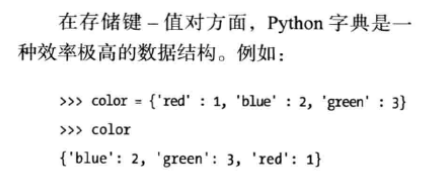
# 字典与集合

## 字典dict

字典是另一种可变容器模型，且可存储任意类型对象。



### 创建字典

字典用{}表示。

d = {} #空字典

字典的每个键值(key=>value)对用冒号(**:**)分割，每个对之间用逗号(**,**)分割。

d = {'Alice': '2341', 'Beth': '9102', 'Cecil': '3258'}

不可变类型可以用来作为键，如：如字符串，数字或元组。字典值可以是任何的 python 对象，既可以是标准的对象，也可以是用户定义的。

键不能重复。在创建字典的时候，如果键重复，后面的会覆盖前面的，即前面的键值对被忽略掉。

键不一定要是同样的类型：

dict2 = { 'abc': 123, 98.6: 37 }

### 访问字典

字典也被称为关联数组、映射、散列表。字典可以通过键值来访问值。

dict = {'Name': '飞入蓝天', 'Age': 7, 'Class': 'First'}

print(dict['Age']); # 更新 Age

### 修改字典

向字典添加新内容的方法是增加新的键/值对，修改已有键/值对如下实例:

dict = {'Name': '飞入蓝天', 'Age': 7, 'Class': 'First'}

dict['Age'] = 8; # 更新 Age

dict['School'] = "菜鸟教程" # 添加信息

删除字典或字典的某个键

del dict['Name'] # 删除键 'Name'，连同它的值一起删掉

dict.clear() # 清空字典

del dict # 删除字典

### 字典的嵌套

字典的键的值又是字典。

cities={

'北京':{

'朝阳':['国贸','CBD','天阶','我爱我家','链接地产'],

'海淀':['圆明园','苏州街','中关村','北京大学'],

'昌平':['沙河','南口','小汤山',],

'怀柔':['桃花','梅花','大山'],

'密云':['密云A','密云B','密云C']

},

'河北':{

'石家庄':['石家庄A','石家庄B','石家庄C','石家庄D','石家庄E'],

'张家口':['张家口A','张家口B','张家口C'],

'承德':['承德A','承德B','承德C','承德D']

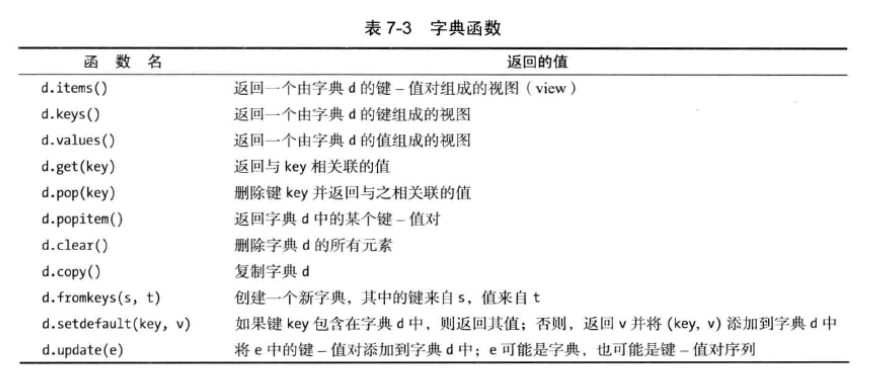
}

}

print(cities['北京']['海淀']

### 常用的字典操作方法

常用的字典方法：



d = {'Name': '飞入蓝天', 'Age': 7, 'Class': 'First'}

val = d.pop('Age')

print(val)

print(d)

## 集合set

集合是一系列不重复的元素。它和字典类似，但是只包含键，没有相应的关联值。

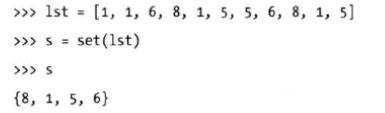
### 创建集合

（1）、利用{}

myset = {3, 4, 5, 1}

（2）、用set语句

转化已存在的类型：



（3）、创建空的集合

创建空集合比较特殊。在 Python 中，空花括号（{}）用于创建空字典。要创建一个没有任何元素的集合，使用 set() 函数（不要包含任何参数）。

s = set()

type(s)

（4）、集合的元素，它们可以是不同的类型（例如：数字、元组、字符串等）。但是，集合不能有可变元素（例如：列表、集合或字典）。

>>> s = {1.0, 'Python', (1, 2, 3)} # 混合类型的集合

>>>

>>> s = {1, 2, [3, 4]} # 不能有可变元素

...

TypeError:

unhashable type: 'list'

### 集合的特性

无序性：一个集合中，每个元素的地位都是相同的，**元素之间是无序的**。   
集合上可以定义序关系，定义了序关系后，元素之间就可以按照序关系排序。但就集合本身的特性而言，元素之间没有必然的序。

互异性：一个集合中，任何**两个元素都认为是不相同**的，即每个元素只能出现一次。   
有时需要对同一元素出现多次的情形进行刻画，可以使用多重集，其中的元素允许出现多次。

确定性：给定一个集合，任给一个元素，该元素或者属于或者不属于该集合，二者必居其一，不允许有模棱两可的情况出现。

# 无序性

>>> s = set('Python')

>>> s

{'y', 'n', 'h', 'o', 'P', 't'}

>>>

>>> s[0] # 不支持索引

...

