

第3章 基本数据类型

本章学习目标

- 理解数字类型、字符串类型和逻辑类型相关的概念
- 掌握数字类型、字符串类型和逻辑类型的表示用法
- 掌握数字类型、字符串类型和逻辑类型的运算操作符、内置函数和内置方法

本章依次介绍 Python 的 3 种基本数据类型及其操作,即数字类型及其操作、字符串类型及其操作和逻辑类型及其操作,期间穿插介绍 4 个应用简单数据类型的案例。

3.1 数字类型及其操作

数字类型及其操作主要包括以下 5 方面内容。

- ① 数字类型的概念;
- ② 数字类型的表示;
- ③ 数字类型的运算操作符;
- ④ 内置的数字类型的函数;
- ⑤ 数字类型的标准库——math 库。

3.1.1 数字类型的概念

Python 的数字类型大致对应数学中“数”的概念,包括整数、浮点数和复数 3 种类型。其中,整数的概念和数学中的“整数”略有不同,Python 的整数除十进制整数外,还包括二进制整数、八进制整数和十六进制整数。浮点数和复数分别对应数学中的实数和复数。

3.1.2 数字类型的表示

数字类型包括整数、浮点数和复数,下面分别介绍每种数字类型的表示方法。

1. 整数的表示

Python 中,十进制、二进制、八进制和十六进制整数的表示方法各不相同。

- 十进制整数表示

十进制整数不需要引导符。书写十进制整数时,直接书写正负号和十进制数字序列即可。例如:1080、99 和 -217 都是 Python 合法的十进制整数。

- 二进制整数表示

二进制整数的引导符是“0b”或“0B”。其中,“b”是英文二进制“binary”的首字母,出现在引导符中的时候,字母“b”大、小写完全等价。书写二进制整数时,数字序列中只可包含“0”和“1”两个数字。例如:0b010 和 -0B101 都是 Python 合法的二进制整数。



- 八进制整数表示

八进制整数的引导符是“0o”或“0O”。其中，“o”是英文八进制“octal”的头字母，出现在引导符中的时候，字母“o”大、小写完全等价。书写八进制整数时，数字序列中只可包含“0”、“1”、“2”、“3”、“4”、“5”、“6”和“7”这 8 个数字。例如：0o123 和 0O456 都是 Python 合法的八进制整数。

- 十六进制整数表示

十六进制整数的引导符是“0x”或“0X”。其中，“x”表示英文十六进制“hexadecimal”，出现在引导符中的时候，“x”大、小写完全等价。书写十六进制整数时，数字序列中只可以包含“0”、“1”、“2”、“3”、“4”、“5”、“6”、“7”、“8”、“9”、“a”、“b”、“c”、“d”、“e”和“f”。其中，“a”表示 10，“b”表示 11，……，以此类推，“f”表示 15。另外，十六进制整数中，“a”、“b”、“c”、“d”、“e”和“f”大、小写等价。例如：0x1a3 和 0X9f 都是 Python 合法的十六进制整数。

Python 整数类型理论上的取值范围是 $[-\infty, \infty]$ ，实际上，Python 整数的表示范围受到程序员计算机内存的限制，内存配置不同的机器取值范围也不同。

2. 浮点数的表示

Python 中浮点数的表示方法有两种：十进制法和类科学计数法。

- 十进制法

十进制法的特点是数据中必须包含小数点。例如：0.0、-77.、1000. 和 -2.17 都是十进制法表示的浮点数。

- 类科学计数法

类科学计数法把一个浮点数分解成 a 与 10 的 n 次幂相乘的形式，其中 a 称为尾数， n 称为指数。用类科学计数法书写浮点数时，先写 a ，然后写“e”或“E”，最后写 n 。例如：68e4、4.4e-3 和 8.6E5 都是类科学计数法表示的浮点数。

Python 中，浮点数的表示范围仅受程序员计算机内存的限制，因此浮点数表示范围大，应用广泛，但 Python 的浮点数运算存在误差（一般小于 10^{-16} ），典型的例子如 0.1 和 0.2 的和不等于 0.3，以下是 Python 中求 0.1 和 0.2 的和的代码及其运行结果。

```
>>> 0.1 + 0.2
0.30000000000000004
```

