1. **What I Understood – Asymptotic Notation**

While working on this task, I understood that asymptotic notation (especially Big O notation) is a way to measure how efficient an algorithm is, especially when the input size becomes very large.

It tells us how the time or steps taken by an algorithm grow as more data is added.

For example, in linear search, the time increases linearly with more products — if there are 1000 products, it may check each one.

But in binary search, the algorithm cuts the search space in half each time, so it's much faster.

I also learned the difference between:

* Best Case: When the product is found quickly (e.g., first try).
* Average Case: Normal expected performance.
* Worst Case: When the product is not found, or it’s at the very end.

Understanding this helped me realize why binary search is more efficient for sorted data.

2. **What I Understood – Analysis**

After writing both search algorithms, I compared their performance and understood the following:

Linear Search checks every item one by one. It’s simple and works even if the products aren’t sorted. But for large data, it becomes slow.

Binary Search is faster because it uses a divide-and-conquer approach, but it needs the data to be sorted first.

I learned that:

* Linear Search = O(n) → Slower with large datasets
* Binary Search = O(log n) → Much faster for large sorted datasets

So, for a real-world e-commerce platform:

* If the product list is not sorted and small → Linear search is okay.
* If the product list is sorted and big → Binary search is way more efficient.

This comparison helped me understand which search method to use and when — based on data size and whether it's sorted or not.