TypeError: 'set' object does not support indexing

# 互异性

>>> s = set('Hello')

>>> s

{'e', 'H', 'l', 'o'}

### 集合的运算

>>> A = set('abcd')

>>> B = set('cdef')

>>>C=set('ab')

**（1）、子集**

>>>

>>> C < A

True

>>>

>>> C < B

False

>>>

>>> C.issubset(A)

True

**（2）、**并集

>>> A | B

{'e', 'f', 'd', 'c', 'b', 'a'}

>>>

>>> A.union(B)

{'e', 'f', 'd', 'c', 'b', 'a'}

**（3）、**交集

>>> A & B

{'d', 'c'}

>>>

>>> A.intersection(B)

{'d', 'c'}

**（4）、**差集

>>> A - B

{'b', 'a'}

>>>

>>> A.difference(B)

{'b', 'a'}

**（5）、对称差**

两个集合的对称差是只属于其中一个集合，而不属于另一个集合的元素组成的集合。

>>> A ^ B

{'b', 'e', 'f', 'a'}

>>>

>>> A.symmetric\_difference(B)

{'b', 'e', 'f', 'a'}

### 集合的更改

虽然集合不能有可变元素，但是集合本身是可变的。也就是说，可以添加或删除其中的元素。

可以使用 add() 方法添加单个元素，使用 update() 方法添加多个元素，update() 可以使用元组、列表、字符串或其他集合作为参数。

（1）、增加元素

>>> s = {'P', 'y'}

>>>

>>> s.add('t') # 添加一个元素

>>> s

{'P', 'y', 't'}

>>>

>>> s.update(['h', 'o', 'n']) # 添加多个元素

>>> s

{'y', 'o', 'n', 't', 'P', 'h'}

>>>

>>> s.update(['H', 'e'], {'l', 'l', 'o'}) # 添加列表和集合

>>> s

{'H', 'y', 'e', 'o', 'n', 't', 'l', 'P', 'h'}

（2）、删除元素

>>> s = {'P', 'y', 't', 'h', 'o', 'n'}

>>>

>>> s.discard('t') # 去掉一个存在的元素

>>> s

{'y', 'o', 'n', 'P', 'h'}

>>>

>>> s.remove('h') # 删除一个存在的元素

>>> s

{'y', 'o', 'n', 'P'}

>>>

>>> s.discard('w') # 去掉一个不存在的元素（正常）

>>> s

{'y', 'o', 'n', 'P'}

>>>

>>> s.remove('w') # 删除一个不存在的元素（引发错误）

...

KeyError: 'w'

类似地，可以使用 pop() 方法删除和返回一个项目。

还可以使用 clear() 删除集合中的所有元素。

>>> s = set('Python')

>>>

>>> s.pop() # 随机返回一个元素

'y'

>>>

>>> s.clear() # 清空集合

>>> s

set()

### 其他集合方法

可以用dir(set)查看。

| **方法** | **描述** |
| --- | --- |
| add() | 将元素添加到集合中 |
| clear() | 删除集合中的所有元素 |
| copy() | 返回集合的浅拷贝 |
| difference() | 将两个或多个集合的差集作为一个新集合返回 |
| difference\_update() | 从这个集合中删除另一个集合的所有元素 |
| discard() | 删除集合中的一个元素（如果元素不存在，则不执行任何操作） |
| intersection() | 将两个集合的交集作为一个新集合返回 |
| intersection\_update() | 用自己和另一个的交集来更新这个集合 |
| isdisjoint() | 如果两个集合有一个空交集，返回 True |
| issubset() | 如果另一个集合包含这个集合，返回 True |
| issuperset() | 如果这个集合包含另一个集合，返回 True |
| pop() | 删除并返回任意的集合元素（如果集合为空，会引发 KeyError） |
| remove() | 删除集合中的一个元素（如果元素不存在，会引发 KeyError） |
| symmetric\_difference() | 将两个集合的对称差作为一个新集合返回 |
| symmetric\_difference\_update() | 用自己和另一个的对称差来更新这个集合 |
| union() | 将集合的并集作为一个新集合返回 |
| update() | 用自己和另一个的并集来更新这个集合 |

### 集合与内置函数

下述内置函数通常作用于集合，来执行不同的任务。

| **函数** | **描述** |
| --- | --- |
| all() | 如果集合中的所有元素都是 True（或者集合为空），则返回 True。 |
| any() | 如果集合中的所有元素都是 True，则返回 True；如果集合为空，则返回 False。 |
| enumerate() | 返回一个枚举对象，其中包含了集合中所有元素的索引和值（配对）。 |
| len() | 返回集合的长度（元素个数） |
| max() | 返回集合中的最大项 |
| min() | 返回集合中的最小项 |
| sorted() | **从集合中的元素返回新的排序列表（不排序集合本身）** |
| sum() | 返回集合的所有元素之和 |

## 不可变集合frozenset

frozenset 是一个具有集合特征的新类，但是一旦分配，它里面的元素就不能更改。

>>> s = frozenset('Python')

>>> type(s)

<class 'frozenset'>

frozenset 也提供了一些列方法，和 set 中的类似。由于 frozenset 是不可变的，所以没有添加或删除元素的方法。