Regex正则表达式

覃雄派

Regex正则表达式

提纲

- Python字符串常用函数
- 为何需要正则表达式
- 正则表达式的核心内容
- 正则表达式练习

Regex正则表达式

- Python字符串常用函数
- 首先介绍python字符串的几个重要处理函数
  - 在一般情况下可以满足大部分常用需求
  - Replace
  - Split
  - Strip
  - upper/lower

Regex正则表达式

- Python常用字符串函数replace

描述

Python replace() 方法把字符串中的 old（旧字符串） 替换成 new(新字符串)，如果指定第三个参数max，则替换不超过 max 次。

语法

replace()方法语法：

```
str.replace(old, new[, max])
```

参数

- old -- 将被替换的子字符串。
- new -- 新字符串，用于替换old子字符串。
- max -- 可选字符串, 替换不超过 max 次

返回值

返回字符串中的 old（旧字符串） 替换成 new(新字符串)后生成的新字符串，如果指定第三个参数max，则替换不超过 max 次。

```
string = "this is string example ... wow!!!this is really string;"
newstring = string.replace("is", "was")#全部替换
print("newstring: ", newstring)

newstring = string.replace("is", "was",3)#替换不超过3次
print("newstring: ", newstring)

newstring = newstring.replace(".", " ").\
    replace("!", " ").\
    replace(";", " ")#替换
print("newstring: ", newstring)
```

```
newstring:  thwas was string example ... wow!!!thwas was really string;
newstring:  thwas was string example ... wow!!!thwas is really string;
newstring:  thwas was string example     wow   thwas is really string
```

Regex正则表达式

- Python常用字符串函数split

描述

Python **split()** 通过指定分隔符对字符串进行切片，如果参数 num 有指定值，则分隔 num+1 个子字符串

语法

split() 方法语法：

```
str.split(str="", num=string.count(str)).
```

参数

- str -- 分隔符，默认为所有的空字符，包括空格、换行(\n)、制表符(\t)等。
- num -- 分割次数。默认为 -1, 即分隔所有。

返回值

返回分割后的字符串列表。

```
string = "this is string example ... wow!!!this is really string;"
print(string.split(" "))#按照空格分割
print(string.split(" ",3))#按照空格分割3次
```

```
['this', 'is', 'string', 'example', '...', 'wow!!!this', 'is', 'really', 'string;']
['this', 'is', 'string', 'example ... wow!!!this is really string;']
```

Regex正则表达式

- Python常用字符串函数strip

描述

Python strip() 方法用于移除字符串头尾指定的字符（默认为空格或换行符）或字符序列。

注意：该方法只能删除开头或是结尾的字符，不能删除中间部分的字符。

语法

strip()方法语法：

```
str.strip([chars]);
```

参数

- chars -- 移除字符串头尾指定的字符序列。只要头尾含有指定字符串中的任意一个字符就删除。

返回值

返回移除字符串头尾指定的字符生成的新字符串。

```
string = "010ruc data science 101"
print(string.strip("10"))#删除首尾1,0,以及空格

string = "      ruc data science      "
print(string.strip(" "))#删掉首尾的空格
```

```
ruc data science
ruc data science
```

