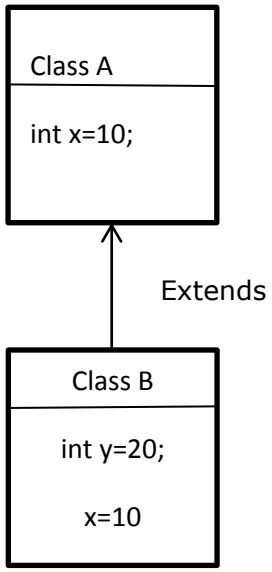
Inheritance(Is-A relationship)

Inheritance is one of the oops principal of java. The process of one class is acquiring properties of another class is known as inheritance. Inheritance can be achieved using extends keyword.

The class which is inheriting properties of another class is called a sub-class or derived class. The class from which property is inherited I known as super-class or base class.
There are four types of inheritance in java:
1. Single-level inheritance
2. Multiple inheritance
3. Multilevel inheritance
4. Hierarchial inheritance

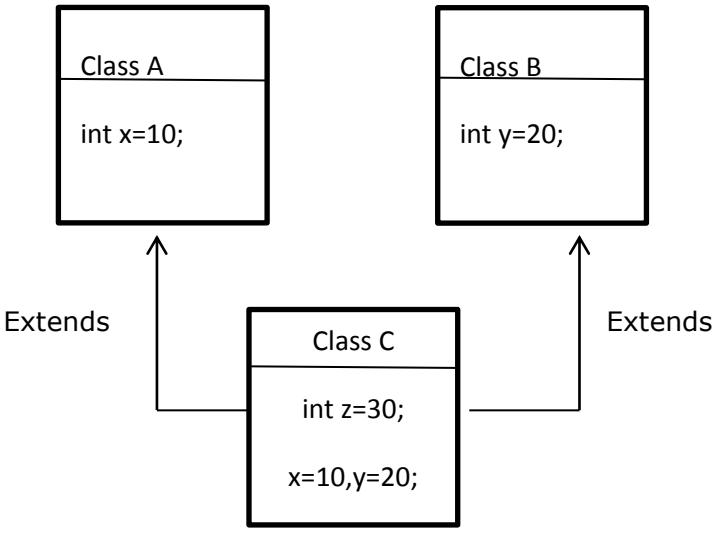
Single-level inheritance:

Process of one sub-class inheriting properties of one super class is known as single-level inheritance.



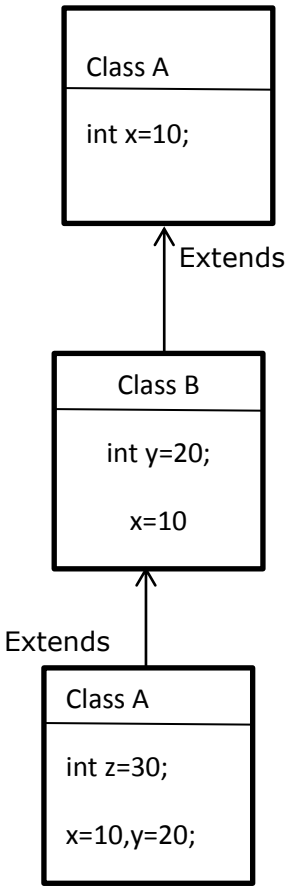
Multiple Inheritance:

The process of one sub-class inheriting properties from multiple super-class is known as multiple inheritance.



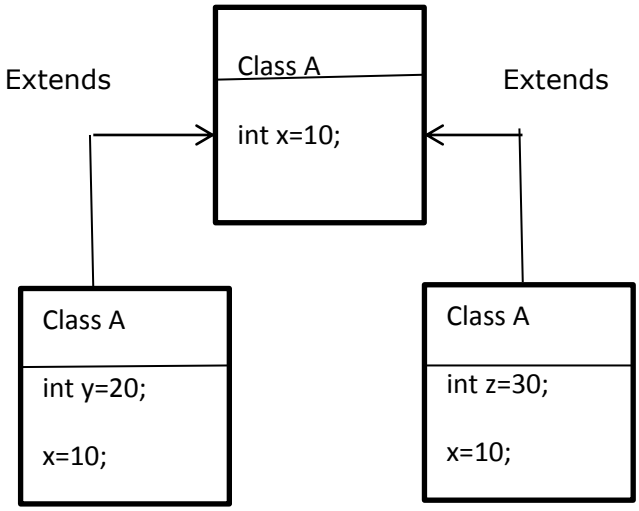
Multi-level inheritance

The process of one class inheriting properties of another class and that class is subclass of some other class is known as multilevel inheritance.



Hierarchial Inheritance

Process of one superclass is inherited by multiple subclasses is known as hierarchial inheritance.



Following properties cannot be inherited to sub-class

1. Private members

2. Constructor

Inheritance is also called as Is-A- Relationship.

