1.12 序列中出现次数最多的元素¶

问题¶

怎样找出一个序列中出现次数最多的元素呢？

解决方案¶

`collections.Counter` 类就是专门为这类问题而设计的， 它甚至有一个有用的 `most_common()` 方法直接给了你答案。

为了演示，先假设你有一个单词列表并且想找出哪个单词出现频率最高。你可以这样做：

```
words = [
    'look', 'into', 'my', 'eyes', 'look', 'into', 'my', 'eyes',
    'the', 'eyes', 'the', 'eyes', 'the', 'eyes', 'not', 'around', 'the',
    'eyes', "don't", 'look', 'around', 'the', 'eyes', 'look', 'into',
    'my', 'eyes', "you're", 'under'
]
from collections import Counter
word_counts = Counter(words)
# 出现频率最高的3个单词
top_three = word_counts.most_common(3)
print(top_three)
# Outputs [('eyes', 8), ('the', 5), ('look', 4)]
```

讨论¶

作为输入， `Counter` 对象可以接受任意的由可哈希（`hashable`）元素构成的序列对象。 在底层实现上，一个 `Counter` 对象就是一个字典，将元素映射到它出现的次数上。比如：

```
>>> word_counts['not']
1
>>> word_counts['eyes']
8
>>>
```

如果你想手动增加计数，可以简单的用加法：

```
>>> morewords = ['why','are','you','not','looking','in','my','eyes']
>>> for word in morewords:
...     word_counts[word] += 1
...
>>> word_counts['eyes']
9
>>>
```

或者你可以使用 `update()` 方法：

```
>>> word_counts.update(morewords)
>>>
```

`Counter` 实例一个鲜为人知的特性是它们可以很容易的跟数学运算操作相结合。比如：

```
>>> a = Counter(words)
>>> b = Counter(morewords)
>>> a
Counter({'eyes': 8, 'the': 5, 'look': 4, 'into': 3, 'my': 3, 'around': 2,
"you're": 1, "don't": 1, 'under': 1, 'not': 1})
>>> b
Counter({'eyes': 1, 'looking': 1, 'are': 1, 'in': 1, 'not': 1, 'you': 1,
'my': 1, 'why': 1})
>>> # Combine counts
>>> c = a + b
>>> c
```

```
Counter({'eyes': 9, 'the': 5, 'look': 4, 'my': 4, 'into': 3, 'not': 2,
'around': 2, "you're": 1, "don't": 1, 'in': 1, 'why': 1,
'looking': 1, 'are': 1, 'under': 1, 'you': 1})
>>> # Subtract counts
>>> d = a - b
>>> d
Counter({'eyes': 7, 'the': 5, 'look': 4, 'into': 3, 'my': 2, 'around': 2,
"you're": 1, "don't": 1, 'under': 1})
>>>
```

毫无疑问， `Counter` 对象在几乎所有需要制表或者计数数据的场合是非常有用的工具。 在解决这类问题的时候你应该优先选择它，而不是手动的利用字典去实现。