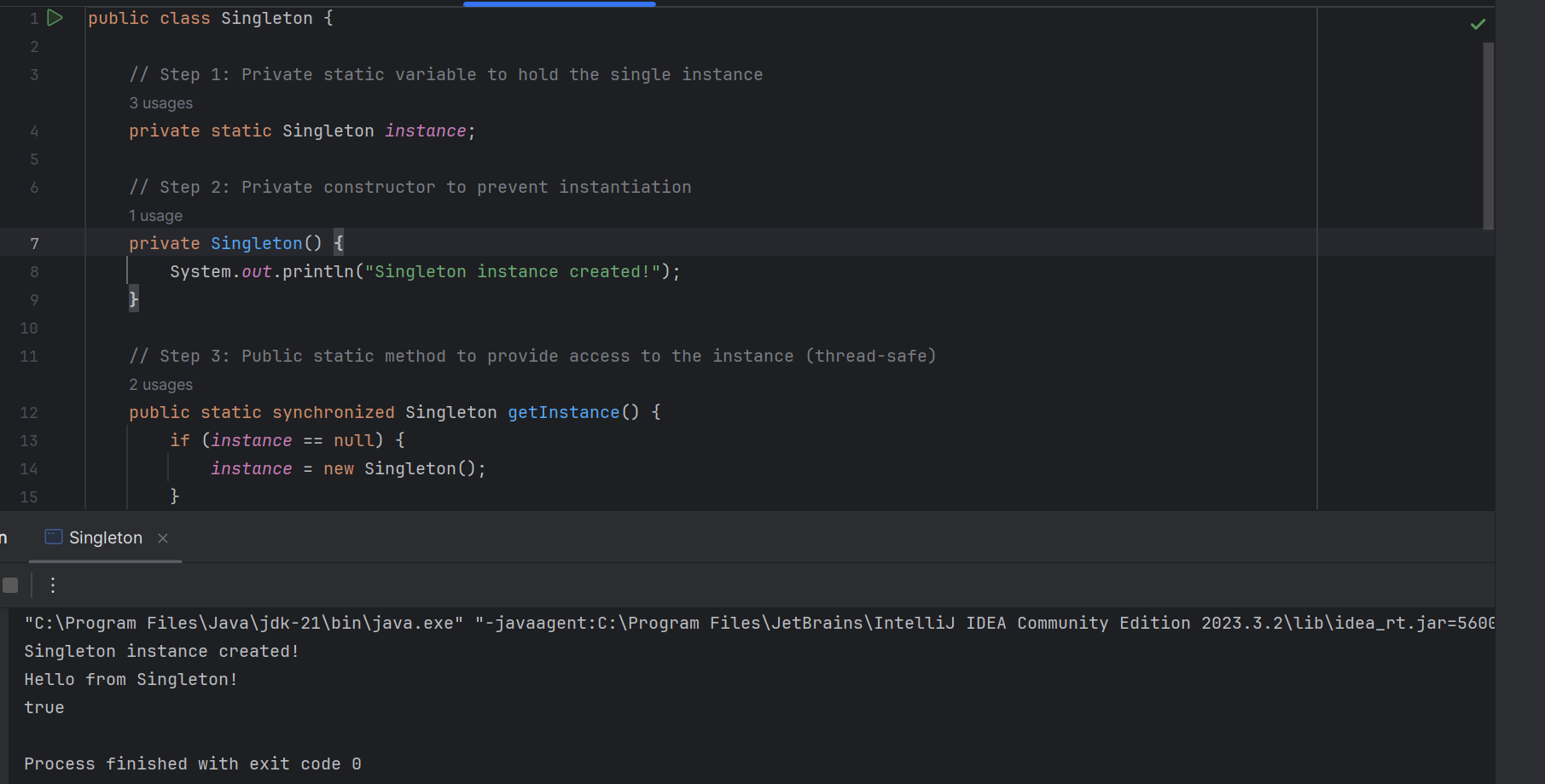
Cognizant : Week 1 :

Exercise 1 : Implementing the Singleton Pattern

public class Singleton {  
  
 // Step 1: Private static variable to hold the single instance  
 private static Singleton *instance*;  
  
 // Step 2: Private constructor to prevent instantiation  
 private Singleton() {  
 System.*out*.println("Singleton instance created!");  
 }  
  
