Python-CookBook：40、对数值进行取整

作者：zhaolixiang
原文链接：https://ld246.com/article/1567063069535
来源网站：链滴

问题

我们想将一个浮点数取整到固定的小数位。

3.1.2 解决方案

对于简单的取整操作，使用内建的round(value, ndigits)函数即可。示例如下：

```
>>> round(1.23, 1)
1.2
>>> round(1.27, 1)
1.3
>>> round(-1.27, 1)
-1.3
>>> round(1.25361,3)
1.254
>>>
```

当某个值恰好等于两个整数间的一半时，取整操作会取到离该值最接近的那个偶数上。也就是说，像1.5或2.5这样的值都会取整到2。

传递给round()的参数ndigits可以是负数，在这种情况下会相应地取整到十位、百位、千位等。示例下：

```
>>> a = 1627731
>>> round(a, -1)
1627730
>>> round(a, -2)
1627700
>>> round(a, -3)
1628000
>>>
```

3.1.3 讨论

在对值进行输出时别把取整和格式化操作混为一谈。如果只是将数值以固定的位数输出，一般来说是不着round()的。相反，只要在格式化时指定所需要的精度就可以了。示例如下：

```
>>> x = 1.23456
>>> format(x, '0.2f')
'1.23'
>>> format(x, '0.3f')
'1.235'
>>> 'value is {:0.3f}'.format(x)
'value is 1.235'
>>>
```

此外，不要采用对浮点数取整的方式来“修正”精度上的问题。比如，我们可能会倾向于这样做：

```
>>> a = 2.1
>>> b = 4.2
>>> c = a + b
```

```
>>> c
6.300000000000001
>>> c = round(c, 2)       # "Fix" result (???)
>>> c
6.3
>>>
```

对于大部分涉及浮点数的应用程序来说，一般来讲都不必（或者说不推荐）这么做。尽管这样会引入些小误差，但这些误差是可理解的，也是可容忍的。如果说避免出现误差的行为非常重要（例如在金类应用中），那么可以考虑使用decimal模块，这也正是下一节的主题。