## 正则测试

正则表达式测试链接：<https://tool.oschina.net/regex/?optionGlobl=global>

正则表达式参考文档：https://www.cnblogs.com/bilaisheng/p/10211045.html

对于URL来说，我们就可以用下面的正则表达式匹配：

[a-zA-z]+://[^\s]\*



比如a-z代表匹配任意的小写字母，\s表示匹配任意的空白字符，\*就代表匹配前面的字符任意多个.

好处：那么对于网页来说，如果我们想找出网页源代码里有多少URL，就可以用匹配URL的正则表达式去匹配，就可以得到源码中的URL了。

### 1 匹配小案例1：

Hello, my phone number is 010-86432100 and email is cqc@cuiqingcai.com, and my website is <http://cuiqingcai.com>.

1. 匹配 010-86432100



2． [匹配 cqc@cuiqingcai.com](mailto:匹配%20%20%20%20%20%20cqc@cuiqingcai.com)

[a-zA-z]+@[^\s]\*

3． 匹配 <http://cuiqingcai.com>

[a-zA-z]+://[^\s]\*

### 2 匹配小案例2：

Hello 123 4567 World\_This is a Regex Demo

1. 匹配 Hello 123 4567 World\_This

^Hello\s\d\d\d\s\d{4}\s\w{10}

这样看起来就很复杂，还有一个万能匹配可以用，也就是.\*，.可以匹配任意字符(除换行符)，\*又代表匹配前面的字符无限次，所以它们组合在一起就可以匹配任意的字符了

1. 匹配 上面所有

^Hello.\*Demo$

将中间的部分直接省略，全部用.\*来代替，最后加一个结尾字符串就好了

1. 匹配 123 4567

\d+\s+\d{4}

## 匹配规则：

模式描述  
\w匹配字母数字及下划线  
\W匹配非字母数字及下划线  
\s匹配任意空白字符，等价于 [\t\n\r\f].  
\S匹配任意非空字符  
\d匹配任意数字，等价于 [0-9]  
\D匹配任意非数字  
\A匹配字符串开始：  
\Z匹配字符串结束，如果是存在换行，只匹配到换行前的结束字符串  
\z匹配字符串结束  
\G匹配最后匹配完成的位置  
\n匹配一个换行符  
\t匹配一个制表符  
^匹配字符串的开头  
$匹配字符串的末尾。  
.匹配任意字符，除了换行符，当re.DOTALL标记被指定时，则可以匹配包括换行符的任意字符。  
[...]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。  
\*匹配0个或多个的表达式。 那么= .\* 代表匹配任意的字符了，一般还需要再最后加一个结尾字符串就好了  
+匹配1个或多个的表达式。  
?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  
{n}精确匹配n个前面表达式。  
{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  
( )匹配括号内的表达式，也表示一个组

