第5章 字典与集合

学习目标

- 理解字典元素结构。
- 掌握字典常用的方法。
- 理解集合元素无序、不重复的特点。
- 掌握集合常见运算及常用方法，能使用集合解决实际问题。

5.1 字典

5.1.1 字典的概念与特性

1. 字典的概念

字典(Dict)是 Python 中唯一的映射类型，是无序可变序列，里面可以包括若干个元素，每个元素是一个键值对，包含键(Key)和值(Value)两个部分，中间使用冒号分隔，表示一种映射关系或对应关系，所有的元素使用一对花括号“{ }”括起来。字典的格式如下。

```
d = {key1 : value1, key2 : value2, … }
```

2. 字典特性

字典的键不可出现两次。字典的键必须是唯一的，不可以重复。值是可以重复的，这一点与数据库原理里讲的实体间一对多的联系类似。

字典中元素的键必须是不可变的，可以是任意的不可变类型，如整数、实数、复数、字符串或元组等，不可以使用列表、集合、字典等可变类型作字典的键。

5.1.2 字典的创建与删除

1. 字典的创建

创建字典有 3 种方法：一种是使用赋值语句直接创建字典，一种是使用 dict()函数创建字典，最后一种是使用 fromkeys()方法创建字典。

(1) 使用赋值语句直接创建字典。

使用赋值运算符“=”将一个字典赋值给一个变量即可创建一个字典变量。例如：

```
scores = {'数学':85,'物理':78,'化学':92}
scores
```

结果为：

```
{'数学': 85, '物理': 78, '化学': 92}
```

也可以在赋值的时候创建一个不含任何元素的空字典。

(2) 使用 dict()函数创建字典。

使用 dict()函数可以将以键值对形式的列表或元组创建为字典。例如，下面的例子都是使用 dict()函数创建的字典对象。

```
scores = dict([['数学',85],['物理',78],['化学',92]])
scores = dict((('数学',85),('物理',78),('化学',92)))
scores = dict(数学 = 85,物理 = 78,化学 = 92)
```

第一个是将列表转换为字典，第二个是将元组转换为字典，第三个是通过指定关键字参数创建字典。

也可以通过使用 zip()函数将多个序列作为参数，返回由元组构成的列表，然后使用 dict()函数创建字典对象。例如：

```
keys = ['数学','物理','化学']
values = [85,78,92]
scores = zip(keys,values)
scores2 = dict(scores)
```

(3) 使用 fromkeys()方法创建字典。

fromkeys()方法创建字典的语法格式为：

```
dict.fromkeys(seq[,value])
```

其中，参数 seq 表示字典的键值序列，参数 value 是可选项，表示字典所有键对应的初始值，如果指定了，那么所有键都有这个相同的值，如果不指定，则所有键都取空值 None。

```
scores3 = {}
a = scores3.fromkeys(['英语','语文'],99)
print(a)
print(scores3)
```

结果为：

```
{'英语': 99, '语文': 99}
{}
```

不指定 value 参数则会对所有键取 None 值。例如：

```
scores3 = {}
a = scores3.fromkeys(['英语','语文'])
print(a)
print(scores3)
```

结果为：

```
{'英语':None, '语文': None}
{}
```

注意：上面的两个例子都说明使用 fromkeys()方法会创建新字典，但并不会改变调用此方法的字典内容。另外，创建字典时如果一个键在字典中出现了两次，那么后一个值会覆盖前面的值。

2. 字典删除

使用 del 命令可以删除字典。例如：

```
del scores3
```

注意：删除字典时，del 命令后面跟随的是字典名称，不能是字典元素。

5.1.3　字典元素访问

1. 通过字典键访问

字典元素的访问可以通过字典键进行。与列表元素访问类似，都通过括号“[]”的形式，只不过访问字典元素时，括号里面的内容是键，而列表访问时括号里面的是索引下标。在访问字典元素时，如果键不存在的话则会抛出异常，因此，常常结合选择结构或异常处理结构访问字典，以防代码崩溃。例如：

```
scores = {'数学': 85, '物理': 78, '化学': 92}
key = input('请输入要查看的课程: ')
if key in scores:
    print(scores[key])
else:
    print(f'课程{key}不存在')
```

