CLASS Diagram Sample Questions:

1.Create a class diagram that represents the primary classes involved in the system (for example, Customer, Order, Restaurant, Menu, Payment, and Delivery). Illustrate the relationships between these classes (such as associations, inheritance, and aggregations) and include important attributes and methods for each class.

Classes and Relationships:

1. **Customer**
    - Attributes: customerID, name, email, phoneNumber
    - Methods: placeOrder(), viewOrderHistory()
2. **Order**
    - Attributes: orderID, orderDate, status, totalAmount
    - Methods: calculateTotal(), updateStatus()
    - **Associations:**
        - A Customer places multiple Orders (1-to-many).
        - An Order contains multiple MenuItems (many-to-many).
        - An Order is processed by a Restaurant (1-to-1).
3. **Restaurant**
    - Attributes: restaurantID, name, location
    - Methods: manageMenu(), processOrder()
    - **Aggregation:** A Restaurant has a Menu (1-to-1).
4. **Menu**
    - Attributes: menuID, menuType
    - Methods: addMenuItem(), removeMenuItem()
    - **Composition:** A Menu consists of multiple MenuItems (1-to-many).
5. **MenuItem**
    - Attributes: itemID, name, price, category
    - Methods: getDetails()
    - **Associations:** An Order contains multiple MenuItems (many-to-many).
6. **Payment**
    - Attributes: paymentID, paymentMethod, paymentStatus
    - Methods: processPayment()

- **Association:** An Order is paid via Payment (1-to-1).

7. **Delivery**
    - Attributes: deliveryID, deliveryStatus, estimatedTime
    - Methods: updateDeliveryStatus()
    - **Association:** An Order is associated with Delivery (1-to-1).

2. In an e-commerce system, how would you represent multiple levels of product categories (e.g., Electronics → Mobile Phones → Apple)?

Classes and Relationships:

1. **Category (Abstract Class or Interface)**
    - Attributes: categoryID, name, description
    - Methods: addSubCategory(), removeSubCategory(), getProducts()
    - **Self-Association:** A Category can have multiple subcategories (1-to-many).
2. **ProductCategory (Concrete Class)**
    - Inherits from Category
    - Represents actual product categories like "Electronics" or "Mobile Phones".
3. **Product**
    - Attributes: productID, name, price, brand, stockQuantity
    - Methods: getDetails()
    - **Association:** A Product belongs to **one** Category (1-to-1).

Hierarchy Example:

- **Electronics (ProductCategory)**
    - **Mobile Phones (ProductCategory)**
        - **Apple (ProductCategory)**
            - Product: iPhone 15
        - **Samsung (ProductCategory)**
            - Product: Galaxy S24

UML Notation:

- Use **generalization** (inheritance) from Category to ProductCategory.
- Use **composition** (solid diamond) from Category to ProductCategory (categories are part of a hierarchy).
- Use **association** from ProductCategory to Product (each product belongs to a category).

3.Design a class diagram for a hotel management system where customers can book rooms. The system should handle room availability, payments, and customer details. Consider classes like Hotel, Room, Customer, Booking, and Payment.

Classes and Relationships:

1. **Hotel**
    - Attributes: hotelID, name, location
    - Associations:
        - *Has* multiple Room objects
        - *Manages* multiple Booking objects
2. **Room**
    - Attributes: roomID, roomNumber, type, price, isAvailable
    - Associations:
        - *Belongs to* one Hotel
        - *Can be booked* through Booking
3. **Customer**
    - Attributes: customerID, name, email, phoneNumber
    - Associations:
        - *Can make* multiple Booking objects
4. **Booking**
    - Attributes: bookingID, checkInDate, checkOutDate, status
    - Associations:
        - *Links* Customer to Room
        - *Has* one Payment
5. **Payment**
    - Attributes: paymentID, amount, paymentDate, paymentMethod
    - Associations:
        - *Belongs to* one Booking

4. Design a class diagram for a library system that manages books, members, and borrowing records. Consider classes like Library, Book, Member, Loan, and Librarian.

Classes and Relationships:

1. **Library**
    - Attributes: name, address
    - Associations:
        - *Has* multiple Book objects
        - *Has* multiple Member objects
        - *Has* multiple Librarian objects
2. **Book**

- Attributes: bookID, title, author, ISBN, isAvailable
- Associations:
  - *Belongs to* one Library
  - *Can be borrowed* by Member through Loan

3. **Member**
   - Attributes: memberID, name, email, phoneNumber
   - Associations:
     - *Borrows* books through Loan
     - *Belongs to* one Library
4. **Loan**
   - Attributes: loanID, issueDate, dueDate, returnDate, fineAmount
   - Associations:
     - *Links* Book and Member
     - *Managed by* Librarian
5. **Librarian**
   - Attributes: librarianID, name, email, phoneNumber
   - Associations:
     - *Manages* Loan
     - *Works at* one Library