1.16 过滤序列元素¶

问题¶

你有一个数据序列，想利用一些规则从中提取出需要的值或者是缩短序列

解决方案¶

最简单的过滤序列元素的方法就是使用列表推导。比如：

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> [n for n in mylist if n > 0]
[1, 4, 10, 2, 3]
>>> [n for n in mylist if n < 0]
[-5, -7, -1]
>>>
```

使用列表推导的一个潜在缺陷就是如果输入非常大的时候会产生一个非常大的结果集，占用大量内存。 如果你对内存比较敏感，那么你可以使用生成器表达式迭代产生过滤的元素。比如：

```
>>> pos = (n for n in mylist if n > 0)
>>> pos
<generator object <genexpr> at 0x1006a0eb0>
>>> for x in pos:
... print(x)
...
1
4
10
2
3
>>>
```

有时候，过滤规则比较复杂，不能简单的在列表推导或者生成器表达式中表达出来。 比如，假设过滤的时候需要处理一些异常或者其他复杂情况。这时候你可以将过滤代码放到一个函数中， 然后使用内建的 `filter()` 函数。示例如下：

```
values = ['1', '2', '-3', '-', '4', 'N/A', '5']
def is_int(val):
    try:
        x = int(val)
        return True
    except ValueError:
        return False
ivals = list(filter(is_int, values))
print(ivals)
# Outputs ['1', '2', '-3', '4', '5']
```

`filter()` 函数创建了一个迭代器，因此如果你想得到一个列表的话，就得像示例那样使用 `list()` 去转换。

讨论¶

列表推导和生成器表达式通常情况下是过滤数据最简单的方式。 其实它们还能在过滤的时候转换数据。比如：

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> import math
>>> [math.sqrt(n) for n in mylist if n > 0]
[1.0, 2.0, 3.1622776601683795, 1.4142135623730951, 1.7320508075688772]
>>>
```

过滤操作的一个变种就是将不符合条件的值用新的值代替，而不是丢弃它们。 比如，在一列数据中你可能不仅想找到正数，而且还想将不是正数的数替换成指定的数。 通过将过滤条件放到条件表达式中去，可以很容易的解决这个问题，就像这样：

```
>>> clip_neg = [n if n > 0 else 0 for n in mylist]
>>> clip_neg
[1, 4, 0, 10, 0, 2, 3, 0]
>>> clip_pos = [n if n < 0 else 0 for n in mylist]
>>> clip_pos
[0, 0, -5, 0, -7, 0, 0, -1]
>>>
```

另外一个值得关注的过滤工具就是 `itertools.compress()`， 它以一个 `iterable` 对象和一个相对应的 `Boolean` 选择器序列作为输入参数。 然后输出 `iterable` 对象中对应选择器为 `True` 的元素。 当你需要用另外一个相关联的序列来过滤某个序列的时候，这个函数是非常有用的。 比如，假如现在你有下面两列数据：

```
addresses = [
    '5412 N CLARK',
    '5148 N CLARK',
    '5800 E 58TH',
    '2122 N CLARK',
    '5645 N RAVENSWOOD',
    '1060 W ADDISON',
    '4801 N BROADWAY',
    '1039 W GRANVILLE',
]
counts = [ 0, 3, 10, 4, 1, 7, 6, 1]
```

现在你想将那些对应 `count` 值大于5的地址全部输出，那么你可以这样做：

```
>>> from itertools import compress
>>> more5 = [n > 5 for n in counts]
>>> more5
[False, False, True, False, False, True, True, False]
>>> list(compress(addresses, more5))
['5800 E 58TH', '1060 W ADDISON', '4801 N BROADWAY']
>>>
```

这里的关键点在于先创建一个 `Boolean` 序列，指示哪些元素符合条件。 然后 `compress()` 函数根据这个序列去选择输出对应位置为 `True` 的元素。

和 `filter()` 函数类似， `compress()` 也是返回的一个迭代器。因此，如果你需要得到一个列表， 那么你需要使用 `list()` 来将结果转换为列表类型。