**SCENARIO:**

You are developing a payment system where different payment methods (e.g., Credit Card, PayPal) can be selected at runtime. Use the Strategy Pattern to achieve this.

**STEPS:**

1. **Create a New Java Project:**
   * Create a new Java project named StrategyPatternExample.
2. **Define Strategy Interface:**
   * Create an interface PaymentStrategy with a method pay().
3. **Implement Concrete Strategies:**
   * Create classes CreditCardPayment, PayPalPayment that implement PaymentStrategy.
4. **Implement Context Class:**
   * Create a class PaymentContext that holds a reference to PaymentStrategy and a method to execute the strategy.
5. **Test the Strategy Implementation:**
   * Create a test class to demonstrate selecting and using different payment strategies**.**

**SOLUTION:**

**Step1:** Create a New Java Project

**Step 2:** Define Strategy Interface

// PaymentStrategy interface

public interface PaymentStrategy {

void pay(int amount);

}

**Step 3:** Implement Concrete Strategies:

// Concrete implementation for CreditCardPayment

public class CreditCardPayment implements PaymentStrategy {

private String userName;

private String cardNumber;

public CreditCardPayment(String uname, String cno) {

this.userName = uname;

this.cardNumber = cno;

}

@Override

public void pay(int amount) {

System.out.println("The amount " + amount + " was paid successfully by " + userName + " using Credit Card.");

}

}

// Concrete implementation for PayPalPayment

public class PayPalPayment implements PaymentStrategy {

private String userName;

private String upiId;

private String phone;

public PayPalPayment(String uname, String id, String no) {

this.userName = uname;

this.upiId = id;

this.phone = no;

}

@Override

public void pay(int amount) {

System.out.println("The amount " + amount + " was paid successfully by " + userName + " using PayPal.");

}

}

**Step 4:** Implement Context Class:

// PaymentContext class

public class PaymentContext {

private PaymentStrategy paymentStrategy;

public PaymentContext(PaymentStrategy paymentStrategy) {

this.paymentStrategy = paymentStrategy;

}

public void executeStrategy(int amount) {

paymentStrategy.pay(amount);

}

}

**Step 5:** Test the Strategy Implementation:

// Main class

public static void main(String[] args) {

// Create an instance of StrategyPatternExample to access non-static inner classes

StrategyPatternExample example = new StrategyPatternExample();

// Create a payment strategy

PaymentStrategy payment1 = example.new CreditCardPayment("Jackson", "9876543210");

PaymentStrategy payment2 = example.new PayPalPayment("Joshua","joshua27@abc","8978675645");

// Create a payment context with the strategy

PaymentContext pay1 = example.new PaymentContext(payment1);

PaymentContext pay2 = example.new PaymentContext(payment2);

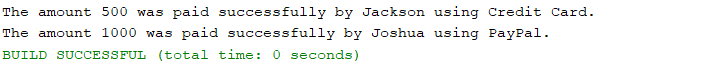
// Execute the payment strategy

pay1.executeStrategy(500);

pay2.executeStrategy(1000);

}

**SAMPLE OUTPUT:**

****