Python核心数据类型-字符串

什么是字符串？
一个有序的字符集合，用来表现和存储基于文本的信息

特点：

1.支持的操作　合并　分片 索引，　这些操作对于任意一种序列对象都可以使用 如 列表 元组

X+Y将会创建一个包含了两个操作对象内容的新的序列对象。

X*N将会创建一个包含操作对象X内容N份拷贝的新的序列对象。

字符串操作:

字符串可以通过 + 进行拼接，　也可以通过 * 进行重复

```
In [28]: x = 'apple'
In [29]: b = 'banana'

In [30]: x + b
Out[30]: 'applebanana'

In [31]: x * 4
Out[31]: 'appleappleappleapple'
```

2. 字符串是一个序列，　可以进行迭代

3. 不可变类型:字符串所包含的字符存在从左到右的顺序且不可以在原处修改
   字符串是不可变对象

   ```
   In [43]: x[0] = 'cat'
   ----------------------------------------------------------------------
   ------
   ```

```
TypeError                                 Traceback (most recent call last)
<ipython-input-43-ab916fbb5227> in <module>()
----> 1 x[0] = 'cat'

TypeError: 'str' object does not support item assignment
```

1. upper #转换string中的小写字母为大写

```
In [1]: a = 'apple'
In [2]: a.upper()
Out[2]: 'APPLE'
```

2.lower #转换string中的大写字母为小写

```
In [5]: a
Out[5]: 'APPLE'
In [6]: a.lower()
Out[6]: 'apple'
```

3. isdigit() ##string中只包含数字则为 true

```
In [8]: a = '123123'
In [9]: a.isdigit()
Out[9]: True
In [10]: a = '123a123'
In [11]: a.isdigit()
Out[11]: False
```

4. isallpha() #string至少有一个字符且所有字符都是字母则返回true

```
In [12]: a = 'apple'
In [13]: b = 'apple1'
In [14]: a.isalpha()
Out[14]: True
In [16]: b.isalpha()
```

Out[16]: False

5. isspace() #string中都是空格则返回true

In [17]: a
Out[17]: 'apple'
In [18]: b
Out[18]: 'apple1'
In [19]: c = '   '
In [20]: a.isspace()
Out[20]: False
In [21]: b.isspace()
Out[21]: False
In [22]: c.isspace()
Out[22]: True

6. isalnum() ## string中至少有一个字符并且所有字符都是字母或数字则返回true

In [23]: a
Out[23]: 'apple'
In [24]: b
Out[24]: 'apple1'
In [25]: c
Out[25]: '
In [26]: a.isalnum()
Out[26]: True
In [27]: b.isalnum()
Out[27]: True
In [28]: c.isalnum()
Out[28]: False

7. islower() 如果字符串都是小写字母则返回 true

In [59]: a
Out[59]: 'apple'
In [60]: a.islower()

Out[60]: True
In [61]: a = 'Apple()'
In [62]: a.islower()
Out[62]: False

8.istitle() ##检测字符串中所有单词拼写的首字母是否为大写 切其他的字母为小写
In [63]: a = 'apple'
In [64]: b = 'Big'
In [65]: c = 'CAT'
In [66]: a.istitle()
Out[66]: False
In [67]: b.istitle()
Out[67]: True
In [68]: c.istitle()
Out[68]: False

9.isupper 所有字母都是大写字母则返回true
In [71]: b
Out[71]: 'Big'
In [72]: c
Out[72]: 'CAT'
In [73]: c.isupper()
Out[73]: True
In [74]: b.isupper()
Out[74]: False

10. lstrip() ##截掉string左边的空格

In [30]: a
Out[30]: ' apple '
In [31]: a.lstrip()
Out[31]: 'apple '

11. rstrip() ##截掉string右边的空格

In [32]: a
Out[32]: ' apple   '
In [33]: a.rstrip()
Out[33]: ' apple'

12. strip()   ##截掉string两边的空格

In [34]: a
Out[34]: ' apple   '
In [35]: a.strip()
Out[35]: 'apple'

13. startswith() # 字符串是否以 XX 开头

In [42]: a
Out[42]: 'apple'
In [44]: a.startswith('a')
Out[44]: True
In [45]: a.startswith('e')
Out[45]: False

14. endswith() #字符串是否以 XX结尾

In [42]: a
Out[42]: 'apple'
In [46]: a.endswith('e')
Out[46]: True
In [47]: a.endswith('a')
Out[47]: False

15. split() #通过指定的分隔符对字符串进行切片 默认值为空格

In [54]: a = 'apple'
In [55]: a.split('l')   ##指定以l作为分割
Out[55]: ['app', 'e']

16. splitlines() ##按照行分隔 返回一个包含各行作为元素的列表

```
In [61]: a
Out[61]: 'Line1-a b c d e f\nLine2- a b c\n\nLine4- a b c d'
In [62]: a.splitlines()
Out[62]: ['Line1-a b c d e f', 'Line2- a b c', '', 'Line4- a b c d']
```

17.rsplit()

```
In [7]: x = 'a b c d e f g'
In [3]: x.rsplit(' ', 1)
Out[3]: ['a b c d', 'e']
```

