**DESIGN PATTERNS AND PRINCIPLES:**

**Exercise 4: Implementing the Adapter Pattern**

**Step 1: Target Interface**

**PaymentProcessor.java**

public interface PaymentProcessor {

void processPayment(double amount);

}

**Step 2: Adaptee Classes – Payment Gateways**

**// File: PayPalGateway.java**

public class PayPalGateway {

public void sendPayment(double money) {

System.out.println("Payment of ₹" + money + " processed through PayPal.");

}

}

**// File: StripeGateway.java**

public class StripeGateway {

public void makePayment(double value) {

System.out.println("Payment of ₹" + value + " processed through Stripe.");

}

}

**Step 3: Adapter Classes:**

**// File: PayPalAdapter.java**

public class PayPalAdapter implements PaymentProcessor {

private PayPalGateway paypal;

public PayPalAdapter(PayPalGateway paypal) {

this.paypal = paypal;

}

public void processPayment(double amount) {

paypal.sendPayment(amount);

}

}

**// File: StripeAdapter.java**

public class StripeAdapter implements PaymentProcessor {

private StripeGateway stripe;

public StripeAdapter(StripeGateway stripe) {

this.stripe = stripe;

}

public void processPayment(double amount) {

stripe.makePayment(amount);

}

}

**Step 4: Test Class –**

**TestAdapterPattern.java**

public class TestAdapterPattern {

public static void main(String[] args) {

// Using PayPal through adapter

PayPalGateway paypal = new PayPalGateway();

PaymentProcessor paypalProcessor = new PayPalAdapter(paypal);

paypalProcessor.processPayment(1500.00);

// Using Stripe through adapter

StripeGateway stripe = new StripeGateway();

PaymentProcessor stripeProcessor = new StripeAdapter(stripe);

stripeProcessor.processPayment(2500.00);

}

}

**OUTPUT:**

