**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 concept used to express the upper limit of an algorithm’s runtime or space complexity. It is generally represented as f(n)=O(g(n)), where f(n) signifies the number of operations (steps) an algorithm performs to solve a problem of size n.

(i) It allows for comparing the performance of various algorithms and data structures and predicting their behavior as the input size grows.

(ii) It aids in understanding the scalability of algorithms and forecasting their performance with increasing input sizes.

(iii) It helps developers in optimizing code and enhancing overall performance.

(iv) It enables programmers to assess different algorithms and select the most efficient one for a particular problem.

* Describe the best, average, and worst-case scenarios for search operations.
* **Best Case:** The search element is at the first position.
* **Average Case:** The search element is located somewhere in the middle of the dataset.
* **Worst Case:** The search element is either not present in the dataset or is located at the last position.

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 element to be found is the first one in the array.
* Average Case: O(n/2) ~ O(n) - The element to be found is somewhere in the middle of the array.
* Worst Case: O(n) - The element to be found is either the last one or not present in the array.

1. Binary Search:

* Best Case: O(1) - The element to be found is the middle element.
* Average Case: O(log n) - The search interval is halved with each step.
* Worst Case: O(log n) - The element to be found is not present or is at the end of the search interval.
* 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. Binary search has a time complexity of O(log⁡n)O(\log n)O(logn), making it significantly faster and more efficient than linear search for large datasets due to its divide-and-conquer approach.
2. E-commerce platforms often require sorting products based on user preferences, so binary search, which works efficiently with sorted data, would be beneficial.