**Exercise 1: Inventory Management System**

**Scenario:**

You are developing an inventory management system for a warehouse. Efficient data storage and retrieval are crucial.

import java.util.\*;

public class Excercise {

    public static void main(String[] args) {

        Product[] products = {

            new Product(101, "Phone", "Electronics"),

            new Product(102, "Shoes", "Fashion"),

            new Product(103, "Laptop", "Electronics"),

            new Product(104, "Watch", "Accessories")

        };

        System.out.println("Linear Search:");

        Product result1 = linearSearch(products, 103);

        System.out.println(result1 != null ? result1 : "Product not found");

        // Sorting the array by productId before using binary search

        Arrays.sort(products, Comparator.comparingInt(p -> p.productId));

        System.out.println("Binary Search:");

        Product result2 = binarySearch(products, 103);

        System.out.println(result2 != null ? result2 : "Product not found");

    }

    public static Product linearSearch(Product[] arr, int targetId) {

        for (Product p : arr) {

            if (p.productId == targetId) {

                return p;

            }

        }

        return null;

    }

    public static Product binarySearch(Product[] arr, int targetId) {

        int left = 0, right = arr.length - 1;

        while (left <= right) {

            int mid = (left + right) / 2;

            if (arr[mid].productId == targetId)

                return arr[mid];

            else if (arr[mid].productId < targetId)

                left = mid + 1;

            else

                right = mid - 1;

        }

        return null;

    }

}

class Product {

    public int productId; // Keep public for direct access

    public String productName;

    public String category;

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

        this.productId = productId;

        this.productName = productName;

        this.category = category;

    }

    public String toString() {

        return "[" + productId + "] " + productName + " - " + category;

    }

}



**Exercise 7: Financial Forecasting**

**Scenario:**

You are developing a financial forecasting tool that predicts future values based on past data.

public class Excercise7 {

    // Recursive method to calculate future value

    public static double predictFutureValue(double presentValue, double rate, int years) {

        if (years == 0) {

            return presentValue;

        }

        return predictFutureValue(presentValue, rate, years - 1) \* (1 + rate);

    }

    public static void main(String[] args) {

        double presentValue = 10000; // Starting value

        double growthRate = 0.05;    // 5% annual growth

        int years = 5;

        double futureValue = predictFutureValue(presentValue, growthRate, years);

        System.out.printf("Future Value after %d years: %.2f%n", years, futureValue);

    }

}