Regex正则表达式

- Python常用字符串函数upper,lower

描述

Python upper() 方法将字符串中的小写字母转为大写字母。

Python lower() 方法将字符串中的大写字母转为小写字母。

语法

upper() / lower()方法语法：

```
str.upper()
str.lower()
```

返回值

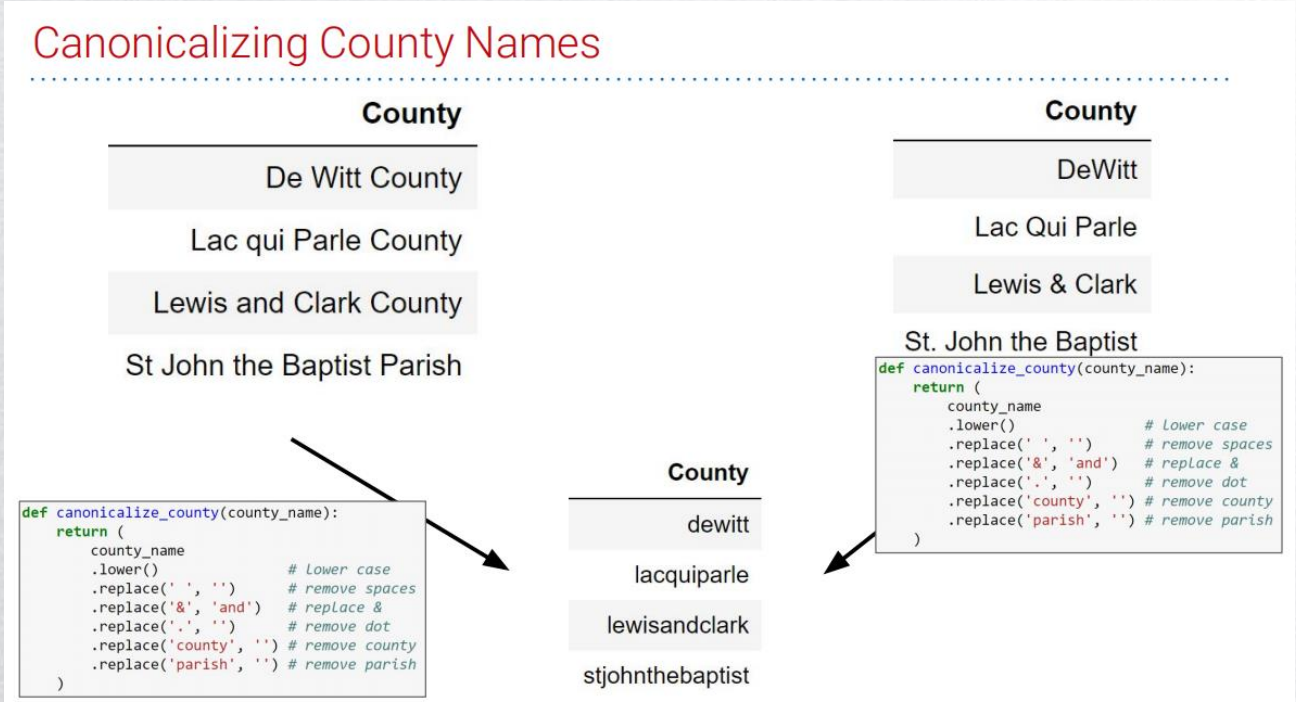
返回大小写字母转换后的字符串。

```
string = "Renmin University of China"
print(string.upper())
print(string.lower())
```

```
RENMIN UNIVERSITY OF CHINA
renmin university of china
```

Regex正则表达式

- Python字符串处理实例



Regex正则表达式

Regex正则表达式

- 为何需要正则表达式
  - 本节通过一个例子说明
  - 为何需要正则表达式来参与字符串操作

Regex正则表达式

- 为何需要正则表达式
- 要求从这个日志数据中提取出年、月、日、时、分、秒，该如何操作

```
169.237.46.168 - - [26/Jan/2014:10:47:58
-0800] "GET /stat141/Winter04/ HTTP/1.1" 200
2585 "http://anson.ucdavis.edu/courses/"
```

```
line = "169.237.46.168 - - [26/Jan/2014:10:47:58 -0800] \"GET /stat141/Winter04/ HTTP/1.1
        \" 200 2585 \"http://anson.ucdavis.edu/courses/\" "

day,month, rest = line.split('[')[1].split(']')[0].split('/')
print("day: ",day)
print("month: ",month)
print("rest: ",rest)
year,hour,minute,seconds = rest.split(' ')[0].split(':')
print("year: ",year)
print("hour: ",hour)
print("minute: ",minute)
print("seconds: ",seconds)
time_zone = rest.split(' ')[1]
print("time_zone: ",time_zone)
```

```
day:  26
month:  Jan
rest:  2014:10:47:58 -0800
year:  2014
hour:  10
minute:  47
seconds:  58
time_zone:  -0800
```

用split不断切割
很麻烦

Regex正则表达式

- 为什么需要正则表达式
  - 除了刚才的例子，
  - 如果验证一个字符串是否是电子邮箱呢？依然自己写规则判断吗？

34sdg4gfwr4@ruc.edu.cn
653564309@qq.com
hehehe@gmail.com
chigua@sina.cn
34gdggfdgrtyhyr@163.com
... ...