18. join() ##用来连接字符串

```
In [68]: a
Out[68]: 'apple'
In [69]: ','.join(a)
Out[69]: 'a,p,p,l,e'

In [73]: a = ['a', 'b', 'c']
In [74]: ''.join(a)
Out[74]: 'abc'
In [75]: ' '.join(a)
Out[75]: 'a b c'
```

19.replace() #替换字符串中的元素

```
In [15]: a
Out[15]: 'apple'
In [16]: a.replace('p', 'ee')
```

Out[16]: 'aeeeele'

20.capitalize() ##字符串首字母大写　其他小写
In [13]: a
Out[13]: 'apple'
In [14]: a.capitalize()
Out[14]: 'Apple'

21. center ##原字符居中并且使用空格填充长度

In [22]: a
Out[22]: 'apple'
In [23]: a.center(10)
Out[23]: '　apple　‘　　#在左右两边填充空格
In [24]: a.center(10, 's') ##在 a 的两边填充10个字符s
Out[24]: 'ssapplesss'

22.expandtabs()　##把tab转换成空格 默认一个tab = 8个空格
In [33]: a
Out[33]: '\tapple'
In [34]: b = a.expandtabs()
In [35]: len(b)
Out[35]: 13
In [36]: len(a)
Out[36]: 6
In [37]: b
Out[37]: '　　apple'

23. find 检查str是否包含在string中 ， 返回他第一次出现位置的索引， 如果str不在string中 将返回-1
In [41]: a = 'apple'
In [42]: a.find('e')
Out[42]: 4

24. x.rfind()　##从右侧查找Str是否在string中 返回它第一次出现的位置

25. index() 返回 字符在字符串中的位置

```
In [49]: a
Out[49]: 'apple'
In [50]: a.index('l')
Out[50]: 3
```

26. x.rindex()

27.format() # 格式化字符串输出

```
In [43]: name = 'liushuo'
In [44]: age = 25
In [45]: print 'name %s age is %s' %(name, age)
name liushuo age is 25
In [47]: print 'name {0} age is {1}'.format(name, age)       #格式化字符串输出  {0}  代表format中的第            一个元素  {1}  代表format中的第二个元素
name liushuo age is 25
```

28. rjust #返回原字符串右对齐 并且使用空格填充长度

```
In [80]: a
Out[80]: 'apple'
In [83]: a.rjust(10, '%')
Out[83]: '%%%%%apple'
```

29.ljust #返回原字符串左对齐 并且使用空格填充长度

```
In [76]: a
Out[76]: 'apple'
In [77]: a.ljust(10)
Out[77]: 'apple       '
In [78]: a.ljust(10, '#')
```

Out[78]: 'apple#####'

30. partition() ##从str出现的第一个位置起 把字符串string分成三个元素的元组(string_pre str, str_post)
In [85]: a
Out[85]: 'apple'
In [86]: a.partition('l')
Out[86]: ('app', 'l', 'e')
In [87]: a.partition('p')
Out[87]: ('a', 'p', 'ple')

31. rpartition() #与partition功能类似 不过是从右边查找

32.swapcase #反转String中的大小写

In [90]: a
Out[90]: 'apple'
In [91]: a = 'Apple'
In [92]: a.swapcase()
Out[92]: 'aPPLE'

33. title ## 把字符串的首字母更改为大写 其他字母为小写

In [93]: a = 'Apple'
In [94]: b = 'APple'
In [95]: a.title()
Out[95]: 'Apple'
In [96]: b.title()
Out[96]: 'Apple'

capitalize()与 title()都是可以使字符串首字母大写.

主要区别在于:

>>> a = "this is a test string"

>>> a.title()

'This Is A Test String'

```
>>> a.capitalize()
'This is a test string'
>>>
```

与 istitle的区别是 istitle只会显示 True or False

34. translate

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import string      #  导入string模块

intab = "aeiou"
outtab = "12345"
deltab = "thw"

trantab = string.maketrans(intab,outtab) #  创建字符映射转换表

test = "this is string example....wow!!!";

print test.translate(trantab);
print test.translate(trantab,deltab); # Python2中，删除指定字符在
```

translate() 方法中
以上实例输出结果如下：

```
th3s 3s str3ng 2x1mpl2....w4w!!!
3s 3s sr3ng 2x1mpl2....4!!!
```

35. zfill
指定最终字符串的长度 原字符右对齐 前面填充0

```
In [102]: a
Out[102]: 'Apple'
In [103]: a.zfill(40)
Out[103]: '0000000000000000000000000000000000Apple'
In [104]: a.zfill(10)
Out[104]: '00000Apple'
```

36. x.decode()

37 x.encode()

38.x.count()
返回某个字符出现的次数

Len() ##返回字符串的长度

str()

```
In [44]: x = [1,2,3,4]
In [45]: str(x)
Out[45]: '[1, 2, 3, 4]'
```

索引与分片

```
In [32]: x
Out[32]: 'apple'

In [33]: x[1]
Out[33]: 'p'

In [34]: x[1:4]
Out[34]: 'ppl'
```

特殊的分片：分片表达式增加了一个可选的第三个索引， 用作步进

In [35]: x
Out[35]: 'apple'

In [38]: x[::2]
Out[38]: 'ape'

字符串反转

In [39]: x
Out[39]: 'apple'

In [41]: x[::-1]
Out[41]: 'elppa'