**ALGORITHMS AND DATA STRUCTURES**

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

**1.Product.java**

package ecommerce;

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: " + productId + ", Name: " + productName + ", Category: " + category;

}

}

**2. SearchService.java**

package ecommerce;

import java.util.Arrays;

import java.util.Comparator;

public class SearchService {

// Linear Search by product name

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

for (Product product : products) {

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

return product;

}

}

return null;

}

// Binary Search by product name

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

Arrays.sort(products, Comparator.comparing(p -> p.productName.toLowerCase())); // Sort by name

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

while (left <= right) {

int mid = (left + right) / 2;

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

if (comparison == 0) {

return products[mid];

} else if (comparison < 0) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return null;

}

}

**3.Main.Java**

package ecommerce;

public class Main {

public static void main(String[] args) {

Product[] products = {

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

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

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

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

// Test Linear Search

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

Product foundLinear = SearchService.linearSearch(products, "Mobile");

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

// Test Binary Search

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

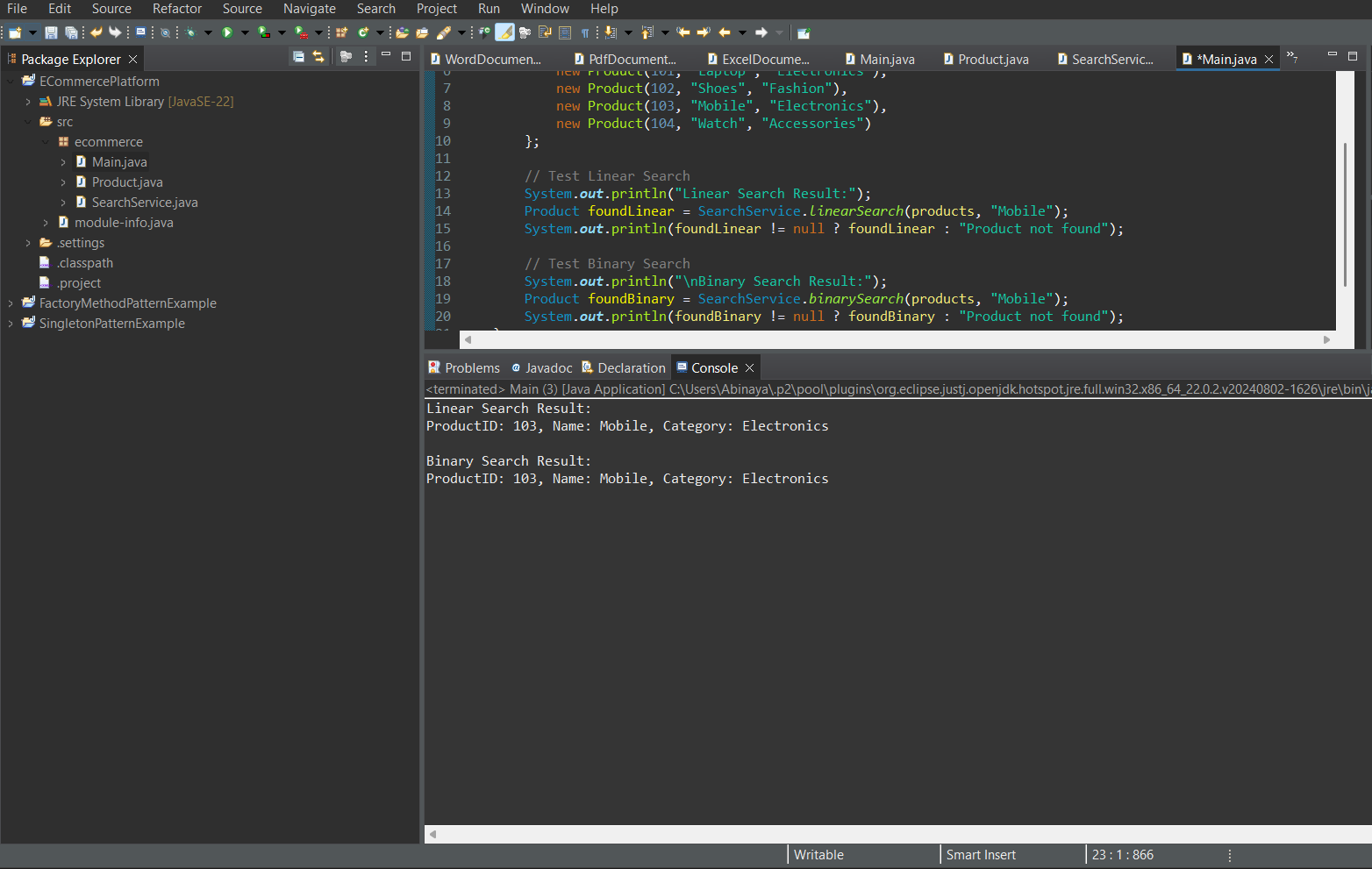
Product foundBinary = SearchService.binarySearch(products, "Mobile");

