**构造器**

子类是不继承父类的构造器（构造方法或者构造函数）的，它只是调用（隐式或显式）。如果父类的构造器带有参数，则必须在子类的构造器中显式地通过 super 关键字调用父类的构造器并配以适当的参数列表。

如果父类构造器没有参数，则在子类的构造器中不需要使用 super 关键字调用父类构造器，系统会自动调用父类的无参构造器。

实例

class SuperClass {

private int n;

SuperClass(){

System.out.println("SuperClass()");

}

SuperClass(int n) {

System.out.println("SuperClass(int n)");

this.n = n;

}

}

// SubClass 类继承

class SubClass extends SuperClass{

private int n;

SubClass(){ // 自动调用父类的无参数构造器

System.out.println("SubClass");

}

public SubClass(int n){

super(300); // 调用父类中带有参数的构造器

System.out.println("SubClass(int n):"+n);

this.n = n;

}

}

// SubClass2 类继承

class SubClass2 extends SuperClass{

private int n;

SubClass2(){

super(300); // 调用父类中带有参数的构造器

System.out.println("SubClass2");

}

public SubClass2(int n){ // 自动调用父类的无参数构造器

System.out.println("SubClass2(int n):"+n);

this.n = n;

}

}

public class TestSuperSub{

public static void main (String args[]){

System.out.println("------SubClass 类继承------");

SubClass sc1 = new SubClass();

SubClass sc2 = new SubClass(100);

System.out.println("------SubClass2 类继承------");

SubClass2 sc3 = new SubClass2();

SubClass2 sc4 = new SubClass2(200);

}

}

输出结果为：

------SubClass 类继承------

SuperClass()

SubClass

SuperClass(int n)

SubClass(int n):100

------SubClass2 类继承------

SuperClass(int n)

SubClass2

SuperClass()

SubClass2(int n):200