**Algorithms\_Data Structures**

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

CODE

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;

    }

    @Override

    public String toString() {

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

    }

}

import java.util.Arrays;

import java.util.Comparator;

public class SearchEngine {

    // Linear search by productName

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

        for (Product product : products) {

            if (product.productName.equalsIgnoreCase(targetName)) {

                return product;

            }

        }

        return null;

    }

    // Binary search by productName

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

        int low = 0;

        int high = products.length - 1;

        while (low <= high) {

            int mid = (low + high) / 2;

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

            if (cmp == 0) {

                return products[mid];

            } else if (cmp < 0) {

                low = mid + 1;

            } else {

                high = mid - 1;

            }

        }

        return null;

    }

    public static void sortProducts(Product[] products) {

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

    }

}

public class Main1 {

    public static void main(String[] args) {

        Product[] products = {

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

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

            new Product(103, "Watch", "Accessories"),

            new Product(104, "Phone", "Electronics"),

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

        };

        // Linear Search

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

        Product result1 = SearchEngine.linearSearch(products, "Phone");

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

        // Binary Search

        SearchEngine.sortProducts(products);

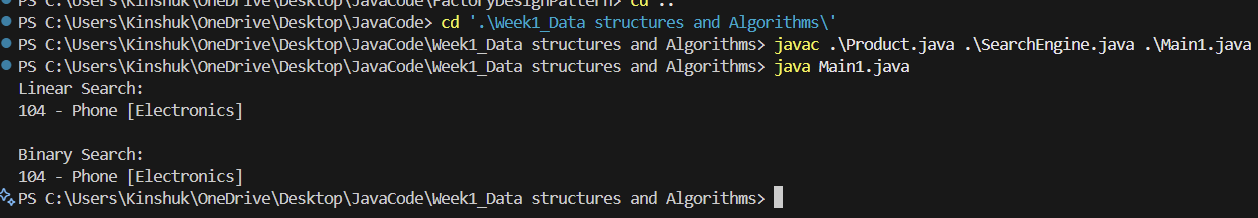
        System.out.println("\nBinary Search:");

        Product result2 = SearchEngine.binarySearch(products, "Phone");

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

    }

}



**Exercise 7: Financial Forecasting**

public class FinancialForecast {

    public static double forecast(double currentValue, double growthRate, int years) {

        if (years == 0) {

            return currentValue;

        }

        return forecast(currentValue \* (1 + growthRate / 100), growthRate, years - 1);

    }

}

public class Main2 {

    public static void main(String[] args) {

        double initialValue = 10000;

        double growthRate = 5;

        int years = 5;

        double predictedValue = FinancialForecast.forecast(initialValue, growthRate, years);

        System.out.printf("Predicted value after %d years: %.2f\n", years, predictedValue);

    }

}

OUTPUT

