**Exercise 2: E-commerce Platform Search Function**

**Product.java**

public class Product {

    int productId;

    String productName;

    String category;

    public Product(int productId, String productName, String category) {

        this.productId = productId;

        this.productName = productName;

        this.category = category;

    }

    public int getProductId() {

        return productId;

    }

    public String getProductName() {

        return productName;

    }

    public String getCategory() {

        return category;

    }

}

**ProductList.java**

import java.util.\*;

public class ProductList {

    private List<Product> idSortedList = new ArrayList<>();

    private List<Product> nameSortedList = new ArrayList<>();

    private Map<String, List<Product>> categoryMap = new HashMap<>();

    public void addProduct(Product product) {

        idSortedList.add(product);

        idSortedList.sort(Comparator.comparingInt(Product::getProductId));

        nameSortedList.add(product);

        nameSortedList.sort(Comparator.comparing(Product::getProductName, String.CASE\_INSENSITIVE\_ORDER));

        String category = product.getCategory().toLowerCase();

        categoryMap.putIfAbsent(category, new ArrayList<>());

        categoryMap.get(category).add(product);

    }

    public void removeProduct(Product product) {

        idSortedList.remove(product);

        nameSortedList.remove(product);

        String category = product.getCategory().toLowerCase();

        if (categoryMap.containsKey(category)) {

            categoryMap.get(category).remove(product);

            if (categoryMap.get(category).isEmpty()) {

                categoryMap.remove(category);

            }

        }

    }

    public List<Product> getIdSortedList() {

        return idSortedList;

    }

    public List<Product> getNameSortedList() {

        return nameSortedList;

    }

    public Map<String, List<Product>> getCategoryMap() {

        return categoryMap;

    }

}

**SearchUtil.java**

import java.util.List;

import java.util.Map;

public class SearchUtil {

    public static Product binarySearchById(List<Product> idSortedList, int targetId) {

        int low = 0, high = idSortedList.size() - 1;

        while (low <= high) {

            int mid = low + (high - low) / 2;

            Product midProduct = idSortedList.get(mid);

            if (midProduct.getProductId() == targetId) {

                return midProduct;

            } else if (midProduct.getProductId() < targetId) {

                low = mid + 1;

            } else {

                high = mid - 1;

            }

        }

        return null;

    }

    public static Product binarySearchByName(List<Product> nameSortedList, String targetName) {

        int low = 0, high = nameSortedList.size() - 1;

        while (low <= high) {

            int mid = low + (high - low) / 2;

            Product midProduct = nameSortedList.get(mid);

            int cmp = midProduct.getProductName().compareToIgnoreCase(targetName);

            if (cmp == 0) {

                return midProduct;

            } else if (cmp < 0) {

                low = mid + 1;

            } else {

                high = mid - 1;

            }

        }

        return null;

    }

    public static List<Product> linearSearchByCategory(Map<String, List<Product>> categoryMap, String category) {

        return categoryMap.getOrDefault(category.toLowerCase(), null);

    }

}

**Test.java**

import java.util.List;

public class Test {

    public static void main(String[] args) {

        ProductList productList = new ProductList();

        productList.addProduct(new Product(101, "iPhone", "Electronics"));

        productList.addProduct(new Product(102, "Shoes", "Footwear"));

        productList.addProduct(new Product(103, "Samsung Galaxy", "Electronics"));

        productList.addProduct(new Product(104, "Dell Laptop", "Computers"));

        productList.addProduct(new Product(105, "Adidas Sneakers", "Footwear"));

        int searchId = 103;

        Product byId = SearchUtil.binarySearchById(productList.getIdSortedList(), searchId);

        if (byId != null) {

            System.out.println("Product found by ID " + searchId + ": " + byId.getProductName());

        } else {

            System.out.println("Product with ID " + searchId + " not found.");

        }

        String searchName = "iPhone";

        Product byName = SearchUtil.binarySearchByName(productList.getNameSortedList(), searchName);

        if (byName != null) {

            System.out.println("Product found by Name \"" + searchName + "\": " + byName.getProductName());

        } else {

            System.out.println("Product with Name \"" + searchName + "\" not found.");

        }

        String searchCategory = "Footwear";

        List<Product> categoryList = SearchUtil.linearSearchByCategory(productList.getCategoryMap(), searchCategory);

        if (categoryList != null && !categoryList.isEmpty()) {

            System.out.println("Products in category \"" + searchCategory + "\":");

            for (Product p : categoryList) {

                System.out.println(p.getProductName());

            }

        } else {

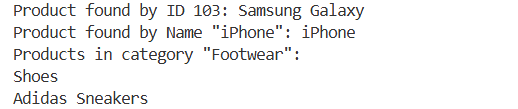
            System.out.println("No products found in category \"" + searchCategory + "\".");

        }

    }

}

**Output**



**Exercise 7: Financial Forecasting**

**FinancialForecast.java**

public class FinancialForecast {

    public static double futureValue(double initialValue, double rate, int years) {

        if (years == 0) {

            return initialValue;

        }

        return futureValue(initialValue, rate, years - 1) \* (1 + rate);

    }

    public static void main(String[] args) {

        double initial = 1000.0;

        double rate = 0.10;

        int years = 5;

        double resultRecursive = futureValue(initial, rate, years);

        System.out.printf("Recursive Forecast after %d years: Rs. %.2f%n", years, resultRecursive);

    }

}

**Output**

