**Mandatory hands-on**

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

**Code:**

//ECommerceSearch.java

import java.util.Arrays;

import java.util.Comparator;

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 "Product{" +

               "ID=" + productId +

               ", Name='" + productName + '\'' +

               ", Category='" + category + '\'' +

               '}';

    }

}

public class ECommerceSearch {

    // Linear Search Method

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

        for (Product product : products) {

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

                return product;

            }

        }

        return null;

    }

    // Binary Search Method (requires sorted array)

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

        int low = 0, high = products.length - 1;

        while (low <= high) {

            int mid = (low + high) / 2;

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

            if (comparison == 0) {

                return products[mid];

            } else if (comparison < 0) {

                low = mid + 1;

            } else {

                high = mid - 1;

            }

        }

        return null;

    }

    public static void main(String[] args) {

        Product[] products = {

            new Product(1, "Laptop", "Electronics"),

            new Product(2, "Shoes", "Footwear"),

            new Product(3, "Smartphone", "Electronics"),

            new Product(4, "Backpack", "Accessories")

        };

        // Linear Search Example

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

        Product result1 = linearSearch(products, "phone");

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

        // Sort the array for Binary Search

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

        // Binary Search Example

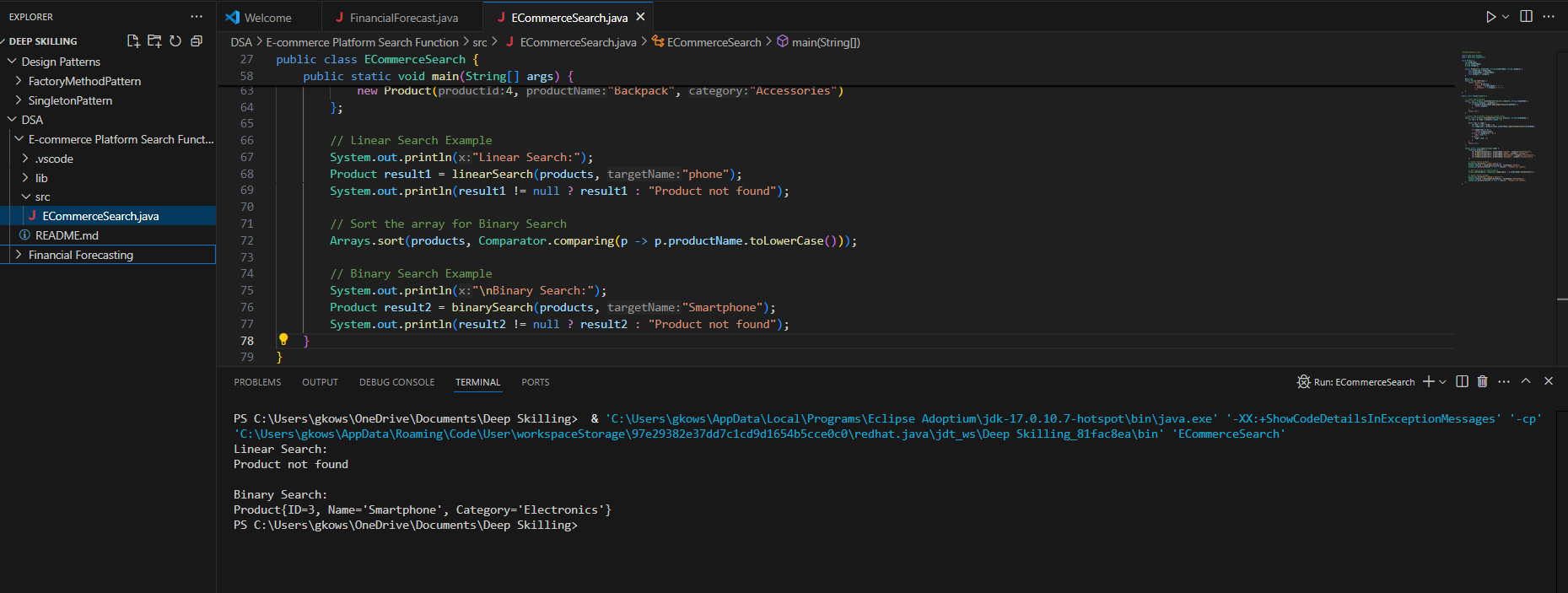
        System.out.println("\nBinary Search:");

        Product result2 = binarySearch(products, "Smartphone");

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

    }

}

**Output:**  


**Exercise 7: Financial Forecasting**

**Code:**

public class FinancialForecast {

// Recursive method to calculate future value

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

// Base case

if (years == 0) {

return presentValue;

}

// Recursive case

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

}

public static void main(String[] args) {

double pv = 10000.0; // Initial investment

double annualRate = 0.08; // 8% growth

int years = 5;

double result = futureValue(pv, annualRate, years);

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

}

}

**Output:**

