1.Understand Asymptotic Notation:

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

Ans: **Big O Notation** expresses an algorithm's time or space complexity in terms of input size n. It describes the upper bound of an algorithm's growth rate, providing a worst-case scenario. Big O notation analysis of algorithms helps in the selection of the most effective algorithm by developers, guaranteeing peak performance, particularly for huge datasets.

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

Ans: 1. Best Case: The shortest amount of time an algorithm needs to finish, typically when the requested element is located at the initial position. For instance, if the element is the first item in a linear search then it is O(1).   
2. Average Case: The anticipated amount of time needed for an algorithm to finish using all potential input. For instance, O(n/2) for linear search.

3. Worst Case: The longest an algorithm can run, usually when the last or desired element cannot be located. When the element is not in the array then linear search time is O(n).

4. Analysis:

Compare the time complexity of linear and binary search algorithms.

Ans:

Linear Search

Best case – O(1), Average case – O(n), Worst case - O(n)

Binary Search

Best case – O(1), Average case – O(log n) , Worst case - O(log n)

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

Ans: **Binary Search** is more suitable for the e-commerce platform due to its superior time complexity, allowing for faster search operations on large, sorted datasets. While linear search is simpler and works on unsorted data, the requirement for sorting data in binary search is manageable and often necessary for optimizing search performance.