1.9 查找两字典的相同点¶

问题¶

怎样在两个字典中寻寻找相同点（比如相同的键、相同的值等等）？

解决方案¶

考虑下面两个字典：

```
a = {
    'x' : 1,
    'y' : 2,
    'z' : 3
}

b = {
    'w' : 10,
    'x' : 11,
    'y' : 2
}
```

为了寻找两个字典的相同点，可以简单的在两字典的 `keys()` 或者 `items()` 方法返回结果上执行集合操作。比如：

```
# Find keys in common
a.keys() & b.keys() # { 'x', 'y' }
# Find keys in a that are not in b
a.keys() - b.keys() # { 'z' }
# Find (key,value) pairs in common
a.items() & b.items() # { ('y', 2) }
```

这些操作也可以用于修改或者过滤字典元素。 比如，假如你想以现有字典构造一个排除几个指定键的新字典。 下面利用字典推导来实现这样的需求：

```
# Make a new dictionary with certain keys removed
c = {key:a[key] for key in a.keys() - {'z', 'w'}}
# c is {'x': 1, 'y': 2}
```

讨论¶

一个字典就是一个键集合与值集合的映射关系。 字典的 `keys()` 方法返回一个展现键集合的键视图对象。 键视图的一个很少被了解的特性就是它们也支持集合操作，比如集合并、交、差运算。 所以，如果你想对集合的键执行一些普通的集合操作，可以直接使用键视图对象而不用先将它们转换成一个 set。

字典的 `items()` 方法返回一个包含 (键，值) 对的元素视图对象。 这个对象同样也支持集合操作，并且可以被用来查找两个字典有哪些相同的键值对。

尽管字典的 `values()` 方法也是类似，但是它并不支持这里介绍的集合操作。 某种程度上是因为值视图不能保证所有的值互不相同，这样会导致某些集合操作会出现问题。 不过，如果你硬要在值上面执行这些集合操作的话，你可以先将值集合转换成 set，然后再执行集合运算就行了。