**SKILLS : ALGORITHMS\_DATA STRUCTURES**

**Qn 1 : E-commerce Platform Search Function**

**Code :**

**ProductSearch.java**

import java.util.Arrays;

import java.util.Comparator;

class Product {

    int productId;

    String productName;

    String category;

    public Product(int id, String name, String category) {

        this.productId = id;

        this.productName = name.trim();

        this.category = category.trim();

    }

    @Override

    public String toString() {

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

    }

}

public class ProductSearch {

    // Linear Search

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

        name = name.trim();

        for (Product product : products) {

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

                return product;

            }

        }

        return null;

    }

    // Binary Search (requires sorted array)

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

        name = name.trim();

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

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

        while (low <= high) {

            int mid = (low + high) / 2;

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

            if (cmp == 0) {

                return products[mid];

            } else if (cmp < 0) {

                high = mid - 1;

            } else {

                low = mid + 1;

            }

        }

        return null;

    }

    // Main method

    public static void main(String[] args) {

        Product[] products = {

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

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

            new Product(3, "Smart Phone", "Electronics"),

            new Product(4, "Table with chair", "Furniture"),

            new Product(5, "Smart Watch", "Accessories")

        };

        String searchName = "Smart Phone"; //

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

        Product foundLinear = linearSearch(products, searchName);

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

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

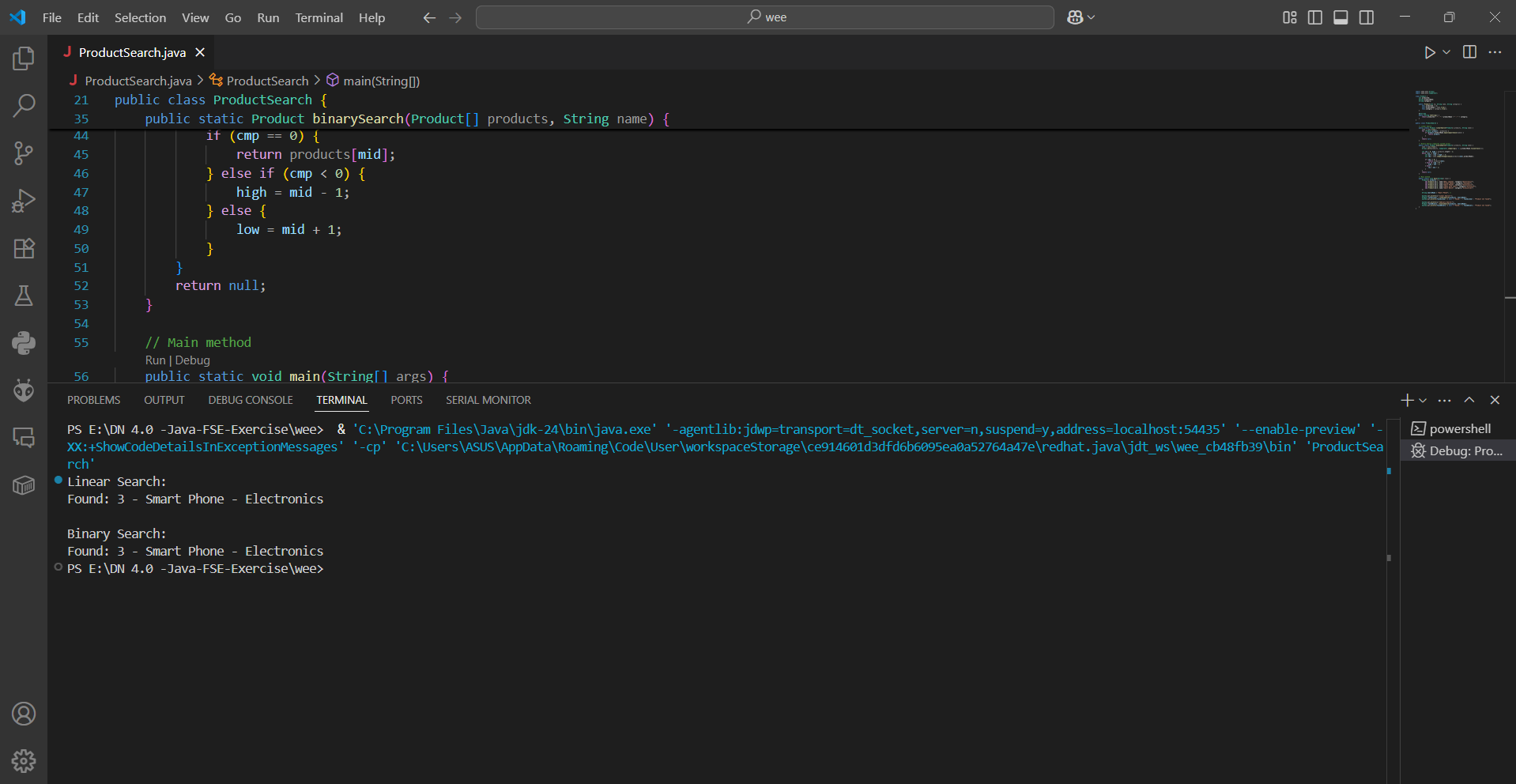
        Product foundBinary = binarySearch(products, searchName);