Regex正则表达式

- 为什么需要正则表达式

```
169.237.46.168 - - [26/Jan/2014:10:47:58
-0800] "GET /stat141/Winter04/ HTTP/1.1" 200
2585 "http://anson.ucdavis.edu/courses/"
```

用正则表达式匹配
很方便
需要了解正则表达式的写法

```
import re

line = "169.237.46.168 - - [26/Jan/2014:10:47:58 -0800] \"GET
     /stat141/Winter04/ HTTP/1.1\" 200 2585 \"http://anson.ucdavis.edu/courses/\""

pattern = r'\[(\d+)/(\w+)/(\d+):(\d+):(\d+):(\d+) (.+)\]'

match = re.search(pattern, line)
print (match.group())

day,month,year,hour,minute,seconds,time_zone = re.search( pattern, line).groups()
print("year: ",year)
print("month: ",month)
print("day: ",day)
print("hour: ",hour)
print("minute: ",minute)
print("seconds: ",seconds)
print("time_zone: ",time_zone)
```

```
[26/Jan/2014:10:47:58 -0800]
year:  2014
month:  Jan
day:  26
hour:  10
minute:  47
seconds:  58
time_zone:  -0800
```

Regex正则表达式

- 利用在线测试构造Regex

https://www.regextester.com/

Regex正则表达式

- 思考题
  - 判断是否是合法邮箱（格式正确即可，不管是否真的存在）
- 邮箱的规则为：
  - 输入的只能是字母、数字、@以及.等
  - @和.前后只能是字母或者数字
  - 最后一个.后只能是com或者cn

这里先思考
自己构造试试
答案在PPT末尾

标识符：由字母数字构成
邮箱的格式为
标识符
跟着@
跟着多个标识符.
最后一个是com或者cn

Regex正则表达式

Regex正则表达式

- 正则表达式的写法
  - 正则表达式主要用于查找、筛选、替换、检查格式等复杂操作
  - 本节通过简单的教程来说明正则表达式的用法，让大家尽快掌握正则表达式

Regex正则表达式

- 正则表达式的写法
  - 首先介绍pattern
- pattern即为正则表达式，可以是一个普通的字符串，也可以是re.compile(string)生成的对象
  - 当我们在Python中使用正则表达式时，re模块内部会干两件事情
  - 1.编译正则表达式，如果正则表达式的字符串本身不合法，会报错
  - 2.用编译后的正则表达式去匹配字符串
    - 所以如果调用次数较多时，建议提前编译好

Regex正则表达式

- 正则表达式的写法

模式	描述
^	匹配字符串的开头。比如："^hhh"可以匹配"hhhhhehehe", 但不可以匹配"ehehehhh"。
$	匹配字符串的末尾。用法同上，不再赘述
. (这里有个dot)	匹配任意字符，除了换行符。如果要加换行符，应该用[.\n]
[...]	用来表示一组字符,单独列出：[amk] 匹配 'a'，'m'或'k'
[^...]	不在[]中的字符：[^abc] 匹配除了a,b,c之外的字符。
re*	匹配0个或多个的表达式。例如："a*"表示匹配0或多次a
re+	匹配1个或多个的表达式。例如："a+"表示匹配1或多次a
re?	匹配0个或1个由前面的正则表达式定义的片段，非贪婪方式。非贪婪方式的意思是匹配范围尽可能小，例如字符串为"<hhh>hehehe>"，正则表达式为"<.*>"的话，则匹配到"<hhh>hehehe>"整个，如果为"<.*?>"的话，则匹配到<hhh>即停止。

