Day-16

Overriding Vs Overloading, super & final keywords

Overriding in Java

- if subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in Java**.
- In other words, if a subclass provides the specific implementation of the method that has been declared by one of its parent class, it is known as **method overriding**.
- Overriding possible only if you have at **least 2 classes (Parent and Child).**
- Method overriding is used for **runtime polymorphism.**

Rules for overriding methods:

- The method in the **subclass must have the**
  - Same name
  - Same return type
  - Same parameters as in the superclass.
- It's used for achieving runtime polymorphism.
- The **@Override** annotation is often used to indicate that a method is being overridden.

Overloading with inheritance

- We can achieve overloading with inheritance.
- Overloading can be achieved with single class and multiple classes(inheritance).

Important Note:

- We can override only methods but not constructors. (Cannot override constructors)
- We can overload methods and constructors within the same class or in sub class.

How to Achieve Runtime Polymorphism

- **Runtime polymorphism** (or dynamic binding) in Java is achieved through **method overriding**.
- It allows a subclass to provide a specific implementation of a method that is already defined in its superclass.
- The method that gets called is determined at **runtime** based on the **object's type**, **not the reference type.**

Method Overloading Vs Method Overriding

Aspect	Overloading	Overriding
Definition	Defining multiple methods with the same name but different parameter lists (number, type, or both).	Redefining a method in a subclass that is already present in the parent class.
Parameters	Must have different parameters (either in number, type, or order).	Must have the same parameters as the method in the parent class.
Compile-Time vs Runtime	Resolved during compile-time (Static Polymorphism).	Resolved during runtime (Dynamic Polymorphism).
Return Type	Can have the same or different return types.	Must have the same return type (or a covariant return type from Java 5+).
Inheritance Requirement	No inheritance relationship is required.	Requires inheritance; subclass must inherit from the parent class.

Super keyword in java

Usage of Java super Keyword

1. **super** can be used to refer immediate parent class instance variable.
2. **super** can be used to invoke immediate parent class method.
3. **super()** can be used to invoke immediate parent class constructor.

Final keyword in java

1) final Variables : When a variable is declared as final, its value cannot be changed once it is initialized. This makes the variable a constant.

2) final Methods : A method declared as final cannot be overridden by subclasses. This is useful when you want to prevent a subclass from altering the functionality of the parent class's method.

3) final Classes

A class declared as final cannot be extended. This is useful when you want to prevent inheritance, ensuring that no other class can alter its behavior.

Quiz on Java Inheritance, Overriding, super and final keywords

1. **What is inheritance in Java?**
    - a) A method of calling a superclass method
    - b) A mechanism where one class acquires the properties of another
    - c) A feature that only allows overriding methods
    - d) None of the above

      Answer: b) A mechanism where one class acquires the properties of another
2. **Which keyword is used to inherit a class in Java?**
    - a) this
    - b) super
    - c) extends
    - d) implements

      Answer: c) extends
3. **Which of the following is true about inheritance?**
    - a) A class can inherit multiple classes
    - b) Inheritance allows method overriding
    - c) Private members of a superclass are inherited
    - d) All of the above

      Answer: b) Inheritance allows method overriding
4. **How many types of inheritance are supported in Java?**

- a) 1
- b) 2
- c) 4
- d) 5

  Answer: d) 5

5. **Which of the following is an example of single inheritance?**
   - a) class B extends class A
   - b) class B extends A, C
   - c) class B implements A, C
   - d) class B extends D and class C

   Answer: a) class B extends class A

6. **Which type of inheritance is NOT supported by Java?**
   - a) Single inheritance
   - b) Multilevel inheritance
   - c) Multiple inheritance through classes
   - d) Hierarchical inheritance

   Answer: c) Multiple inheritance through classes

7. **In which inheritance type can one class be a subclass of two classes simultaneously?**
   - a) Single inheritance
   - b) Multiple inheritance
   - c) Hybrid inheritance
   - d) None of the above

   Answer: b) Multiple inheritance

8. **Why can't we extend multiple classes in Java?**
   - a) To maintain simplicity and avoid complexity in inheritance
   - b) To support multiple inheritance in a structured way

- c) Java allows multiple inheritance
- d) It causes memory leak issues

  Answer: a) To maintain simplicity and avoid complexity in inheritance

9. **What problem does multiple inheritance cause that Java aims to avoid?**
   - a) Multiple instantiation
   - b) Diamond problem
   - c) Performance issues
   - d) Compilation errors

     Answer: b) Diamond problem

10. **How does Java handle the diamond problem?**
    - a) It uses the final keyword
    - b) It allows extending only one class
    - c) It uses multiple classes for inheritance
    - d) None of the above

      Answer: b) It allows extending only one class

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

      Answer: a) Compile-time polymorphism

12. **Method overriding is an example of:**
    - a) Compile-time polymorphism
    - b) Runtime polymorphism
    - c) Constructor overloading

- d) Class loading

  Answer: b) Runtime polymorphism

13. **Which of the following is true for method overloading?**
    - a) Overloaded methods must have different method names
    - b) Overloaded methods must have the same parameter list
    - c) Overloaded methods must differ in their parameter type or number
    - d) Overloaded methods cannot change the return type

