字符串和正则表达式

字符串常用操作

1. 拼接字符串

   使用+运算符完成多个字符串拼接，产生一个新的字符串

```
str1 = "hello"
str2 = 'Python'

print("str1 + str2:",str1 + str2)
```

```
str1 + str2: helloPython
```

2. 计算字符串的长度

```
#计算字符串的长度
str1 = "人生苦短，我用Python"
print("str1的长度: ", len(str1)) #str1的长度:  13
```

3. 截取字符串

   语法：

> string[start: end : step]
>
> string: 原始字符串
>
> start:开始位置（索引）
>
> end:结束位置
>
> step:步长

```
#截取字符串
print("原始字符串:", str1)
print("第一个字符:", str1[0])
print("第3个到第5个字符: ", str1[2:5])
```

```
str1的长度:  13
原始字符串: 人生苦短，我用Python
第一个字符: 人
第3个到第5个字符:  苦短，
```

4. 分割字符串

   分割字符串是把字符串分割成列表。语法：

> str.split(sep, maxsplit)
>
> str:原始字符串

> sep:用于指定分隔符，可以使多个，默认为None，即所有空字符(空格，换行"\n" ,制表符"\t"等)

```
# 分割字符串
str1 = "I love Python"
list1 = str1.split()
print(list1) #['I', 'love', 'Python']

str2 = "I love Python,java,C#"
list2 = str2.split(',')
print(list2) # ['I love Python', 'java', 'C#']
```

5. 检索字符串
    - count () 函数

```
# 检索字符串
# 利用count函数
str1 = "@Python @java @C#"
print('字符串',str1,"中@符号出现了", str1.count('@'), '次')
# 打印结果：字符串 @Python @java @C# 中@符号出现了 3 次
```

    - find () 函数

```
# find 获取指定字符首次出现的索引位置
print("字符串:", str1, "中@第一次出现的位置是：", str1.find("@"))
# 打印结果：字符串: @Python @java @C# 中@第一次出现的位置是：  0
```

    - index()函数

```
# index 获取指定字符首次出现的索引位置
print("字符串:", str1, "中@第一次出现的位置是：", str1.index("@"))
# 打印结果：字符串: @Python @java @C# 中@第一次出现的位置是：  0
```

      和find很像，但是有区别，如果用find找一个不存在元素，不会报错，但是index会

```
# 在str1中查找! 号
print(str1.find('!')) # 结果-1

print(str1.index('!')) # ValueError: substring not found
```

    - startwidth() 判断字符串是否是以指定字符开头

```
# startwith() 判断是否以指定字符开头
str1 = "@Python @java @C#"
print(str1.startswith("@")) # True
```

    - endswith()判断是否以指定的字符结尾

```
# endswidth() 判断是否以指定字符结尾
print(str1.endswith('#')) #True
```

6. 字母大小写转换

- lower 大写变小写

```
# 大写变小写
str1 = "ABCDEFG"
print(str1.lower()) # 打印结果abcdefg
```

- upper()小写变大写

```
# 小写变大写
str1 = "qwertyuiop"
print(str1.upper()) # 打印结果：QWERTYUIOP
```

7. 去除字符串中的空格和特殊字符

- strip()去掉左右空格

```
#strip 去左右空格
str1 = "     Python    "
print(str1.strip()) # 打印结果Python
```

- lstrip()去左边空格 rstrip()去右边空格

```
# lstrip 去左边空格
print(str1.lstrip()) #'Python    '

#rstrip 去右边空格
print(str1.rstrip()) #      Python
```

如果在括号中加入指定字符，那么就可以去掉指定的字符

```
# 去掉左右的#好
str1 = "# asdfgh #"
print(str1.strip('#')) #  asdfgh
```

8. 格式化字符串

- 使用% 号格式化 这个方法现在不推荐使用了，大家可以自行百度一下用法就好。
- format()方法 推荐使用这个方法来格式化

```
name = '路飞'
age = 18
print("{}的年龄是{}".format(name, age)) # 路飞的年龄是18
```

该方法还有很多其他参数可以设置，大家自行查文档了解一下。

正则表达式基础

1. 行定位符

   “^ ”表示行的开始，"$"表示行的结尾

   例如: ^qq 可以和“qq:1234444“匹配，但不能和“1234的qq“匹配

2. 元字符

代码	说明
.	匹配除了行符意外的任意字符
\w	匹配字母，数字，下划线或汉字
\W	匹配字母，数字，下划线或汉字以外的字符
\s	匹配单个空白字符
\S	匹配单个空白字符以外的字符
\d	匹配数字
\b	匹配单词的开始或结束，单词的分界符通常是空格，标点或者换行。
^	匹配字符的开始
$	匹配字符的结束

   如匹配8位QQ号：^\d{8}$

3. 限定符

限定符	说明	举例
?	匹配前面的字符0次或1次	colou?r,可以匹配到colour和color
+	匹配前面的字符1次或者多次	go+gle可以匹配到：gogle,或go...gle(省略多个o)
*	匹配前面字符零次或多次	go*gle,可以匹配到ggle,或go...gle(省略多个o)
{n}	匹配前面n次	go{2}gle,可以匹配到google
{n,}	匹配前面至少n次	go{2，}gle,可以匹配到google或go...gle(省略多个o)
{n,m}	pp匹配前面至少n次，最多m次	go{2，6}gle,可以匹配到google或go...gle(省略多个o，但最多6个)

4. 排除字符

   使用的是^ 比如[^ a-zA-Z] 表示匹配一个不是字母的字符

5. 转义字符

   转义用\ 比如要匹配. 就要转义 \. 才行

6. 在Python中使用正则表达式语法

   需要将表达式中的所有\ 用两个\\替换。如：'\\ bm'

   或者在表达式前面加 r 如：r'\bm'

使用re模块实现正则表达式操作

在使用这个模块时需要引入re模块

```
import re
```

1. 匹配字符串
   - 使用match()
   - 使用search()
   - 使用findall
2. 替换字符串

```
# 替换字符串 sub方法
pattern = r'1[34578]\d{9}'
string = '中奖号码为: python0001 联系电话为：13698989802'
result = re.sub(pattern, '1pythoner', string)
print(result) # 中奖号码为: python0001 联系电话为：1pythoner
```