Match as few as possible

Regex正则表达式

- 正则表达式的写法

模式	描述
re{ n}	精确匹配 n 个前面表达式。例如， `o{2}` 不能匹配 "Bob" 中的 "o"，但是能匹配 "food" 中的两个 o。
re{ n,}	匹配 n 个前面表达式。例如， o{2,} 不能匹配"Bob"中的"o"，但能匹配 "foooood"中的所有 o。"o{1,}" 等价于 "o+"。"o{0,}" 则等价于 "o*"。
re{ n, m}	匹配 n 到 m 次由前面的正则表达式定义的片段，贪婪方式
a| b	匹配a或b
(re)	对正则表达式分组并记住匹配的文本。分组之后看作一个整体，比如"abc+"可以匹配"abccccccc", 而"(abc)+"可以匹配"abcabcabcabc"
转义字符\	涉及到匹配规则用到的特殊字符时，前面加\表示其本身。例如要找到一个字符串"mit(eth)cmu"括号内的内容, 需要用"\(.*?\)"

Regex正则表达式

- 正则表达式的写法

特殊字符类实例	描述
\d	匹配一个数字字符。等价于 [0-9]。
\D	匹配一个非数字字符。等价于 [^0-9]。
\s	匹配任何空白字符，包括空格、制表符、换页符等等。等价于 [\f\n\r\t\v]。
\S	匹配任何非空白字符。等价于 [^ \f\n\r\t\v]。
\w	匹配包括下划线的任何单词字符。等价于'[A-Za-z0-9_]'。
\W	匹配任何非单词字符。等价于 '[^A-Za-z0-9_]'。

Regex正则表达式

- 转义字符

Escaped Characters

Class	Explanation
\. * \\	escaped special characters
\t \n \r	tab, linefeed, carriage return

Regex正则表达式

- 一些简单的实例

一些常用的简单实例	描述
[Pp]ython	匹配 "Python" 或 "python"
rub[ye]	匹配 "ruby" 或 "rube"
[aeiou]	匹配中括号内的任意一个字母
[0-9]	匹配任何数字。类似于 [0123456789]
[a-z]	匹配任何小写字母
[A-Z]	匹配任何大写字母
[a-zA-Z0-9]	匹配任何字母及数字
[^aeiou]	除了aeiou字母以外的所有字符
[^0-9]	匹配除了数字外的字符

Regex正则表达式

- 一些简单实例

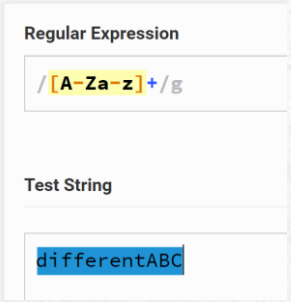
example	matches	does not match
AABAAB	AABAAB	every other string
AA|BAAB	AA BAAB	every other string
AB*A	AA ABBBBBBA	AB ABABA
A(A|B)AAB	AAAAB ABAAB	every other string
(AB)*A	A ABABABABA	AA ABBA

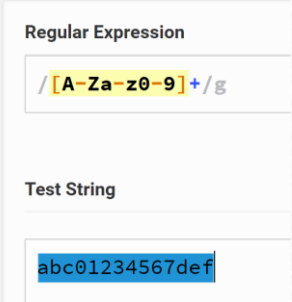
匹配

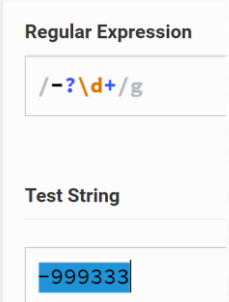
Regex正则表达式

- 经典实例

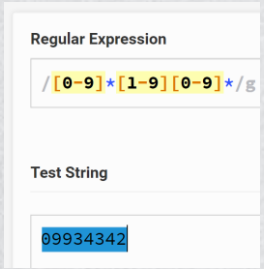
[A-Za-z]+ 由26个字母组成的字符串
[A-Za-z0-9]+ 由26个字母和数字组成的字符串
-?\d+ 整数形式的字符串
[0-9]*[1-9][0-9]* 正整数形式的字符串
[1-9]\d{5} 中国境内邮政编码,6位
[\u4e00-\u9fa5] 匹配中文字符
\d{3}-\d{8}|\d{4}-\d{7}电话010-62513264或者0779-8875269

