**WEEK-01**

**MODULE-02:DATA STRUCTURES AND ALGORITHM**

**MANDATORY HANDSON**

**EXERCISE -02: E-commerce Platform Search Function**

Product.java

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;

}

public void displayProduct() {

System.out.println("Product ID: " + productId + ", Name: " + productName + ", Category: " + category);

} }

SearchDemo.java

import java.util.Arrays;

import java.util.Comparator;

public class SearchDemo {

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

for (int i = 0; i < products.length; i++) {

if (products[i].productName.equalsIgnoreCase(targetName)) {

return i;

} }

return -1;

}

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

int left = 0;

int right = products.length - 1;

while (left <= right) {

int mid = (left + right) / 2;

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

if (result == 0) {

return mid;

} else if (result > 0) {

left = mid + 1;

} else {

right = mid - 1;

} }

return -1;

}

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")

};

String target = "Phone";

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

int linearIndex = linearSearch(products, target);

if (linearIndex != -1) {

System.out.println("Product found at index " + linearIndex);

products[linearIndex].displayProduct();

} else {

System.out.println("Product not found.");

}

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

System.out.println("\n=== Binary Search ===");

int binaryIndex = binarySearch(products, target);

if (binaryIndex != -1) {

System.out.println("Product found at index " + binaryIndex);

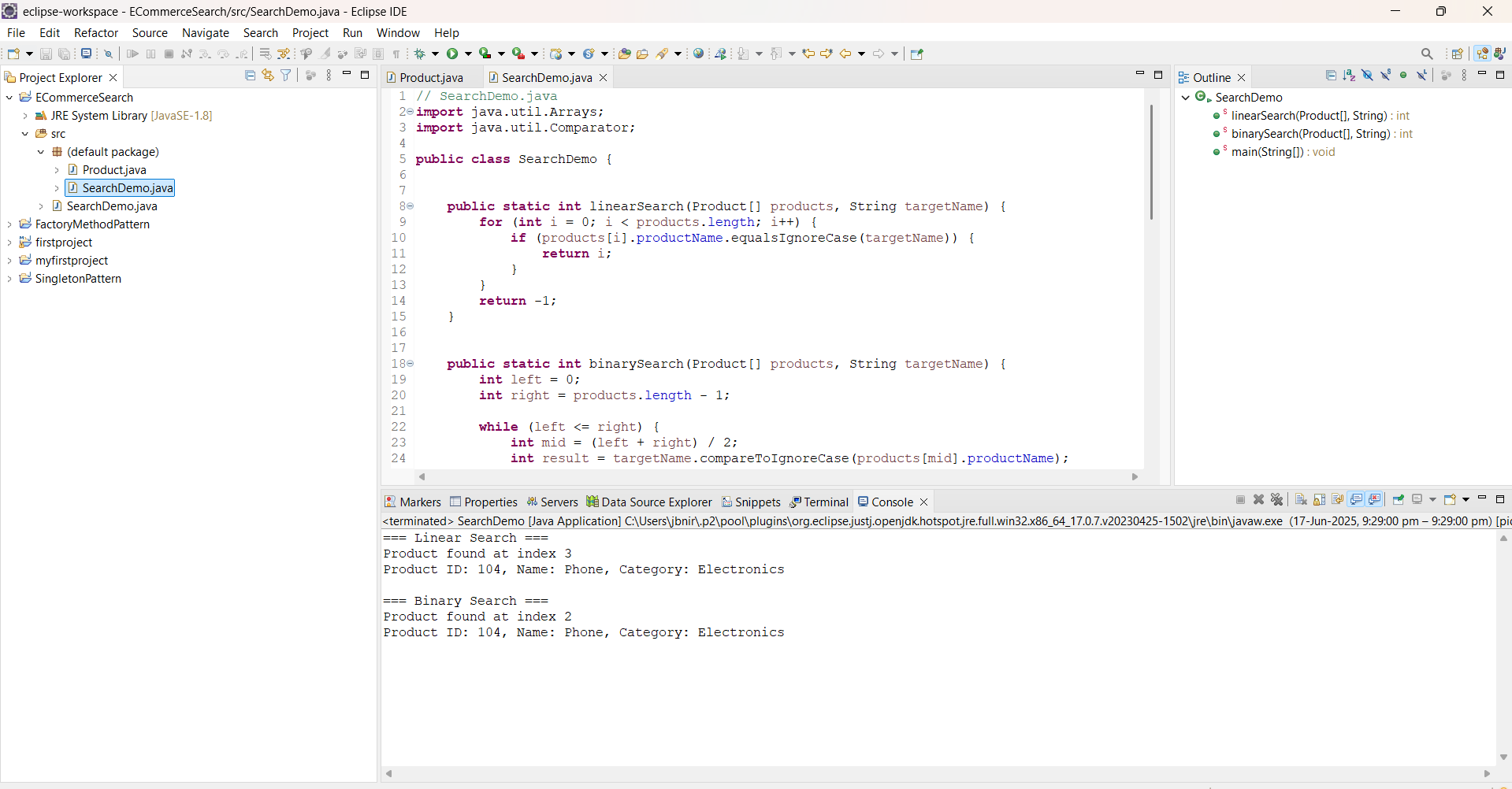
products[binaryIndex].displayProduct();

