JavaJDK：小心使用boxingJava认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/561/2021_2022_JavaJDK_EF_BC_9A_c104_561648.htm 自动装箱与拆箱的功能事实上是编译器来帮您的忙，编译器在编译时期依您所编写的语法，决定是否进行装箱或拆箱动作。例如：Integer i = 100. 相当于编译器自动为您作以下的语法编译：Integer i = new Integer(100). 所以自动装箱与拆箱的功能是所谓的“ 编译器蜜糖 ”(Compiler Sugar)，虽然使用这个功能很方便，但在程序运行阶段您得了解Java的语义。例如下面的程序是可以通过编译的： Integer i = null. int j = i. 这样的语法在编译时期是合法的，但是在运行时期会有错误，因为这种写法相当于： Integer i = null. int j = i.intValue(). null表示i没有参考至任何的对象实体，它可以合法地指定给对象参考名称。由于实际上i并没有参考至任何的对象，所以也就不可能操作intValue()方法，这样上面的写法在运行时会出现NullPointerException错误。 自动装箱、拆箱的功能提供了方便性，但隐藏了一些细节，所以必须小心。再来看范例4.6，您认为结果是什么呢？ 范例4.6 AutoBoxDemo2.java public class AutoBoxDemo2 { public static void main(String[] args) { Integer i1 = 100. Integer i2 = 100. if (i1 == i2) System.out.println("i1 == i2"). else System.out.println("i1 != i2"). } } 从自动装箱与拆箱的机制来看，可能会觉得结果是显示i1 == i2，您是对的。那么范例4.7的这个程序，您觉得结果是什么？ 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com