In [16]:

```
import time
print("少数字符串拼接:")
start=time.perf_counter()
str1="hello" + "world"
end=time.perf_counter()
print("'+'的使用时间：")
print(end-start)                                    #对于少数字符拼接，+更快速
start=time.perf_counter()
str2 = " ".join(["hello", "world"])
end=time.perf_counter()
print("'join'的使用时间：")
print(end-start)
```

```
少数字符串拼接:
'+'的使用时间：
6.919999998444837e-05
'join'的使用时间：
7.110000001375738e-05
```

In [24]:

```
#大量字符串拼接
s = []
for n in range(0,1000):
    s.append(str(n))
str3=""
start = time.perf_counter()
for j in s:
    str3=str3+j
end = time.perf_counter()
print("'+'的使用时间：")
print(end-start)                                    #对于很多字符拼接，join更快

start = time.perf_counter()
str4=''.join(s)
end = time.perf_counter()
print("'join'的使用时间：")
print(end-start)
```

```
'+'的使用时间：
0.0004618000000003914756
'join'的使用时间：
0.00015609999991283985
```

In [40]:

```
print("123python".isalnum(),"123!=  @@".isalnum() )      #isalnum:如果 string 至少有一个字符并且所有字

print("abcdef".isalpha(),"123abc".isalpha(),"abc! !=...".isalpha())#isalpha:如果字符串至少有一个字符

print("1234".isdigit(),"123abc".isdigit(),"123! !=...".isdigit())#isdigit:如果字符串只包含数字则返回

print(u"37913719".isdecimal(),u"1515abcd".isdecimal()) #isdecimal() 方法检查字符串是否只包含十进制字

print(u"123pyhton".isnumeric(),u"1515abcd".isnumeric(),u"四".isnumeric(),) #如果字符串中只包含数字字

print("      ".isspace(),"This is string example....wow!!".isspace())#如果字符串中只包含空格，则返回

print("ABCDEF".isupper(),"AbcdEFG".isupper())#isupper() 方法检测字符串中所有的字母是否都为大写。

print("abcdefg".islower(),"AbcdEFG".islower())#islower() 方法检测字符串中所有的字母是否都为小写。
```

```
True False
True False False
True False False
True False
False False True
True False
True False
True False
```

In [51]:

```
str = "this is string example....wow!!!"       # zfill() 方法返回指定长度的字符串，原字符串右对齐，前
print(str.zfill(40))
print(str.zfill(50))

str = 'runoob'                                  #center() 返回一个原字符串居中,并使用空格填充至长度 w
print(str.center(20, '*'))
print(str.center(20))

str = "this is string example....wow!!!"    #ljust() 方法返回一个原字符串左对齐,并使用空格填充至指定
print(str.ljust(50, '0'))                    #如果指定的长度小于原字符串的长度则返回原字符串。

str = "this is string example....wow!!!"    #rjust() 方法返回一个原字符串右对齐,并使用空格填充至指定
print(str.rjust(50, '0'))                    #如果指定的长度小于原字符串的长度则返回原字符串。

str = "this is string example....wow!!!"   #startswith() 方法用于检查字符串是否是以指定子字符串开头,
print(str.startswith( 'this' ))              #如果参数 beg 和 end 指定值，则在指定范围内检查。
print (str.startswith( 'is', 2, 4 ))
print (str.startswith( 'this', 2, 4 ))

suffix = "is"
print(str.endswith(suffix, 2, 4))           #endswith() 方法用于判断字符串是否以指定后缀结尾，如果以指
print (str.endswith(suffix, 2, 6))          #可选参数"start"与"end"为检索字符串的开始与结束位置。
```

```
00000000this is string example....wow!!!
0000000000000000000this is string example....wow!!!
*******runoob*******
       runoob
this is string example....wow!!!000000000000000000
0000000000000000000this is string example....wow!!!
True
True
False
True
False
```

In [67]:

```
import re
html="""<div id="topics">
    <div class = "post">
        <h1 class = "postTitle">
            <a id="cb_post_title_url" class="postTitle2" href="https://www.cnblogs.com/cq90/p/695956
        </h1>
        <div class="clear"></div>
        <div class="postBody">
            <div id="cnblogs_post_body" class="blogpost-body"><p>u/U:表示unicode字符串 <br>不是
<p>r/R:非转义的原始字符串 <br>与普通字符相比，其他相对特殊的字符，其中可能包含转义字符，即那些，
<p>b:bytes <br>python3.x里默认的str是(py2.x里的)unicode, bytes是(py2.x)的str, b" "前缀代表的就
<p> </p>
<p>参考：http://blog.csdn.net/u010496169/article/details/70045895</p></div><div id="MySignature"></d
<div class="clear"></div>
<div id="blog_post_info_block">
<div id="BlogPostCategory"></div>
<div id="EntryTag"></div>
<div id="blog_post_info">
</div>
<div class="clear"></div>
<div id="post_next_prev"></div>
</div>"""
str = re.compile(r'<[^>]+>',re.S)
result = str.sub('', html)
print(result)
```

python学习-字符串前面添加u,r,b的含义

u/U:表示unicode字符串 不是仅仅是针对中文, 可以针对任何的字符串，代表是对字符串进行unicode编码。 一般英文字符在使用各种编码下, 基本都可以正常解析, 所以一般不带u；但是中文, 必须表明所需编码, 否则一旦编码转换就会出现乱码。 建议所有编码方式采用utf8
r/R:非转义的原始字符串 与普通字符相比，其他相对特殊的字符，其中可能包含转义字符，即那些，反斜杠加上对应字母，表示对应的特殊含义的，比如最常见的"
"表示换行，"　"表示Tab等。而如果是以r开头，那么说明后面的字符，都是普通的字符了，即如果是“
”那么表示一个反斜杠字符，一个字母n，而不是表示换行了。 以r开头的字符，常用于正则表达式，对应着re模块。
b:bytes python3.x里默认的str是(py2.x里的)unicode, bytes是(py2.x)的str, b"
“前缀代表的就是bytes python2.x里, b前缀没什么具体意义， 只是为了兼容python 3.x的这种写法

参考：http://blog.csdn.net/u010496169/article/details/70045895