 // Step 3: Public static method to provide access to the instance (thread-safe)  
 public static synchronized Singleton getInstance() {  
 if (*instance* == null) {  
 *instance* = new Singleton();  
 }  
 return *instance*;  
 }  
  
 // Example method  
 public void showMessage() {  
 System.*out*.println("Hello from Singleton!");  
 }  
  
 // Main method to test  
 public static void main(String[] args) {  
 Singleton obj1 = Singleton.*getInstance*();  
 Singleton obj2 = Singleton.*getInstance*();  
  
 obj1.showMessage();  
  
 // Verifying both objects are same  
 System.*out*.println(obj1 == obj2); // Output: true  
 }  
}



Exercise 2 : Implementing the factory Method pattern

Shape.java – Interface

public interface Shape {

void draw();

}

Circle.java – Concrete class

public class Circle implements Shape {

@Override

public void draw() {

System.out.println("Drawing a Circle");

}

}

Rectangle.java – Concrete class

public class Rectangle implements Shape {

@Override

public void draw() {

System.out.println("Drawing a Rectangle");

}

}  
  
ShapeFactory.java – Factory class

public class ShapeFactory {

// Factory Method

public Shape getShape(String shapeType) {

if (shapeType == null) {

return null;

}

if (shapeType.equalsIgnoreCase("CIRCLE")) {

return new Circle();

} else if (shapeType.equalsIgnoreCase("RECTANGLE")) {

return new Rectangle();

}

return null;

}

}

FactoryPatternDemo.java – Main class

public class FactoryPatternDemo {

public static void main(String[] args) {

ShapeFactory factory = new ShapeFactory();

Shape shape1 = factory.getShape("CIRCLE");

shape1.draw(); // Output: Drawing a Circle

Shape shape2 = factory.getShape("RECTANGLE");

shape2.draw(); // Output: Drawing a Rectangle

Shape shape3 = factory.getShape("TRIANGLE"); // Not implemented

if (shape3 != null) {

shape3.draw();

} else {

System.out.println("Invalid shape type");

}

}

}

OUTPUT :

Drawing a Circle

Drawing a Rectangle

Invalid shape type

Exercise 2 : E-commerce Platform Search Function

Product.java

public class Product {

private int id;

private String name;

private double price;

// Constructor

public Product(int id, String name, double price) {

this.id = id;

this.name = name;

this.price = price;

}

// Getters

public int getId() { return id; }

public String getName() { return name; }

public double getPrice() { return price; }

// Display product details

public void display() {

System.out.println("ID: " + id + ", Name: " + name + ", Price: ₹" + price);

}

}

ECommercePlatform.java

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class ECommercePlatform {

private List<Product> productList;

// Constructor to initialize products

public ECommercePlatform() {

productList = new ArrayList<>();

addSampleProducts();

}

// Add some products to the list

private void addSampleProducts() {

productList.add(new Product(1, "Samsung Galaxy S21", 69999));

productList.add(new Product(2, "Apple iPhone 14", 79999));

productList.add(new Product(3, "OnePlus Nord", 29999));

productList.add(new Product(4, "Sony Headphones", 4999));

productList.add(new Product(5, "Apple MacBook Air", 99999));

}

// Search method

public void searchProduct(String keyword) {

boolean found = false;

System.out.println("\nSearch Results for: " + keyword);

for (Product product : productList) {

if (product.getName().toLowerCase().contains(keyword.toLowerCase())) {

product.display();

found = true;

}

}

if (!found) {

System.out.println("No products found matching: " + keyword);

}

}

// Main method

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

ECommercePlatform platform = new ECommercePlatform();

System.out.println("Welcome to the E-Commerce Platform");

System.out.print("Enter product keyword to search: ");

