# ✅ Interfaces

## 🔷 Direct MCQs

1. **Which keyword is used to define an interface in Java?**
   a) class
   b) **interface**
   c) implements
   d) abstract
   ✅ **Answer:** b

2. **Can an interface contain method implementations in Java?**
   a) Yes, always
   b) No, never
   c) **Only default and static methods**
   d) Only abstract methods
   ✅ **Answer:** c

3. **Which keyword is used by a class to inherit an interface?**
   a) extends
   b) **implements**
   c) inherits
   d) interface
   ✅ **Answer:** b

4. **Can a Java interface extend another interface?**
   a) No
   b) **Yes**
   c) Only if the interface is abstract
   d) Only one interface
   ✅ **Answer:** b

5. **What is the default access modifier for methods in an interface?**
   a) protected
   b) **public**
   c) private
   d) package-private
   ✅ **Answer:** b

## 🔷 Scenario-Based MCQs

1. **Class `CardPayment` implements `Payment` but forgets to override `process()`. What happens?**
   a) **Compilation error**
   b) Runtime error
   c) Warning only

d) Nothing happens

✅ **Answer:** a

2. **Class `Vehicle` implements `Engine` and `Transmission`, both with method `start()`. How to resolve ambiguity?**
   a) **Override `start()` in `Vehicle`**
   b) Extend one interface
   c) Rename methods
   d) Java throws runtime error

   ✅ **Answer:** a

3. **How to give a method in an interface a default implementation?**
   a) **default void methodName() {}**
   b) public void methodName() {}
   c) void methodName() {}
   d) static void methodName() {}

   ✅ **Answer:** a

4. **Multiple classes share logging behavior. What is the best Java approach?**
   a) Abstract class
   b) **Interface with default method**
   c) Utility class
   d) Constructor chaining

   ✅ **Answer:** b

5. **Class `Printer` implements `Device` but adds extra methods. Is it allowed?**
   a) No
   b) Yes, with override
   c) **Yes**
   d) Only if methods are abstract

   ✅ **Answer:** c

# ✅ Composition

## 🔷 Direct MCQs

1. **Composition is a _____ relationship.**
   a) is-a
   b) **has-a**
   c) uses-a
   d) none

   ✅ **Answer:** b

2. **What is a key benefit of composition over inheritance?**
   a) Simpler code
   b) **More flexibility and reusability**
   c) Easier syntax

d) Faster runtime

✅ **Answer:** b

3. **Which describes composition best?**
   a) Extending a class
   b) Reusing code via interfaces
   c) **Including one class within another**
   d) Using static methods

   ✅ **Answer:** c

4. **Can a class have multiple compositions?**
   a) No
   b) Only with inheritance
   c) **Yes**
   d) Only through interface

   ✅ **Answer:** c

5. **Preferred access modifier for composed objects?**
   a) public
   b) **private**
   c) protected
   d) static

   ✅ **Answer:** b

## 🔷 Scenario-Based MCQs

1. **A class `Car` has an `Engine`. What is the relationship?**
   a) Inheritance
   b) **Composition**
   c) Aggregation
   d) Interface

   ✅ **Answer:** b

2. **If `Engine` is destroyed when `Car` is destroyed, it's a...**
   a) Weak composition
   b) **Strong composition**
   c) Inheritance
   d) Association

   ✅ **Answer:** b

3. **Why use composition in a `Printer` class with multiple formats?**
   a) Avoid circular dependency
   b) **Allow runtime changes of components**
   c) Increase compile-time errors
   d) Force interface use

   ✅ **Answer:** b

4. **Class `Team` includes `Player`. Team controls lifecycle. What is this?**
   a) Aggregation
   b) Association
   c) **Composition**
   d) Inheritance
   ✅ **Answer:** c

5. **What does declaring a composed object as `final` mean?**
   a) Can't be accessed
   b) **Can't be reassigned**
   c) Can be inherited
   d) Becomes abstract
   ✅ **Answer:** b

# ✅ Overloading and Overriding

## 🔷 Direct MCQs

1. **What is method overloading?**
   a) Defining method in subclass
   b) Redefining method body
   c) **Same method name with different parameters**
   d) Declaring abstract method
   ✅ **Answer:** c

2. **What is method overriding?**
   a) Hiding a static method
   b) Changing method name
   c) **Redefining inherited method**
   d) Overloading with same name
   ✅ **Answer:** c

3. **Can a static method be overridden?**
   a) Yes
   b) **No**
   c) Only in final classes
   d) Only with private methods
   ✅ **Answer:** b

4. **Which annotation is used to override a method?**
   a) @Overload
   b) **@Override**
   c) @Inherited
   d) @Implements
   ✅ **Answer:** b

5. **Which of the following can be overloaded?**
   a) Constructors
   b) Private methods
   c) Static methods
   d) **All of the above**
   ✅ **Answer:** d

## 🔷 Scenario-Based MCQs

1. **`MathUtil` has `add()` methods with different params. This is...**
   a) Overriding
   b) Abstract method
   c) **Overloading**
   d) Interface method
   ✅ **Answer:** c

2. **`Animal` class has `speak()`, `Dog` overrides it. What is this?**
   a) Method hiding
   b) Overloading
   c) **Overriding**
   d) Abstraction
   ✅ **Answer:** c

3. **Two methods: `print(String)` and `print(int)`. This is...**
   a) Casting
   b) **Overloading**
   c) Inheritance
   d) Shadowing
   ✅ **Answer:** b

4. **Subclass omits @Override annotation but overrides method. Result?**
   a) Compilation error
   b) Runtime error
   c) **Still compiles**
   d) Logs warning
   ✅ **Answer:** c

5. **Override a method but reduce visibility. Result?**
   a) **Compilation error**
   b) Runtime error
   c) Logs warning
   d) No effect
   ✅ **Answer:** a

# ✅ Aggregation

## 🔷 Direct MCQs

1. **Aggregation is also called:**
   a) Inheritance
   b) Composition
   c) **Weak association**
   d) Association
   ✅ **Answer:** c

2. **Lifecycle of child object in aggregation:**
   a) Depends on parent
   b) **Is independent**
   c) Must be overridden
   d) Must be abstract
   ✅ **Answer:** b

3. **Which is an example of aggregation?**
   a) **Library and Book**
   b) Car and Engine
   c) Bird and Fly
   d) Student and Subject
   ✅ **Answer:** a

4. **Aggregation is a form of:**
   a) Polymorphism
   b) Inheritance
   c) **Association**
   d) Abstraction
   ✅ **Answer:** c

5. **Aggregation allows:**
   a) Only one-to-one relationships
   b) Strong ownership
   c) **Shared references**
   d) Method overriding
   ✅ **Answer:** c

## 🔷 Scenario-Based MCQs

1. **Team has Players, but Players can exist without Team. This is:**
   a) Composition
   b) **Aggregation**
   c) Inheritance
   d) Association
   ✅ **Answer:** b

2. **Professors work in multiple departments. Relationship type?**
   a) Composition
   b) **Aggregation**

c) Inheritance
d) Interface

✅ **Answer:** b

3. **Teachers transfer between schools. This is:**
   a) Association
   b) **Aggregation**
   c) Inheritance
   d) Composition

   ✅ **Answer:** b

4. **Library contains books, books can exist outside. This is:**
   a) **Aggregation**
   b) Composition
   c) Abstraction
   d) Interface

   ✅ **Answer:** a

5. **You want shared ownership without dependent lifecycle. Use:**
   a) **Aggregation**
   b) Composition
   c) Inheritance
   d) Polymorphism

   ✅ **Answer:** a