1) We are looking for a Java-based application that will help us efficiently manage product records using the Collections framework. The system should allow us to:

* Store and manage product data in a structured format.
* Perform key operations such as adding, retrieving, updating, and deleting product records.
* Sort products dynamically based on criteria like product id, product name.
* Prevent duplicate entries to maintain data integrity.

Product entity should contain the following:

Product ID

Product Name

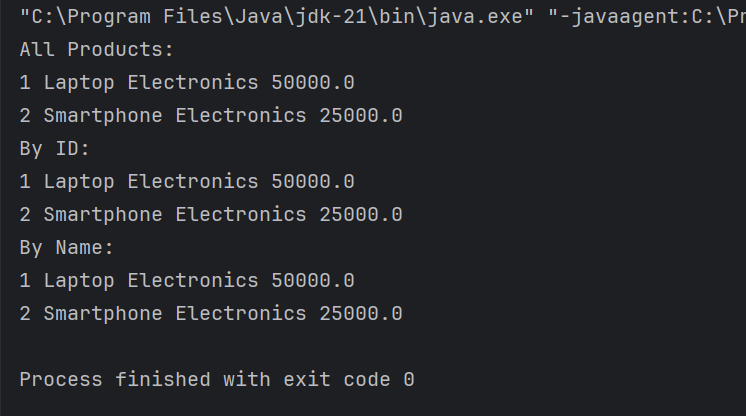
Category

Price

***Code –***

package Assignment;  
import java.util.\*;  
  
class Product { int productId; String productName; String category; double price;  
 Product(int productId, String productName, String category, double price) {  
 this.productId = productId;  
 this.productName = productName;  
 this.category = category;  
 this.price = price;  
 }  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 Product product = (Product) o;  
 return productId == product.productId;  
 }  
 public int hashCode() {  
 return Objects.*hash*(productId);  
 }  
}  
  
class ProductManager { Set<Product> products = new HashSet<>();  
 void addProduct(Product p) {  
 products.add(p);  
 }  
 Product getProductById(int id) {  
 for (Product p : products) {  
 if (p.productId == id) return p;  
 }  
 return null;  
 }  
 void updateProduct(int id, String name, String category, double price) {  
 Product p = getProductById(id);  
 if (p != null) {  
 p.productName = name;  
 p.category = category;  
 p.price = price;  
 }  
 }  
 void deleteProduct(int id) {  
 products.removeIf(p -> p.productId == id);  
 }  
 List<Product> getSortedById() {  
 List<Product> list = new ArrayList<>(products);  
 list.sort(Comparator.*comparingInt*(p -> p.productId));  
 return list;  
 }  
 List<Product> getSortedByName() {  
 List<Product> list = new ArrayList<>(products);  
 list.sort(Comparator.*comparing*(p -> p.productName));  
 return list;  
 }  
 void displayAll() {  
 for (Product p : products) {  
 System.*out*.println(p.productId + " " + p.productName + " " + p.category + " " + p.price);  
 }  
 }  
}  
  
public class ProductCat { public static void main(String[] args) { ProductManager manager = new ProductManager();  
 manager.addProduct(new Product(1, "Laptop", "Electronics", 50000));  
 manager.addProduct(new Product(2, "Tablet", "Electronics", 20000));  
 manager.addProduct(new Product(3, "Chair", "Furniture", 3000));  
 manager.updateProduct(2, "Smartphone", "Electronics", 25000);  
 manager.deleteProduct(3);  
 System.*out*.println("All Products:");  
 manager.displayAll();  
 System.*out*.println("By ID:");  
 for (Product p : manager.getSortedById()) {  
 System.*out*.println(p.productId + " " + p.productName + " " + p.category + " " + p.price);  
 }  
 System.*out*.println("By Name:");  
 for (Product p : manager.getSortedByName()) {  
 System.*out*.println(p.productId + " " + p.productName + " " + p.category + " " + p.price);  
 }  
}  
}

***Output –***



2) Create a product catalogue key as a product and value as quantity:

* Store and manage product data in a structured format.
* Perform key operations such as adding, retrieving, updating, and deleting product records.
* Sort products dynamically based on criteria like product id, product name.
* Prevent duplicate entries to maintain data integrity.

Product entity should contain the following:

Product ID

Product Name

Category

Price

***Code –***

***Output –***