2. 通过 get()方法访问

字典对象还提供了一个 get()方法用来返回指定键对应的值，但需要注意访问的键不存在时应设置默认值提示，否则返回空值。如下面的语句，如果存在课程就返回其值，不存在课程就给出“没有该课程”的提示。

```
scores.get('高等数学','没有该课程')
```

3. 通过 setdefault()方法访问

setdefault()方法也可以获取字典中元素的值或者增加新的元素。例如：

```
scores = {'数学': 85, '物理': 78, '化学': 92}
print(scores.setdefault('语文','80'))
print(scores)
```

结果为：

```
80
{'数学': 85, '物理': 78, '化学': 92, '语文': '80'}
```

4. 字典遍历

字典元素的访问，除了上述 3 种方法外，还支持 for 循环遍历访问字典元素。例如：

```
scores = {'数学': 85, '物理': 78, '化学': 92}
for item in scores:
    print(item,end = ' ')                #打印键,用空格分隔
print('\n')
print(scores.keys())                     #打印键
print(scores.values())                   #打印值
print(scores.items())                    #打印元素
```

结果为：

```
数学 物理 化学

dict_keys(['数学', '物理', '化学'])
dict_values([85, 78, 92])
dict_items([('数学', 85), ('物理', 78), ('化学', 92)])
```

5.1.4 字典元素的增加、修改与删除

1. 字典元素增加与修改

可以使用键为下标来为字典元素赋值。例如：

```
scores = {'数学': 85, '物理': 78, '化学': 92}
scores['生物'] = 77
print(scores)
```

结果为：

```
{'数学': 85, '物理': 78, '化学': 92, '生物': 77}
```

这里要注意一个问题，如果指定的键在字典中不存在，则表示要向字典中添加一个新的元素。但如果指定的键在字典中存在，表示修改指定的键对应的元素值。例如：

```
scores = {'数学': 85, '物理': 78, '化学': 92}
scores['数学'] = 77
print(scores)
```

结果为：

```
{'数学':77, '物理': 78, '化学': 92}
```

除了使用键下标的方式对字典元素增加和修改以外，还可以使用字典的 update()方法将另一个字典或可迭代对象（每个元素都包含两个值的元组或类似结构）中的元素一次性全部添加到调用 update()方法的字典对象中，如果两个字典中有相同的键，就会以另一个字典中的值为准对调用 update()方法的字典进行更新。例如：

```
score1 = {'数学': 85, '物理': 78, '化学': 92}
score2 = {'数学': 80, '英语': 78, '生物': 92}
score1.update(score2)
print(score1)
```

结果为：

```
{'数学': 80, '物理': 78, '化学': 92, '英语': 78, '生物': 92}
```

2. 字典元素的删除

前面已经提到过可以使用 del 命令来删除字典的元素，除此以外，还可以使用字典对象的 pop()方法删除指定的键对应的元素，同时返回对应的值，实际上就是删除元素并执行弹出值操作，也可以使用 popitem()方法按照弹栈的方式（后进先出）删除元素并返回一个包含两个元素的元组（此元组元素对应于字典元素的键和值）。例如：

```
score1 = {'数学': 80, '物理': 78, '化学': 92, '英语': 78, '生物': 92}
print(score1.pop('数学'))
length = len(score1)
for i in range(length):
    print(score1.popitem())
```

结果为：

```
80
('生物', 92)
('英语', 78)
('化学', 92)
('物理', 78)
```

5.1.5　字典内置函数与方法

表 5-1 和表 5-2 展示了字典对象的内置函数和常用方法。

表 5-1　字典内置函数及功能

字典内置函数	功 能 描 述
cmp(dict1,dict2)	比较两个字典元素
len(dict)	计算字典元素个数及键的数量
str(dict)	输出字典可打印的字符串表示
type(variable)	返回变量类型，如果变量是字典就返回字典类型

表 5-2　字典常用方法及功能

字典常用方法	功 能 描 述
clear()	没有参数，删除当前字典对象中所有元素，无返回值
copy()	没有参数，返回调用此方法的字典对象的浅复制
fromkeys(iterable,value=None,/)	以参数 iterable 中的元素为键，以参数 value 为值创建并返回字典对象
get(key,default=None)	返回当前字典对象中以参数 key 为键对应的元素的值，如果没有键为 key 的元素则返回 default 值
item()	无参数，返回调用此方法的字典对象的元素，以元组(key,value)的形式
keys()	无参数，返回调用此方法的字典对象的键
pop(k[,d])	删除以 k 为键的元素，返回对应的值，d 表示字典中没有对应键为 k 的元素时默认返回的值

