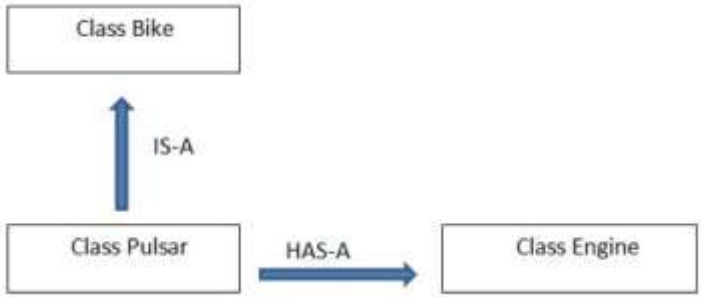
Is-A RELATIONSHIP

Is-A RELATIONSHIP :

- The relationship between two objects which is similar to the parent and child relation is known as the Is-A relationship.
- In an Is-A relationship, the child object will acquire all properties of the parent object, and the child object will have its own extra properties.
- In an Is-A relationship, we can achieve generalization and specialization.



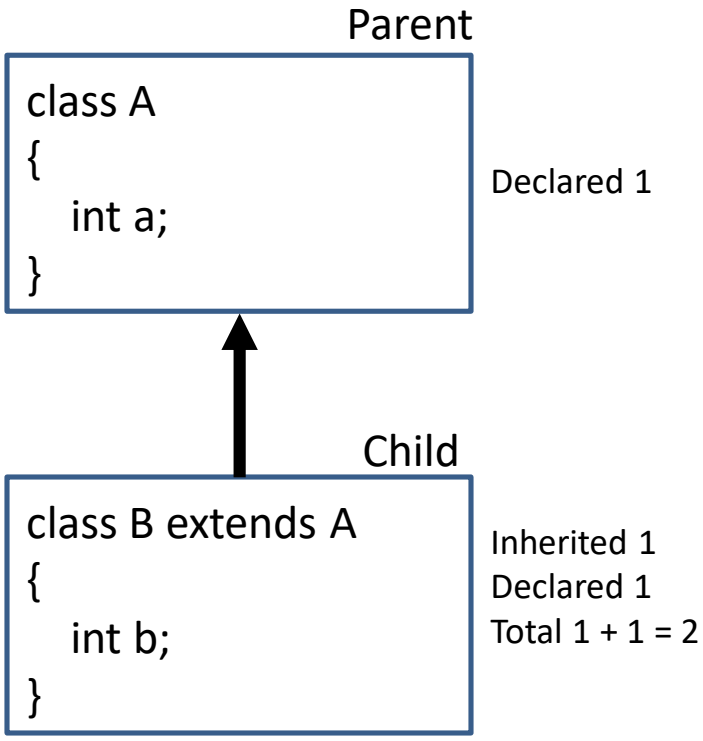
NOTE: 1

- **Parents are called generalized.**
- **Children are called specialized.**

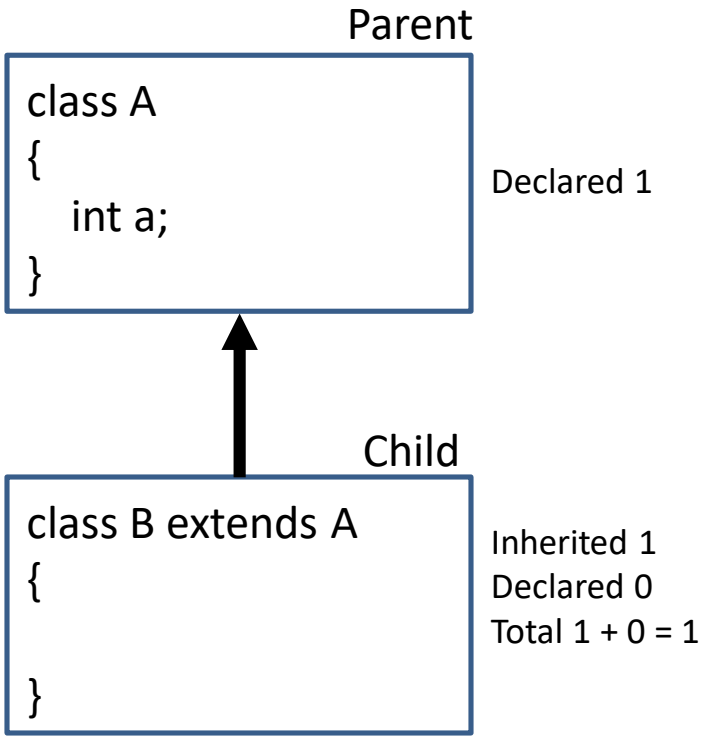
NOTE: 2

private members, constructors, and initializers are not inherited to the child class.

Example 1



Example 2



EXAMPLE :

- With the help of the child class reference type, we can use the members of the parent's class as well as the child.
- With the help of parent class reference, we can use only the members of a parent but not the child class.

PARENT CLASS :

The parent class is also known as a superclass or base class.

CHILD CLASS :

