



马哥教育

IT 人的高薪职业学院

Python内置数据结构

讲师：Wayne

从业十余载，漫漫求知路

字符串

- 一个个字符组成的有序的序列，是字符的集合
- 使用单引号、双引号、三引号引住的字符序列
- 字符串是**不可变**对象
- Python3起，字符串就是Unicode类型

字符串定义 初始化

□ 举例

```
s1 = 'string'
```

```
s2 = "string2"
```

```
s3 = '''this's a "String" '''
```

```
s4 = 'hello \n magedu.com'
```

```
s5 = r"hello \n magedu.com"
```

```
s6 = 'c:\windows\nt'
```

```
s7 = R"c:\windows\nt"
```

```
s8 = 'c:\windows\\nt'
```

```
sql = """select * from user where name='tom' """
```



字符串元素访问——下标

□ 字符串支持使用索引访问

```
sql = "select * from user where name='tom'"
```

```
sql[4] # 字符串'c'
```

```
sql[4] = 'o'
```

□ 有序的字符集合，字符序列

```
for c in sql:
```

```
    print(c)
```

```
    print(type(c)) # 什么类型？
```

□ 可迭代

```
lst = list(sql)
```



字符串join连接*

□ "string".join(iterable) -> str

- 将可迭代对象连接起来，使用string作为分隔符
- 可迭代对象本身元素都是字符串
- 返回一个新字符串

```
lst = ['1','2','3']
```

```
print("\\".join(lst)) # 分隔符是双引号
```

```
print(" ".join(lst))
```

```
print("\n".join(lst))
```

```
lst = ['1',['a','b'],'3']
```

```
print(" ".join(lst))
```



字符串+连接

□ + -> str

- 将2个字符串连接在一起
- 返回一个新字符串



字符串分割

- 分割字符串的方法分为2类

- split系

- 将字符串按照分隔符分割成若干字符串，并返回列表

- partition系

- 将字符串按照分隔符分割成2段，返回这2段和分隔符的元组

字符串分割*

□ `split(sep=None, maxsplit=-1)` -> list of strings

□ 从左至右

□ `sep` 指定分割字符串，缺省的情况下空白字符串作为分隔符

□ `maxsplit` 指定分割的次数，-1 表示遍历整个字符串

```
s1 = "I'm \ta super student."
```

```
s1.split()
```

```
s1.split('s')
```

```
s1.split('super')
```

```
s1.split('super ')
```

```
s1.split(' ')
```

```
s1.split(' ',maxsplit=2)
```

```
s1.split('\t',maxsplit=2)
```


字符串分割

❑ `rsplit(sep=None, maxsplit=-1)` -> list of strings

❑ 从右向左

❑ `sep` 指定分割字符串，缺省的情况下空白字符串作为分隔符

❑ `maxsplit` 指定分割的次数，-1 表示遍历整个字符串

```
s1 = "I'm \ta super student."
```

```
s1.rsplit()
```

```
s1.rsplit('s')
```

```
s1.rsplit('super')
```

```
s1.rsplit('super ')
```

```
s1.rsplit(' ')
```

```
s1.rsplit(' ',maxsplit=2)
```

```
s1.rsplit('\t',maxsplit=2)
```

字符串分割

□ `splitlines([keepends])` -> list of strings

□ 按照行来切分字符串

□ `keepends` 指的是是否保留行分隔符

□ 行分隔符包括 `\n`、`\r\n`、`\r` 等

```
'ab c\n\nde fg\rkl\r\n'.splitlines()
```

```
'ab c\n\nde fg\rkl\r\n'.splitlines(True)
```

```
s1 = '''I'm a super student.
```

```
You're a super teacher.'''
```

```
print(s1)
```

```
print(s1.splitlines())
```

```
print(s1.splitlines(True))
```



字符串分割*

□ partition(sep) -> (head, sep, tail)

