# Exercise 2: E-commerce Platform Search Function

import java.util.Arrays;

import java.util.Comparator;

class Product {

    int productId; String productName; String category;

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

        this.productId = productId; this.productName = productName; this.category = category;

    }

    public String toString() {

        return "Product ID: "+productId+", Name: "+productName+", Category: "+category;

    }

}

public class EcommerceSearch {

    public static Product linearSearch(Product[] products, String name) {

        for(Product p : products) if(p.productName.equalsIgnoreCase(name)) return p;

        return null;

    }

    public static Product binarySearch(Product[] products, String name) {

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

        while(left<=right) {

            int mid=(left+right)/2;

            int cmp=name.compareToIgnoreCase(products[mid].productName);

            if(cmp==0) return products[mid];

            else if(cmp<0) right=mid-1;

            else left=mid+1;

        }

        return null;

    }

    public static void main(String[] args) {

        Product[] products = {

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

            new Product(102,"Smartphone","Electronics"),

            new Product(103,"Shoes","Fashion"),

            new Product(104,"Book","Education"),

            new Product(105,"Watch","Accessories")

        };

        Product[] sorted = Arrays.copyOf(products, products.length);

        Arrays.sort(sorted, Comparator.comparing(p -> p.productName.toLowerCase()));

        String search = "Shoes";

        Product linear = linearSearch(products, search);

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

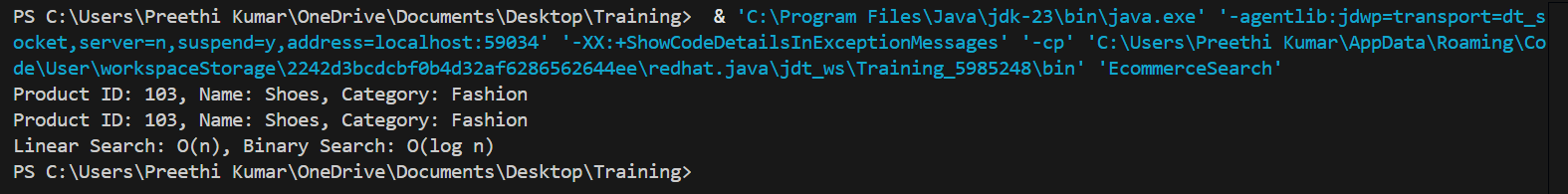
        Product binary = binarySearch(sorted, search);

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

        System.out.println("Linear Search: O(n), Binary Search: O(log n)");

    }

}



# Exercise 7: Financial Forecasting

public class FinancialForecasting {

    public static double futureValue(double presentValue, double growthRate, int years) {

        if (years == 0) return presentValue;

        return futureValue(presentValue, growthRate, years - 1) \* (1 + growthRate);

    }

    private static double[] memo;

    public static double futureValueMemo(double presentValue, double growthRate, int years) {

        if (years == 0) return presentValue;

        if (memo[years] != 0) return memo[years];

        memo[years] = futureValueMemo(presentValue, growthRate, years - 1) \* (1 + growthRate);

        return memo[years];

    }

    public static void main(String[] args) {

        double presentValue = 1000, growthRate = 0.05; int years = 5;

        System.out.printf("Recursive: %.2f%n", futureValue(presentValue, growthRate, years));

        memo = new double[years+1];

        System.out.printf("Memoized: %.2f%n", futureValueMemo(presentValue, growthRate, years));

    }

}

