**Why Data Structures & Algorithms Are Essential:**

* **Efficiency**: Managing thousands of products requires fast lookup, insertion, and deletion operations.
* **Scalability**: Good algorithms and structures ensure that performance remains optimal as the inventory grows.
* **Organization**: Helps categorize, sort, and retrieve products systematically.
* **Real-time Access**: Allows for quick updates and checks (e.g., low stock alerts).

**Suitable Data Structures:**

* **HashMap (Dictionary in Python, HashMap in Java)**: Best for fast lookup, insertion, and deletion using productId as the key.
* **ArrayList (List in Python)**: Useful for ordered data but has slower lookups if used for searching by productId.
* **TreeMap**: If sorted order by key is needed.

**Best Choice**: HashMap — O(1) average case for add, update, delete.

Time Complexity:

| **Operation** | **Time Complexity (HashMap)** |
| --- | --- |
| Add Product | O(1) on average |
| Update Product | O(1) on average |
| Delete Product | O(1) on average |
| Search Product | O(1) on average |

**Optimization Ideas:**

* **Load Factor and Rehashing**: Ensure the load factor stays within optimal range to avoid collisions.
* **Database Integration**: For very large inventories, move storage from memory to a database with indexed access.
* **Concurrent Access**: Use concurrent hash maps or synchronization to handle multi-threaded environments.
* **Batch Operations**: For efficiency, batch insert or update when processing multiple items.