

# python

## data process

张海宁

贵州大学

*hnzhang1@gzu.edu.cn*

March 26, 2019

dictionary、 set  
○○○○○○

read and write file  
○○○○

Regular expression operations  
○○○

homework  
○

Q&A  
○

# Overview

dictionary、 set

read and write file

Regular expression operations

homework

Q&A

# 映射

两个非空集合  $A$  与  $B$  间存在着对应关系  $f$ ，而且对于  $A$  中的每一个元素  $x$ ， $B$  中总有有唯一的一个元素  $y$  与它对应，就这种对应为从  $A$  到  $B$  的映射，记作  $f: A \rightarrow B$ 。其中， $b$  称为元素  $a$  在映射  $f$  下的象，记作： $b = f(a)$ 。 $a$  称为  $b$  关于映射  $f$  的原象。集合  $A$  中所有元素的象的集合称为映射  $f$  的值域，记作  $f(A)$ 。

# 字典

*Python* 中用来处理映射关系的数据结构叫做字典(dictionary)。

- 在字典中，涉及两个集合，一个是键(key)的集合，一个是值(value)的集合。
- 在字典中，元素是以“键：值”对的方式存储的。

## 定义一个包含 3 个元素的字典

1

```
pac = {'0851': '贵阳', '023': '重庆', '028': '成都'}
```

## 注意事项

- 键必须是不可变对象
- 值可以是任何类型的数据
- 键是唯一的，值不必唯一
- 字典中的元素是无序的

Table: 字典常用操作

操作	描述	操作	描述
<code>len(pac)</code> <code>x in pac</code>	求元素个数 <code>x</code> 是否是 <code>pac</code> 的一个键	<code>pac[key]</code> <code>x:y not in pac</code>	返回 <code>key</code> 对应的值 <code>x:y</code> 这个元素是否不在 <code>pac</code> 中
<code>pac[k1]=v1</code>	if <code>k1 in pac</code> : 更新 <code>k1</code> 对应的值；否则将 <code>k1:v1</code> 这个元素添加到 <code>pac</code> 中	<code>del pac[k1]</code>	删除键为 <code>k1</code> 的元素
<code>pac[k1,dft]</code>	if <code>k1 in pac</code> : return <code>pac[k1]</code> else: return <code>dft</code>	<code>d.clear()</code>	移除字典中所有的元素
<code>list(pac.keys())</code>	返回 <code>pac</code> 中的所有键组成的列表	<code>list(pac.values())</code>	返回 <code>pac</code> 中的所有值组成的列表
<code>list(pac.items())</code>	返回 <code>(key,value)</code> 形式的二元组组成的列表	<code>pac.update(c)</code>	将字典 <code>c</code> 中所有元素合并入 <code>pac</code> , 若拥有相同的键, 则使用 <code>c</code> 中的值替换 <code>pac</code> 中的值

# 集合

集合也是 *python* 中用来存储一系列元素的一种数据结构，其有以下特点：

- 其中的元素是无序的
- 不允许重复元素
- 可以包含数值、字符串、元组和布尔变量，不可以容纳列表或集合

定义一个包含 6 个元素的集合

1

```
ns = {1,2,3,' 贵阳',' 重庆',True}
```

Table: 集合常用操作

操作	描述
<code>len(ns)</code>	求集合 <code>ns</code> 中的元素个数
<code>ns.add(e)</code>	向集合 <code>ns</code> 中添加元素 <code>e</code>
<code>ns.clear()</code>	移除集合 <code>ns</code> 中的所有元素
<code>e in ns</code>	元素 <code>e</code> 是否在集合 <code>ns</code> 中
<code>set1.union(set2)</code>	$set1 \cup set2 = \{x   x \in set1, \text{ or } x \in set2\}$
<code>set1.intersection(set2)</code>	$set1 \cap set2 = \{x   x \in set1, \text{ and } x \in set2\}$
<code>set1.difference(set2)</code>	$set1 - set2 = \{x   x \in set1, \text{ and } x \notin set2\}$