The child class is also known as a subclass or derived class.

NOTE: Is-A relationship is achieved with the help of inheritance.

INHERITANCE

INHERITANCE :

The process of one class acquiring all the properties and behaviour from the other class is called inheritance.

In java, we can achieve inheritance with the help of

1. **extends keyword**
2. **implements keyword**

extends keyword :

extends keyword is used to achieve inheritance between two classes.

EXAMPLE :

```
class A
{
int i = 10;
}
class B extends A
{
int j = 20;
public static void main(String args[])
{
B b = new B();
S.o.pln( b.i ); //CTS
S.o.pln( b.j ); //CTS
A a = new A();
S.o.pln( a.i ); //CTS
S.o.pln( a.j ) //CTE
}
}
```

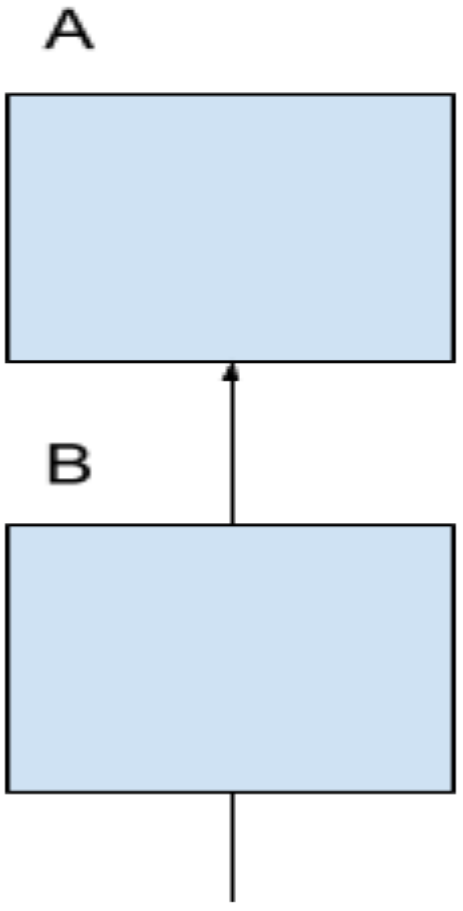
EXAMPLE :

```
class A
{
int i = 10;
}
class B extends A
{
int j = 20;
public static void main(String args[])
{
B b = new B();
S.o.pln( b.i ); //CTS
S.o.pln( b.j ); //CTS
A a = new A();
S.o.pln( a.i ); //CTS
S.o.pln( a.j ) //CTE with the help of parent variable we can only access
parent members
}
}
```