因此，如果数据处理不允许有误差，就应该转换为整数进行运算，运算结束后再处理小数点的问题。

3. 复数的表示

Python 中，复数的表示和数学中复数的表示相近，写作 $a+bj$ 或 $a+bj$ 。其中， a 是实部， b 是虚部， a 和 b 都是浮点数类型。例如：26.2+3j、-5.6+7j 和 1.25e-6+5.68e+9j 都是 Python 合法的复数。

书写复数时，等于整数的浮点数实部或虚部可以略写为整数形式。例如：12.3+4j 和 -5.6+7j 都是 Python 合法的复数。但在计算机内部，虚部 4 和 7 仍然作为浮点数处理。

任何一个 Python 复数都有两个属性：real 和 imag。对于一个特定的复数 $z=a+bj$ ， z .real 的返回值为实部 a ， z .imag 的返回值为虚部 b 。

例如：以下是求一个复数的实部和虚部的代码及其运行结果。

```
>>> z = 1.23e-4 + 5.6e + 89j
>>> z.real
0.000123
>>> z.imag
5.6e + 89
```

3.1.3 数字类型的运算操作符

Python 数字类型的运算操作符主要包括两大类：基本运算操作符和增强运算操作符。

1. 基本运算操作符

Python 数字类型的基本运算操作符主要有 9 个, 详见表 3.1。

表 3.1 Python 数字类型的基本运算操作符

运算表达式	运 算 结 果	结 合 性
$x**y$	x 的 y 次幂, 即 x^y	右结合
$+x$	求 x 本身的正数, 即 $x * 1$	右结合
$-x$	求 x 的负数, 即 $x * (-1)$	右结合
$x * y$	x 与 y 之积	左结合
x / y	x 除以 y 的商	左结合
$x // y$	x 除以 y 的整商	左结合
$x \% y$	x 除以 y 的余数, 也称为模运算	左结合
$x + y$	x 与 y 之和	左结合
$x - y$	x 与 y 之差	左结合

以上 9 个基本运算操作符中, 求正数和求负数只需要 1 个操作数, 是一元运算操作符, 其余 7 个运算操作符都是二元运算操作符。

对于这 7 个二元运算操作符($**$ 、 $*$ 、 $/$ 、 $//$ 、 $\%$ 、 $+$ 、 $-$), Python 都提供了相应的增强运算操作符。

2. 增强运算操作符

Python 的增强运算操作符共 7 个, 分别是 $**=$ 、 $*=$ 、 $/=$ 、 $//=$ 、 $\%=$ 、 $+=$ 和 $-=$ 。增强运算操作的过程是先将两个操作数做运算, 然后创建操作符左边的变量, 并把运算结果赋值给操作符左边的变量。

例如：以下代码展示了增强运算操作符“ $**=$ ”的运算过程。

```
>>> y = 3
>>> x = 2
>>> # 计算 y 的 x 次方得到 9, 然后创建变量 y, 并把 9 赋值给变量 y
>>> y**= x
>>> y
9
```



3. 不同运算的优先级

数字类型的运算操作符很多,同一表达式中出现 2 个以上运算操作符时,需要考虑运算的优先顺序。数字类型运算的优先顺序详见表 3.2。

表 3.2 数字类型运算的优先顺序

运算操作符	运 算 描 述
**	乘方(最高优先级)
~	按位取反
+, -	求正和取负
*, /, %, //	乘、除、取余、求整商
+, -	加法、减法
>>, <<	右移位、左移位
&	位与
^	位或、位异或
=, %=, /=, //=, -=, +=, *=, **=	创建变量及赋值运算、增强运算

其中,运算的优先顺序对应表中行的先后顺序,即排在第一行的乘方运算优先级最高,排在第二行的按位取反运算优先级次之,……,以此类推,排在最后一行的创建变量及赋值运算、增强运算优先级最低。

同一行的多个运算优先级相同,表达式中哪个运算先出现哪个运算优先执行。

4. 运算结果的类型

数字类型的数据进行运算后,运算结果的数据类型遵循以下 3 个原则:

- ① 整数和浮点数运算,结果是浮点数;
- ② 整数或浮点数与复数运算,结果是复数;
- ③ 相同类型的数据运算,结果一般保持类型不变;但整数除法例外,结果是浮点数。

3.1.4 内置的数字类型的函数

所谓内置的函数,指的是定义语句集成在 Python 解释器里,Python 使用者可以直接调用的函数。Python 主要提供了 13 个内置的数字类型的函数,这些函数可以分为 3 大类:数值运算函数、类型转换函数和类型判断函数。

