python

data process

张海宁

贵州大学

hnzhang1@gzu.edu.cn

March 27, 2019

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
len(pac)	求元素个数	pac[key]	返回 key 对应的值
x in pac	x 是否是 pac 的一个键	x:y not in pac	x:y 这个元素是否不在 pac 中
pac[k1]=v1	if k1 in pac: 更新 k1 对应的值；否则将 k1:v1 这个元素添加到 pac 中	del pac[k1]	删除键为 k1 的元素
pac[k1,dft]	if k1 in pac: return pac[k1] else: return dft	d.clear()	移除字典中所有的元素
list(pac.keys())	返回 pac 中的所有键组成的列表	list(pac.values())	返回 pac 中的所有值组成的列表
list(pac.items())	返回 (key,value) 形式的二元组组成的列表	pac.update(c)	将字典 c 中所有元素合并入 pac，若拥有相同的键，则使用 c 中的值替换 pac 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
len(ns)	求集合 ns 中的元素个数
ns.add(e)	向集合 ns 中添加元素 e
ns.clear()	移除集合 ns 中的所有元素
e in ns	元素 e 是否在集合 ns 中
set1.union(set2)	$set1 \cup set2 = \{x \mid x \in set1, or\ x \in set2\}$
set1.intersection(set2)	$set1 \cap set2 = \{x \mid x \in set1, and\ x \in set2\}$
set1.difference(set2)	$set1 - set2 = \{x \mid x \in set1, and\ x \notin set2\}$

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  for k,v in pac.items():
5      print(k,':',v)
6
7  ns={1,2,3,'r'}
8  for i in ns:
9      print(i)
10
11 1
12 2
13 3
14 r
15 0851 :: 贵阳
16 023 :: 重庆
17 028 :: 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。

```
open(file, mode='r', encoding=None)
===== ========================================
Character Meaning
----- ----------------------------------------
'r'   open for reading (default)
'w'   open for writing, truncating the file first
'x'   create a new file and open it for writing
'a'   open for writing, appending to the end of the file
       if it exists
'b'   binary mode
't'   text mode (default)
===== ========================================
```

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  out.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[1]

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-z0-9A-Z]	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

[1]正则表达式

模式字符串[2]

符号	意义	符号	意义
^	匹配字符串的开头	$	匹配字符串的结尾
*	匹配 0 个或多个表达式	+	匹配 1 个或多个表达式
?	使用非贪婪模式，即匹配到合适的就结束	()	匹配括号内的表达式，也表示一个组
{m}	匹配 m 个表达式	{m,n}	匹配 m 到 n 个表达式

[2]正则表达式

匹配

在实际应用中，首先要构造一个模式字符串(*re.compile*(*patternString*))，然后使用这个模式去匹配给定字符串中的内容。
匹配的函数有：*re.match*(), *re.search*(), *re.findall*():

```
1 match(pattern, string, flags=0)
2     Try to apply the pattern at the start of the string,
3     returning a Match object, or None if no match was found.
4 search(pattern, string, flags=0)
5     Scan through string looking for a match to the pattern,
6     returning a Match object, or None if no match was found.
7 findall(pattern, string, flags=0)
8     Return a list of all non-overlapping matches in the string.
```

MatchObjects[3]

MatchObjects 支持以下方法：

Match.group() 默认返回整个匹配的字符串

Match.span() 默认返回一个二元组，这个二元组的第一个和第二个元素分别是匹配字符串在原字符串中的索引起始值

[3]相对地，有一个 Regular Expression Objects, 支持 Pattern.match(), Pattern.search(), Pattern.findall() 等方法。

匹配示例

```
1  import re
2  obj = 'ython, We are learn python language.' \
3        ' Python is very useful.'
4  pattern = re.compile('[pP]ython')
5  match1 = re.match(pattern,obj)
6  print('re.match():',match1)
7  match2 = re.search(pattern,obj)
8  print('re.search():',match2)
9  print('Match.group():',match2.group())
10 print('Match.span():',match2.span())
11 match3 = re.findall(pattern,obj)
12 print('re.findall():',match3)
13
14 re.match(): None
15 re.search(): <re.Match object; span=(20, 26), match='python'>
16 Match.group(): python
17 Match.span(): (20, 26)
18 re.findall(): ['python', 'Python']
```

Homework

1. 统计一个给定文档 *aboutUN.txt* 的词频，并将结果写入到 *wordsFrequence.txt* 中。[4]样式如下所示：

```
UN     5
China     6
...
```

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：

```
China
America
...
```

[4]不统计标点符号，不考虑有单词换行的情况。

Q&A