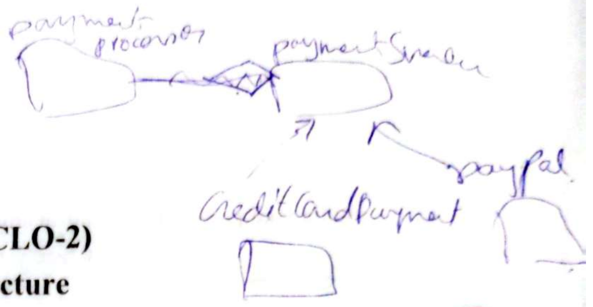

Quiz-1: Strategy Pattern- (CLO-2)

Software Design and Architecture

Information Technology University

5/10

Name Zunaira Abdul Aziz **Roll No.** BSSE23058

Total Marks: 10 (2 marks per question)

Question 1: Define the Strategy Interface

Fill in the missing parts to complete the **Strategy interface** for a payment system where different payment methods can be used dynamically.

```
public interface PaymentStrategy {
public void ~~signature()~~ pay(). signature();  ; // (Fill in: method signature for processing payment)      signature().
}
```

Question 2: Implement a Concrete Strategy

Complete the missing code to define the **CreditCardPayment** strategy.

```
public class CreditCardPayment implement PaymentStrategy {
  private String cardNumber;
  public CreditCardPayment(String cardNumber) {
    this.cardNumber = cardNumber;
  }
  public void pay(int amount) {
    System.out.println("Paid " + amount + " using Credit Card ending in" + cardNumber);  // (Fill in:
last 4 digits of card)
  }
}
```

Question 3: Implement the Context Class

Complete the missing code to define the **PaymentProcessor** class, which will allow dynamic strategy switching.

```
public class PaymentProcessor {
  private PaymentStradegy  payment$Stradegy
  public void setPaymentStrategy(PaymentStrategy paymentStrategy) {
    this.paymentStradedy = paymentStrategy;  // (Fill in: set the strategy)
  }
```

```
    public void processPayment(int amount) {
        paymentStradegy.~~amo~~pay (amount) ;
    }
}
```

Question 4: Client Code Using Strategy Pattern

Complete the missing code in the **main( ) method** to demonstrate dynamic strategy selection at runtime.

```
public class StrategyPatternDemo {
    public static void main(String[] args) {
        PaymentProcessor processor = new PaymentProcessor();
        // Select payment method at runtime
        processor.creditCardPaymat. ( 1234567830124); // (Fill in: Set the strategy as CreditCardPayment with card number
"1234567890123456")
        processor.processPayment(100);
    }
}
```

Question 5: Implement Another Concrete Strategy

Complete the missing code to implement **PayPalPayment**, another strategy for payment processing.

```
public class PayPalPayment implements PaymentStrategy {
    private String email;
    public PayPalPayment(String email) {
        this.email. = email;
    }
    @Override
    public void pay(int amount) {
        System.out.println("Paid " + amount + " using PayPal (Email: " + email. + ")");
    }
}
```