



第五章

讲笑话游戏

本章内容:

- 使用 `print()` 函数的 `end` 关键字参数
- 转义字符
- 单引号和双引号的使用

`print()` 函数全解析

本书中的大多数游戏都会用到文本的输入和输出。输入是指用户通过键盘向电脑键入内容。输出则是显示在屏幕上的文本。在 Python 中，`print()` 函数用来在屏幕上输出文本。我们已经学过 `print()` 函数的一些基本用法，但是关于 Python 中的字符串和 `print()` 函数的用法，我们还有很多要学的。

游戏运行演示

```
What do you get when you cross a snowman with a vampire?
```

```
Frostbite!
```

```
What do dentists call an astronaut's cavity?
```

```
A black hole!
```

```
Knock knock.
```

```
Who's there?
```

Interrupting cow.

Interrupting cow wh-MOO!

讲笑话游戏源代码

这是我们讲笑话小游戏的源代码。你要把它输入到文本编辑器中，然后保存为 `jokes.py`。如果你不想自己输入这些代码可以到本书的官方网站去下载。网址：<http://inventwithpython.com/chapter5>。

注意！ 确定你是在 Python 3 而不是 Python 2 下运行你的程序。本书使用的是 Python 3，如果你在 Python 2 下运行会得到错误信息。想知道自己的 Python 的版本可以点击 **Help** 然后 **About IDLE** 查看。

jokes.py

这些代码可以从 <http://inventwithpython.com/jokes.py> 下载

如果输入完后得到错误信息，可以用在线 `diff` 工具(<http://inventwithpython.com/diff>)把你输入的代码和书中的代码对比，或者给作者发邮件 al@inventwithpython.com

```
1. print('What do you get when you cross a snowman with a vampire?')
2. input()
3. print('Frostbite!')
4. print()
5. print('What do dentists call a astronaut\'s cavity?')
6. input()
7. print('A black hole!')
8. print()
9. print('Knock knock.')
10.  input()
11.  print("Who's there?")
12.  input()
13.  print('Interrupting cow.')
14.  input()
15.  print('Interrupting cow wh', end='')
16.  print('-MOO!')
```

如果你不理解这个程序的代码，别担心。你只要保存并运行它就可以了。记住，如果你的程序有漏洞（`bug`），你可以用本书的在线 `diff` 工具，地址：<http://inventwithpython.com/chapter5>

它是怎么工作的

我们来仔细看看这些代码。

```
1. print('What do you get when you cross a snowman with a vampire?')
2. input()
3. print('Frostbite!')
4. print()
```

这里我们调用了 `print()` 函数三次。因为我们不想玩家直接看到笑话的妙语部分，所以我们在调用了 `print()` 函数之后调用了 `input()` 函数。这样玩家就会先看到第一行字，按了回车之后才会看到笑话的妙语。

玩家也可以输入一个字符串后再按回车，不过由于我们没有把这个字符串保存在变量中，所以程序不会理会它，而会直接执行下一行代码。

我们没有给最后一个 `print()` 函数传参数。这是为了让程序输出空白行。空白行很有用，它可以让文本之间有合理的距离，而不会堆在一起。

转义字符

```
5. print('What do dentists call a astronaut\'s cavity?')
6. input()
7. print('A black hole!')
8. print()
```

你也许注意到了，在上面的第一个 `print()` 函数中的单引号前面有一个斜杠。这个反斜杠 (`\` 是反斜杠，`/` 是正斜杠) 告诉我们斜杠后面的字符是一个**转义字符**。转义字符可以帮助我们输出那些在源代码中不方便输入的字符。转义字符有很多种，我们在 `print()` 函数中用到的是单引号。

在这里我们必须使用单引号转义字符，否则 `Python` 解释器会误以为这是字符串结束的标志，但事实上这个引号是字符串的一部分。当我们输出这个字符串的时候，那个反斜杠不会输出。

