Java字符谜题1:最后的笑声 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_138391.htm 下面的程序将打印出什么呢？ public class LastLaugh{ public static void main(String[] args){ System.out.print("H" "a"). System.out.print(’ H ’ ’ a ’). }} 你可能会认为这个程序将打印HaHa。该程序看起来好像是用两种方式连接了H和a，但是你所见为虚。如果你运行这个程序，就会发现它打印的是Ha169。那么，为什么它会产生这样的行为呢？ 正如我们所期望的，第一个对System.out.print的调用打印的是Ha：它的参数是表达式"H" "a"，显然它执行的是一个字符串连接。而第二个对System.out.print的调用就是另外一回事了。问题在于 ’ H ’ 和 ’ a ’ 是字符型字面常量，因为这两个操作数都不是字符串类型的，所以 操作符执行的是加法而不是字符串连接。 编译器在计算常量表达式 ’ H ’ ’ a ’ 时，是通过我们熟知的拓宽原始类型转换将两个具有字符型数值的操作数（ ’ H ’ 和 ’ a ’ ）提升为int数值而实现的。 从char到int的拓宽原始类型转换是将16位的char数值零扩展到32位的int。对于 ’ H ’ ，char数值是72，而对于 ’ a ’ ，char数值是97，因此表达式 ’ H ’ ’ a ’ 等价于int常量72 97，或169。 站在语言的立场上，若干个char和字符串的相似之处是虚幻的。语言所关心的是，char是一个无符号16位原始类型整数仅此而已。对类库来说就不尽如此了，类库包含了许多可以接受char参数，并将其作为Unicode字符处理的方法。 那么你应该怎样将字符连接在一起呢？你可以使用这些类库。例如，你可以使用一个字符串缓冲区： StringBuffer sb = new

StringBuffer(). sb.append(’ H ’). sb.append(’ a ’). System.out.println(sb). 这么做可以正常运行，但是显得很丑陋。其实我们还是有办法去避免这种方式所产生的拖沓冗长的代码。 你可以通过确保至少有一个操作数为字符串类型，来强制 操作符去执行一个字符串连接操作，而不是一个加法操作。这种常见的惯用法用一个空字符串（""）作为一个连接序列的开始，如下所示： System.out.println("" ’ H ’ ’ a ’). 这种惯用法可以确保子表达式都被转型为字符串。尽管这很有用，但是多少有一点难看，而且它自身可能会引发某些混淆。你能猜到下面的语句将会打印出什么吗？如果你不能确定，那么就试一下： System.out.print("2 2 = " 2 2). 如果使用的是JDK 5.0，你还可以使用 System.out.printf("%c%c", ’ H ’ , ’ a ’). 总之，使用字符串连接操作符使用格外小心。 操作符当且仅当它的操作数中至少有一个是String类型时，才会执行字符串连接操作；否则，它执行的就是加法。如果要连接的没有一个数值是字符串类型的，那么你可以有几种选择： 预置一个空字符串； 将第一个数值用String.valueOf显式地转换成一个字符串； 使用一个字符串缓冲区； 或者如果你使用的JDK 5.0，可以用printf方法。 这个谜题还包含了一个给语言设计者的教训。操作符重载，即使在Java中只在有限的范围内得到了支持，它仍然会引起混淆。为字符串连接而重载 操作符可能就是一个已铸成的错误。