JAVA字符谜题4:转义字符的溃败 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_138382.htm 下面的程序使用了两个Unicode的转义字符，它们是用其十六进制代码来表示Unicode字符。那么，这个程序会打印什么呢？ public class EscapeRout{ public static void main(String[] args){ // \u0022 是双引号的Unicode转义字符 System.out.println("a\u0022.length() \u0022b".length()). }} 对该程序的一种很肤浅的分析会认为它应该打印出26，因为在由两个双引号"a\u0022.length() \u0022b"标识的字符串之间总共有26个字符。 稍微深入一点的分析会认为该程序应该打印16，因为两个Unicode转义字符每一个在源文件中都需要用6个字符来表示，但是它们只表示字符串中的一个字符。因此这个字符串应该比它的外表看其来要短10个字符。 如果你运行这个程序，就会发现事情远不是这么回事。它打印的既不是26也不是16，而是2。 理解这个谜题的关键是要知道：Java对在字符串字面常量中的Unicode转义字符没有提供任何特殊处理。编译器在将程序解析成各种符号之前，先将Unicode转义字符转换成为它们所表示的字符[JLS 3.2]。因此，程序中的第一个Unicode转义字符将作为一个单字符字符串字面常量（"a"）的结束引号，而第二个Unicode转义字符将作为另一个单字符字符串字面常量（"b"）的开始引号。程序打印的是表达式"a".length() "b".length()，即2。 如果该程序的作者确实希望得到这种行为，那么下面的语句将要清楚得多： System.out.println("a".length() "b".length()). 更有可能的情况是该作者希望将两个双引号字符置于字符串字面常

量的内部。使用Unicode转义字符你是不能实现这一点的，但是你可以使用转义字符序列来实现[JLS 3.10.6]。表示一个双引号的转义字符序列是一个反斜杠后面紧跟着一个双引号（\ ”）。如果将最初的程序中的Unicode转义字符用转义字符序列来替换，那么它将打印出所期望的16：System.out.println("a\".length() \"b".length()). 许多字符都有相应的转义字符序列，包括单引号（\ ’）、换行（\n）、制表符（\t）和反斜线（\\）。你可以在字符字面常量和字符串字面常量中使用转义字符序列。 实际上，你可以通过使用被称为八进制转义字符的特殊类型的转义字符序列，将任何ASCII字符置于一个字符串字面常量或一个字符字面常量中，但是最好是尽可能地使用普通的转义字符序列。 普通的转义字符序列和八进制转义字符都比Unicode转义字符要好得多，因为与Unicode转义字符不同，转义字符序列是在程序被解析为各种符号之后被处理的。 ASCII是字符集的最小公共特性集，它只有128个字符，但是Unicode有超过65,000个字符。一个Unicode转义字符可以被用来在只使用ASCII字符的程序中插入一个Unicode字符。一个Unicode转义字符精确地等价于它所表示的字符。 Unicode转义字符被设计为用于在程序员需要插入一个不能用源文件字符集表示的字符的情况。它们主要用于将非ASCII字符置于标识符、字符串字面常量、字符字面常量以及注释中。偶尔地，Unicode转义字符也被用来在看起来颇为相似的数个字符中明确地标识其中的某一个，从而增加程序的清晰度。 总之，在字符串和字符字面常量中要优先选择的是转义字符序列，而不是Unicode转义字符。Unicode转义字符可能会因为它们在编译序列中被处理得过

早而引起混乱。不要使用Unicode转义字符来表示ASCII字符。在字符串和字符字面常量中，应该使用转义字符序列；对于除这些字面常量之外的情况，应该直接将ASCII字符插入到源文件中。 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