**Assignment** 3: Inventory Management System Integration in Java

**Duration: 30 minutes**

**Assignment Description**:

This  **Assignment** challenges students to integrate array-based data structures into a broader software system in Java. It requires not only data structure and algorithm expertise but also system-level thinking.

**Questions**:

Integrated Inventory Class: In an Inventory Management System, design a Java class that integrates arrays for inventory management with additional features like sales tracking, reporting, and alerts for restocking. Provide a high-level overview of the class structure and its methods.

Solution:

java

class InventorySystem {

private InventoryItem[] inventory;

private Map<String, Integer> sales;

Other necessary data structures

public InventorySystem(int capacity) {

inventory = new InventoryItem[capacity];

sales = new HashMap<>();

Initialize other data structures

}

public void addItem(String name, double price, int quantity) {

Add item to inventory

}

public void recordSale(String name, int quantity) {

Record sale

}

public void generateReport() {

Generate sales report

}

public void restockAlert() {

Generate restocking alerts

}

}

Restocking Optimization Algorithm: Develop an algorithm in Java that optimizes the restocking process in the inventory system. Consider factors like delivery times, demand forecasting, and storage capacity. Explain the algorithm and its benefits.

Solution:

You can use a predictive restocking algorithm that considers past sales data, lead times, and current stock levels to predict the optimal restocking quantity and time. This algorithm optimizes the restocking process by minimizing overstock and understock situations, leading to cost savings and improved customer satisfaction.

Automatic Reordering: Create a feature in the system to automatically reorder items based on predefined stock levels and priorities using Java. Explain the data structures and algorithms used to ensure efficient, timely restocking while minimizing inventory costs.

Solution:

You can use a priority queue (e.g., MinHeap) to manage items that need restocking based on predefined stock levels and priorities. The priority is determined by factors such as urgency, demand, and lead time. The system checks this priority queue regularly and initiates restocking when necessary. This ensures timely restocking of critical items and minimizes costs by optimizing stock levels.