JAVA字符谜题5:令人晕头转向的Hello PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/138/2021_2022_JAVA_E5_AD_97_E7_AC_A6_c97_138385.htm 下面的程序是对一个老生常谈的例子做出了稍许的变化之后的版本。那么，它会打印出什么呢？ /** * Generated by the IBM IDL-to-Java compiler, version 1.0 * from F:\TestRoot\apps\a1\units\include\PolicyHome.idl * Wednesday, June 17, 1998 6:44:40 o ’ clock AM GMT 00:00 */ public class Test{ public static void main(String[] args){ System.out.print("Hell"). System.out.println("o world"). } } 这个谜题看起来相当简单。该程序包含了两条语句，第一条打印Hell，而第二条在同一行打印o world，从而将两个字符串有效地连接在了一起。因此，你可能期望该程序打印出Hello world。但是很可惜，你犯了错，实际上，它根本就通不过编译。 问题在于注释的第三行，它包含了字符\units。这些字符以反斜杠（\）以及紧跟着的字母u开头的，而它（\u）表示的是一个Unicode转义字符的开始。遗憾的是，这些字符后面没有紧跟四个十六进制的数字，因此，这个Unicode转义字符是病构的，而编译器则被要求拒绝该程序。Unicode转义字符必须是良构的，即使是出现在注释中也是如此。 在注释中插入一个良构的Unicode转义字符是合法的，但是我们几乎没有什么理由去这么做。程序员有时会在JavaDoc注释中使用Unicode转义字符来在文档中生成特殊的字符。 // Unicode转义字符在JavaDoc注释中有问题的用法 /** * This method calls itself recursively, causing a * StackOverflowError to be thrown. * The algorithm is due to Peter

von der Ah\u00E9. */ 这项技术表示了Unicode转义字符的一种没什么用处的用法。在Javadoc注释中，应该使用HTML实体转义字符来代替Unicode转义字符： /** * This method calls itself recursively, causing a * StackOverflowError to be thrown. * The algorithm is due to Peter von der Ah é . */ 前面的两个注释都应该是的在文档中出现的名字为“ Peter der Ah é ”，但是后一个注释在源文件中还是可理解的。 可能你会感到很诧异，在这个谜题中，问题出在注释这一信息来源自一个实际的bug报告。该程序是机器生成的，这使得我们很难追踪到问题的源头IDL-to-Java编译器。为了避免让其他程序员也陷入此境地，在没有将Windows文件名进行预先处理，以消除的其中的反斜杠的情况下，工具应该确保不将Windows文件名置于所生成的Java源文件的注释中。 总之，要确保字符\u不出现在一个合法的Unicode转义字符上下文之外，即使是在注释中也是如此。在机器生成的代码中要特别注意此问题。