**Exercise 7: Financial Forecasting**

import java.util.Scanner;

public class FinancialForecast {

public static double forecastValue(double presentValue, double growthRate, int years) {

if (years == 0) {

return presentValue;

}

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

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("=== Financial Forecast Calculator ===");

while (true) {

System.out.print("Enter present value (in rupees): ");

double present = sc.nextDouble();

System.out.print("Enter annual growth rate (in percent): ");

double ratePercent = sc.nextDouble();

double growthRate = ratePercent / 100;

System.out.print("Enter number of years: ");

int years = sc.nextInt();

if (present < 0 || ratePercent < 0 || years < 0) {

System.out.println("Please enter non-negative values for all inputs.");

continue;

}

double future = forecastValue(present, growthRate, years);

System.out.printf("Projected value after %d years: ₹%.2f\n", years, future);

System.out.print("Do you want to forecast again? (yes/no): ");

String choice = sc.next().toLowerCase();

if (!choice.equals("yes")) {

System.out.println("Thank you for using the Financial Forecast Calculator!");

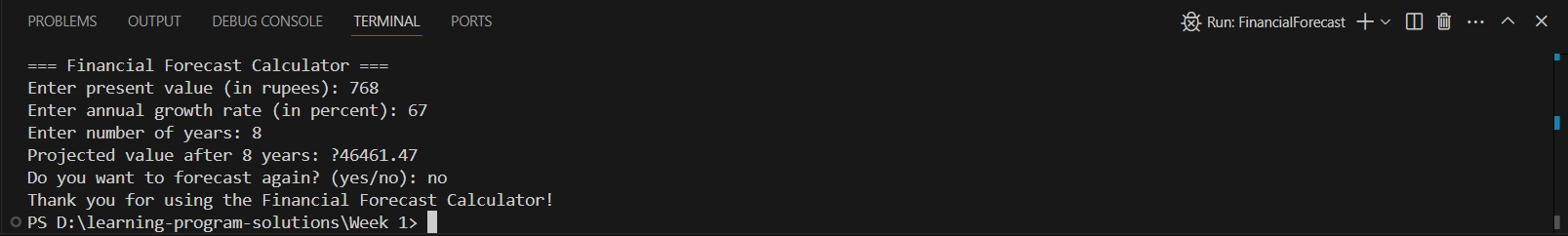
break;

}

}

sc.close();

}

}

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

**Product.java**

public class Product {

int productId;

String productName;

String category;

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

this.productId = id;

this.productName = name;

this.category = category;

}

public String toString() {

return "[" + productId + "] " + productName + " (" + category + ")";

}

}

**SearchDemo.java**

import java.util.Arrays;

import java.util.Comparator;

import java.util.Scanner;

public class SearchDemo {

public static void main(String[] args) {

Product[] products = {

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

new Product(205, "Shoes", "Fashion"),

new Product(150, "Phone", "Electronics"),

new Product(303, "Watch", "Accessories"),

new Product(175, "Bag", "Fashion")

};

Scanner sc = new Scanner(System.in);

System.out.println("=== Available Products ===");

displayAllProducts(products);

System.out.print("\nEnter Product ID to search: ");

int targetId = sc.nextInt();

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

Product result1 = linearSearch(products, targetId);

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

Arrays.sort(products, Comparator.comparingInt(p -> p.productId));

System.out.println("\nSorted Products by ID:");

displayAllProducts(products);

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

Product result2 = binarySearch(products, targetId);

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

System.out.print("\nEnter a category to filter products (e.g., Fashion): ");

sc.nextLine(); // consume leftover newline

String categoryInput = sc.nextLine();

System.out.println("\nProducts in category '" + categoryInput + "':");

filterByCategory(products, categoryInput);

sc.close();

}

static void displayAllProducts(Product[] items) {

for (Product p : items) {

System.out.println(p);

}

}

static Product linearSearch(Product[] items, int targetId) {

for (Product item : items) {

if (item.productId == targetId) return item;

}

return null;

}

static Product binarySearch(Product[] items, int targetId) {

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

while (left <= right) {

int mid = (left + right) / 2;

if (items[mid].productId == targetId) return items[mid];

if (items[mid].productId < targetId) left = mid + 1;

else right = mid - 1;

}

return null;

}

static void filterByCategory(Product[] items, String category) {

boolean found = false;

for (Product p : items) {

if (p.category.equalsIgnoreCase(category)) {

System.out.println(p);

found = true;

}

}

if (!found) {

System.out.println("No products found in this category.");

}

}

}

