**9. Inheritance and Polymorphism**

• **Theory:**

**1 . Inheritance Types and Benefits:-**

* **Types of Inheritance in Java**

1. **Single Inheritance**
   * A class inherits from only one superclass.
2. **Multilevel Inheritance**
   * A class inherits from a superclass, which is itself a subclass of another superclass.
3. **Hierarchical Inheritance**
   * Multiple classes inherit from the same superclass.
4. **Multiple Inheritance (through Interfaces)**
   * Java does not support multiple inheritance with classes to avoid ambiguity (the "diamond problem"). However, it can be achieved using interfaces.

**2.Method Overriding:-** is same name as class name same as inheritance. Method overriding is a core concept in Java that allows a subclass to provide a specific implementation of a method that is already defined in its superclass. This mechanism enables polymorphism, allowing you to call overridden methods based on the object's runtime type rather than its reference type.

**Syntax:**- class Superclass { // Method in the superclass returnType methodName(parameters) { // method body } } class Subclass extends Superclass { @Override // Overriding method returnType methodName(parameters) { // new method body } }

**3.Dynamic Binding (Run-Time Polymorphism):**Method overriding is one of the ways in which Java supports Runtime Polymorphism. Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time. At run-time, it depends on the type of the object being referred to (not the type of the reference variable) that determines which version of an overridden method will be executed

**4.Super Keyword and Method Hiding:-**

**Super Keyword:-** The super keyword in Java is a reference variable that refers to the immediate parent class object.

* When a method in a subclass overrides a method in the superclass, you can use super to call the superclass method.

**Method Hiding** :- Method hiding occurs when a subclass defines a static method with the same name and parameters as a static method in its superclass. In this case, the method in the subclass "hides" the method in the superclass.