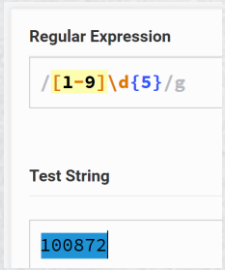


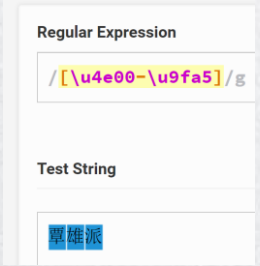


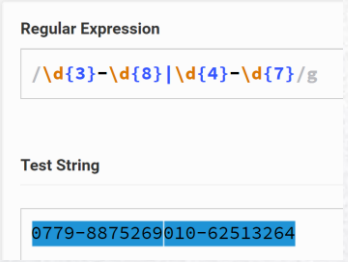


https://www.regextester.com/
在线测试









Regex正则表达式

Regex正则表达式

- 一些练习
  - 1.匹配moon，moooon，mooooooon等，要求中间o为正偶数个
  - 2. 匹配moon，muun，moooon，muuuun，要求中间可以为正偶数个连续的o或u
  - 3. 第2题扩展为也可以匹配moouuuuoouun等，即可以o或u穿插，但连续的部分必须为偶数
  - 4.匹配11位手机号，规律为第一位一定是1
  - 5.匹配ip地址，分为四节，每节最少1位数，最多三位数，形如110.242.68.4

Regex正则表达式

- 一些练习
  - 1.匹配moon，moooon，mooooooon等，要求中间o为正偶数个
  - 2. 匹配moon，muun，moooon，muuuun，要求中间可以为正偶数个连续的o或u
  - 3. 第2题扩展为也可以匹配moouuuuoouun等，即可以o或u穿插，但连续的部分必须为偶数
  - 4.匹配11位手机号，规律为第一位一定是1
  - 5.匹配ip地址，分为四节，每节最少1位数，最多三位数，形如110.242.68.4

答案：

1. m(oo)+n
2. m((oo)+|(uu)+)n
3. m((oo)+|(uu)+)+n
4. 1\d{10}或1[0-9]{10}
5. (\d{1,3}\.){3}\d{1,3}

Regex正则表达式

- 更多练习
  - 正则表达式在线练习
  - https://regex101.com/quiz

Regex正则表达式

Regex正则表达式

- Python的正则表达式
- 最常用的四个函数

match	sub	search	findall
检查格式	替换子串	查找位置	找到全部

用法：

- re.match(pattern, string)，从字符串开头进行匹配
- re.sub(pattern, repl, string)，repl为替换的字符串，也可为一个函数
- re.search(pattern, string)，可以在字符串任何位置匹配
- re.findall(pattern, string)，找到所有满足匹配条件的子串

Regex正则表达式

- Python的正则表达式

match

```
s = "110.242.68.245"
if re.match("(\d{1,3}\.){3}\d{1,3}", s):
    print(True)
```

```
True
```

search

```
s = "I think Renmin University is the best university!"
span = re.search("\w+ University", s)
print(span)
```

```
<re.Match object; span=(8, 25), match='Renmin University'>
```

sub

```
s = "h****a####p@@@@p$$$$y"
s = re.sub("[*$#@]", "", s)
print(s)
```

```
happy
```

findall

```
html = """
<ul>
    <ol class="first">RUC</ol>
    <ol class="second">PKU</ol>
    <ol class="third" style="height:10px">THU</ol>
    <br></br>
    <ol>NYU</ol>
    <ol>SJTU</ol>
</ul>
"""
found = re.findall("<ol.*?>(.*?)</ol>", html)
print(found)
```

```
['RUC', 'PKU', 'THU', 'NYU', 'SJTU']
```

split

```
s="happy$new#year@!"
slist = re.split("[*$#@]",s)
print(slist)
```

```
['happy', 'new', 'year', '!']
```

