**Exercise 1: Inventory Management System**

**Understand the Problem**

Why are Data Structures and Algorithms Essential?

* Large warehouses have thousands of products.
* Efficient search, update, and deletion are key to keeping performance high.
* Without proper structures, operations may become slow and memory-heavy.

**Suitable Data Structures:**

| **Data Structure** | **Use Case** | **Why Suitable?** |
| --- | --- | --- |
| ArrayList | Small inventory | Fast insertion at end, but slow search/update by productId. |
| HashMap | Large inventory (preferred) | Allows fast lookup by productId. All operations are O(1) average case. |
| TreeMap | If sorted order is required | Keeps keys sorted, but slower than HashMap (O(log n) operations). |

**Analysis:**

| Operation | Average Time Complexity | Why? |
| --- | --- | --- |
| Add | O(1) | HashMap.put() inserts by key instantly. |
| Update | O(1) | Lookup by productId is O(1). |
| Delete | O(1) | Remove by key is fast. |
| List Inventory | O(n) | You loop through all products. |

**Optimization Ideas:**

* If you need sorted listing: use TreeMap instead of HashMap.
* For range queries (e.g., products with price > 100), use secondary indexes (like TreeSet by price).
* For bulk operations, consider batching updates.