**1. Understand Asymptotic Notation**

**Big O Notation**

* **Big O notation** is used to describe the upper bound of an algorithm's running time. It gives the worst-case scenario of how an algorithm performs as the input size grows.
* Big O notation helps in understanding the efficiency and scalability of an algorithm by focusing on the term that grows the fastest as the input size increases.

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

* **Best-case scenario:** The minimum time an algorithm takes to complete, often occurring when the element being searched for is the first element in the data structure.
* **Average-case scenario:** The expected time an algorithm takes to complete, averaged over all possible inputs.
* **Worst-case scenario:** The maximum time an algorithm takes to complete, often occurring when the element being searched for is the last element or not present at all.

### 4. Analysis

#### Time Complexity Comparison

* **Linear Search:**
  + Best-case: O(1) (if the product is the first element)
  + Average-case: O(n)
  + Worst-case: O(n) (if the product is the last element or not present)
* **Binary Search:**
  + Best-case: O(1) (if the product is the middle element)
  + Average-case: O(log n)
  + Worst-case: O(log n) (if the product is not present)

#### Which Algorithm is More Suitable?

* **Linear Search:**
  + Suitable for small datasets or unsorted data.
  + Easy to implement and requires no preprocessing.
* **Binary Search:**
  + Suitable for large datasets that are sorted.
  + More efficient for large datasets due to its logarithmic time complexity.
  + Requires data to be sorted, which can add preprocessing time (O(n log n) for sorting).