## 遍历字典和集合中的元素

```
1 pac={'0851':' 贵阳','023':' 重庆','028':' 成都'}
2 for i in pac.keys():
3     print(i,"::",pac[i])
4
5 ns={1,2,3,'r'}
6 for i in ns:
7     print(i)
8
9 1
10 2
11 3
12 r
13 0851 :: 贵阳
14 023 :: 重庆
15 028 :: 成都
```



# 文件

截至目前，程序的运行结果显示在屏幕上，并最终随着关闭程序或系统，结果会消失。在实际应用中，通常希望能够将程序运行结果长久地保存在磁盘的文件中，以供以后使用。

## open 函数

*open()* 函数可以用来打开或创建一个文件。

```
1 open(file, mode='r', encoding=None)
2 =====
3 Character Meaning
4 -----
5 'r'    open for reading (default)
6 'w'    open for writing, truncating the file first
7 'x'    create a new file and open it for writing
8 'a'    open for writing, appending to the end of the file
9         if it exists
10 'b'    binary mode
11 't'    text mode (default)
12 =====
```

# 读取文件

```
1 f = open('./toolbox.py')
2 line_number = 1
3 for line in f:
4     print(line_number, ': ', line)
5     line_number += 1
6 f.close()
```

line1 使用 open() 函数打开文本文件

line3 使用 for 循环遍历文件的每一行

line6 使用 close() 函数关闭文件

## 写文件

```
1 out = open('./open4Writing.py','w')
2 out.write("test line 1\n")
3 out.close()
4 out1 = open('./open4Writing.py','w')
5 out1.write("test line 2\n")
6 out1.close()
7
8 out2 = open('appendFile.txt','a')
9 out2.write('first\n')
10 out2.close()
11 out3 = open('appendFile.txt','a')
12 out3.write('second\n')
13 out3.close()
```

line1 使用 `open()` 函数打开文本文件  
注意使用 `'w'` 模式  
即 open a file for writing

line234 使用 `write()` 函数写入字符串  
注意 `\n` 的使用，是为了换行

line5 使用 `close()` 函数关闭文件

在电脑上进行写文件的测试：  
比较 `'w'` 和 `'a'` 两种模式有何异同。

# 正则表达式

## 概念

正则表达式是一个特殊的字符序列，是一个模式字符串，可以检测一个字符串是否与这个特殊的字符序列的模式相匹配。

## *re*

*Python* 中，*re* 模块提供了正则表达式的功能。

# 模式字符串<sup>1</sup>

h[ea]llo	匹配 hello 或 hallo	\W	匹配任一非单词字符
[a-z]	匹配任一小写字母	\s	匹配任一空白字符
[A-Z]	匹配任一大写字母	\S	匹配任一非空白字符
[0-9]	匹配任一数字	\d	匹配任一数字
[a-zA-Z0-9]	匹配任一数字或字母	\w	匹配任一数字、字母、下划线
[^abc]	匹配 abc 以外的字符	.	匹配\n 以外的字符
[^0-9]	匹配数字以外的字符	\D	匹配数字以外的字符

---

<sup>1</sup>正则表达式

dictionary、set  
○○○○○○

read and write file  
○○○○

Regular expression operations  
○○●

homework  
○

Q&A  
○

# 匹配

## Homework

1. 统计一个给定文档 *aboutUN.txt* 的词频，并将结果写入到 *wordsFrequency.txt* 中。<sup>2</sup>样式如下所示：

```
1 UN      5
2 China    6
3 ...
```

2. 通过 *UN.txt* 文件，获取目前联合国的 193 个会员国名字，并将其写入 *UNmembers.txt* 中。样式如下所示：

```
1 China
2 America
3 ...
```

---

<sup>2</sup>不统计标点符号，不考虑有单词换行的情况。



dictionary、set  
oooooo

read and write file  
oooo

Regular expression operations  
ooo

homework  
o

Q&A  
●

# Q&A