**JAVA:**

Java is a rich, object-oriented programming language with various key concepts that are essential for building effective and efficient applications. Here's an overview of the core concepts in Java:

**1. Object-Oriented Programming (OOP) Concepts**

Java is an object-oriented programming language, meaning it revolves around objects that represent real-world entities. The main OOP principles in Java include:

* **Classes and Objects**:
  + A **class** is a blueprint or template for creating objects.
  + An **object** is an instance of a class. It has attributes (fields) and behaviors (methods).
* **Encapsulation**:
  + Encapsulation refers to bundling the data (fields) and methods that operate on the data into a single unit (class).
  + It also involves restricting access to some of the object's components using access modifiers (private, protected, public).
* **Inheritance**:
  + Inheritance allows one class (subclass or derived class) to inherit the properties and behaviors of another class (superclass or base class).
  + It promotes reusability and establishes a hierarchical relationship between classes.
* **Polymorphism**:
  + Polymorphism allows objects of different classes to be treated as objects of a common superclass.
  + It can be achieved through **method overriding** (runtime polymorphism) and **method overloading** (compile-time polymorphism).
* **Abstraction**:
  + Abstraction allows hiding complex implementation details and exposing only essential features of an object.
  + It can be implemented using **abstract classes** (cannot be instantiated) or **interfaces** (defines a contract for other classes to implement).

Example:

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello, World!");

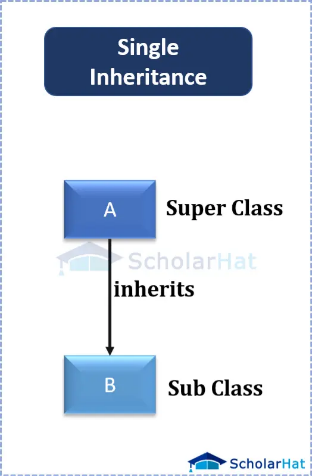
TYPES OF INHERITETENCE

In Java, **inheritance** is a fundamental concept of Object-Oriented Programming (OOP) that allows a class (called the **subclass** or **child class**) to inherit the properties and behaviors (fields and methods) of another class (called the **superclass** or **parent class**). This promotes code reusability and establishes a relationship between classes.

There are **several types of inheritance** in Java:

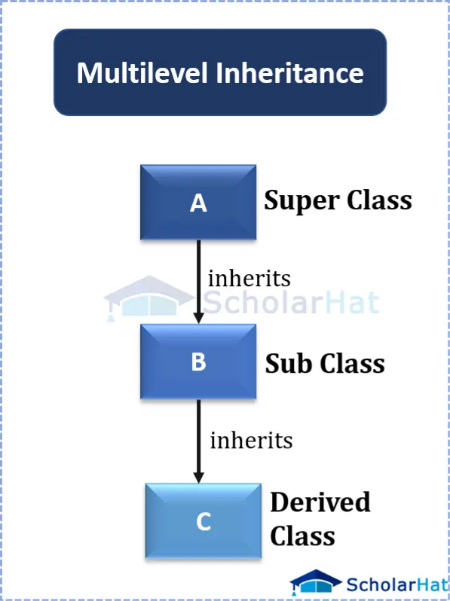
**1. Single Inheritance**

* **Single inheritance** occurs when a subclass inherits from only one superclass.
* It is the most straightforward type of inheritance.
* This type of inheritance prevents the complexities associated with inheriting from multiple classes.



**Example:**

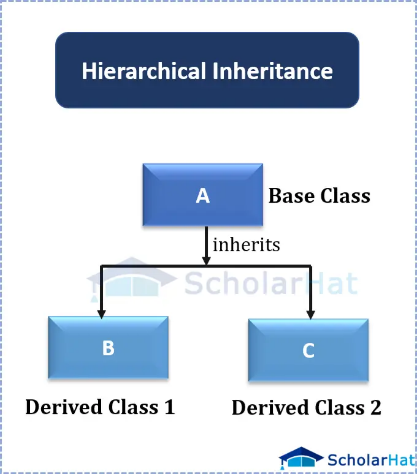
**2. Multilevel Inheritance**

* In **multilevel inheritance**, a class derives from another class, which is itself derived from another class.
* This creates a chain of inheritance, where the child class of one class becomes the parent class for the next class.
* Java supports multilevel inheritance directly.
* 

