**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.

Solution:

Big O notation is a mathematical representation used to describe the upper bound of an algorithm's runtime or space requirements in terms of the input size. It helps in understanding the worst-case scenario of how an algorithm performs as the input size grows.

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

Solution:

Best Case: The scenario where the algorithm performs the fewest possible operations. For example, in searching, this could be finding the target at the first position.

Average Case: The expected runtime over all possible inputs of size nnn.

Worst Case: The scenario where the algorithm performs the most possible operations. In searching, this is typically not finding the target until the last position or not finding it at all.

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.

Solution:

Setup and Implemention is in attached java file.

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

**Time Complexity**

* **Linear Search**: O(n)
  + In the worst case, linear search will check each element once, leading to a time complexity of O(n).
* **Binary Search**: O(logn)
  + Binary search repeatedly divides the array in half, leading to a time complexity of O(logn).
  + Discuss which algorithm is more suitable for your platform and why.

Solution:

**Suitability for the Platform**

* **Linear Search**: Simpler and doesn't require the array to be sorted, but is less efficient for large datasets.
* **Binary Search**: More efficient for large datasets due to its logarithmic time complexity, but requires the array to be sorted.