Q1.Explain Big O notation and how it helps in analyzing algorithms.

**ANS: Big O notation** is a mathematical way to describe how the **performance of an algorithm scales** with the size of the input. It focuses on the **growth rate** rather than exact timing, allowing you to compare algorithms regardless of hardware or implementation details.

* Helps in evaluating and comparing algorithm efficiency.
* Predicts scalability and performance bottlenecks.
* Aids in choosing the best algorithm for large datasets.

Q2.Describe the best, average, and worst-case scenarios for search operations.

ANS:

1. Linear Search (Unsorted Array)

* Best Case (O(1)): Element is at the first position.
* Average Case (O(n)): Element is somewhere in the middle.
* Worst Case (O(n)): Element is at the last position or not found at all.

**2.** Binary Search (Sorted Array)

* Best Case (O(1)): Middle element is the target.
* Average Case (O(log n)): Repeatedly halves the search space.
* Worst Case (O(log n)): Element not found after log₂(n) comparisons.