4.7 迭代器切片¶

问题¶

你想得到一个由迭代器生成的切片对象，但是标准切片操作并不能做到。

解决方案¶

函数 `itertools.islice()` 正好适用于在迭代器和生成器上做切片操作。比如：

```
>>> def count(n):
...     while True:
...         yield n
...         n += 1
...
>>> c = count(0)
>>> c[10:20]
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'generator' object is not subscriptable

>>> # Now using islice()
>>> import itertools
>>> for x in itertools.islice(c, 10, 20):
...     print(x)
...
10
11
12
13
14
15
16
17
18
19
>>>
```

讨论¶

迭代器和生成器不能使用标准的切片操作，因为它们的长度事先我们并不知道(并且也没有实现索引)。 函数 `islice()` 返回一个可以生成指定元素的迭代器，它通过遍历并丢弃直到切片开始索引位置的所有元素。 然后才开始一个个的返回元素，并直到切片结束索引位置。

这里要着重强调的一点是 `islice()` 会消耗掉传入的迭代器中的数据。 必须考虑到迭代器是不可逆的这个事实。 所以如果你需要之后再次访问这个迭代器的话，那你就得先将它里面的数据放入一个列表中。