# Inheritance

* Inheritance is a process of inheriting the properties from one class into another class.
* Existing is said to be a superclass and the new class is said to be a subclass.
* Superclass object creates memory only for superclass members.
* Subclass objects create a memory for both super and sub-class members.

Graphical user interface, text, application

Description automatically generated

## Types of Inheritance:

1. Sigle Inheritance:

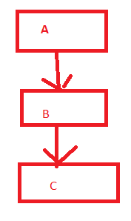
* Java will Support
* Deriving a class from only one superclass.

Diagram, schematic

Description automatically generated

1. Multilevel Inheritance

* Java will support it.
* Deriving a class from a superclass when the superclass is a subclass for another superclass.



1. Hierarchical Inheritance

* Java will support it.
* Derivation of several classes from only one superclass is called hierarchical inheritance.

Diagram

Description automatically generated

1. Multiple Inheritance

* Java does not support it.
* Derivation of a class from more than one superclass is called multiple Inheritance.

Diagram, schematic

Description automatically generated

1. Hybrid Inheritance

* Java does not support it.
* Derivation of a class involving more than one form of inheritance is called Hybrid inheritance.
* Diagram

  Description automatically generated

1. Multipath inheritance:

* Java does not support it.
* Derivation of a class from more than one sub class, those sub classes get inherited from the same super class is called as multi path inheritance

Diagram, schematic

Description automatically generated  
Text, whiteboard

Description automatically generated

**Refer Code Snippet:** InheriteceExample1\_Class.java

**package** inheritence;

/\*Program to demonstrate

\* B Class Inherited A Class and Object is created to B Class\*/

**class** InheriteceExample1\_Class {

**public** **static** **void** main(String[] args)

{

B obj = **new** B();

//===========> Object is created to B Class and Assigned to B Class Reference Variable

System.***out***.println("Class x Variable Value - " + obj.x);

//==> if Both Derived and Super class has same variable name, Super class will get overridden by base class.

// ==> Here B Class X Value will get displayed.

System.***out***.println("\nClass y Variable Value - " + obj.y);

//==> Here A Class Y value will get displayed because Derived class does not have Y.

System.***out***.println("\nCall Print Method");

obj.print();

//==> if Both Derived and Super class has same method name, Super class method will get overridden by Derived class.

// ==> Here B Class Print method will be called.

System.***out***.println("\nCall show method");

obj.show();

// ==> Here B Class Show method will be called. It is available only in B Class, No need to override.

System.***out***.println("\nCall sample method");

obj.sample();

//==> Here A Class Sample method will be called. It is available only in A Class, No need to override.

}

}

**class** A

{

**int** x = 100;

**int** y = 200;

**public** **void** print()

{

System.***out***.println("A Class");

}

**public** **void** sample()

{

System.***out***.println("A Class");

}

}

**class** B **extends** A

{

**int** x = 1000;

**public** **void** print()

{

System.***out***.println("B Class");

}

**public** **void** show()

{

System.***out***.println("B Class");

}

}

//Output:

Class x Variable Value - 1000

Class y Variable Value - 200

Call Print Method

B Class

Call show method

B Class

Call sample method

A Class

**Refer Code Snippet:** InheriteceExample2\_Class.java

**package** inheritence;

/\*Program to demonstrate

\* C Class Inherited to D Class and Object is created to D Class (Derived class)

\* and assigned to C class (Base class) reference variable.\*/

**class** InheriteceExample2\_Class {

**public** **static** **void** main(String[] args)

{

C obj = **new** D();

//===========> Object is created to D Class (Derived Class) and Assigned to B Class (Superclass) Reference Variable

System.***out***.println("Class x Variable Value - " + obj.x);

//==> if Both Derived and Super class has same variable name, Only super class variable will get display. Variable will not be overridden.

// ==> Here C Class X Value will get displayed.

System.***out***.println("\nClass y Variable Value - " + obj.y);

//==> Here C Class Y value will get displayed because derived class does not have Y.

System.***out***.println("\nCall Print Method");

obj.print();

//==> if Both Derived and Super class has same method name, Super class will get overridden by base class.

// ==> Here D Class Print method will be called.

System.***out***.println("\nCall show method");

//obj.show();

// ==> Here D Class Show method is not accessible.

System.***out***.println("\nCall sample method");

obj.sample();

//==> Here C Class Sample method will be called.

}

}

**class** C

{

**int** x = 100;

**int** y = 200;

**public** **void** print()

{

System.***out***.println("C Class");

}

**public** **void** sample()

{

System.***out***.println("C Class");

}

}

**class** D **extends** C

{

**int** x = 1000;

**public** **void** print()

{

System.***out***.println("D Class");

}

**public** **void** show()

{

System.***out***.println("D Class");

}

}

//OUTPUT

Class x Variable Value - 100

Class y Variable Value - 200

Call Print Method

D Class

Call show method

Call sample method

C Class

Note: Assigning a Super Class Object to Sub Class reference Variable is not allowed in JAVA.

Ex : Assume A => Super Class & B => Derived Class

B Obj = new A(); ==🡺 Not Allowed in JAVA.