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

    }

}

**Output :**

****

**Qn 2 : Financial Forecasting**

**Code:**

**FinancialForecast.java**

public class FinancialForecast {

    //  calculate future value

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

        // Base case

        if (years == 0) {

            return initialValue;

        }

        // Recursive case

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

    }

    public static void main(String[] args) {

        double initialValue = 10000;  // Starting investment

        double rate = 0.05;           // 5% annual growth

        int years = 5;

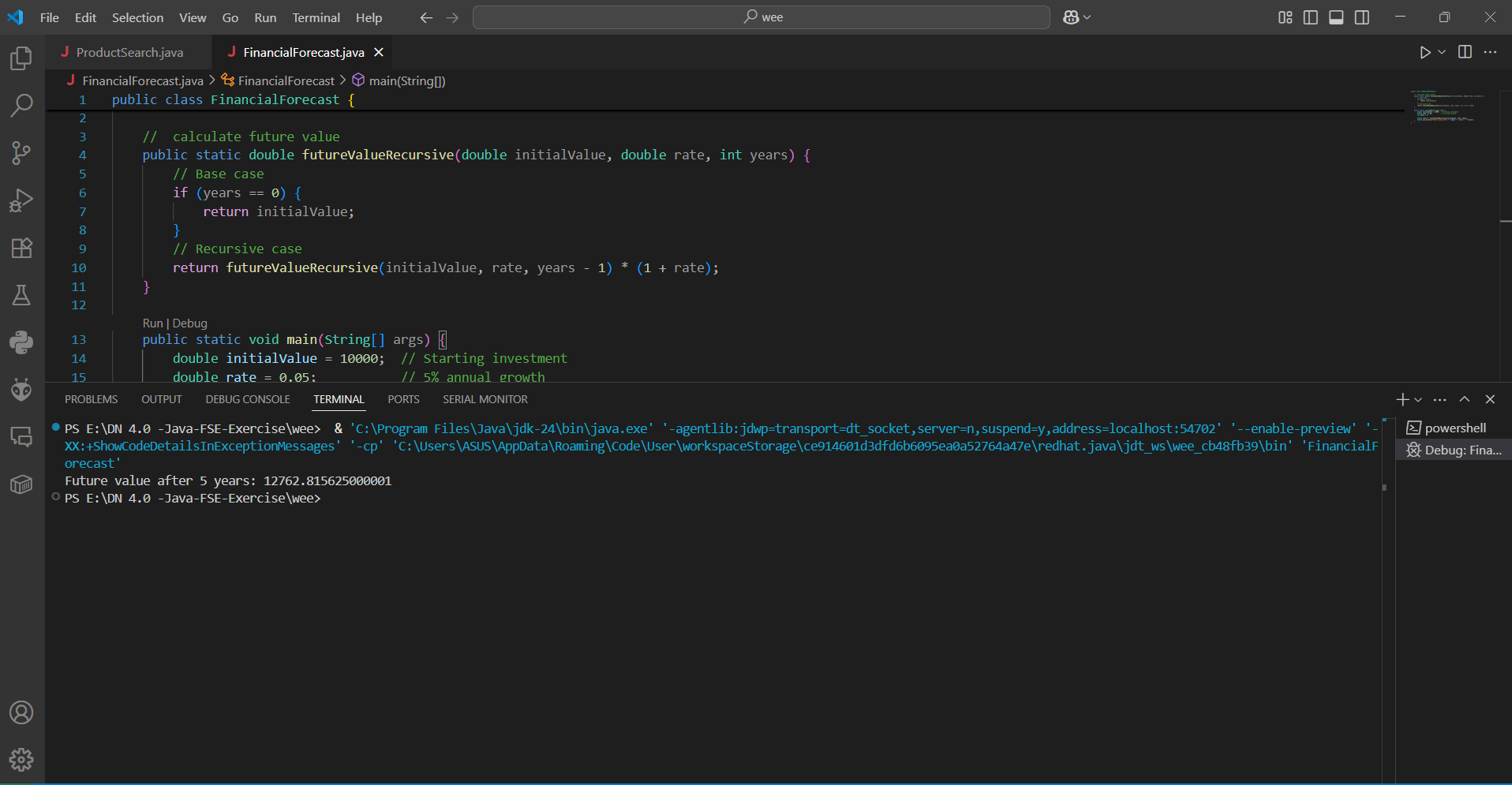
        double result = futureValueRecursive(initialValue, rate, years);

        System.out.println("Future value after " + years + " years: " + result);

    }

}

**Output:**

****