**Comparison of Time Complexity:**

1. **Linear Search:**

**Time Complexity:** O(n)

**Explanation:** In the worst-case scenario (and average case), linear search has to iterate through every element in the list to find the target or determine it's not present. If there are n elements, it might perform up to n comparisons. Its performance scales linearly with the size of the input.

2.**Binary Search:**

**Time Complexity:** O(logn)

**Explanation:** Binary search works by repeatedly dividing the search interval in half. With each comparison, it eliminates half of the remaining elements. This means that the number of comparisons grows logarithmically with the number of elements. For instance, if you double the number of elements, you only need one more comparison. This makes it significantly faster for large datasets.

**Suitability of Searching Algorithm:**

For an e-commerce platform's search functionality, **Binary Search is overwhelmingly more suitable than Linear Search, especially for searching by a unique identifier like** productId **where the data can be sorted.**

****Reasons for suitability:****

**Speed for Large Datasets:** E-commerce platforms typically deal with millions of products. Linear search would become incredibly slow and unresponsive as the number of products grows. A search operation that takes seconds or even minutes for a linear search would take milliseconds with binary search. The difference between O(n) and O(logn) becomes massive with large values of n.

**Scalability:** As the product catalog expands, binary search maintains its efficiency much better than linear search. Adding more products only marginally increases the search time for binary search, whereas linear search time would increase proportionally.

**User Experience:** Fast search results are crucial for a good user experience on an e-commerce site. Users expect immediate feedback. Binary search provides this speed.