Types of inheritance

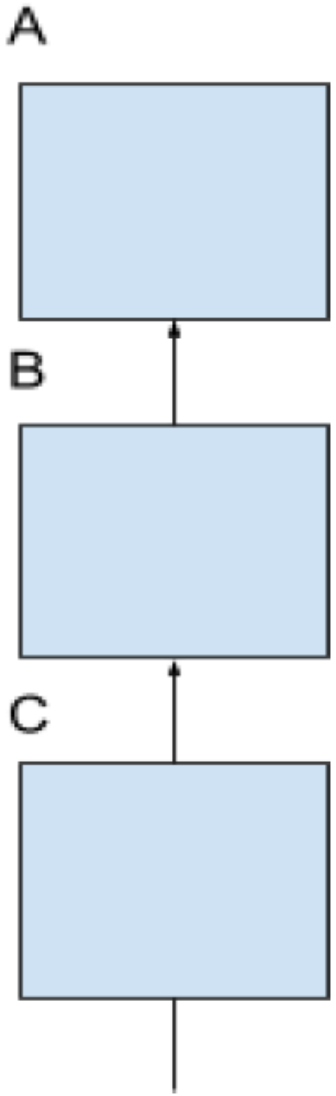
1. SINGLE LEVEL INHERITANCE:

- Inheritance of only one level is known as single-level inheritance.



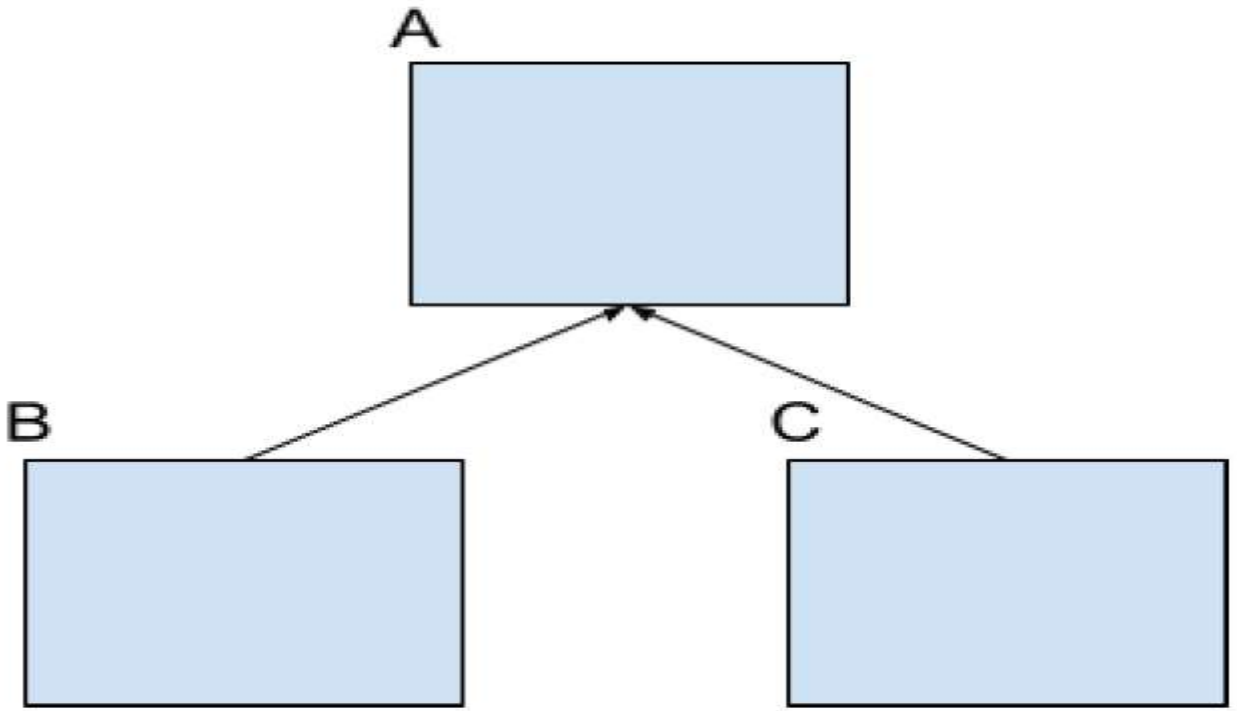
2. MULTILEVEL INHERITANCE

- Inheritance of more than one level is known as multi-level inheritance.