□ 从左至右，遇到分隔符就把字符串分割成两部分，返回头、分隔符、尾三部分的三元组；如果没有找到分隔符，就返回头、2个空元素的三元组

□ sep 分割字符串，必须指定

```
s1 = "I'm a super student."
```

```
s1.partition('s')
```

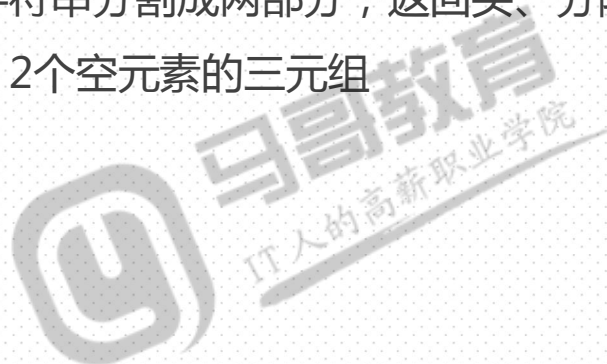
```
s1.partition('stu')
```

```
s1.partition('')
```

```
s1.partition('abc')
```

□ rpartition(sep) -> (head, sep, tail)

□ 从右至左，遇到分隔符就把字符串分割成两部分，返回头、分隔符、尾三部分的三元组；如果没有找到分隔符，就返回2个空元素和尾的三元组



字符串大小写

- upper()
 - 全大写
- lower()
 - 全小写
- 大小写，做判断的时候用
- swapcase()
 - 交互大小写



字符串排版

- ❑ title() -> str
 - ❑ 标题的每个单词都大写
- ❑ capitalize() -> str
 - ❑ 首个单词大写
- ❑ center(width[, fillchar]) -> str
 - ❑ width 打印宽度
 - ❑ fillchar 填充的字符
- ❑ zfill(width) -> str
 - ❑ width 打印宽度，居右，左边用0填充
- ❑ ljust(width[, fillchar]) -> str 左对齐
- ❑ rjust(width[, fillchar]) -> str 右对齐
- ❑ 中文用的少，了解一下



字符串修改*

□ `replace(old, new[, count]) -> str`

□ 字符串中找到匹配替换为新子串，返回新字符串

□ `count`表示替换几次，不指定就是全部替换

`'www.magedu.com'.replace('w','p')`

`'www.magedu.com'.replace('w','p',2)`

`'www.magedu.com'.replace('w','p',3)`

`'www.magedu.com'.replace('ww','p',2)`

`'www.magedu.com'.replace('www','python',2)`

字符串修改*

□ strip([chars]) -> str

- 从字符串两端去除指定的字符集chars中的所有字符
- 如果chars没有指定，去除两端的空白字符

```
s = "\r\n\t Hello Python \n\t"
```

```
s.strip()
```

```
s = " I am very very very sorry "
```

```
s.strip('ly')
```

```
s.strip('ly ')
```

□ lstrip([chars]) -> str

- 从左开始

□rstrip([chars]) -> str

- 从右开始



字符串查找*

❑ find(sub[, start[, end]]) -> int

❑ 在指定的区间[start, end), 从左至右, 查找子串sub。找到返回索引, 没找到返回-1

❑ rfind(sub[, start[, end]]) -> int

❑ 在指定的区间[start, end), 从右至左, 查找子串sub。找到返回索引, 没找到返回-1

s = "I am very very very sorry"

s.find('very')

s.find('very', 5)

s.find('very', 6, 13)

s.rfind('very', 10)

s.rfind('very', 10, 15)

s.rfind('very', -10, -1)

字符串查找*

□ `index(sub[, start[, end]]) -> int`

□ 在指定的区间[start, end), 从左至右, 查找子串sub。找到返回索引, 没找到抛出异常ValueError

□ `rindex(sub[, start[, end]]) -> int`

□ 在指定的区间[start, end), 从左至右, 查找子串sub。找到返回索引, 没找到抛出异常ValueError

`s = "I am very very very sorry"`

`s.index('very')`

`s.index('very', 5)`

`s.index('very', 6, 13)`

`s.rindex('very', 10)`

`s.rindex('very', 10, 15)`

`s.rindex('very', -10, -1)`

字符串查找

□ 时间复杂度

- index和count方法都是 $O(n)$
- 随着列表数据规模的增大，而效率下降

□ len(string)