Regex正则表达式

- 不同的匹配方法
- 匹配对象方法
  - span(): 返回匹配的范围 (start, end)
  - group(num): 在正则表达式中用()可以分组，group() 可以一次输入多个组号，在这种情况下它将返回一个包含那些组所对应值的元组
  - groups(): 返回一个包含所有小组字符串的元组，从 1 到所含的小组号

```
import re
line = "Cats are smarter than dogs"
matchObj = re.match(r'(.*) are (.*?) than (.*)', line)
if matchObj:
    print("matchObj.group(): ",matchObj.group())
    print("matchObj.span(): ",matchObj.span())
    print("matchObj.group(1): ",matchObj.group(1))
    print("matchObj.group(2): ",matchObj.group(2))
    print("matchObj.group(2,3): ",matchObj.group(2,3))
    print("matchObj.groups(): ",matchObj.groups())
else:
    print("No match!!!")
```

```
matchObj.group():  Cats are smarter than dogs
matchObj.span():  (0, 26)
matchObj.group(1):  Cats
matchObj.group(2):  smarter
matchObj.group(2,3):  ('smarter', 'dogs')
matchObj.groups():  ('Cats', 'smarter', 'dogs')
```

Regex正则表达式

Regex正则表达式

- 正则表达式练习——邮箱

34sdg4gfwr4@ruc.edu.cn

653564309@qq.com

wdddangghwz@.com

10.44.75.123

Trer.fsew@sdcs.com

hehehe@gmail.com

Sfw$$$dfgdg@126.com

ghhhh@gmail.ussr

Chigua@sina.cn

34gdggfdgrtyhyr@163.com

Regular Expression

```
/[A-Za-z0-9.]*@([A-Za-z0-9]*.)+(com|cn)/g
```

Test String

```
34sdg4gfwr4@ruc.edu.cn
653564309@qq.com
wdddangghwz@.com
10.44.75.123
Trer.fsew@sdcs.com
hehehe@gmail.com
```

Regular Expression

```
/[A-Za-z0-9.]*@([A-Za-z0-9]*.)+(com|cn)/g
```

Test String

```
Sfw$$$dfgdg@126.com
ghhhh@gmail.ussr
Chigua@sina.cn
34gdggfdgrtyhyr@163.com
```

Regex正则表达式

- 抽取人民日报某天某版的要闻标题
  - http://paper.people.com.cn/rmrb/html/2021-03/25/nbs.D110000renmrb_01.htm

人民日报图文数据库（1946-2021）

人民日报 2021年03月25日 星期四　　上一期　下一期

01版：要闻	02版：要闻	03版：要闻	04版：要闻
05版：奋斗百年路 启航新征程	06版：要闻	07版：评论	08版：广告
09版：理论	10版：文件	11版：综合	12版：经济
13版：政治	14版：文化	15版：社会	16版：生态
17版：体育	18版：民主政治	19版：法治	20版：副刊

