Department of Computer Science and Engineering

KARPAGAM COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
(Autonomous)

ACADEMIC YEAR: 2024-25

Name & Register No of the Candidate: MADHAN S & 717824P129						
Course Code & Title: 23CSR306 JAVA PROGRAMMING						
Date of Issue: 14.09.2025		**Date of Submission:**				
Year/Dept./Sem/Section: II/CSE/III/A						
Assignment: I						
Reference(s):						
Marks Details						
Q. No						**Total (100)**
COs	CO1	CO2				
Marks Obtained						

Course In-charge

QUESTION 29:

29	Design and implement a console-based Marketplace system to onboard sellers, manage catalogs, and publish products with pricing rules using OOP in Java. Requirements: 1. Create at least 4 classes: ○ Seller – sellerId, name, email, rating, catalog. ○ Product – sku, title, basePrice, category, stock. ○ Catalog – list of products, category filters, bulk ops. ○ MarketplaceService – onboarding, listing, pricing, search. 2. Each class must include: ○ ≥4 instance/static variables. ○ A constructor to initialize values. ○ ≥5 methods (getters/setters, addProduct(), updatePrice(), publish(), search()). 3. Demonstrate OOPS Concepts: ○ Inheritance → ApparelProduct/ElectronicProduct extend Product with rules. ○ Method Overloading → search() by title/category/price range. ○ Method Overriding → finalPrice() differs by product type (GST, warranty). ○ Polymorphism → compute cart totals from List<Product>. ○ Encapsulation → protect stock and pricing updates. 4. Write a Main class (MarketplaceAppMain) to test: ○ Onboard sellers, create catalogs, add products. ○ Publish listings, update stock/price. ○ Run searches and print category-wise price lists.

CODE:

Product.java

```
public class Product {
   private String sku;
   private String title;
   private double basePrice;
   private int stock;

   public Product(String sku, String title, double basePrice, int stock) {
      this.sku = sku;
      this.title = title;
      this.basePrice = basePrice;
      this.stock = stock;
   }

   public double finalPrice() {
      return basePrice * 1.18;
   }

   public String getTitle() { return title; }
   public double getBasePrice() { return basePrice; }
   public int getStock() { return stock; }

   public void addStock(int qty) { stock += qty; }
}
```

ApparelProduct.java

```
public class ApparelProduct extends Product {
    public ApparelProduct(String sku, String title, double basePrice, int stock) {
        super(sku, title, basePrice, stock);
    }

    @Override
    public double finalPrice() {
        return getBasePrice() * 1.12;
    }
}
```

ElectronicProduct.java

```
public class ElectronicProduct extends Product {
    public ElectronicProduct(String sku, String title, double basePrice, int stock) {
        super(sku, title, basePrice, stock);
    }

    @Override
    public double finalPrice() {
        return getBasePrice() * 1.18 + 500;
    }
}
```

Seller.java

```
import java.util.ArrayList;
import java.util.List;

public class Seller {
    public String name;
    public List<Product> catalog = new ArrayList<>();

    public Seller(String name) {
        this.name = name;
    }

    public void addProduct(Product p) {
        catalog.add(p);
        System.out.println("Product added: " + p.getTitle());
    }
}
```

MarketplaceAppMain.java

```
import java.util.ArrayList;
import java.util.List;

public class MarketplaceAppMain {
    public static void main(String[] args) {
        Seller seller1 = new Seller("TechStore");
        Seller seller2 = new Seller("FashionHub");
```

```
        Product p1 = new ElectronicProduct("E001", "Smartphone", 15000, 50);
        Product p2 = new ApparelProduct("A001", "T-Shirt", 500, 200);

        seller1.addProduct(p1);
        seller2.addProduct(p2);

        System.out.println("\n--- Catalog ---");
        for (Product p : seller1.catalog) {
            System.out.printf("%s - Base: %.2f | Final Price: %.2f\n", p.getTitle(),
p.getBasePrice(), p.finalPrice());
        }
        for (Product p : seller2.catalog) {
            System.out.printf("%s - Base: %.2f | Final Price: %.2f\n", p.getTitle(),
p.getBasePrice(), p.finalPrice());
        }

        List<Product> cart = new ArrayList<>();
        cart.add(p1);
        cart.add(p2);

        double total = 0;
        for (Product p : cart) {
            total += p.finalPrice();
        }

        System.out.printf("\nCart total price: %.2f\n", total);
    }
}
```

OUTPUT:

```
Product added: Smartphone
Product added: T-Shirt

--- Catalog ---
Smartphone - Base: 15000.00 | Final Price: 18200.00
T-Shirt - Base: 500.00 | Final Price: 560.00

Cart total price: 18760.00
```