1. 数值运算函数

数值运算函数见表 3.3。

表 3.3 数值运算函数

函 数	描 述
abs(x)	参数是整数或实数时,返回值为绝对值; 参数是复数时,返回值是该复数和其共轭复数的积的平方根
divmod(x, y)	返回值是元组($x//y$, $x\%y$)



续表

函 数	描 述
<code>pow(x, y[, z])</code>	返回值是 $(x**y)\%z$, 其中 z 是可选参数, z 缺省时, 函数的返回值为 $x**y$; 如果同时给 3 个参数传参, 要求 3 个实参必须都取整数
<code>round(x[, ndigits])</code>	返回值是对 x 精确到 $ndigits$ 位小数的取值, 其中 $ndigits$ 是可选参数, 其默认值是 0
<code>max(x₁, x₂, ..., x_n)</code>	返回 $x_1, x_2, \dots, x_n$ 中的最大值或一个可迭代类型数据的最大元素
<code>min(x₁, x₂, ..., x_n)</code>	返回 $x_1, x_2, \dots, x_n$ 中的最小值或一个可迭代类型数据的最小元素

其中, 调用 `round()` 函数对数据取整时, 函数按照 4 舍 6 入 5 成双的原则对 x 取整。
例如: 以下代码及其运行结果说明了 4 舍 6 入 5 成双的原则。

```
>>> # 4 舍
>>> round(2.4)
2
>>> # 6 入
>>> round(2.6)
3
>>> # 2 是双数, 按照 5 成双原则结果为 2
>>> round(2.5)
2
>>> # 4 是双数, 按照 5 成双原则结果为 4
>>> round(3.5)
4
```

2. 类型转换函数

类型转换函数见表 3.4。

表 3.4 类型转换函数

函 数	描 述
<code>bin(x)</code>	返回 x 对应的二进制整数的字符串
<code>oct(x)</code>	返回 x 对应的八进制整数的字符串
<code>hex(x)</code>	返回 x 对应的十六进制整数的字符串
<code>int(x)</code>	返回 x 对应的十进制整数。参数 x 可以是浮点数, 仅包含数字的数字字符串、二、八、十六进制数
<code>float(x)</code>	返回 x 对应的浮点数。参数 x 可以是整数、整数字符串和浮点数字字符串
<code>complex(re[, im])</code>	返回一个复数, 实部为 re , 虚部为 im 。参数 re 可以是整数、浮点数、整数字符串、浮点数字字符串和复数字字符串; 参数 im 可以是整数或浮点数, 但不能为字符串

3. 类型判断函数

Python 内置的类型判断函数是 `type(x)`, 其返回值是 x 的类型, 参数 x 可以是 Python 支持的所有类型的数据。



例如：以下是两个调用函数 `type()` 的例子。

```
>>> type(3e2)
<class 'float'>
>>> type(abs(3+4j))
<class 'float'>
```

3.1.5 标准库 2: math 库的使用方法

操作数字类型除可以使用运算操作符、内置的数字类型的函数外,还常用数字类型的标准库——math 库。

标准库由负责 Python 开发和维护的组织 PSF 开发,随 Python 解释器一起发布,并且随 Python 的安装同步安装于计算机中。因此,math 库不需要安装即可使用。

math 库中主要定义了 5 个数学常数和 44 个数值运算函数。

1. math 库的常数

math 库的常数详见表 3.5。

表 3.5 math 库的常数

常 数	数学表示	描 述
math.pi	π	圆周率,值为 3.141592653589793
math.e	e	自然常数 e,值为 2.718281828459045
math.inf	∞	正无穷大
math.nan		浮点数类型的一个特殊数据,表示“不是数”
math.tau	τ	2π ,值为 6.283185307179586

2. math 库的函数

math 库总计定义了 44 个函数,分为数值计算函数、幂对数函数、三角双曲函数和高等特殊函数 4 大类。

- 数值计算函数

math 库的数值计算函数共 16 个,详见表 3.6。

表 3.6 math 库的数值计算函数

函 数	数学表示	描 述
math.fabs(x)	$ x $	返回 x 的绝对值,其中 x 是整数或浮点数
math.fmod(x, y)	$x \% y$	返回 x 与 y 的模
math.fsum([x, y, ...])	$x + y + \dots$	浮点数精确求和
math.ceil(x)	$\lceil x \rceil$	向上取整,返回不小于 x 的最小整数
math.floor(x)	$\lfloor x \rfloor$	向下取整,返回不大于 x 的最大整数
math.factorial(x)	$x!$	返回 x 的阶乘,如果 x 是小数或负数,就返回 ValueError