· 进一步深化税收征管改革

· 中国共产党成立100周年庆祝活动标识

· 中国共产党成立100周年庆祝活动标识使用说明

· 李克强主持召开国务院常务会议

· 脱贫攻坚民主监督成效显著经验宝贵 新时代多党合作事业前景广阔大有可为

· 持之以恒纠“四风”树新风 为“十四五”开好局起好步提供有力保障

· 扎实做好财税重点工作 积极开展财税政策研究 确保“十四五”开好局起好步

· 穿行花海

```
325 〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_1-01.htm〉进一步深化税收征管改革 〈/a〉〈/li〉
326 〈li class="one"〉
327  〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_2-01.htm〉中国共产党成立100周年庆祝活动标识 〈/a〉〈/li〉
328 〈li〉
329 〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_3-01.htm〉中国共产党成立100周年庆祝活动标识使用说明 〈/a〉〈/li〉
330 〈li class="one"〉
331  〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_4-01.htm〉李克强主持召开国务院常务会议 〈/a〉〈/li〉
332 〈li〉
333 〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_5-01.htm〉脱贫攻坚民主监督成效显著经验宝贵 新时代多党合作事业前景广阔大有可为 〈/a〉〈/li〉
334 〈li class="one"〉
335  〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_6-01.htm〉持之以恒纠“四风”树新风 为“十四五”开好局起好步提供有力保障 〈/a〉〈/li〉
336 〈li〉
337 〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_7-01.htm〉扎实做好财税重点工作 积极开展财税政策研究 确保“十四五”开好局起好步 〈/a〉〈/li〉
338 〈li class="one"〉
339  〈span〉·〈/span〉〈a href=nw.D110000renmrb_20210325_8-01.htm〉穿行花海 〈/a〉〈/li〉
340
```

Regex正则表达式

- 抽取人民日报某天某版的要闻标题
  - http://paper.people.com.cn/rmrb/html/2021-03/25/nbs.D110000renmrb_01.htm

```
import urllib.request
req = urllib.request.Request("http://paper.people.com.cn/rmrb/html/2021-03/25/nbs.D110000renmrb_01.htm")
response = urllib.request.urlopen(req)
content = response.read().decode("utf-8")
print(content)
```

```
<li>
<span>·</span><a href=nw.D110000renmrb_20210325_1-01.htm>进一步深化税收征管改革 </a></li>
<li class="one">
 <span>·</span><a href=nw.D110000renmrb_20210325_2-01.htm>中国共产党成立100周年庆祝活动标识 </a></li>
<li>
<span>·</span><a href=nw.D110000renmrb_20210325_3-01.htm>中国共产党成立100周年庆祝活动标识使用说明 </a></li>
<li class="one">
 <span>·</span><a href=nw.D110000renmrb_20210325_4-01.htm>李克强主持召开国务院常务会议 </a></li>
<li>
<span>·</span><a href=nw.D110000renmrb_20210325_5-01.htm>脱贫攻坚民主监督成效显著经验宝贵 新时代多党合作事业前景广阔大有可为 </a><
>
<li class="one">
 <span>·</span><a href=nw.D110000renmrb_20210325_6-01.htm>持之以恒纠“四风”树新风 为“十四五”开好局起好步提供有力保障 </a></li>
<li>
<span>·</span><a href=nw.D110000renmrb_20210325_7-01.htm>扎实做好财税重点工作 积极开展财税政策研究 确保“十四五”开好局起好步 </a>
i>
<li class="one">
 <span>·</span><a href=nw.D110000renmrb_20210325_8-01.htm>穿行花海 </a></li>
```

Regex正则表达式

- 抽取人民日报某天某版的要闻标题
  - http://paper.people.com.cn/rmrb/html/2021-03/25/nbs.D110000renmrb_01.htm

拿到每个新闻的标题

```
In [50]:
found=re.findall( "<span>·</span><a.*?>(.*?)</a></li>",content)
print(found)
```

```
['进一步深化税收征管改革 ', '中国共产党成立100周年庆祝活动标识 ', '中国共产党成立100周年庆祝活动标识使用说明 ', '李克强主持召开国务院常务会议 ', '脱贫攻坚民主监督成效显著经验宝贵 新时代多党合作事业前景广阔大有可为 ', '持之以恒纠“四风”树新风 为“十四五”开好局起好步提供有力保障 ', '扎实做好财税重点工作 积极开展财税政策研究 确保“十四五”开好局起好步 ', '穿行花海 ']
```

```
In [53]:
found=re.findall( "<span>·</span><a href=(.*?)>.*?</a></li>",content)
print(found)
```

```
['nw.D110000renmrb_20210325_1-01.htm', 'nw.D110000renmrb_20210325_2-01.htm', 'nw.D110000renmrb_20210325_3-01.htm', 'nw.D110000renmrb_20210325_4-01.htm', 'nw.D110000renmrb_20210325_5-01.htm', 'nw.D110000renmrb_20210325_6-01.htm', 'nw.D110000renmrb_20210325_7-01.htm', 'nw.D110000renmrb_20210325_8-01.htm']
```

拿到每个新闻的URL