第4章 字符串

授课老师：刘国旭

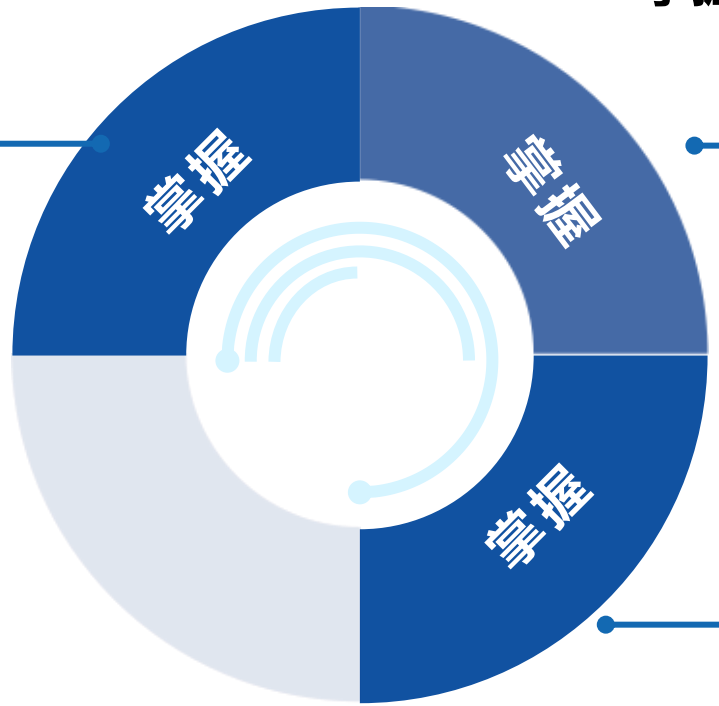
潍坊科技学院

- 字符串定义方法
- 格式化字符串
- 字符串的常见操作

学习目标

1 掌握 字符串的定义方法

掌握 字符串格式化 2



熟悉 字符串的常见操作 3

目录页

目录页

4.1 字符串介绍

思考：

什么是字符串？

4.1 字符串介绍

字符串是由字母、符号或者数字组成的字符序列。

4.1 字符串介绍

Python支持使用单引号、双引号和三引号定义字符串，其中单引号和双引号通常用于定义单行字符串，三引号通常用于定义多行字符串。

使用单引号	使用双引号	使用三引号
'hello itcast'	"hello itcast"	"""my name is itcast my name is itcast"""

4.1 字符串介绍

Python使用反斜杠“\”转义。例如，在字符串中的引号前添加“\”，此时Python解释器会将“\”之后的引号视为解释为一个普通字符，而非特殊符号。

示例

```
print('let\'s learn Python')
```

结果

```
let's learn Python
```

多学一招：转义字符

一些普通字符与反斜杠组合后将失去原有意义，产生新的含义。类似这样的由“\”和而成的、具有特殊意义的字符就是转义字符。转移字符通常用于表示一些无法显示的字符，例如空格、回车等。

转义字符	功能说明
\b	退格（Backspace）
\n	换行
\v	纵向制表符
\t	横向制表符
\r	回车

多学一招：转义字符

在一段字符串中如果包含多个转义字符，但又不希望转义字符产生作用，此时可以使用原始字符串。原始字符串即在字符串开始的引号之前添加r或R，使它成为原始字符串。

```
print(r'转义字符中:\n表示换行;\r表示回车;\b表示退格')
```