// SIngle level inheritance

```
class Faculty
{
double salary=60000;
void sal()
{
System.out.println("Faculty method");
}
}

class Mfaculty extends Faculty
{

double bonus=60000;

void bon()
{
System.out.println("MathFaculty method");
}
}

class Inheritance3
{
public static void main(String args[])
{
Mfaculty f=new Mfaculty();
double Total=f.salary+f.bonus;
System.out.println("The salary total is" +Total);
f.bon();
f.sal();
}
}
```

2. Example for multilevel inheritance

```
class Faculty
{
double salary=60000;
void sal()
{
System.out.println("Faculty method");
}
}

class Mfaculty extends Faculty
{
double bonus=3000;
```

```
void bon()
{
System.out.println("MathFaculty method");
}
}

class HRA extends Mfaculty
{
double Hra=2000;
void hr()
{
System.out.println("HRA method");
}
}

class Inheritance4
{
public static void main(String args[])
{
HRA h=new HRA();
double Total=h.salary+h.bonus+h.Hra;
System.out.println("The salary total is" +Total);
h.bon();
h.sal();
h.hr();
}
}
```

3. Hierarchical inheritance

```
class Vehicle
{
String Ins_Name;
String Reg_Number;
}

class Bike extends Vehicle
{
String company;
Bike(String a, String b, String c)
{
Ins_Name=a;
Reg_Number=b;
company=c;
}
}
class Car extends Vehicle
{
String Type;
Car(String a, String b, String c)
{
Ins_Name=a;
Reg_Number=b;
Type=c;
}
}
```

```
class Inheritance1
{
public static void main(String args[])
{
Bike b=new Bike("New_India","KA-09","Honda");
System.out.println(" Bike insurance name :" +b.Ins_Name);
System.out.println(" Bike registration:"+b.Reg_Number);
System.out.println(" Bike company:"+b.company);

Car c=new Car("New_India","KA-09","Automatic");

System.out.println("================================");
System.out.println(" car insurance name :" +c.Ins_Name);
System.out.println(" car registration:"+c.Reg_Number);
System.out.println(" car Type:"+c.Type);
}
}
```

This and super keyword

This is a keyword in java used to access the current class members. This keyword can be used inside non-static context and constructor.

Super is a keyword in java used to access the super class members. Super keyword can be used inside non-static context and constructor.

This and super keywords cannot be used inside static context.

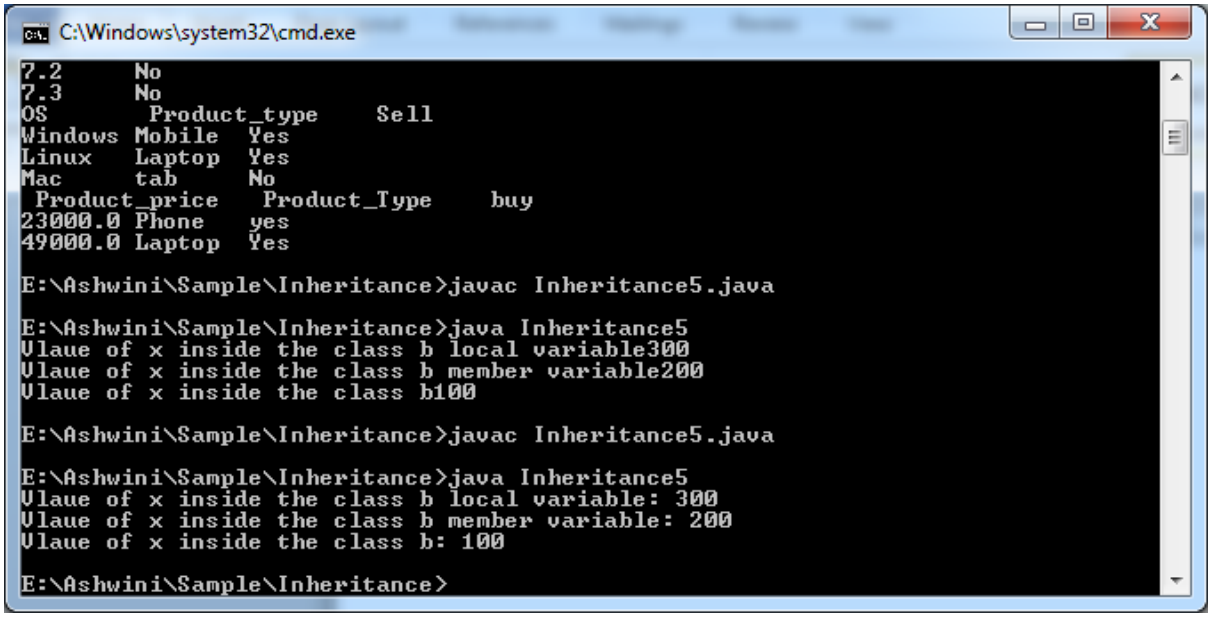
If identifier are same only then super and this can be used.

```
class a
{
int x=100;
}

class b extends a
{
int x=200;
void test()
{
int x=300;
System.out.println("Vlaue of x inside the class b local variable: "+x);
System.out.println("Vlaue of x inside the class b member variable: "+this.x);
System.out.println("Vlaue of x inside the class b: "+super.x);
}
}

class Inheritance5
{
public static void main(String[] args)
{
b b1=new b();
b1.test();
}
}
```

O/p:



2. Prog using this and super inside constructor

```
class person
{
String name;
void m1()
{
System.out.println(“This is method inside person");
}
}
class employee extends person
{
double salary;
employee(String name,double salary)
{
this.salary=salary;
super.name=name;
}
void m1()
{
System.out.println("This is method inside employee");
}
void test()
{
this.m1();
super.m1();
}
}
class Inheritance7
{
public static void main(String[] args)
{
employee e=new employee("Ashwini",60000.00);
System.out.println("The name of employee: "+e.name);
System.out.println("The salary of employee: "+e.salary);
e.test();
}
}
```