**Summary of Key Topics in Java Inheritance and Advanced OOP Concepts**

**1. Types of Inheritance in Java**

* **Single Inheritance**: A subclass inherits from one superclass.
  + Example: Dog extends Animal to inherit methods like eat().
* **Multilevel Inheritance**: A subclass inherits from another subclass.
  + Example: Puppy extends Dog, which in turn extends Animal.
* **Hierarchical Inheritance**: Multiple subclasses inherit from a single superclass.
  + Example: Dog and Cat both extend Animal.
* **Multiple Inheritance (via Interfaces)**: Java doesn’t allow multiple inheritance with classes, but interfaces can achieve this.

**2. Method Overriding**

* **Definition**: A subclass provides a specific implementation of a method in the superclass.
* **Rules**:
  + The method signature must remain the same.
  + Achieves runtime polymorphism.
* **Example**: Overriding the sound() method in the Dog subclass.

**3. super Keyword**

* **Purpose**:
  + Access superclass methods, constructors, and fields.
* **Usage**:
  + Call a superclass method using super.methodName().
  + Example: super.eat() in the overridden eat() method in Dog.

**4. Abstract Classes and Methods**

* **Abstract Class**: Cannot be instantiated directly.
  + May include both abstract and concrete methods.
* **Abstract Method**: Declared without a body; must be implemented in a subclass.
* **Example**:
  + Abstract class Animal with abstract method sound().
  + Concrete implementation in Dog.

**5. Interfaces**

* **Definition**: A contract that specifies methods a class must implement.
* **Key Features**:
  + Contains abstract methods by default.
  + Allows multiple inheritance.
* **Example**:
  + Interface Animal with method sound().
  + Dog implements Animal.

**6. Multiple Inheritance Using Interfaces**

* **Purpose**: Achieve multiple inheritance through interfaces.
* **Example**:
  + Dog implements Animal and Mammal, providing methods for both.

**7. Default and Static Methods in Interfaces**

* **Default Methods**:
  + Provide a default implementation in an interface.
  + Can be overridden in implementing classes.
* **Static Methods**:
  + Defined in interfaces but cannot be inherited.
  + Called using InterfaceName.methodName().
* **Example**:
  + default void eat() in Animal interface.
  + static void breathe() in the same interface.

**Key Takeaways**

* Inheritance supports code reuse and hierarchical relationships.
* Method overriding enables dynamic method dispatch.
* The super keyword facilitates access to the parent class.
* Abstract classes and interfaces define blueprints for other classes, supporting flexibility and extensibility.
* Java achieves multiple inheritance only through interfaces, avoiding ambiguity issues.
* Default and static methods in interfaces enhance functionality and usability.