示例

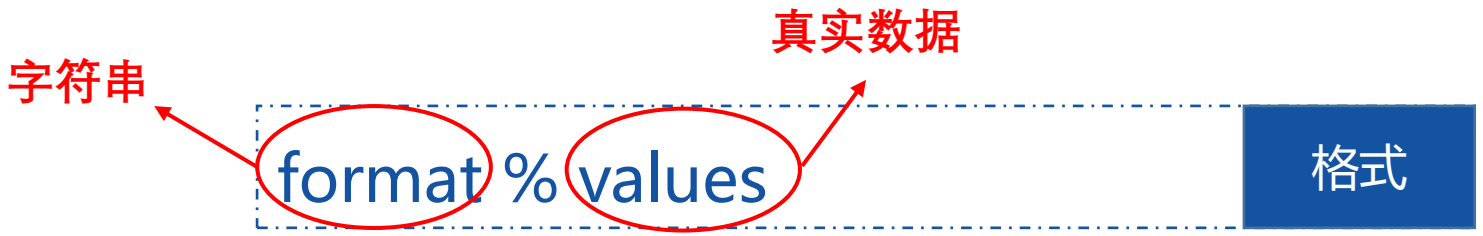
```
转义字符中:\n表示换行;\r表示回车;\b表示退格
```

结果

目录页

4.2.1 使用%格式化字符串

字符串具有一种特殊的内置操作，它可以使用%进行格式化。



- format表示一个字符串，该字符串中包含单个或多个为真实数据占位的格式符；
- values表示单个或多个真实数据；
- %代表执行格式化操作，即将format中的格式符替换为values。

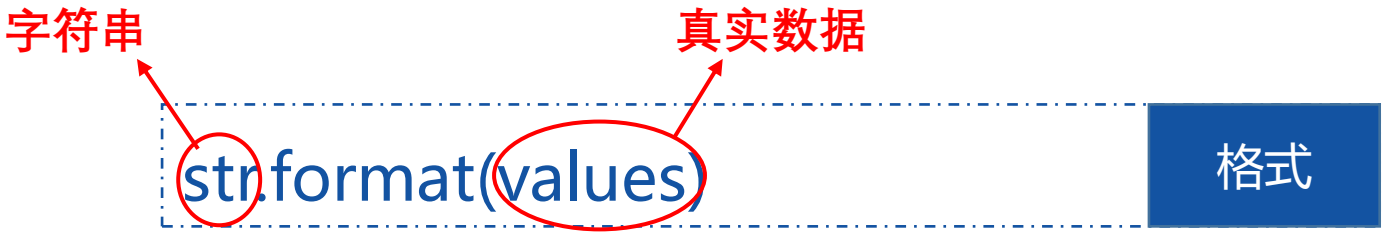
4.2.1 使用%格式化字符串

不同的格式符为不同类型的数据预留位置，常见的格式符如下所示。

格式符	格式说明
%c	将对应的数据格式化为字符
%s	将对应的数据格式化为字符串
%d	将对应的数据格式化为整数
%u	将对应的数据格式化为无符号整型
%o	将对应的数据格式化为无符号八进制数
%x	将对应的数据格式化为无符号十六进制数
%f	将对应的数据格式化为浮点数，可指定小数点后的精度（默认保留6位小数）

4.2.2 使用format()方法格式化字符串

虽然使用%可以对字符串进行格式化，但是这种方式并不是很直观，一旦开发人员遗漏了替换数据或选择了不匹配的格式符，就会导致字符串格式化失败。为了能更直观、便捷地格式化字符串，Python为字符串提供了一个格式化方法format()。



- str表示需要被格式化的字符串，字符串中包含单个或多个为真实数据占位的符号{}；
- values表示单个或多个待替换的真实数据，多个数据之间以逗号分隔。

4.2.2 使用format()方法格式化字符串

字符串中可以包含多个{}符号，字符串被格式化时Python解释器默认会按从左到右的顺序将{}逐个替换为真实的数据

示例

```
name = '张倩'
age = 25
string = "姓名：{}\n年龄：{}"
print(string.format(name, age))
```

结果

```
姓名：张倩
年龄：25
```

4.2.2 使用format()方法格式化字符串

字符串的{}中可以明确地指定编号，格式化字符串时解释器会按编号取values中相应位置的值替换{}，values中元素的索引从0开始。

示例

```
name = '张倩'
age = 25
string = "姓名：{1}\n年龄：{0}"
print(string.format(age, name))
```

