**Inventory Management System**

**Step 1: Understand the problem**

* **Importance of Data Structures & Algorithms**:  
  Efficient handling of large inventories requires quick access, updates, and deletions of product data.
* Good data structures reduce memory usage and improve lookup times.
* Optimized algorithms improve performance when processing thousands of products.
* **Suitable Data Structures**:
  + **ArrayList**:
    - Maintains order of insertion.
    - Good for iterating through all products.
    - Not ideal for searching by productId.
  + **HashMap** (HashMap<String, Product>):
    - Keys: productId
    - Values: Product objects
    - Offers O(1) time complexity for add, update, and delete operations.
    - Best for fast lookups by ID.

**Step 2: Setup**

* Create a new Java project named: InventoryManagementSystem.

**Step 3: Implementation**

import java.util.HashMap;

import java.util.Scanner;

class Product {

    String productId;

    String productName;

    int quantity;

    double price;

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

        this.productId = productId;

        this.productName = productName;

        this.quantity = quantity;

        this.price = price;

    }

    public void display() {

        System.out.println("Product ID: " + productId);

        System.out.println("Name      : " + productName);

        System.out.println("Quantity  : " + quantity);

        System.out.println("Price     : ₹" + price);

        System.out.println();

    }

}

class InventoryManager {

    HashMap<String, Product> inventory = new HashMap<>();

    public void addProduct(Product product) {

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

            System.out.println("Product ID already exists. Use update instead.");

        } else {

            inventory.put(product.productId, product);

            System.out.println("Product added successfully.");

        }

    }

    public void updateProduct(Product product) {

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

            inventory.put(product.productId, product);

            System.out.println("Product updated successfully.");

        } else {

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

        }

    }

    public void deleteProduct(String productId) {

        if (inventory.containsKey(productId)) {

            inventory.remove(productId);

            System.out.println("Product deleted successfully.");

        } else {

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

        }

    }

    public void displayInventory() {

        if (inventory.isEmpty()) {

            System.out.println("Inventory is empty.");

        } else {

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

                product.display();

            }

        }

    }

}

public class InventoryManagementSystem {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        InventoryManager manager = new InventoryManager();

        while (true) {

            System.out.println("\nINVENTORY LIST");

            System.out.println("1. Add Product");

            System.out.println("2. Update Product");

            System.out.println("3. Delete Product");

            System.out.println("4. View All Products");

            System.out.println("5. Exit");

            System.out.print("Enter your choice: ");

            int choice = scanner.nextInt();

            scanner.nextLine();

            if (choice == 5) {

                System.out.println("Exiting");

                break;

            }

            switch (choice) {

                case 1:

                case 2:

                    System.out.print("Enter Product ID: ");

                    String id = scanner.nextLine();

                    System.out.print("Enter Name: ");

                    String name = scanner.nextLine();

                    System.out.print("Enter Quantity: ");

                    int qty = scanner.nextInt();

                    System.out.print("Enter Price: ");

                    double price = scanner.nextDouble();

                    Product product = new Product(id, name, qty, price);

                    if (choice == 1) {

                        manager.addProduct(product);

                    } else {

                        manager.updateProduct(product);

                    }

                    break;

                case 3:

                    System.out.print("Enter Product ID to delete: ");

                    String deleteId = scanner.nextLine();

                    manager.deleteProduct(deleteId);

                    break;

                case 4:

                    manager.displayInventory();

                    break;

                default:

                    System.out.println("Invalid choice.");

            }

        }

    }

}

**Output:**

****

**Step 4: Analysis**

* **Time Complexity (with HashMap)**:
  + Add: O(1)
  + Update: O(1)
  + Delete: O(1)
* **Optimization Tips**:
  + Always use unique productId as the key.
  + Use HashMap for fast operations.
  + Avoid duplicates by checking key existence before insertion.