**WEEK 1 Mandatory Hands on**

**Exercise 1: Implementing the Singleton Pattern :**

public class Singleton {

private static volatile Singleton instance;

private Singleton() {

if (instance != null) {

throw new RuntimeException("Use getInstance() to create");

}

}

public static Singleton getInstance() {

if (instance == null) {

synchronized (Singleton.class) {

if (instance == null) {

instance = new Singleton();

}

}

}

return instance;

}

public void showMessage() {

System.out.println("Singleton instance says hi!");

}

}

public class Main {

public static void main(String[] args) {

Singleton s1 = Singleton.getInstance();

Singleton s2 = Singleton.getInstance();

s1.showMessage();

System.out.println(s1 == s2);

}

}

Output :



**Exercise 2 : Implementing the Factory Method Pattern**

interface Product {

void use();

}

class ConcreteProductA implements Product {

public void use() {

System.out.println("Using Product A");

}

}

class ConcreteProductB implements Product {

public void use() {

System.out.println("Using Product B");

}

}

abstract class Creator {

public abstract Product factoryMethod();

public void someOperation() {

Product product = factoryMethod();

product.use();

}

}

class CreatorA extends Creator {

public Product factoryMethod() {

return new ConcreteProductA();

}

}

class CreatorB extends Creator {

public Product factoryMethod() {

return new ConcreteProductB();

}

}

public class Main {

public static void main(String[] args) {

Creator creatorA = new CreatorA();

creatorA.someOperation();

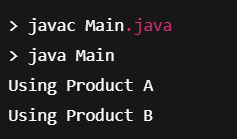
Creator creatorB = new CreatorB();

creatorB.someOperation();

}

}

Output :



**Exercise 3 : E-commerce Platform Search Function**

import java.util.ArrayList;

import java.util.List;

import java.util.stream.Collectors;

class Product {

private String id;

private String name;

private String category;

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

this.id = id;

this.name = name.toLowerCase();

this.category = category.toLowerCase();

}

public String getName() {

return name;

}

public String getCategory() {

return category;

}

public String toString() {

return String.format("Product{id='%s', name='%s', category='%s'}", id, name, category);

}

}

class SearchService {

private List<Product> products;

public SearchService(List<Product> products) {

this.products = products;

}

public List<Product> search(String keyword, String category) {

String kw = keyword == null ? "" : keyword.toLowerCase();

String cat = category == null ? "" : category.toLowerCase();

return products.stream()

.filter(p -> p.getName().contains(kw))

.filter(p -> cat.isEmpty() || p.getCategory().equals(cat))

.collect(Collectors.toList());

}

}

public class Main {

public static void main(String[] args) {

List<Product> productList = new ArrayList<>();

productList.add(new Product("101", "Apple iPhone 13", "Electronics"));

productList.add(new Product("102", "Samsung Galaxy S21", "Electronics"));

productList.add(new Product("103", "Nike Running Shoes", "Footwear"));

productList.add(new Product("104", "Levi's Jeans", "Clothing"));

SearchService searchService = new SearchService(productList);

List<Product> results1 = searchService.search("iphone", null);

System.out.println("Search results for keyword 'iphone':");

results1.forEach(System.out::println);

List<Product> results2 = searchService.search("", "electronics");

System.out.println("\nSearch results for category 'electronics':");

results2.forEach(System.out::println);

List<Product> results3 = searchService.search("shoes", "footwear");

System.out.println("\nSearch results for keyword 'shoes' and category 'footwear':");

results3.forEach(System.out::println);

}

}

**Exercise 4 : Financial Forecasting**

import java.util.Scanner;

public class FinancialForecasting {

public static double forecast(double principal, double annualContribution, double rate, int years) {

double amount = principal;

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

amount = amount \* (1 + rate) + annualContribution;

}

return amount;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

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

double principal = sc.nextDouble();

System.out.print("Enter annual contribution: ");

double contribution = sc.nextDouble();

System.out.print("Enter annual interest rate (in decimal, e.g., 0.07 for 7%): ");

double rate = sc.nextDouble();

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

int years = sc.nextInt();

double futureValue = forecast(principal, contribution, rate, years);

System.out.printf("Projected amount after %d years: %.2f%n", years, futureValue);

}

}

Output :

