✅ Polymorphism

🔷 Direct MCQs

1. **What is polymorphism in Java?**
   a) Multiple constructors
   b) Method overloading and overriding ✅
   c) Class inheritance
   d) Encapsulation

2. **Which keyword is used for method overriding?**
   a) final
   b) static
   c) super ✅
   d) override

3. **What type of polymorphism is resolved at compile time?**
   a) Dynamic
   b) Runtime
   c) Static ✅
   d) Late binding

4. **Method overloading is an example of:**
   a) Runtime polymorphism
   b) Compile-time polymorphism ✅
   c) Inheritance
   d) Encapsulation

5. **Which is not a valid form of polymorphism in Java?**
   a) Method overloading
   b) Method overriding
   c) Constructor overloading
   d) Variable overloading ✅

🔷 Scenario-Based MCQs

1. **If class A has a method `display()`, and class B overrides it, which method is called if an object of B is referenced as type A?**
   a) A's method
   b) B's method ✅
   c) Compile error
   d) None of the above

2. **A developer uses the same method name in a class with different parameters. What is this called?**
   a) Method overriding
   b) Method overloading ✅

c) Constructor chaining
d) Interface implementation

3. **You have a superclass `Animal` and subclasses `Dog` and `Cat`, each with `makeSound()`. What is this an example of?**
   a) Overloading
   b) Abstraction
   c) Polymorphism ✅
   d) Aggregation

4. **Which of the following benefits from polymorphism?**
   a) Security
   b) Code reusability
   c) Performance
   d) Dynamic behavior ✅

5. **Which code demonstrates runtime polymorphism?**
   a) `int a = 10 + 20;`
   b) `Dog d = new Dog();`
   c) `Animal a = new Dog(); a.sound();` ✅
   d) `System.out.println("Hello");`

✅ Abstraction

🔷 Direct MCQs

1. **What is abstraction in Java?**
   a) Hiding internal implementation ✅
   b) Showing all data to user
   c) Inheriting classes
   d) Overriding methods

2. **Which keyword is used to create an abstract class?**
   a) class
   b) final
   c) abstract ✅
   d) static

3. **Can abstract classes have constructors?**
   a) Yes ✅
   b) No
   c) Only if final
   d) Only if public

4. **Can you create objects of abstract classes?**
   a) Yes

b) No ✅
c) Only with interface
d) Only inside the class

5. **Which is true about abstract methods?**
a) Must be static
b) Must be final
c) Have no body ✅
d) Can be private

🔷 Scenario-Based MCQs

1. **A developer wants to define a blueprint for multiple shapes. What should be used?**
a) Interface
b) Concrete class
c) Abstract class ✅
d) Static class

2. **Abstract class A has an abstract method. What must the subclass do?**
a) Inherit as-is
b) Ignore it
c) Implement it ✅
d) Declare as final

3. **Why would a developer use abstraction?**
a) For faster code
b) For memory efficiency
c) To reduce complexity ✅
d) To create GUI

4. **Which of the following is allowed?**
a) Abstract class with constructor ✅
b) Abstract class with object
c) Instantiating abstract class
d) Abstract class with private abstract method

5. **What happens if an abstract method is not implemented in a subclass?**
a) Compile error ✅
b) Runtime error
c) Program runs fine
d) JVM error

✅ Encapsulation

🔷 Direct MCQs

1. **What does encapsulation mean in Java?**
   a) Wrapping code and data together ✅
   b) Inheritance
   c) Method Overloading
   d) Abstraction

2. **Which access modifier is typically used for encapsulation?**
   a) public
   b) protected
   c) private ✅
   d) static

3. **How do you access private variables?**
   a) Direct access
   b) Through public getters/setters ✅
   c) Through interface
   d) Through static methods

4. **Encapsulation improves:**
   a) Performance
   b) Memory
   c) Security ✅
   d) Compilation

5. **Which class structure shows proper encapsulation?**
   a) public variables
   b) static variables
   c) private variables and public methods ✅
   d) abstract class

🔷 Scenario-Based MCQs

1. **A developer hides class fields and provides public methods to access them. What is this?**
   a) Inheritance
   b) Polymorphism
   c) Encapsulation ✅
   d) Overriding

2. **Why are setters used?**
   a) To override constructors
   b) To access abstract methods
   c) To modify private fields ✅
   d) To call static methods

3. **An object's fields are private but accessed through public getters. This is:**
   a) Abstraction
   b) Encapsulation ✅

c) Composition
d) Aggregation

4. **Which approach promotes encapsulation?**
   a) Make all fields public
   b) Use final fields only
   c) Provide getter and setter methods ✅
   d) Use static classes

5. **A banking system restricts access to the balance field using setters/getters. What OOP concept is applied?**
   a) Inheritance
   b) Encapsulation ✅
   c) Aggregation
   d) Overriding