**1.Understand Asymptotic Notation:**

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

Big O Notation:

Definition: Big O notation characterizes the upper bound of time complexity for an algorithm. It characterizes functions based on their growth rates and hence gives a high-level understanding of the efficiency of the algorithm with respect to input size 'n'.

Purpose: It is used in analyzing the performance of algorithms. It gives the insight of the worst-case, hence helping a developer to make a decision on which algorithm to use***.***

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

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

Best Case: This is the situation when the search operation has to compare the target element in the first sight itself. In the case of a linear search, this happens when the target element is at the beginning of the list—O(1). For binary search, this will happen if the searched element is exactly in the middle of a sorted list—O(1).

Average Case: This is the scenario which explains or provides the average performance of the search operation. In the case of a linear search, this is O(n/2), which ultimately simplifies to O(n). For binary searches, it is O(log n).

Worst Case: This refers to when the search operation uses a maximum number of comparisons in finding the target element or to determine that the element is not in the list. For linear search, this can be expressed as O(n). For binary search, it is O(log n).

**2.Analysis**

***Time Complexity:***

Linear Search:

Best Case: O(1) (element found at the beginning)

Average Case: O(n)

Worst Case: O(n) (element not found or at the end)

Binary Search:

Best Case: O(1) (element found at the middle)

Average Case: O(log n)

Worst Case: O(log n) (element not found)

Comparison:

Linear Search: Applicable to small datasets or in unsorted data. It doesn't mandatorily require the array to be sorted.

Binary Search: This technique works pretty good with large data sets where the array has been sorted. This really brings down the number of comparisons to be made against an array as compared to a linear search***.***

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

Binary search is also preferred when dealing with an e-commerce platform with many products, since the time complexity is logarithmic. But for this, the product list has to be sorted on the basis of the attribute on which the search is done, such as productId. If the data are not sorted and a sort is not efficiently done, then linear search can be used, but may not give the best performance on large datasets.