结果

```
姓名：张倩
年龄：25
```

4.2.2 使用format()方法格式化字符串

字符串的{}中可以指定名称，字符串在被格式化时Python解释器会按真实数据绑定的名称替换{}中的变量。

示例

```
name = '张倩'
age = 25
weight = 65
string = "姓名：{name}\n年龄：{age}\n体重：{weight}kg"
print(string.format(name=name, weight=weight, age=age))
```

结果

```
姓名：张倩
年龄：25
体重：65kg
```

4.2.2 使用format()方法格式化字符串

字符串中的{}可以指定替换的浮点型数据的精度，浮点型数据在被格式化时会按指定的精度进行替换。

示例

```
points = 19
total = 22
print('所占百分比: {:.2%}'.format(points/total))
```

结果

```
所占百分比: 86.36%
```

4.2.3 使用f-string格式化字符串

f-string提供了一种更为简洁的格式化字符串的方式，它在形式上以f或F引领字符串，在字符串中使用“{变量名}”标明被替换的真实数据和其所在位置。

f('{变量名}') 或F('{变量名}')	格式

目录页

4.3.1 进制转换

十进制是实际应用中最常使用的计数方式，除此之外，还可以采用二进制、八进制或十六进制计数。

本实例要求编写代码，实现将用户输入的十进制整数转换为指定进制的功能。

4.3.2 文本进度条

进度条一般以图形的方式显示已完成任务量和未完成任务量，并以动态文字的方式显示任务的完成度。

本实例要求编写程序，实现如图所示的文本进度条。

```
==========开始下载==========
64%[**********************************.....................]

==========开始下载==========
100%[**************************************************]
==========下载完成==========
```

目录页

4.4.1 字符串的查找与替换

Python中提供了实现字符串查找操作的find()方法，该方法可查找字符串中是否包含子串，若包含则返回子串首次出现的位置，否则返回-1。

格式

```
str.find(sub[, start[, end]])
```

- sub：指定要查找的子串。
- start：开始索引，默认为0。
- end：结束索引，默认为字符串的长度。

示例

```
word = 't'
string = 'Python'
result = string.find(word)
print(result)
```

结果

```
2
```

4.4.1 字符串的查找与替换

Python中提供了实现字符串替换操作的replace()方法，该方法可将当前字符串中的指定子串替换成新的子串，并返回替换后的新字符串。

格式

```
str.replace(old, new[, count])
```

- old：被替换的旧子串。
- new：替换旧子串的新子串。
- count：表示替换旧字符串的次数。

示例

```
string = 'He said, "you have to go forward, ' \
'Then turn left, Then go forward, and Then turn right."'
# 指定替换两次
new_string = string.replace("Then", "then",2)
print(new_string)
```

结果

```
He said, "you have to go forward, then turn l
 then go forward, and Then turn right."
```

4.4.2 字符串的分隔与拼接

split()方法可以按照指定分隔符对字符串进行分割，该方法会返回由分割后的子串组成的列表。

格式

```
str.split(sep=None, maxsplit=-1)
```

- sep：分隔符，默认为空字符。
- maxsplit：分割次数，默认值为-1，表示不限制分割次数。

示例

```
string= "Hello, my name is Wang Hong"
# 以空格作为分割符，并分割2次
print(string.split(' ', 2))
```