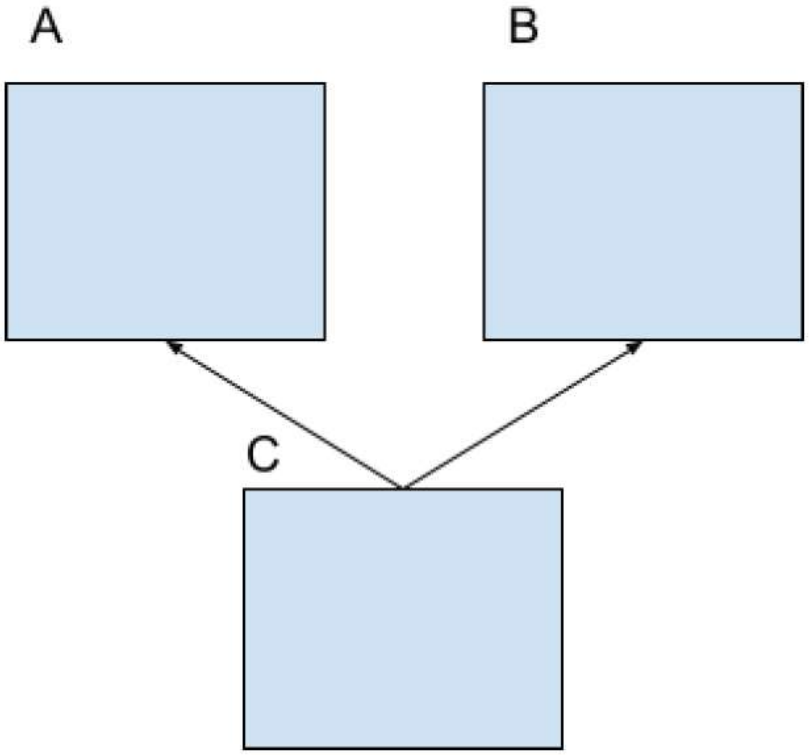
HIERARCHICAL INHERITANCE

- If a parent (superclass) has more than one child (subclass) at the same level then it is known as hierarchical inheritance



4. MULTIPLE INHERITANCE:

- If a subclass (child) inherits more than one parent (Superclass)then it is known as multiple inheritance.

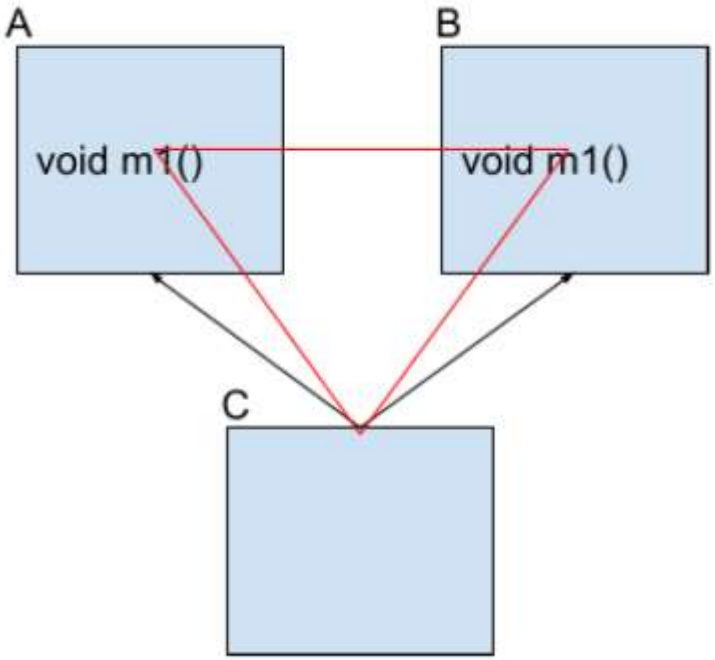


NOTE:

- **Multiple inheritance has a problem known as the diamond problem.**
- **Because of the diamond problem, we can't achieve multiple inheritance with the help of class.**
- **In java, we can achieve multiple inheritance with the help of an interface.**

