10.集合数据类型讲解

集合：数学上，把set称做由不同的元素组成的集合，集合（set）的成员通常称做集合元素。python把这个概念引入到它的集合类型对象里。集合对象是一组无序排列的可哈希的值（也就是说集合中的元素都是不可变类型，因此像列表和字典等可变类型对象是不能作为集合的元素的）。集合的成员可以做字典中的键。

集合有两种不同的类型，可变集合（set）和不可变集合（frozenset）。可变集合可以进行增加和删除元素，不可变集合就不可以。可变集合是不可哈希的，因此不能作为字典的键，不可变集合正好相反。

一、集合的创建以及赋值

空集合的创建只能通过集合的实例set和frozenset创建。

```
# 创建空的可变集合以及不可变集合
s = set()
s1 = frozenset()  # 创建一个空的不可变集合基本上是没什么意义的
# 给集合初始化
s2 = set('helloworld')
s3 = frozenset('bookshop')
# 注意：使用set和frozenset方法创建集合的时候，里面的参数必需是可迭代类型，同时类型里面的成员必需是可哈希的，否则会报错
s4 = {1,2,3,4,5}
```

二、访问集合中的值

因为集合不是序列类型，也就是跟字典一样都是无序的，因此不能通过下标的方式进行访问里面的元素，所以只能通过成员操作符in和not in来判断某个元素是否在集合里面，或者通过for循环把集合中的元素遍历出来。

```
>>> s = set([1,2,3,4,5])
>>> 1 in s
True
>>> 'a' in s
False
>>> 9 not in s
True
>>> for i in s:
...     print i
```

```
...
1
2
3
4
5
```

三、对可变集合进行增删操作

用各种集合内建的方法和操作符添加和删除集合的成员

集合操作符和关系符号

python符号	说明
in	是……的成员
not in	不是……的成员
==	等于
!=	不等于
<	是……的真子集
<=	是……的子集
>	是……的真超集
>=	是……的超集
&	交集
|	合集（并集）
- (减号)	差补或相对补集也叫差集
^	对称差分

```
# 使用add方法给集合添加新的元素
s = set('hello world')
s.add('haha')  # 将haha添加到集合s里面
# 使用update方法添加多个元素进去,update方法接收的是一个可迭代类型参数，要求跟前面的一样
s.update('python')  # 迭代python，将里面的元素一个个的添加到集合s中
# 使用remove方法删除集合中的元素
s.remove('a')  # 将字符串a从集合中删除
#  使用-=符号一下子删除集合中的多个元素，其实就是求两个集合的差集
```

```
s -= set("hello")  # 把hello这几个字符串从集合s中删除
# 使用clear方法清空集合中的所有成员
s.clear()
# 注意，对于不可变集合是没有增加和删除元素的操作的，只能访问其中的元素
# 删除整个集合用del关键字就可以了
del s
```

型操作符的使用

1. 并集（|）

   并集（union）操作和集合的or其实是等价的，两个集合的并集是一个新集合，该集合中的每个元素至少都是一种一个集合的成员，即，属于两个集合其中之一的成员。并集符号有一个等价的方法: union()。

   ```
   >>> s = set('hello')
   >>> s1 = set('python')
   >>> s | s1
   ```

2. 交集（&）

   你可以把交集操作符必做成集合的and操作，两个集合的交集是一个新的集合，该集合中的每个成员同时属于两个集合中的成员。交集符号有一个等价的方法：intersection().

   ```
   >>> s = set('hello')
   >>> s1 = set('python')
   >>> s & s1
   ```

3. 差集（-）

   两个集合的差集是一个新的集合，该集合中的成员只属于前者，不属于后者，就是谁在符号的左边就属于谁。差集符号又一个等价的方法: difference()。

   ```
   >>> s = set('hello')
   >>> s1 = set('python')
   >>> s - s1
   ```

4. 对称差分（^）

   两个集合（如：s和s1）的对称差分是一个新的集合，还集合中的元素是能属于集合s或者集合t的成员，不能同时属于两个集合。对称差分符号有一个等价的方法: symmetric_difference()。

   ```
   >>> s = set('hello')
   >>> s1 = set('python')
   >>> s ^ s1
   ```

5. 使用|=操作符给集合添加一个或多个成员，与update方法等价

```
>>> s = set('hello')
>>> s1 = set('python')
>>> s |= s1
```

6. 使用-=操作符删除集合中一个或多个成员

```
>>> s = set('hello')
>>> s1 = set('python')
>>> s -= s1
```

7. 使用^=更新集合中的元素，使得其中的元素只属于原集合s或仅是另一个集合s1中的成员。此操作符与方法：symmetric_difference_update()等价。

```
>>> s = set('hello')
>>> s1 = set('python')
>>> s ^= s1
```

8. 使用&=求两个集合的交集

```
>>> s = set('hello')
>>> s1 = set('python')
>>> s &= s1
```

四、集合类型的内建方法

1. 适用于所有集合的方法

方法名称	说明
s.issubset(t)	如果s是t的子集，则返回True，否则返回False等价于小于等于号（<=）
s.issuperset(t)	如果s是t的超集，则返回True，否则返回False,等价于大于等于号(>=)
s.union(t)	返回一个新的集合，该集合是s和t的并集
s.intersection(t)	返回一个新的集合，该集合是s和t的交集
s.difference(t)	返回一个新的集合，该集合是s和t的差集
s.symmetric_diffence(t)	返回一个新的集合，该集合是s或t的成员，但不是s和t共有的成员
s.copy()	返回一个新的集合，该集合是s的浅拷贝

2. 仅适用于可变集合的方法

方法名	说明
s.update(t)	用t中的元素修改s，其实就是求两个集合的并集
s.intersection_update(t)	求集合s和t的交集
s.diffence_update(t)	s中的成员属于s但不包含在t中的元素（求差集）
s.symmetric_diffence_update(t)	求两个集合的对称差补，就是s中的成员属于s或t，但不同时属于s或t
s.add(obj)	在集合s中添加对象obj
s.remove(obj)	从集合s中删除对象obj，如果obj不存在则抛出KeyError异常
s.discard(obj)	如果obj是集合s中的元素，则从集合s中删除对象obj
s.pop()	删除集合s中的任意一个元素，并返回它
s.clear()	删除集合s中所有的元素

3. 操作符与内建方法的比较

很多的内建方法几乎和操作符等价。我们说“几乎等价”，意思是他们之间有一个重要的区别：当用操作符时，操作符两边的操作数必须是集合。在使用内建方法时，对象也可以是迭代类型的。为什么有了操作符还要内建函数呢？主要就是内建函数可以避免操作符使用不当而报错。

作业

1. 如果有一百万的关键词，如何快速的进行去重？写出你的关键词去重代码
2. 将第一题中的关键词进行分类，包含某些关键词的就归为一类，该如何实现？

   例如游戏行业词： 如果关键词里面，包含技能，法术等词那么就归属技能这个分类

   结巴分词安装：

   ```
   pip install jieba -i https://pypi.doubanio.com/simple
   ```