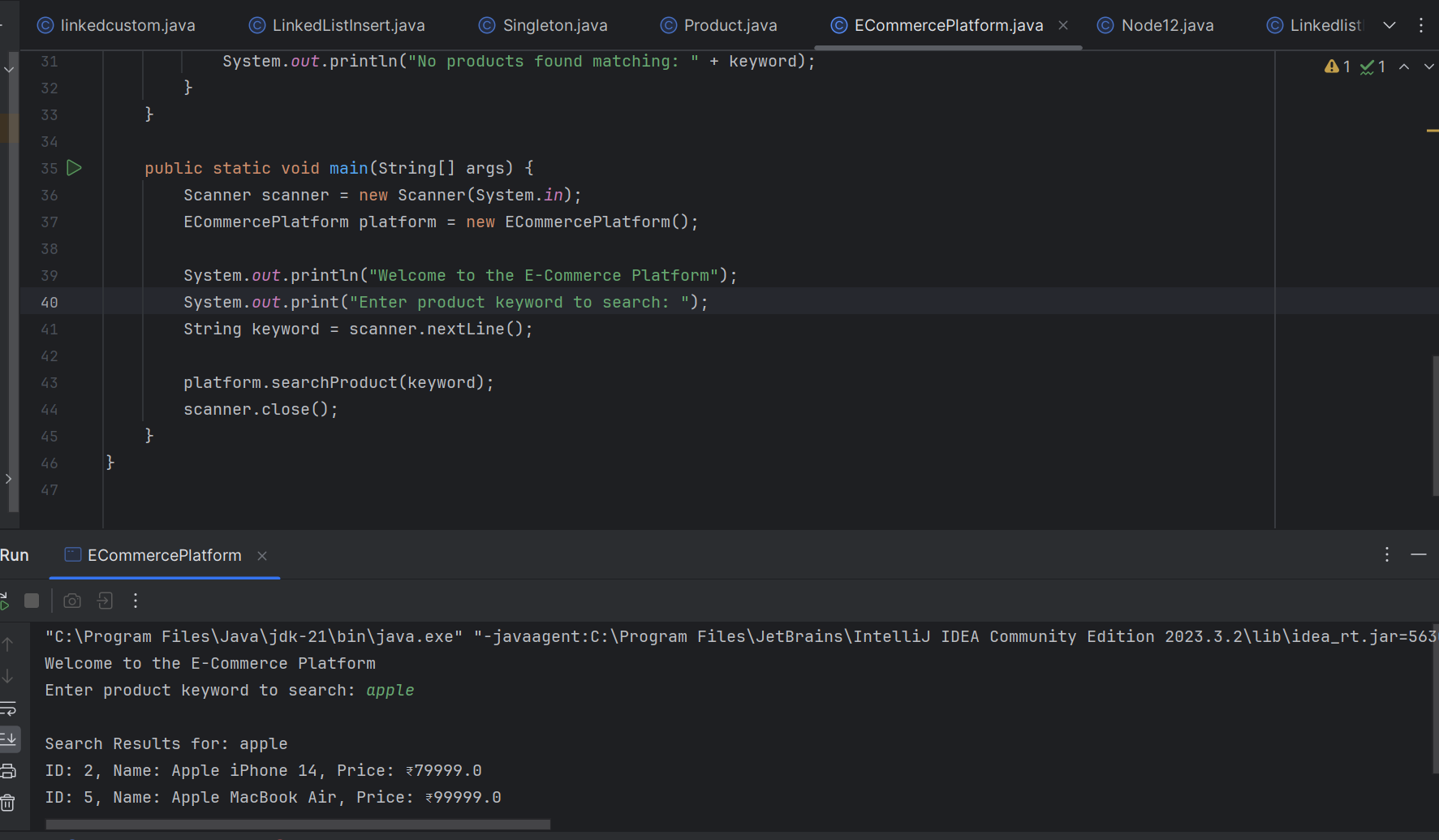
String keyword = scanner.nextLine();

platform.searchProduct(keyword);

scanner.close();

}

}



Exercise 7 : Financial Forecasting

FinancialForecast.java

import java.util.Scanner;

import java.text.DecimalFormat;

public class FinancialForecast {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

DecimalFormat df = new DecimalFormat("₹###,###.00");

// Input

System.out.println("📈 Financial Forecasting Tool");

System.out.print("Enter initial investment amount (₹): ");

double principal = scanner.nextDouble();

System.out.print("Enter annual interest rate (%): ");

double rate = scanner.nextDouble();

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

int years = scanner.nextInt();

System.out.println("\nYear-wise Financial Forecast:");

System.out.println("--------------------------------");

// Forecast

for (int year = 1; year <= years; year++) {

double futureValue = principal \* Math.pow(1 + (rate / 100), year);

System.out.println("Year " + year + ": " + df.format(futureValue));

}

scanner.close();

}

}

