**EXERCISE 1 – INVENTORY MANAGEMENT SYSTEM**

**Explain why data structures and algorithms are essential in handling large inventories.**

Data structures and algorithms form the backbone of system design, they enable the efficient handling and processing of data in a variety of applications. They are crucial for designing scalable and performant systems.

The main reasons would include,

* Efficiency in storage, retrieval and other operations that are expected to be performed in the inventories
* Scalability to accommodate growing size of the inventory
* Proper organization of the data
* Memory management

**Discuss the types of data structures suitable for this problem**

The data structures suitable for this operation are,

* HashMap
* ArrayList

**ArrayList**

ArrayList is a part of the Java collections framework and it is a class of java.util package. It provides us with dynamic arrays.

This data structure can be preferred when order of items is important.

The time complexity of using array list

* Insertion: O(1)
* Removal: O(N)
* Traversal: O(N)
* Replacement: O(1)

**HashMap**

HashMap stores the data in (Key, Value) pairs, and we can access them by an index of another type (e.g. an Integer). One object is used as a key (index) to another object (value).

This key-value mapping is ideal for quick lookup by key, where the key can be the product ID and the value is the product details.

The HashMap maintains a constant time complexity of O(1) for Insertion, Removal and Extraction.

**Analyze the time complexity of each operation (add, update, delete) in your chosen data structure. Discuss how you can optimize these operations.**

* **Add Operation:**
  + O(1) on average.
  + HashMap operations are generally O(1) due to direct access through hashing. However, in the worst case (rare), it can be O(n) if many elements hash to the same bucket, leading to a linked list or tree traversal.
* **Update Operation:**
  + O(1) on average.
  + Similar to the add operation, finding the product using its ID and updating the value is generally O(1).
* **Delete Operation:**
  + O(1) on average.
  + Removing an item by key in a HashMap involves finding the key and removing the entry

**Optimization**

HashMap provides optimal performance for the operations required in an inventory management system. To further optimize:

* Ensure the HashMap has an optimal load factor to maintain performance.
* Implement batch add/update/delete operations to handle multiple products efficiently.