} else {

System.out.println("Product not found.");

} } }

**OUTPUT**

****



**Exercise 7: Financial Forecasting**

FinancialForecast.java

**import** java.util.Scanner;

**public** **class** FinancialForecast {

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

**if** (years == 0) {

**return** presentValue;

} **else** {

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

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter Present Value:");

**double** presentValue = sc.nextDouble();

System.***out***.println("Enter Growth Rate (in percentage):");

**double** growthRate = sc.nextDouble() / 100;

System.***out***.println("Enter Number of Years:");

**int** years = sc.nextInt();

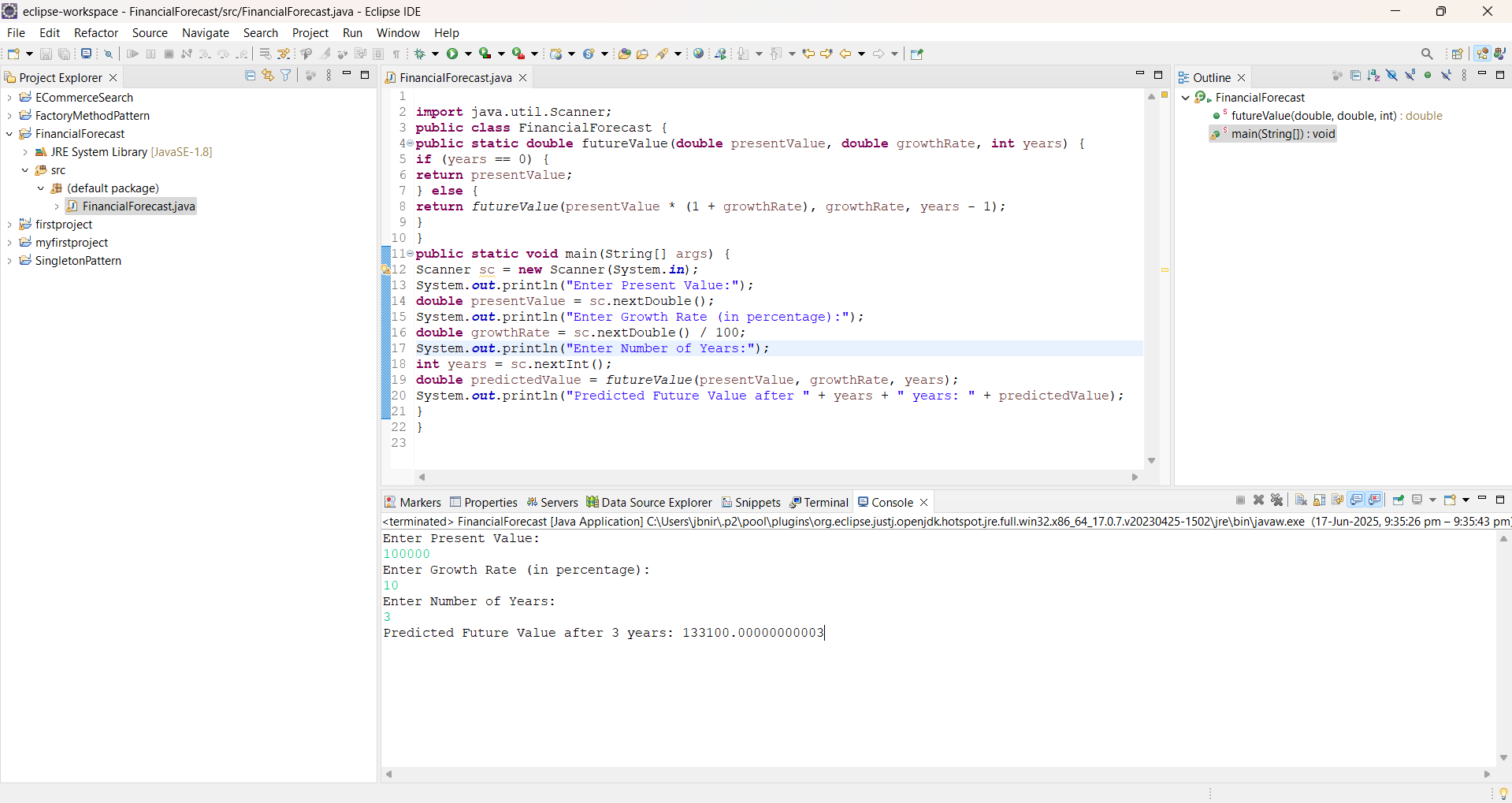
**double** predictedValue = *futureValue*(presentValue, growthRate, years);

System.***out***.println("Predicted Future Value after " + years + " years: " + predictedValue);

}

}

**OUTPUT**

****