如下是一个小测试：



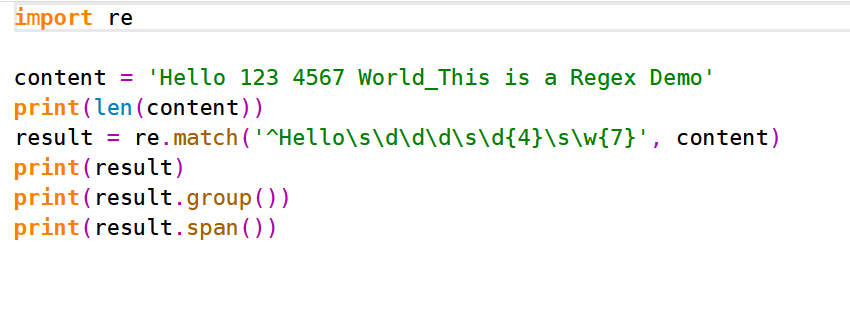
## 3．Python中使用

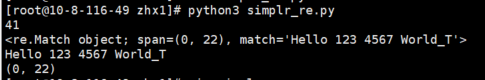
re库提供了整个正则表达式的实现

### match() simplr\_re.py

在这里首先介绍第一个常用的匹配方法，match()方法，我们向这个方法传入要匹配的字符串以及正则表达式，就可以来检测这个正则表达式是否匹配字符串了。

match()方法会尝试从字符串的起始位置匹配正则表达式，如果匹配，就返回匹配成功的结果，如果不匹配，那就返回None





解释：

group()方法可以输出匹配到的内容。这恰好是我们正则表达式规则所匹配的内容，

span()方法可以输出匹配的范围，结果是(0, 22)，这个就是匹配到的结果字符串在原字符串中的位置范围

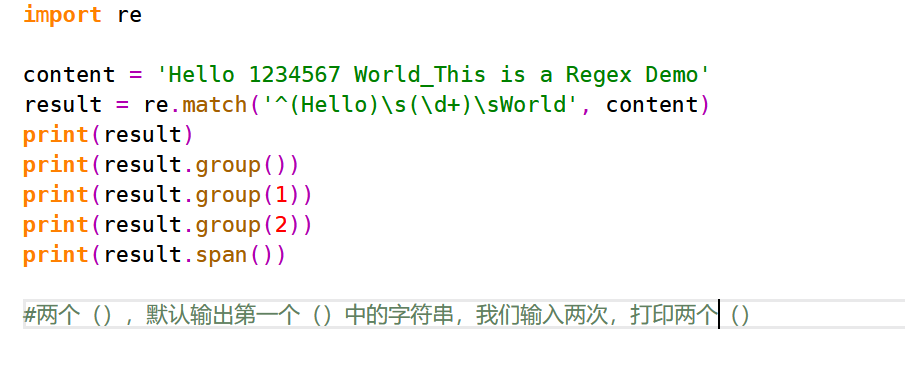
### （）方式，提取的子字符 extract\_string.py

刚才我们用了match()方法可以得到匹配到的字符串内容，但是如果我们想从字符串中提取一部分内容怎么办呢

可以使用()括号来将我们想提取的子字符串括起来，()实际上就是标记了一个子表达式的开始和结束位置



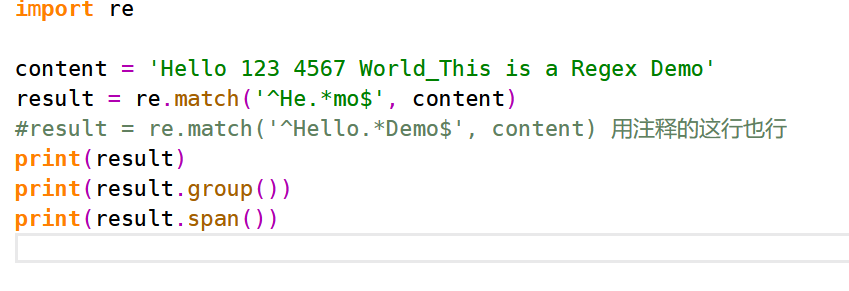
我们测试下：



### 通用匹配 .\* $ currency.py

刚才我们写的正则表达式其实比较复杂，出现空白字符我们就写\s匹配空白字符，出现数字我们就写\d匹配数字，工作量非常大

万能匹配可以用，也就是.\*，.可以匹配任意字符(除换行符)，\*又代表匹配前面的字符无限次，所以它们组合在一起就可以匹配任意的字符了

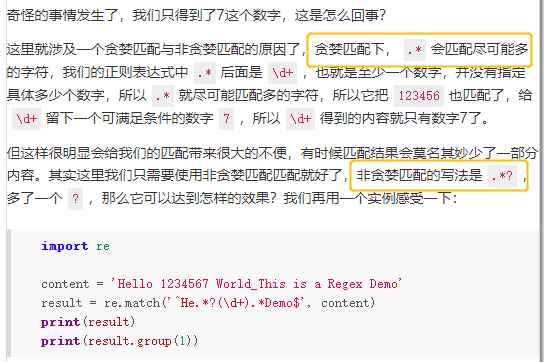




虽然这种方法，可以节约点时间，但是实际上更多的需求是 截取一段话的中间部分到中间部分，并不是所有的都需要截取全部

