目录

1.父类的静态方法能否被子类重写

不能

因为静态方法从程序开始运行后就已经分配了内存，也就是说已经写死了。所有引用到该方法的对象（父类的对象也好子类的对象也好）所指向的都是同一块内存中的数据，也就是该静态方法。

子类中如果定义了相同名称的静态方法，并不会重写，而应该是在内存中又分配了一块给子类的静态方法，没有重写这一说。

2. final，finally，finalize 的区别

就没有什么共同点好嘛。

以下简单分析一下 final,finally,finalize 怎么使用:

(1)final 修饰符（用于关键字）

final 用于控制成员、方法或者是一个类是否可以被重写或者继承等功能。

如果类被声明为 final，意味着它不能再派生出新的子类，不能作为父类被继承。将变量或者方法声明为 final，可以保证他们在使用中不被改变。

其初始化可以在两个地方：一是其定义处，也就是说，在 final 变量定义时直接给其赋值;二是构造函数中。这 2 个地方只能选其一，

要么在定义处直接给其赋值，要么在构造函数中给值，并且在以后的引用中，**只能读取，不可修改**。被声明为 final 的方法也同样只能使用，不能重写。

(2)finally(用于异常处理)

一般是用于异常处理中，提供 finally 块来执行任何的清楚操作，try{} catch(){} finally{}。finally 关键字是**对 java 异常处理模型的最佳补充。finally 结构使代码总会执行，不管有无异常发生**。使得 finally 可以维护对象的内部状态，并可以清理非内存资源。

finally 在 try,catch 中可以有，可以没有。如果 trycatch 中有 finally 则必须执行 finally 块中的操作。一般情况下，用于关闭文件的读写操作，或者是关闭数据库的连接等等。

(3)finalize（用于垃圾回收）

finalize 这个是方法名。在 java 中，允许使用 finalize()方法在垃圾收集器将对象从内存中清理出去之前做必要的清理工作。（被回收之前执行的操作方法）

这个方法是由垃圾收集器在确定这个对象没有被引用是对这个对象调用的。它是 Object 类中定义的，因此，所有的类都继承了它。finalize()方法是在垃圾收集器删除对象之前对这个对象调用的。

3. 静态属性和静态方法是否可以被继承？是否可以被重写？以及原因？

父类的静态属性和方法可以被子类继承

不可以被子类重写：当父类的引用指向子类时，使用对象调用静态方法或者静态变量，是调用的父类中的方法或者变量。并没有被子类改写。

原因：

因为静态方法从程序开始运行后就已经分配了内存，也就是说已经写死了。所有引用到该方法的对象（父类的对象也好子类的对象也好）所指向的都是同一块内存中的数据，也就是该静态方法。

子类中如果定义了相同名称的静态方法，并不会重写，而应该是在内存中又分配了一块给子类的静态方法，没有重写这一说。