**Step1: Understand Asymptotic Notation: -**

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

Solution: -

Big O notation describes how the runtime or space requirements of an algorithm grow with the input size.

It helps in analyzing algorithms by providing a way to compare their efficiency and predict performance as the input size increases.

For example, O(n) means the time grows linearly with the input size, while O(n^2) means it grows quadratically.

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

Solution: -

->Best Case: When the algorithm performs the minimum number of operations.

Example: The element you’re searching for is at the beginning of the list.

->Average Case: When the algorithm performs an average number of operations over all possible inputs.

Example: The element is somewhere in the middle of the list.

->Worst Case: When the algorithm performs the maximum number of operations.

Example: The element is either at the very end of the list or not present at all.

**Step 4: Analysis: -**

Q1) Compare the time complexity of linear and binary search algorithms.

Solution: -

Time Complexity for Linear Search:

->Best Case: O (1)

->Average Case: O(n)

->Worst Case: O(n)

Time Complexity for Binary Search:

->Best Case: O (1)

->Average Case: O (log n)

->Worst Case: O (log n)

Q2) Discuss which algorithm is more suitable for your platform and why.

Solution: -

For an E-commerce platform with large inventory binary search is preferred.

Because of its faster search time. However, the array must be sorted.