**Inheritance**

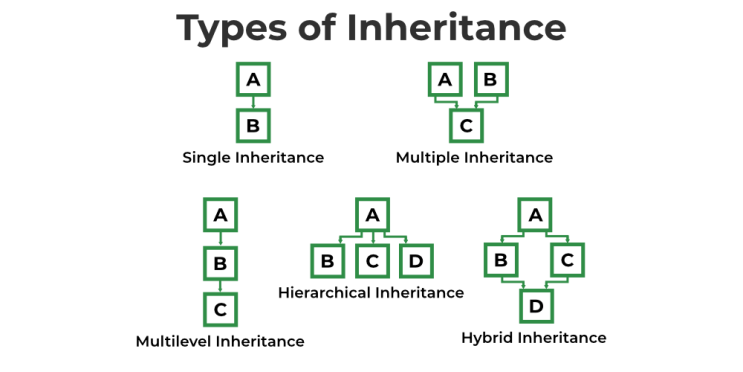
1. **Inheritance** is a mechanism in which one object acquires all the properties and behaviors of a parent object.
2. It is an important part of [OOPs](https://www.javatpoint.com/java-oops-concepts).
3. Inheritance represents the **IS-A relationship** which is also known as a parent-childrelationship*.*
4. A class is derived from another class and uses data and implementation of that other class.
5. The main purpose of Inheritance is to increase code re-usability

### Why use inheritance

1. For [Method Overriding](https://www.javatpoint.com/method-overriding-in-java) (so [runtime polymorphism](https://www.javatpoint.com/runtime-polymorphism-in-java) can be achieved).
2. For Code Reusability.

**Types of inheritance :**

* Single Inheritance
* Multiple Inheritance
* Hierarchical Inheritance
* Multilevel Inheritance
* Hybrid Inheritance



1. **Single Inheritance :-** When a class inherits another class, it is known as a single inheritance**.**
2. **Multilevel Inheritance** :- When there is a chain of inheritance, it is known as multilevel inheritance.
3. **Hierarchical Inheritance** :- When two or more classes inherits a single class, it is known as hierarchical inheritance
4. **Multiple Inheritance** :- When one classes inherits two or more classes, it is known as multiple inheritance.
5. **Hybrid Inheritance** :-
   1. **Single nheritance**

**class A**

**{**

int x, y;

void set(int x, int y)

{

this.x = x;

this.y = y;

}

void sum()

{

System.out.println("Sum : " + (x+y) );

}

**}**

**class B extends A**

**{**

void sub()

{

System.out.println("Sub : " + (x-y) );

}

**}**

**class Q01\_Single\_Inheritance**

**{**

public static void main(String args[])

{

B b = new B();

b.set(100, 20);

b.sum();

b.sub();

}

**}**

**Output :-**

**Sum : 120**

**Sub : 80**

* 1. **Multiple\_Inheritance**

**class A**

**{**

int x, y;

void set(int x, int y)

{

this.x = x;

this.y = y;

}

void sum()

{

System.out.println("Sum : " + (x+y) );

}

**}**

**class B extends A**

**{**

void sub()

{

System.out.println("Sub : " + (x-y) );

}

**}**

**class C extends B**

**{**

void div()

{

System.out.println("div : " + (x/y) );

}

**}**

**class Q02\_Multiple\_Inheritance**

**{**

public static void main(String args[])

{

C c = new C();

c.set(100, 20);

c.sum(); **Sum : 120**

c.sub(); **Sub : 80**

c.div(); **Div : 5**

}

**}**

* 1. **Hierarchical\_Inheritance**

**class A**

**{**

int x, y;

void set(int x, int y)

{

this.x = x;

this.y = y;

}

void sum()

{

System.out.println("Sum : " + (x+y) );

}

**}**

**class B extends A**

**{**

void sub()

{

System.out.println("Sub : " + (x-y) );

}

**}**

**class C extends A**

**{**

void div()

{

System.out.println("div : " + (x/y) );

}

**}**

**class Q03\_Hierarchical\_Inheritance**

**{**

public static void main(String args[])

{

B b = new B();

b.set(100, 10);

b.sum(); Sum : 110

b.sub(); Sub : 90

System.out.println();

C c = new C();

c.set(200, 20); Sum : 220

c.sum(); div : 10

c.div();

}

}

**class A**

**{**

A()

{

System.out.println("Constructor A");

}

**}**

**class B extends A**

**{**

B()

{

System.out.println("Constructor B");

}

**}**

**class C extends B**

**{**

C()

{

System.out.println("Constructor C");

}

**}**

**class Q04\_Inheritance\_With\_Constructor**

**{**

public static void main(String args[])

{

C c = new C();

}

**}**

Constructor A

Constructor B

Constructor C

**class A**

**{**

A()

{

System.out.println("Constructor A");

}

**}**

**class B extends A**

**{**

B()

{

System.out.println("Constructor B");

}

**}**

**class C extends B**

**{**

C()

{

System.out.println("Constructor C");

}

**}**

**class Q05\_Inheritance\_With\_Constructor**

**{**

public static void main(String args[])

{

B b = new B();

}

**}**

Constructor A

Constructor B

**class A**

**{**

int x, y;

A(int x, int y)

{

System.out.println("Sum : " + (x+y) );

}

**}**

**class B extends A**

**{**

B(int x, int y)

{

System.out.println("Sub : " + (x-y) );

}

**}**

**class Q06\_Inheritance\_With\_Constructor**

**{**

public static void main(String args[])

{

B b = new B(10, 20);

}

**}**

**Constructor A in class A cannot be applied to given types;**

**class A**

**{}**

**class B extends A**

**{**

B(int x, int y)

{

System.out.println("Sub : " + (x-y) );

}

**}**

**class Q07\_Inheritance\_With\_Constructor**

**{**

public static void main(String args[])

{

B b = new B(100, 20);

}

**}**

**Sub : 80**

------------------------------------------------------------------------------------------------

**class A**

**{**

A(int x, int y)

{

System.out.println("Sub : " + (x-y) );

}

**}**

**class B extends A**

**{**

**}**

**class Q07\_Inheritance\_With\_Constructor**

**{**

public static void main(String args[])

{

B b = new B(100, 20);

}

**}**

**Error**

**class A**

**{**

A(int x, int y)

{

System.out.println("Sub : " + (x+y) );

}

**}**

**class B extends A**

**{**

B(int x, int y)

{

super(x, y);

System.out.println("Sub : " + (x-y) );

}

**}**

**class Q08\_Super\_Keyword**

**{**

public static void main(String args[])

{

B b = new B(100, 20);

}

**}**

**Sum : 120**

**Sub : 80**

**class A**

**{**

int x = 30;

**}**

**class B extends A**

**{**

int x = 20;

void show()

{

int x = 10;

System.out.println(x);

System.out.println(this.x);

System.out.println(super.x);

}

**}**

**class Q09\_This\_Keyword**

**{**

public static void main(String args[])

{

B b = new B();

b.show();

}

**}**

**10**

**20**

**30**

**class A**

**{**

A(int x, int y)

{

System.out.println( "Sum : " + (x+y) );

}

**}**

**class B extends A**

**{**

B(int x, int y)

{

super(x, y);

System.out.println( "Sub : " + (x-y) );

}

**}**

**class Q10\_Constructor\_Chining**

**{**

public static void main(String args[])

{

B b = new B(100, 20);

}

**}**

Sum : 120

Sub : 80