**Example**:

****

**3. Hierarchical Inheritance**

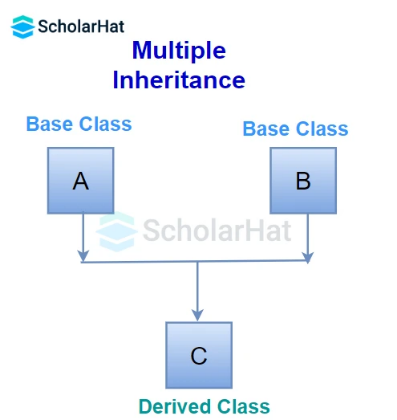
* **Hierarchical inheritance** occurs when a single superclass is inherited by multiple subclasses.
* This means that one parent class can have multiple child classes.
* 

**Example**:

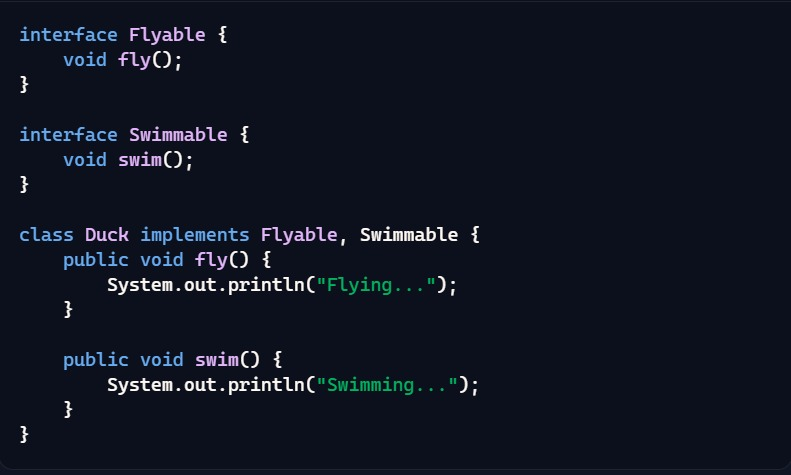


**4. Multiple Inheritance (Through Interfaces)**

* **Multiple inheritance** refers to the ability of a class to inherit from more than one superclass.
* **Java does not support multiple inheritance** directly through classes, because it can lead to ambiguity (known as the **Diamond Problem**).
* However, **multiple inheritance is supported through interfaces**. A class can implement multiple interfaces, allowing it to inherit behaviors from multiple sources.
* This helps avoid the issues of multiple inheritance in classes.

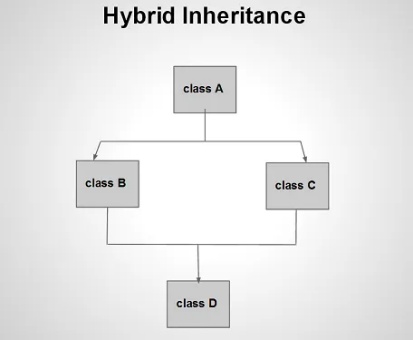


**Example**:



**5. Hybrid Inheritance (Combination of Types)**

* **Hybrid inheritance** refers to the combination of different types of inheritance (like single, multilevel, and hierarchical inheritance) in a program.
* In Java, this concept can be realized when multiple inheritance through interfaces and other forms of inheritance (such as multilevel or hierarchical) are used together.



**Example**:



**Summary of Java Inheritance Types:**

1. **Single Inheritance**: A class inherits from one superclass.
2. **Multilevel Inheritance**: A class inherits from another class, which also inherits from another class.
3. **Hierarchical Inheritance**: Multiple subclasses inherit from a single superclass.
4. **Multiple Inheritance (through interfaces)**: A class implements multiple interfaces (Java does not support multiple inheritance with classes directly).
5. **Hybrid Inheritance**: A combination of different types of inheritance, typically using interfaces along with other inheritance forms.