**抽象类（abstract修饰)：**

在普通类的基础上扩充了⼀些抽象方法，抽象方法指的是只声明而未实现的⽅法（即没有方法体）,所以抽象类无法实例化对象。

**抽象类必须有子类，因此不能被final修饰**

抽象类的子类必须实现抽象类的所有抽象方法（非抽象类)

抽象类的对象可以通过对象多态性（向上转型）

**抽象方法不能使用private定义，因为抽象⽅法必须要能被覆写**

Person per = new Student() ; //实例化⼦类，向上转型
per.getPersonInfo() ; //被⼦类所覆写的⽅法

private与abstract不能同时使⽤

一个抽象类可以没有抽象方法，但有抽象方法一定是抽象类

**内部抽象类允许使用static**

**接口（interface修饰）：**

接口中的方法都是抽象方法

接口中定义的成员变量默认为常量（final修饰）

**接口中的成员变量默认为public static final,方法默认为public abstract**

**接口不可以被实例化**，但可以利用子类的向上转型通过实例化子类来得到接⼝的实例化对象（前提：子类实现接口）

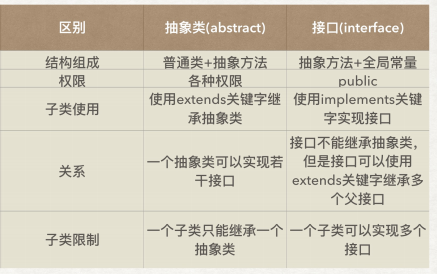
interface IMessage{
public static final String MSG = "I am a biter" ; // 全局常量
public abstract void print() ; // 抽象⽅法
}

5.**子类可以实现多个接口，子类（不是抽象类）必须覆写接口中的全部抽象方法**。

6.**抽象类可以使用implements实现多个接口**，但不实现接口的抽象方法，如果该抽象类再被继承，则该子类需要实现所有抽象方法（包括接口中的）

7.⼀个接口可以继承多个父接口

**抽象类与接口区别：**



对象的实例化步骤：

进行类加载

进行类对象的空间开辟

进行类对象中的属性初始化(构造方法)