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

}

}

**Output :**



**Exercise 7: Financial Forecasting**

**1.FinancialForecast.java**

package forecasting;

public class FinancialForecast {

// Recursive method

public static double predictFutureValue(double initialValue, double growthRate, int years) {

// Base case: if no more years, return current value

if (years == 0) {

return initialValue;

}

//Add growth and call recursively

return predictFutureValue(initialValue \* (1 + growthRate), growthRate, years - 1);

}

public static void main(String[] args) {

double initialAmount = 10000; // starting amount

double growthRate = 0.05; // 5% growth per year

int forecastYears = 10;

double futureValue = predictFutureValue(initialAmount, growthRate, forecastYears);

System.***out***.println("Principal = ₹" + initialAmount);

System.***out***.println("Rate = 10% per year");

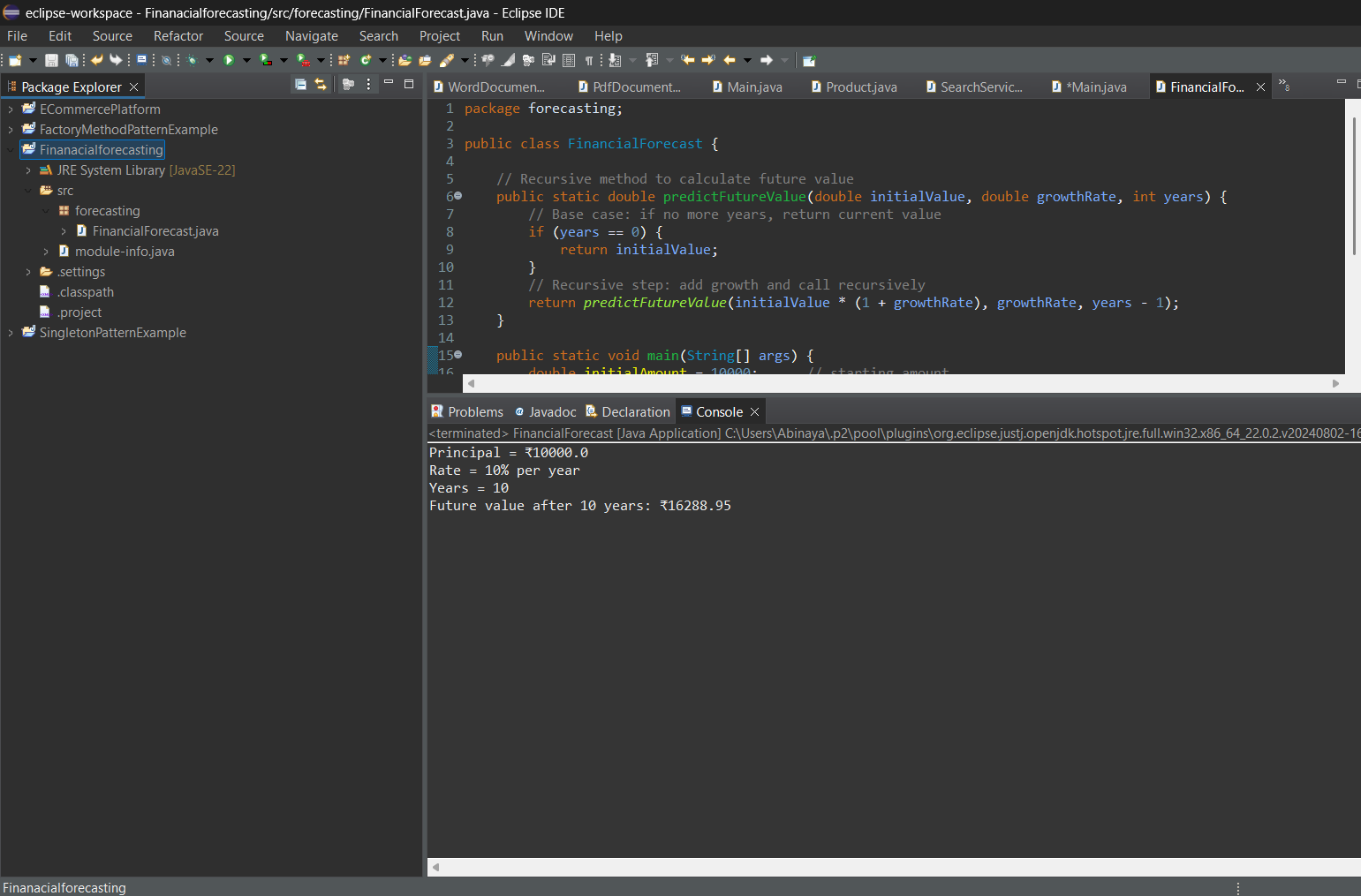
System.***out***.println("Years = " + forecastYears );

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

}

}

**Output:**

****