续表

字典常用方法	功 能 描 述
popitem()	弹出字典最后一个元素，如果字典空则抛出异常
setdefault()	获取字典中元素的值或者增加新的元素
update([E,] ** F)	使用 E 和 F 中的数据对调用此方法的字典对象进行更新，** 表示参数 F 只能接受字典或关键参数
values()	无参数，返回包含调用此方法的字典对象中所有元素值

5.2 集合

5.2.1 集合的概念

集合(set)类型与数学中集合的概念是一致的，也具有无序性、互异性和确定性三个特征。集合是一个无序可变序列，所有元素放在一对花括号“{ }”中，元素之间使用逗号分隔，同一个集合中的元素都是唯一的，不允许重复。集合元素的类型只能是数字、字符串、元组等不可变类型，不可以用列表、字典或集合作为集合的元素。一般来说，使用集合多为了进行成员测试或者删除重复元素。

5.2.2 集合的创建与删除

1. 集合的创建

创建集合有三种方法：一种是使用赋值语句创建，一种是使用 set()函数创建，另一种是使用 frozenset()函数创建。

(1) 使用赋值语句创建。

使用赋值运算符将一个集合赋值给一个变量即可创建一个集合变量。例如：

```
courses = {'数学', '物理', '化学', '英语', '生物'}
print(type(courses))
print(courses)
```

结果为：

```
<class 'set'>
{'数学', '生物', '英语', '化学', '物理'}
```

(2) 使用 set()函数创建。

使用 set()函数可以将列表、元组、字符串、range 对象等可迭代对象转换成集合。例如：

```
set1 = set([1,2,2,3,4,5,6,6,6,6,7])
print(set1)
set2 = set(('a','b','c','d'))
print(set2)
set3 = set('aaabbc')
print(set3)
set4 = set(range(6))
print(set4)
```

结果为：

```
{1, 2, 3, 4, 5, 6, 7}
{'c', 'a', 'd', 'b'}
{'c', 'a', 'b'}
{0, 1, 2, 3, 4, 5}
```

注意：上面的例子表明，如果被转换的可迭代对象存在重复元素，在转换后的集合里只保留一个，集合的元素无序。另外，创建空集合使用 set()，而不是使用{}({}代表空字典)。

(3) 使用 frozenset()函数创建。

使用 frozenset()函数创建的集合是不可变的，不能进行元素的增加、删除，而使用 set()函数创建的集合是可变的。例如：

```
set5 = frozenset('python')
print(set5)
```

结果为：

```
frozenset({'y', 'h', 'n', 'p', 't', 'o'})
```

2. 删除集合

当集合不再使用时，可以使用 del 命令将集合删除。语法格式如下。

```
del 集合变量名
```

5.2.3　集合元素的添加与删除

1. 集合元素的添加

使用集合对象的 add()方法可以向集合中添加一个元素，如果该元素已经存在则忽略该操作，也可以使用 update()方法向集合中添加另外一个集合中的多个元素(自动去重)。例如：

```
set6 = {'数学', '生物', '英语', '化学', '物理'}
set6.add('语文')
set6.update({'政治','历史','生物'})
print(set6)
```

结果为：

```
{'数学', '语文', '历史', '生物', '英语', '化学', '政治', '物理'}
```

2. 集合元素的删除

集合对象提供 remove()、discard()、pop()、clear()等方法来删除集合中的元素。

(1) remove()方法。

使用 remove()方法删除集合元素时，如果元素不存在则会抛出异常。如下例，使用 remove()方法删除元素“物理”后再次删除则会报错。

```
set6 = {'数学', '语文', '历史', '生物', '英语', '化学', '政治', '物理'}
set6.remove('物理')
print(set6)
set6.remove('物理')
```

结果为：

```
{'数学', '语文', '历史', '生物', '英语', '化学', '政治'}
---------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-84-be4f70716e97> in <module>
      2 set6.remove('物理')
      3 print(set6)
----> 4 set6.remove('物理')
KeyError: '物理'
```

(2) 使用 discard()方法。

与使用 remove()方法不同，使用 discard()方法删除集合元素时，如果元素不存在则忽略此操作，不会抛出异常。例如：

```
set6 = {'数学', '语文', '历史', '生物', '英语', '化学', '政治', '物理'}
set6.discard('物理')
print(set6)
set6.discard('物理')
```

