# Time Complexity Analysis of Operations in E-commerce Product Search

## 1. Chosen Data Structure

In the e-commerce product search problem, we used an array (or ArrayList) to store the products. This structure allows easy access to elements via indexing and is simple to implement for small to moderately sized datasets.

## 2. Time Complexity of Operations

Here is the analysis of time complexity for each operation:

- Add Operation: O(1) (if adding at the end of the array without resizing).

- Update Operation: O(n) (if you need to search for the product first by name).

- Delete Operation: O(n) (as you must find the product and shift elements after deletion).

- Search Operation:

- Linear Search: O(n)

- Binary Search: O(log n) (only if the array is sorted by product name)

## 3. Optimizations

To optimize the operations for large-scale product catalogs, consider the following:

- Use a HashMap with productId or productName as key to reduce search, update, and delete operations to O(1) on average.

- Maintain a TreeMap if sorted access is needed, with O(log n) time for most operations.

- For efficient full-text search (e.g., keyword-based), use Trie or a dedicated search library like Apache Lucene.

- Avoid linear search in large datasets; instead, maintain sorted structures and apply binary search or indexed lookup.

In summary, choosing the right data structure based on the access pattern is essential for optimizing performance in an e-commerce system.