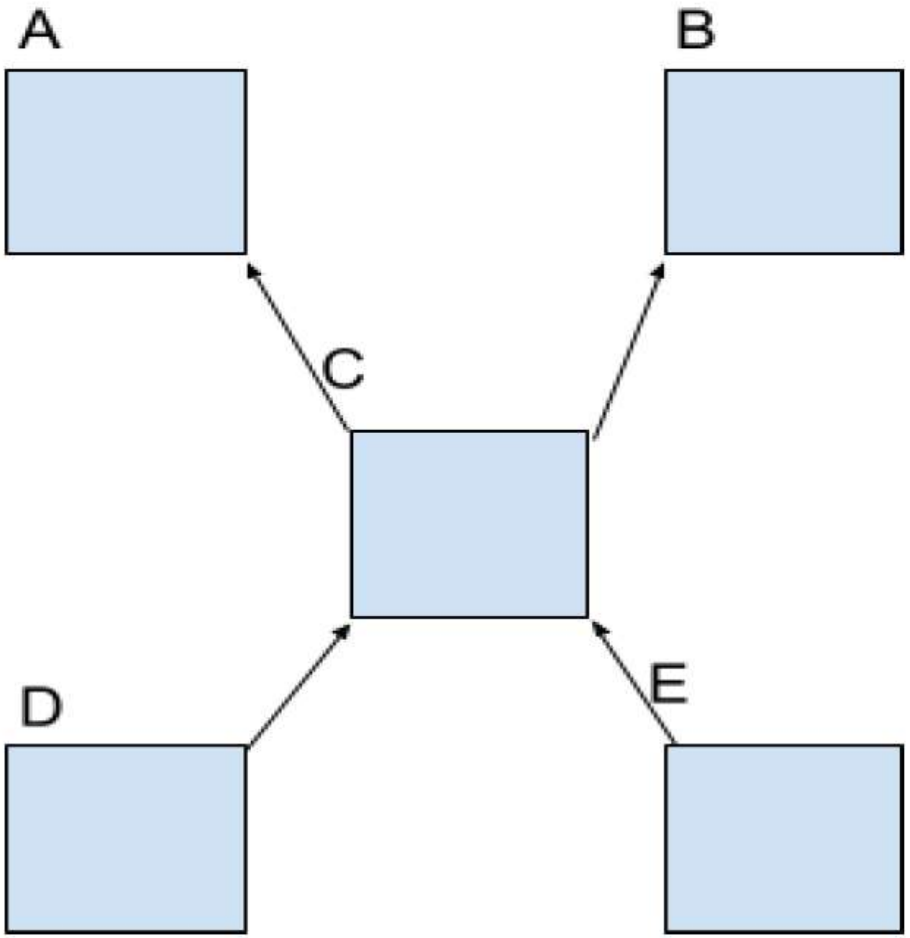
DIAMOND PROBLEM:

- Assume that there are two classes A and B. Both are having the method with the same signature. If class C inherits A and B then these two methods are inherited to C.
- **Now an ambiguity arises when we try to call the superclass method with the help of subclass reference.**
- This problem is known as the **diamond problem.**



5. HYBRID INHERITANCE:

- The combination of multiple inheritance and hierarchical inheritance is known as hybrid inheritance.



super() CALL STATEMENT:

- super is a keyword, it is used to access the members of the superclass.

● super() call statement is used to call the constructor of the parent class from the child class constructor.

PURPOSE OF SUPER() STATEMENT:

● When the object is created, the super call statement helps to load the nonstatic members of the parent class into the child object.

● We can also use the super() call statement to pass the data from the subclass to the parent class.

RULE TO USE SUPER() STATEMENT

● super() call statement should always be the first instruction in the constructor call.

● If a programmer doesn't use the super() call statement, then the compiler will add a no-argument super call statement into the constructor body.

Difference between super and super()

super()	super
➢ It is used to invoke the constructors of parent class from sub/child class	➢ It is used to access variables and methods of parent class from subclass
➢ It is used only inside constructor body	➢ It can be used inside any non static context
➢ super() should always be the first statement	➢ It can be used anywhere inside the constructor body

this()	super()
It is used to call the constructor of the same class	It is used to call the constructor of the parent class(Superclass)
It is used to represent the instance of child class	It is used to represent the instance of parent class
It should be used as the first statement in a constructor block	It should be used as the first statement in a constructor block