结果

```
['Hello,', 'my', 'name is Wang Hong']
```

4.4.2 字符串的分隔与拼接

join()方法使用指定的字符连接字符串并生成一个新的字符串。join()方法的语法格式如下。

格式

```
str.join(iterable)
```

- iterable：表示连接字符串的字符。

示例

```
symbol = '*'
world = 'Python'
print(symbol.join(world))
```

结果

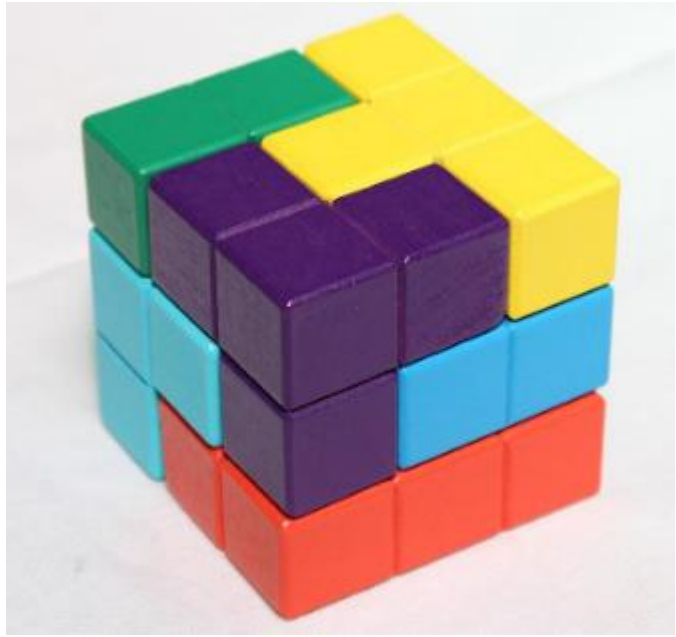
```
P*y*t*h*o*n
```

4.4.2 字符串的分隔与拼接

Python还可以使用运算符“+”拼接字符串。

“Py” + “thon” → “Python”

示 例



4.4.3 删除字符串的指定字符

字符串中可能会包含一些无用的字符（如空格），在处理字符串之前往往需要先删除这些无用的字符。Python中的strip()、lstrip()和rstrip()方法可以删除字符串中的指定字符。

方法	语法格式	功能说明
strip()	str.strip([chars])	移除字符串头尾指定的字符
lstip()	str.lstrip([chars])	移除字符串头部指定的字符
rstrip()	str.rstrip([chars])	移除字符串尾部指定的字符

4.4.4 删除字符串的指定字符

在特定情况下会对英文单词的大小写形式进行要求，表示特殊简称时全字母大写，如CBA；表示月份、周日、节假日时每个单词首字母大写，如Monday。Python中支持字母大小写转换的方法有upper()、lower()、capitalize()和title()。

方法	功能说明
upper()	将字符串中的小写字母全部转换为大写字母
lower()	将字符串中的大写字母全部转换为小写字母
capitalize()	将字符串中第一个字母转换为大写形式
title()	将字符串中每个单词的首字母转换为大写形式

4.4.5 字符串对齐

在使用Word处理文档时可能需要对文档的格式进行调整，如标题居中显示、左对齐、右对齐等。Python提供了center()、ljust()、rjust()这3个方法来设置字符串的对齐方式。

方法	语法格式	功能说明
center()	str.center(width[,fillchar])	返回长度为 width 的字符串，原字符串居中显示
ljust()	str.ljust(width[,fillchar])	返回长度为 width 的字符串，原字符串左对齐显示
rjust()	str.rjust(width[,fillchar])	返回长度为 width 的字符串，原字符串右对齐显示

目录页

4.5.1 敏感词替换

敏感词通常是指带有敏感政治倾向、暴力倾向、不健康色彩的词或不文明的词语，对于文章中出现的敏感词常用的处理方法是使用特殊符号（如“*”）对敏感词进行替换。

本实例要求编写代码，实现具有替换敏感词功能的程序。

4.5.2 文字排版工具

文字排版工具是一款强大的文章自动排版工具，它会将文字按现代汉语习惯及发表出版要求进行规范编排。文字排版工具一般具备删除空格、英文标点替换、英文单词大写功能，

本实例要求编写代码，实现具有上述功能的文字排版工具。

4.6本章小结

本章主要讲解了Python字符串的相关知识，包括什么是字符串、格式化字符串、字符串的常见操作，并结合实训案例演示了字符串的使用。

通过本章的学习，希望读者能够掌握字符串的使用。