结果为：

```
{'数学', '语文', '历史', '生物', '英语', '化学', '政治'}
```

(3) 使用 pop()方法。

使用集合对象的 pop()方法会从集合中随机删除一个元素并返回该元素。

(4) 使用 clear()方法。

使用集合对象的 clear()方法会删除集合的所有元素。

5.2.4 集合常用方法

Python 内置集合类支持 len()、max()、min()、sum()、sorted()、map()、filter()、enumerate()、all()、any()等内置函数以及并集(|)、交集(&)、差集(-)、对称差集(^)、成员测试运算(in)。集合类自身也提供了大量的方法，如表 5-3 所示。

表 5-3 集合对象常用方法

方 法	功 能 描 述
add()	向调用此方法的集合里增加一个可哈希元素(不可哈希会抛出类型错误的异常)，如果集合中已经存在该元素则忽略操作
clear()	删除调用此方法的集合对象的所有元素
copy()	返回调用此方法的集合对象的浅复制
difference()	返回调用此方法的集合对象与参数集合的差集
discard()	接受一个可哈希对象作为参数，从调用此方法的集合中删除该元素，如果不存在则忽略此操作

续表

方 法	功 能 描 述
intersection()	返回调用此方法的集合对象与参数集合的交集
issubset()	测试调用此方法的集合是否为参数集合的子集，如果是则返回 True，否则返回 False
issuperset()	测试调用此方法的集合是否为参数集合的超集，如果是则返回 True，否则返回 False
pop()	删除并返回调用此方法结合对象中的任意一个元素，如果集合为空则抛出异常
remove()	从调用此方法的集合对象中删除一个元素，如果元素不存在则抛出异常（注意与 discard() 的区别）
union()	返回调用此方法的集合对象与参数集合的并集
update()	向调用此方法的集合中添加另外一个集合中的多个元素（自动去重）

5.3 综合例题

定义一个商品字典，里面存放了各个商品的编号和商品名称，定义三个购买记录集合，里面存放购买商品的编号，试着找出所有人都购买的商品、无人购买的商品以及有人购买的商品并输出。

```
商品 = {'01':'鸡蛋','02':'牛奶','03':'面包','04':'酸奶','05':'啤酒','06':'火腿','07':'牙膏','08':
'洗发水'}
账单 1 = {'01','02','05','06','07'}
账单 2 = {'01','05','07'}
账单 3 = {'01','03','06'}
账单 4 = set()
for item in 商品:
    账单 4.add(item)
allbuy = 账单 1& 账单 2& 账单 3
somebuy = 账单 1|账单 2|账单 3
nobodybuy = 账单 4-somebuy
print('所有人都购买的商品：',end = ' ')
for item in allbuy:
    print(商品.get(item),end = ' ')
print()
print('无人购买的商品：',end = ' ')
for item in nobodybuy:
    print(商品.get(item),end = ' ')
print()
print('有人购买的商品：',end = ' ')
for item in somebuy:
    print(商品.get(item),end = ' ')
```

结果为：

```
所有人都购买的商品：鸡蛋
无人购买的商品：酸奶 洗发水
有人购买的商品：牛奶 火腿 面包 啤酒 鸡蛋 牙膏
```

习题

一、选择题

1. 下列不能创建一个字典的语句是(　　)。

A. dict1＝{}　　B. dict2＝{3:5}

C. dict3＝{[1,2,3]: 'sea'}　　D. dict4＝{(1,2,3): 'sea'}

2. 一个通讯录,需要通过人名查找电话号码,使用(　　)数据类型更合理。

A. 集合　　B. 字典　　C. 元组　　D. 列表

3. 下列代码中,可以用于创建一个空集合的是(　　)。

A. set1＝{}　　B. set1＝({})　　C. set1＝set()　　D. set1＝{()}

4. 设字典 dict1 ＝ {1:'a',2:'b',3:'c',4:'d'},执行下列哪个语句不会发生错误?(　　)

A. dict1[4]　　B. dict1.get(4)　　C. del dict1[5]　　D. dict1[2]

二、操作题

将字典{'数学': 85, '物理': 78, '化学': 92, '生物': 77}的键与值互换并输出。