Step 1: Understand Asymptotic Notation\

Big O Notation:

Big O notation is a mathematical notation used to describe the upper bound of an algorithm's running time. It gives an idea of the worst-case scenario in terms of

performance and helps in understanding how an algorithm scales with the input size.

For example, O(n) means the running time increases linearly with the input size, while O(log n) means the running time increases logarithmically

Best, Average, and Worst-Case Scenarios for Search Operations

Best Case: The scenario where the search operation completes in the least amount of time. For example, in a linear search, finding the target element at the first position.

Average Case: The expected time for the search operation over all possible inputs. It represents the expected performance of the algorithm.

Worst Case: The scenario where the search operation takes the maximum amount of time. For example, in a linear search, not finding the element or finding it at the last position

4. Analysis:

Compare the time complexity of linear and binary search algorithms.

Linear Search:

Best Case: O(1) - The element is found at the first position.

Average Case: O(n) - The element is found somewhere in the middle.

Worst Case: O(n) - The element is not found or is at the last position.

Linear search checks each element in the array sequentially until the target element is found or the end of the array is reached. This results in a time complexity that is directly proportional to the size of the array.

Binary Search:

Best Case: O(1) - The element is found at the middle position initially.

Average Case: O(log n) - The search space is halved at each step.

Worst Case: O(log n) - The element is not found, and the search space is minimized to a single element.

Binary search works by repeatedly dividing the search interval in half. If the target value is less than the middle element, the search continues on the left half, otherwise, it continues on the right half. This results in a logarithmic time complexity.

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

Binary Search is More Suitable: For an e-commerce platform, where fast search performance is crucial and the dataset is large, binary search is more suitable due to its logarithmic time complexity. Maintaining a sorted array or using data structures that support efficient sorted insertions (such as balanced trees or binary search trees) can ensure that the search operations remain fast and responsive. This optimization enhances the user experience by providing quick search results and efficiently handling large volumes of data