      Answer: c) Overloaded methods must differ in their parameter type or number

14. **Which of the following is true for method overriding?**
    - a) Overriding methods can change the method name
    - b) The access modifier of the overridden method can be less restrictive
    - c) The return type of an overridden method must be the same or covariant
    - d) Overriding happens during compilation

      Answer: c) The return type of an overridden method must be the same or covariant

15. **What does the super keyword do in Java?**
    - a) Refers to the parent class instance
    - b) Refers to the current class instance
    - c) Refers to the child class instance
    - d) Refers to a static method

      Answer: a) Refers to the parent class instance

16. **Which of the following is NOT a valid use of super?**
    - a) Calling the parent class constructor
    - b) Accessing parent class variables
    - c) Accessing parent class methods
    - d) Accessing private methods of the parent class

      Answer: d) Accessing private methods of the parent class

17. **Where is the super() constructor call used?**

   - a) In any method of the subclass
   - b) In the first statement of the child class constructor
   - c) In the main method
   - d) After the this() call in a constructor

   Answer: b) In the first statement of the child class constructor

18. **What happens if super() is not called explicitly in a constructor?**

   - a) It results in a compile-time error
   - b) The default constructor of the superclass is called implicitly
   - c) The constructor of the child class is not executed
   - d) None of the above

   Answer: b) The default constructor of the superclass is called implicitly

19. **Which of the following is true about the final keyword in Java?**

   - a) A final class can be extended
   - b) A final method can be overridden
   - c) A final variable can be reassigned
   - d) A final variable can be assigned only once

   Answer: d) A final variable can be assigned only once

20. **What is the purpose of the final keyword when applied to a class?**

   - a) To make the class immutable
   - b) To prevent the class from being extended
   - c) To force the class to have only static methods
   - d) To allow only private methods in the class

   Answer: b) To prevent the class from being extended

21. **Which of the following can be marked as final in Java?**

   - a) Variables

- b) Methods
- c) Classes
- d) All of the above

  Answer: d) All of the above

22. **Can a final method be overridden in a subclass?**
    - a) Yes, it can be overridden
    - b) No, it cannot be overridden
    - c) Yes, but only in the same package
    - d) None of the above

    Answer: b) No, it cannot be overridden

23. **Which type of inheritance leads to the diamond problem?**
    - a) Single inheritance
    - b) Multiple inheritance
    - c) Multilevel inheritance
    - d) Hierarchical inheritance

    Answer: b) Multiple inheritance

24. **Which of the following is true for method overriding?**
    - a) It is used for dynamic method dispatch
    - b) It is an example of compile-time polymorphism
    - c) It allows the superclass method to be called by this keyword
    - d) All of the above

    Answer: a) It is used for dynamic method dispatch

25. **When using super in a method, which class method is called?**
    - a) The method of the same class
    - b) The method of the subclass
    - c) The method of the superclass

- d) None of the above

  Answer: c) The method of the superclass

26. **Why is multiple inheritance not supported in Java through classes?**
    - a) Java doesn't allow inheritance
    - b) Java uses interfaces to solve multiple inheritance problems
    - c) It leads to ambiguity in method inheritance
    - d) Both b and c

    Answer: d) Both b and c

27. **What is method overloading based on?**
    - a) Different return types
    - b) Different method names
    - c) Different parameter lists
    - d) Both a and c

    Answer: c) Different parameter lists

28. **What is the significance of the super() constructor?**
    - a) It calls the constructor of the child class
    - b) It calls the constructor of the superclass
    - c) It initializes the private variables
    - d) It overrides the constructor of the superclass

    Answer: b) It calls the constructor of the superclass

29. **Which of the following is true about method overriding?**
    - a) It allows different return types
    - b) It must have the same parameter list as the superclass method
    - c) It can be done with static methods
    - d) It must have a different parameter list from the superclass method

    Answer: b) It must have the same parameter list as the superclass method

30. **What will happen if a class has a final method?**
    - ○ a) The method cannot be inherited
    - ○ b) The method cannot be overridden
    - ○ c) The method can be overridden with a different return type
    - ○ d) The method can be inherited but not overridden

      Answer: b) The method cannot be overridden

31. **Can a constructor be overridden in Java?**

- a) Yes, it can be overridden
- b) No, constructors cannot be overridden
- c) Only if it has the same parameters
- d) Only if it's declared as final

  Answer: b) No, constructors cannot be overridden

32. **Which keyword is used to call the superclass constructor explicitly?**

- a) this
- b) final
- c) super
- d) static

  Answer: c) super

33. **What will happen if a class is declared as final?**

- a) The class cannot have any fields
- b) The class cannot be instantiated
- c) The class cannot be extended
- d) The class can only have private methods

  Answer: c) The class cannot be extended

34. **In method overriding, which version of the method gets called during runtime?**

- a) The method in the parent class

- b) The method in the child class
- c) Both methods are executed
- d) The method in the class where it is called

  Answer: b) The method in the child class

35. **What is the primary benefit of method overriding in Java?**

- a) Improved performance
- b) Runtime polymorphism
- c) Reducing code redundancy
- d) Allows changing return types

  Answer: b) Runtime polymorphism