**Exercise 1: Inventory Management System**

**1. Understand the Problem**

**Importance of Data Structures and Algorithms:**

* **Efficient Storage**: Data structures help manage large volumes of data effectively, enabling quick access and modifications.
* **Fast Retrieval**: Algorithms and data structures can significantly speed up operations like searching, sorting, and updating.
* **Scalability**: Efficient data handling allows the system to scale as the inventory grows.

**Suitable Data Structures:**

1. **ArrayList**:
   * **Pros**: Good for scenarios where you frequently access elements by index.
   * **Cons**: Poor performance for insertions and deletions in the middle of the list due to shifting elements.
2. **HashMap**:
   * **Pros**: Provides constant time complexity (O(1)) for insertions, deletions, and lookups based on the key.
   * **Cons**: Does not maintain any order and has higher memory usage compared to simple lists.
3. **TreeMap**:
   * **Pros**: Maintains order of elements and provides O(log n) time complexity for insertions, deletions, and lookups.
   * **Cons**: Slower compared to HashMap for operations due to the log complexity.

For this scenario, **HashMap** is often a good choice as it provides efficient operations for typical inventory management tasks, where product lookup by ID is common.

**3. Implementation**

**public class Product {**

**private String productId;**

**private String productName;**

**private int quantity;**

**private double price;**

**public Product(String productId, String productName, int quantity, double price) {**

**this.productId = productId;**

**this.productName = productName;**

**this.quantity = quantity;**

**this.price = price;**

**}**

**// Getters and Setters**

**public String getProductId() {**

**return productId;**

**}**

**public void setProductId(String productId) {**

**this.productId = productId;**

**}**

**public String getProductName() {**

**return productName;**

**}**

**public void setProductName(String productName) {**

**this.productName = productName;**

**}**

**public int getQuantity() {**

**return quantity;**

**}**

**public void setQuantity(int quantity) {**

**this.quantity = quantity;**

**}**

**public double getPrice() {**

**return price;**

**}**

**public void setPrice(double price) {**

**this.price = price;**

**}**

**@Override**

**public String toString() {**

**return "Product{" +**

**"productId='" + productId + '\'' +**

**", productName='" + productName + '\'' +**

**", quantity=" + quantity +**

**", price=" + price +**

**'}';**

**}**

**}**

**// InventoryManager.java**

**import java.util.HashMap;**

**public class InventoryManager {**

**private HashMap<String, Product> inventory;**

**public InventoryManager() {**

**inventory = new HashMap<>();**

**}**

**// Add product**

**public void addProduct(Product product) {**

**inventory.put(product.getProductId(), product);**

**}**

**// Update product**

**public void updateProduct(String productId, Product updatedProduct) {**

**if (inventory.containsKey(productId)) {**

**inventory.put(productId, updatedProduct);**

**} else {**

**System.out.println("Product not found.");**

**}**

**}**

**// Delete product**

**public void deleteProduct(String productId) {**

**if (inventory.containsKey(productId)) {**

**inventory.remove(productId);**

**} else {**

**System.out.println("Product not found.");**

**}**

**}**

**// Display product**

**public Product getProduct(String productId) {**

**return inventory.get(productId);**

**}**

**// Display all products**

**public void displayAllProducts() {**

**for (Product product : inventory.values()) {**

**System.out.println(product);**

**}**

**}**

**}**

**4. Analysis**

**Time Complexity:**

* **Add Operation: O(1) - Adding a product to the HashMap is a constant time operation.**
* **Update Operation: O(1) - Updating an existing product is also a constant time operation, assuming the product ID is known and valid.**
* **Delete Operation: O(1) - Removing a product from the HashMap is a constant time operation.**
* **Retrieve Operation: O(1) - Retrieving a product by ID is a constant time operation.**

**Optimisation:**

**By using HashMap, we can ensure efficient data operations for the inventory management system, making it well-suited for handling a large number of products efficiently.**