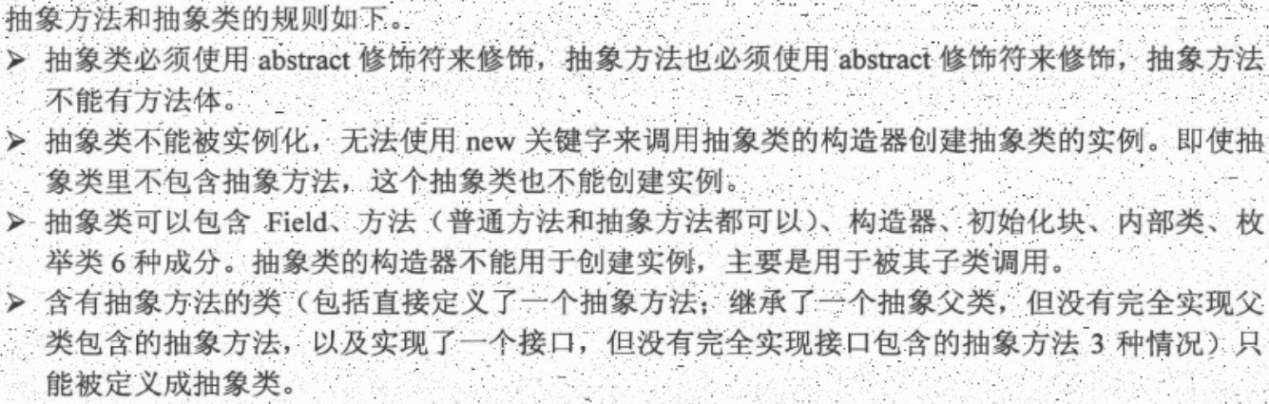
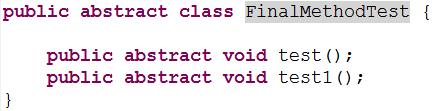
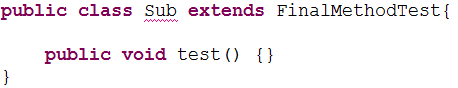
**抽象方法和抽象类**

抽象方法和抽象类必须使用abstract修饰符来定义，有抽象方法的类智能被定义为抽象类，抽象类可以没有抽象方法。



也就是说如果一个继承于抽象类的类要变为普通类则必须重写所有的抽象方法：

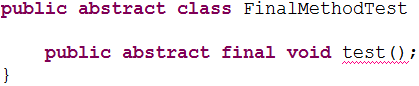




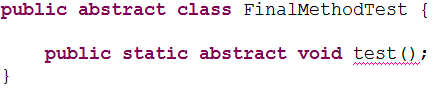
上面的原因就是因为没有重写父类所有的抽象方法造成的，解决办法就是（已验证）：

1. 将子类改写为抽象类
2. 重写父类所有的方法

注意：当使用abstract修饰类时，表明这个类只能被继承，当使用abstract修饰方法时，表明这个方法必须由子类提供实现（即重写），而final修饰的类不能被继承，final修饰的方法不能被重写，因此，final和abstract永远不能同时使用。



除此之外，当使用static修饰一个方法时，表明这个方法属于该类本身，即通过类就可调用该方法，但如果该方法被定义为抽象方法，则将导致通过该类来调用该方法时出现错误（调用一个没有方法体的方法肯定会引起错误），因此static今儿abstract不能同时修饰某个方法，即没有所谓的类抽象方法



**更彻底的抽象：接口**

接口里不能包含普通方法，接口里的所有方法都是抽象方法

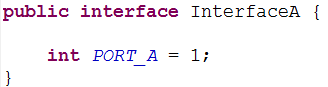
接口里的Field只能是常量，接口里的方法只能是抽象方法

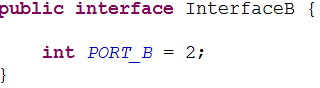
对于接口定义的常量Field而言，它们是接口相关的，而且它们只能是常量，因此系统会自动为这些Field增加static和final两个修饰符，也就是说，在接口中定义Field时，不管是否使用public static final修饰符，接口里的Field总将使用这三个修饰符来修饰。而且，接口里没有构造器和初始化块，因此接口里定义的Field只能在定义时指定默认值。

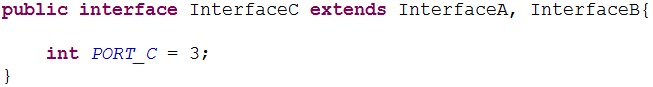
对于接口定义的方法而言，它们只能抽象方法，因此系统会自动为其增加abstract修饰符；由于接口里的方法全部是抽象方法，因此接口里不允许定义静态方法，即不可使用static修饰接口里定义的方法，不管定义接口里方法时是否使用public abstract，接口里的方法总是使用public abstract来修饰。

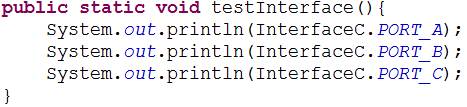
**接口的继承**

接口的继承和类继承不一样，接口完全支持多继承，即一个接口可以有多个直接父接口，和类继承相似，子接口扩展某个父接口，将会获得父接口里定义的所有抽象方法、常量Field、内部类和枚举类定义。











一个类实现了一个或多个接口之后，这个类必须完全实现这些接口所定义的全部抽象方法（也就是重写这些抽象方法）；否则，该类将保留从父接口哪里继承到的抽象方法，该类也必须定义成抽象类。

