**Exercise 2: E-commerce Platform Search Function**

**Scenario:**

You are working on the search functionality of an e-commerce platform. The search needs to be optimized for fast performance.

**Steps:**

1. **Understand Asymptotic Notation:**

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

Big O notation is a mathematical notation used to describe the upper bound of an algorithm's runtime or space complexity. Generally, it is represented as f(n) = O(g(n)) where f(n) is a function that represents the number of operations (steps) that an algorithm performs to solve a problem of size n.

(i) It provides a way to compare the performance of different algorithms and data structures, and to predict how they will behave as the input size increases.

(ii) Helps in understanding the scalability of algorithms and predicting how they will perform as the input size grows.

(iii) Enables developers to optimize code and improve overall performance.

(iv) Allows programmers to compare different algorithms and choose the most efficient one for a specific problem.

* Describe the best, average, and worst-case scenarios for search operations.
* Best Case: For a search algorithm, best case is when the element to be searched is at the first.
* Average case: When the element to be searched in lying somewhere in the middle.
* Worst Case: In the worst case, the element to be searched is either not present or is at the last.

1. **Setup:**
   * Create a class **Product** with attributes for searching, such as **productId, productName**, and **category**.
2. **Implementation:**
   * Implement linear search and binary search algorithms.
   * Store products in an array for linear search and a sorted array for binary search.
3. **Analysis:**

* Compare the time complexity of linear and binary search algorithms.

1. Linear Search:

* Best Case: O(1) - The desired element is the first element.
* Average Case: O(n/2) ~ O(n) - The desired element is somewhere in the middle.
* Worst Case: O(n) - The desired element is the last element or not present.

1. Binary Search:

* Best Case: O(1) - The desired element is the middle element.
* Average Case: O(log n) - The algorithm divides the search interval in half each time.
* Worst Case: O(log n) - The desired element is not present or at the end of the search.
* Discuss which algorithm is more suitable for your platform and why.

For an E-Commerce platform, I think Binary search is more suitable because of the following reasons :-

1. Since binary search has time complexity of O(log n) it is significantly faster and more efficient than linear search for large records(based on divide and conquer).
2. E-Commerce platform often need to sort products based on user's preference so Binary search will be beneficial for sorted data.