**Understanding Asymptotic Notation**

**Big O Notation**

Big O notation is a mathematical notation that describes the complexity of an algorithm, which is the amount of time or space it requires as the size of the input increases. It's used to classify algorithms according to how their run time or space requirements grow as the input size grows.

In the context of search operations, Big O notation helps us analyze the efficiency of different search algorithms. It provides an upper bound on the number of steps an algorithm takes to complete, relative to the size of the input.

**Best, Average, and Worst-Case Scenarios**

When analyzing search algorithms, we consider three scenarios:

* **Best-case scenario**: The algorithm performs optimally, and the search operation is successful in the minimum number of steps.
* **Average-case scenario**: The algorithm performs reasonably well, and the search operation takes an average number of steps.
* **Worst-case scenario**: The algorithm performs poorly, and the search operation takes the maximum number of steps.

For search operations, the best-case scenario is often when the target element is at the beginning of the search space, the average-case scenario is when the target element is somewhere in the middle, and the worst-case scenario is when the target element is at the end or not present at all.

**Time Complexity Comparison**

* **Linear Search**: O(n), where n is the number of products in the array. In the worst-case scenario, the algorithm checks every element in the array.
* **Binary Search**: O(log n), where n is the number of products in the array. In the worst-case scenario, the algorithm divides the search space in half with each iteration.

**Algorithm Suitability**

Based on the time complexity analysis, binary search is more suitable for our e-commerce platform search function. This is because binary search has a much faster search time, especially for large datasets. Additionally, binary search requires the array to be sorted, which can be done efficiently using algorithms like quicksort or mergesort.

In contrast, linear search has a slower search time and is not suitable for large datasets. However, it's simpler to implement and doesn't require the array to be sorted.

Overall, binary search is a better choice for our platform due to its faster search time and scalability.