成员变量 :实例变量和类变量

[修饰符] interface 接口名 [extends 父接口名列表]{

[public] [static] [final] 类常量;  
[public] [abstract] 方法;  
}

interface a

{

void A();  
}

Interface b

{

void B();   
}

Interface c extends a, b

{

void C();

}|

抽象类：

abstract class ClassName {

    abstract void fun();

}

抽象方法是一种特殊的方法：它只有声明，而没有具体的实现

抽象方法必须用abstract关键字进行修饰。如果一个类含有抽象方法，则称这个类为抽象类，抽象类必须在类前用abstract关键字修饰。因为抽象类中含有无具体实现的方法，所以不能用抽象类创建对象。

包含抽象方法的类称为抽象类，但并不意味着抽象类中只能有抽象方法，它和普通类一样，同样可以拥有成员变量和普通的成员方法。注意，抽象类和普通类的主要有三点区别：

　　1）抽象方法必须为public或者protected（因为如果为private，则不能被子类继承，子类便无法实现该方法），缺省情况下默认为public。

　　2）抽象类不能用来创建对象；

　　3）如果一个类继承于一个抽象类，则子类必须实现父类的抽象方法。如果子类没有实现父类的抽象方法，则必须将子类也定义为为abstract

　接口中可以含有 变量和方法。但是要注意，接口中的变量会被隐式地指定为public static final变量（并且只能是public static final变量，用private修饰会报编译错误），而方法会被隐式地指定为public abstract方法且只能是public abstract方法（用其他关键字，比如private、protected、static、 final等修饰会报编译错误），并且接口中所有的方法不能有具体的实现，也就是说，接口中的方法必须都是抽象方法。从这里可以隐约看出接口和抽象类的区别，接口是一种极度抽象的类型，它比抽象类更加“抽象”，并且一般情况下不在接口中定义变量。

1）抽象类可以提供成员方法的实现细节，而接口中只能存在public abstract 方法；

2）抽象类中的成员变量可以是各种类型的，而接口中的成员变量只能是public static final类型的；

　3）接口中不能含有静态代码块以及静态方法，而抽象类可以有静态代码块和静态方法；

　4）一个类只能继承一个抽象类，而一个类却可以实现多个接口。

嵌套在另一个接口中的接口自动就是public的，而不能为private。

当实现某个接口时，并不需要实现嵌套在其内部的任何接口。

子类也并不是完全继承父类的所有方法。  
　　1）能够继承父类的public和protected成员方法；不能够继承父类的private成员方法；  
　　2）对于父类的包访问权限成员方法，如果子类和父类在同一个包下，则子类能够继承；否则，子类不能够继承；  
　　3）对于子类可以继承的父类成员方法，如果在子类中出现了同名称的成员方法，则称为覆盖，即子类的成员方法会覆盖掉父类的同名成员方法。如果要在子类中访问父类中同名成员方法，需要使用super关键字来进行引用。