# Exercise 1: Inventory Management System

***Why Data Structures & Algorithms Matter***

* Efficiency: Large inventories require fast insertion, deletion, and searching operations.
* Scalability: Choosing the right data structure ensures the system can handle growing inventory sizes.
* Performance: Optimized algorithms reduce latency in real-time inventory updates.

***Suitable Data Structures***

| **Data Structure** | **Best For** | **Time Complexity (Search)** | **Time Complexity (Insert/Delete)** |
| --- | --- | --- | --- |
| *ArrayList* | Simple storage, sequential access | O(n) (linear search) | O(1) (end), O(n) (middle) |
| *HashMap* | Fast lookup by key | O(1) average case | O(1) average case |
| *TreeMap* | Sorted storage | O(log n) | O(log n) |
| *HashSet* | Preventing duplicate entries | O(1) average case | O(1) average case |

***Implementation***

Product.java:

package com.example;

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;

}

public String getProductId() {

return productId;

}

public String getProductName() {

return productName;

}

public int getQuantity() {

return quantity;

}

public double getPrice() {

return price;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public void setPrice(double price) {

this.price = price;

}

*@Override*

public String toString() {

return "Product [ID=" + productId + ", Name=" + productName

+ ", Qty=" + quantity + ", Price=Rs" + price + "]";

}

}

InventoryManager.java:

package com.example;

import java.util.\*;

public class InventoryManager {

private Map<String, Product> inventory;

public InventoryManager() {

inventory = new HashMap<>();

}

public void addProduct(Product product) {

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

}

public void updateQuantity(String productId, int newQuantity) {

if (inventory.containsKey(productId)) {

inventory.get(productId).setQuantity(newQuantity);

} else {

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

}

}

public void removeProduct(String productId) {

if (inventory.containsKey(productId)) {

inventory.remove(productId);

} else {

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

}

}

public Product getProduct(String productId) {

return inventory.get(productId);

}

public void displayInventory() {

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

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

}

}

}

Main.java:

package com.example;

public class Main {

public static void main(String[] args) {

InventoryManager manager = new InventoryManager();

manager.addProduct(new Product("P101", "Laptop", 50, 50000.85));

manager.addProduct(new Product("P102", "Mouse", 200, 809.99));

manager.updateQuantity("P101", 45);

Product laptop = manager.getProduct("P101");

System.***out***.println("Found: " + laptop);

System.***out***.println("\nCurrent Inventory:");

manager.displayInventory();

manager.removeProduct("P102");

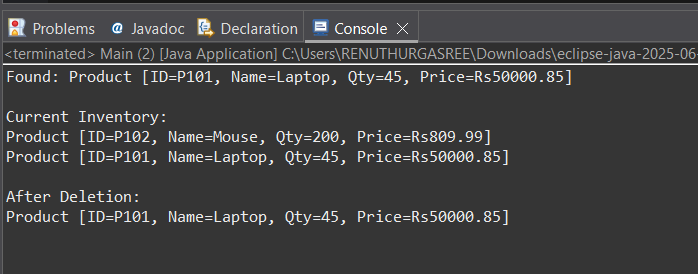
System.***out***.println("\nAfter Deletion:");

manager.displayInventory();

}

}

Output:



***Time complexity***

| **Operation** | **HashMap** |
| --- | --- |
| Add | **O(1)** |
| Update | **O(1)** |
| Delete | **O(1)** |
| Search | **O(1)** |