Big-O, Big-Omega, and Big-Theta Notations

Overview
Asymptotic notations are used to describe the performance of algorithms in terms of time and space as input size grows.

1. Big-O Notation (O)
Big-O represents the **worst-case** time or space complexity of an algorithm. It defines an upper bound on the growth rate.

Example:
A loop that runs n times has time complexity O(n).

2. Big-Omega Notation (Ω)
Big-Omega represents the **best-case** time or space complexity. It defines a lower bound on the growth rate.

Example:
Accessing the first element in an array is Ω(1).

3. Big-Theta Notation (Θ)
Big-Theta represents the **average or exact bound**. It means the algorithm grows at the same rate in both best and worst cases.

Example:
A loop that always runs n times is Θ(n).

Comparison Summary
O(n) → Worst case
Ω(n) → Best case
Θ(n) → Tight bound

Why These Notations Matter
They help developers choose efficient algorithms and compare performance independently from hardware or programming language.