其他转义字符

那如果你真正想输出反斜杠的时候怎么办呢？下面这行代码是不行的：

```
>>> print('He flew away in a green\teal helicopter.')
```

上面这行代码的输出结果如下：

```
He flew away in a green    eal helicopter.
```

这是因为“teal”中的“t”被视为是转义字符，因为它前面有一个反斜杠。转义字符 t 是模拟你键盘上的 Tab 键的。转义字符主要用来表示不能直接输入的字符。

用这行代码试试：

```
>>> print('He flew away in a green\\teal helicopter.')
```

下面是 Python 的转义字符列表：

表 5-1:转义字符

转移字符	实际输出的内容
\\	反斜杠(\)
\'	单引号(')
\"	双引号(")
\n	下一行
\t	Tab 键

单引号和双引号

在 Python 中字符串不一定要位于单引号之间。你也可以把它们放在双引号之间。下面两行代码输出结果相同：

```
>>> print('Hello world')
Hello world
>>> print("Hello world")
Hello world
```

但这两种方式不能混用。如果你试图输入下面这行代码就会出错：

```
>>> print('Hello world")
SyntaxError: EOL while scanning single-quoted string
>>>
```

我喜欢用单引号，因为输入单引号的时候不用按住 shift 键。这样输入起来比较简单，而且用哪种电脑都不会介意。

不过你要记住，正如你想在单引号之间输入单引号时要用转义字符一样，在双引号之间输入双引号也要用转义字符。比如，你可以看看这几行代码：

```
>>> print('I asked to borrow Abe\'s car for a week. He said,
"Sure."')
```

```
I asked to borrow Abe's car for a week. He said, "Sure."
>>> print("He said, \"I can't believe you let him borrow your
car.\")
He said, "I can't believe you let him borrow your car."
```

你有没有注意到？当你在单引号之间输入双引号的时候不用使用转义字符，在双引号之间输入单引号的时候也不用使用转义字符。Python 解释器很聪明，它知道如果一个字符串由一种引号开始，那么另一种引号就不代表它的结束。

end 关键字参数

```
9. print('Knock knock.')
10.     input()
11.     print("Who's there?")
12.     input()
13.     print('Interrupting cow.')
14.     input()
15.     print('Interrupting cow wh', end='')
16.     print('-MOO!')
```

你有没有注意到第 15 行的 `print()` 函数的第二个参数？通常，`print()` 函数在输出一行字符串的同时会换行。（这也是调用一个没有参数的 `print()` 函数会输出一行空白行的原因。）但是 `print()` 函数还有一个可选的参数（这个参数名为 `end`）。我们把 `print()` 函数的这个参数叫做**关键字参数**。这个 `end` 参数很特别，而且如果想使用这个参数我们要用 `end=` 这样的语法。

注意，当你输入关键字和关键字参数的时候只用一个 `=` 号。也就是 `end=`，而不是 `end==`。

给 `end` 一个空白字符串会使 `print()` 函数在它输出的字符串的后面加一个空格而不是换行。这就是 `'-MOO!'` 出现在前一行的后面而不是下一行的原因。也就是在输出字符串 `'Interrupting cow wh'` 后没有换行。

总结

这一章我们研究了 `print()` 函数的不同用法。我们也知道了转义字符的作用是输出那些不方便或者无法输入的字符。转义字符以反斜杠 `\` 开始，后面接需要转义的字母。例如，`\n` 是指换行。如果想在字符串中使用反斜杠，你可以用 `\\` 这个转义字符。

默认情况下 `print()` 函数会自动在输出一行字符串后换行。大多数情况下这是很方便的。但是有时候我们不想换行，这时我们就可以给 `end` 关键字一个空白字

字符串。比如，想输出'spam'并且不换行，那你就可以这样写代码 `print('spam', end='')`。

通过这种控制，我们可以更加灵活和准确的输出字符串，得到我们想要的效果。