. 1. Base + Index 2. Based Register Indirect + Offset **Graded Discussion Board (GDB) Instructions & Guidelines:** **Read the following instructions carefully before posting your comments:** • Your answer should be relevant and concise, as long or off-topic responses will result in mark deductions. Keep your answer focused, with a maximum length of 200 words, and avoid unnecessary details.

- Use "Times New Roman" font style with a font size of 12.
- You may consult books and online resources; however, answers must reflect your own understanding and analysis.
- Post your answer on the Graded Discussion Board (GDB) only, not on the Moderated Discussion Board (MDB).
- GDB submissions via email after the due date will not be accepted.
- There is no grace period for GDB submissions, so ensure you post within the allotted time.
- The GDB will open and close at specific dates and times, with no extensions or additional time provided for participation.

https://whatsapp.com/channel/0029Vb4smybF6smx5admyo47

BY QALB-E- MOMIN 2025

GDB SOLUTION OF CS401:

My Choice:
I would prefer the **Base + Index** addressing mode.

Why I believe it is a better option:
Base + Index is very useful and efficient for looping through a **2D sensor grid** like the 10×10 matrix in this scenario. The base register holds the starting address of the sensor data, while the index register helps access each sensor reading easily, especially when the data is stored row-wise and each reading is 2 bytes.

Reasons:

It keeps the code **simple and easy to read**.

It makes **looping through rows faster and more organized**.

It helps in writing code that is **clean and easier to understand**, especially when working with arrays.

It uses **fewer instructions**, making it more efficient for microcontrollers.

On the other hand, **Based Register Indirect + Offset** can be more complex and may require extra calculations to reach each reading, which can make the code harder to maintain and less efficient.