### 贪婪匹配与非贪婪匹配







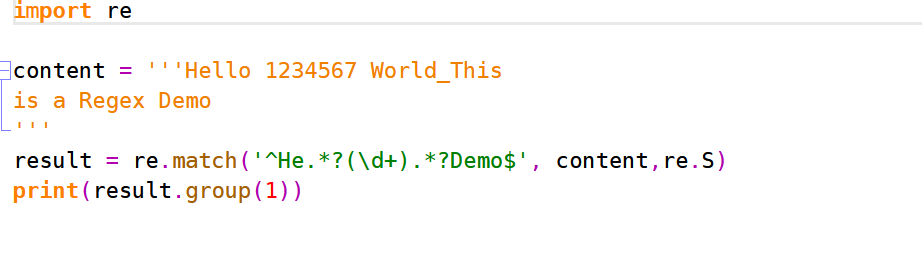
### 修饰符 Modifier.py

.匹配的是除换行符之外的任意字符，当遇到换行符时，.\*?就不能匹配了，所以导致匹配失败

那么在这里我们只需要加一个修饰符re.S，即可修正这个错误。

result = re.match('^He.\*?(\d+).\*?Demo$', content, re.S)

在match()方法的第三个参数传入re.S，它的作用是使.匹配包括换行符在内的所有字符。





### 转义匹配

百度前面的（）也是需要转义的，不然会返回none

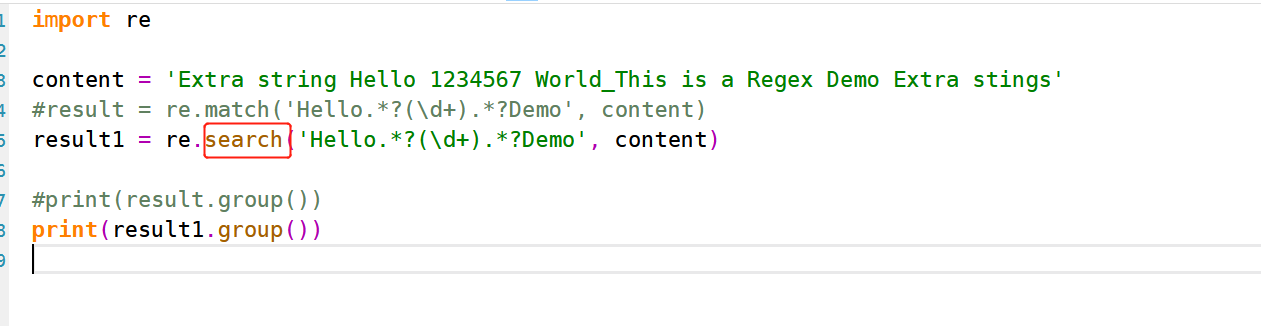


### search() 不从开头开始匹配

我们在前面提到过match()方法是从字符串的开头开始匹配，一旦开头不匹配，那么整个匹配就失败了。我们看下面的例子：

所以match()方法在我们在使用的时候需要考虑到开头的内容，所以在做匹配的时候并不那么方便，它适合来检测某个字符串是否符合某个正则表达式的规则。

所以在这里就有另外一个方法search()，它在匹配时会扫描整个字符串，然后返回第一个成功匹配的结果，也就是说，正则表达式可以是字符串的一部分，在匹配时，search()方法会依次扫描字符串，直到找到第一个符合规则的字符串，然后返回匹配内容，如果搜索完了还没有找到，那就返回None。





### findall()

在前面我们说了search()方法的用法，它可以返回匹配正则表达式的第一个内容，但是如果我们想要获取匹配正则表达式的所有内容的话怎么办？这时就需要借助于findall()方法了



<https://www.cnblogs.com/xieshengsen/p/6727064.html>

如要匹配全部内容，原始数据raw\_data如下：



要如何匹配呢：