- 返回字符串的长度，即字符的个数



字符串查找

□ `count(sub[, start[, end]])` -> int

□ 在指定的区间[start, end), 从左至右, 统计子串sub出现的次数

`s = "I am very very very sorry"`

`s.count('very')`

`s.count('very', 5)`

`s.count('very', 10, 14)`



字符串判断*

❑ `endswith(suffix[, start[, end]]) -> bool`

❑ 在指定的区间[start, end) , 字符串是否是suffix结尾

❑ `startswith(prefix[, start[, end]]) -> bool`

❑ 在指定的区间[start, end) , 字符串是否是prefix开头

`s = "I am very very very sorry"`

`s.startswith('very')`

`s.startswith('very', 5)`

`s.startswith('very', 5, 9)`

`s.endswith('very', 5, 9)`

`s.endswith('sorry', 5)`

`s.endswith('sorry', 5, -1)`

`s.endswith('sorry', 5, 100)`

字符串判断 is系列

- ❑ isalnum() -> bool 是否是字母和数字组成
- ❑ isalpha() 是否是字母
- ❑ isdecimal() 是否只包含十进制数字
- ❑ isdigit() 是否全部数字(0~9)
- ❑ isidentifier() 是不是字母和下划线开头，其他都是字母、数字、下划线
- ❑ islower() 是否都是小写
- ❑ isupper() 是否全部大写
- ❑ isspace() 是否只包含空白字符

字符串格式化

- ❑ 字符串的格式化是一种拼接字符串输出样式的手段，更灵活方便
 - ❑ join拼接只能使用分隔符，且要求被拼接的是可迭代对象
 - ❑ + 拼接字符串还算方便，但是非字符串需要先转换为字符串才能拼接
- ❑ 在2.5版本之前，只能使用printf style风格的print输出
 - ❑ printf-style formatting，来自于C语言的printf函数
 - ❑ 格式要求
 - ❑ 占位符：使用%和格式字符组成，例如%s、%d等
 - ❑ s调用str()，r会调用repr()。所有对象都可以被这两个转换。
 - ❑ 占位符中还可以插入修饰字符，例如%03d表示打印3个位置，不够前面补零
 - ❑ format % values，格式字符串和被格式的值之间使用%分隔
 - ❑ values只能是一个对象，或是一个和格式字符串占位符数目相等的元组，或一个字典



字符串格式化

□ printf-style formatting 举例

"I am %03d" % (20,)

'I like %s.' % 'Python'

'%3.2f%% , 0x%x, 0X%02X' % (89.7654, 10, 15)

"I am %-5d" % (20,)



字符串格式化***

□ format函数格式字符串语法——Python鼓励使用

- "{ } {xxx}".format(*args, **kwargs) -> str
- args是位置参数，是一个元组
- kwargs是关键字参数，是一个字典
- 花括号表示占位符
- {}表示按照顺序匹配位置参数，{n}表示取位置参数索引为n的值
- {xxx}表示在关键字参数中搜索名称一致的
- {} 表示打印花括号



字符串格式化***

□ 位置参数

`"{:}:".format('192.168.1.100',8888)`，这就是按照位置顺序用位置参数替换前面的格式字符串的占位符中

□ 关键字参数或命名参数

`"{server} {1}:{0}".format(8888, '192.168.1.100', server='Web Server Info : ')`，位置参数按照序号匹配，关键字参数按照名词匹配

□ 访问元素

`"{0[0]}.{0[1]}".format(('magedu','com'))`

□ 对象属性访问

```
from collections import namedtuple
```

```
Point = namedtuple('Point','x y')
```

```
p = Point(4,5)
```

```
"{{{0.x},{0.y}}}".format(p)
```

字符串格式化***

□ 对齐

```
'{0}*{1}={2:<2}'.format(3,2,2*3)
```

```
'{0}*{1}={2:<02}'.format(3,2,2*3)
```

```
'{0}*{1}={2:>02}'.format(3,2,2*3)
```

```
'{: ^30}'.format('centered')
```

```
'{: * ^30}'.format('centered')
```

□ 进制

```
"int: {0:d}; hex: {0:x}; oct: {0:o}; bin: {0:b}".format(42)
```

```
"int: {0:d}; hex: {0:#x}; oct: {0:#o}; bin: {0:#b}".format(42)
```

```
octets = [192, 168, 0, 1]
```

```
'{:02X}{:02X}{:02X}{:02X}'.format(*octets)
```



字符串格式化***

请使用format函数格式化字符串



马哥教育
IT人的高薪职业学院

字符串练习

- 用户输入一个数字
 - 判断是几位数
 - 打印每一位数字及其重复的次数
 - 依次打印每一位数字，顺序个、十、百、千、万...位
- 输入5个数字，打印每个数字的位数，将这些数字排序打印，要求升序打印

马哥教育
IT人的高薪职业学院

谢谢

咨询热线 400-080-6560

