DN 4.0 WEEK 1 ANALYSIS

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

Big O notation is a mathematical representation used to describe the time or space complexity of an algorithm in terms of its input size, typically denoted as n. It provides an upper bound on the growth rate of an algorithm's resource usage, allowing developers to compare the efficiency of different algorithms regardless of hardware or implementation details.

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

**·** **Best-case:** This is the most favorable scenario where the desired element is found immediately, such as at the beginning of a list. For example, in linear search, this results in O(1) time.

**· Average-case:** This represents the expected performance over a typical set of inputs, assuming a random distribution. In linear search, it usually takes O(n) time on average since the element might be somewhere in the middle.

**· Worst-case**: This is the least favorable scenario where the element is either at the very end or not present at all, requiring the algorithm to examine every item. For linear search, this is O(n); for binary search on a sorted array, it is O(log n).

1. **Which of the two search algorithms is more suitable (Binary or Linear Search).**

For an e-commerce platform where users expect fast and responsive search results, **binary search is more suitable**—provided that the product list is sorted by the search key (e.g., product name or ID). Binary search significantly reduces the number of comparisons, especially when dealing with large product catalogs, making it ideal for performance-sensitive applications. While linear search can be simpler and works on unsorted data, it becomes inefficient as the dataset grows.