续表

函 数	数学表示	描 述
math.gcd(a, b)		返回 a 与 b 的最大公约数
math.frexp(x)	$x = m * 2^e$	返回 (m, e) , m 和 e 分别是 x 的二进制表示的尾数和指数。 x 的取值为 0 时, 返回 $(0.0, 0)$
math.ldexp(x, i)	$x * 2^i$	返回 $x * 2^i$ 的值, math.frexp(x) 函数的逆运算
math.modf(x)		以元组类型返回 x 的小数部分和浮点数形式的整数部分
math.trunc(x)		返回 x 的整数部分
math.copysign(x, y)	$ x * y /y$	返回用数值 y 的正负号替换数值 x 的正负号得到的数值
math.isclose(a, b)		比较 a 和 b 的相似性, 返回 True 或 False
math.isfinite(x)		当 x 不是无穷大时, 返回 True; 否则返回 False
math.isinf(x)		当 x 为正数无穷大或负数无穷大时, 返回 True; 否则返回 False
math.isnan(x)		当 x 取值 nan 时, 返回 True; 否则返回 False

• 幂对数函数

math 库的幂对数函数共 8 个, 详见表 3.7。

表 3.7 math 库的幂对数函数

函 数	数学表示	描 述
math.pow(x, y)	x^y	返回 x 的 y 次幂
math.exp(x)	e^x	返回 e 的 x 次幂, 其中 e 是自然常数
math.expml(x)	$e^x - 1$	返回 e 的 x 次幂减 1
math.sqrt(x)	$\sqrt{x}$	返回 x 的平方根
math.log(x[, base])	$\log_{\text{base}} x$	返回以 base 为底的 x 的对数值, base 缺省时, 返回 x 的自然对数值
math.log1p(x)	$\ln(1+x)$	返回 $1+x$ 的自然对数值
math.log2(x)	$\log_2 x$	返回 x 的以 2 为底的对数值
math.log10(x)	$\log_{10} x$	返回 x 的以 10 为底的对数值

• 三角双曲函数

math 库的三角双曲函数共 16 个, 详见表 3.8。

表 3.8 math 库的三角双曲函数

函 数	数学表示	描 述
math.degree(x)		返回弧度 x 对应的角度值
math.radians(x)		返回角度 x 对应的弧度值
math.hypot(x, y)	$\sqrt{x^2+y^2}$	返回坐标为 (x, y) 的点到原点 $(0, 0)$ 的距离
math.sin(x)	$\sin x$	返回 x 的正弦函数值, x 是弧度值



续表

函 数	数学表示	描 述
math.cos(x)	$\cos x$	返回 x 的余弦函数值, x 是弧度值
math.tan(x)	$\tan x$	返回 x 的正切函数值, x 是弧度值
math.asin(x)	$\arcsin x$	返回 x 的反正弦函数值, x 是弧度值
math.acos(x)	$\arccos x$	返回 x 的反余弦函数值, x 是弧度值
math.atan(x)	$\arctan x$	返回 x 的反正切函数值, x 是弧度值
math.atan2(y, x)	$\arctan y/x$	返回 y/x 的反正切函数值, x, y 是弧度值
math.sinh(x)	$\sinh x$	返回 x 的双曲正弦函数值
math.cosh(x)	$\cosh x$	返回 x 的双曲余弦函数值
math.tanh(x)	$\tanh x$	返回 x 的双曲正切函数值
math.asinh(x)	$\operatorname{arcsinh} x$	返回 x 的反双曲正弦函数值
math.acosh(x)	$\operatorname{arccosh} x$	返回 x 的反双曲余弦函数值
math.atanh(x)	$\operatorname{arctanh} x$	返回 x 的反双曲正切函数值

• 高等特殊函数

math 库的高等特殊函数共 4 个, 详见表 3.9。

表 3.9 math 库的高等特殊函数

函 数	数学表示	描 述
math.erf(x)	$\frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$	高斯误差函数, 用于概率论、统计学等领域
math.erfc(x)	$\frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt$	余补误差函数, $\text{math.erfc}(x) = 1 - \text{math.erf}(x)$
math.gamma(x)	$\int_0^\infty x^{t-1} e^{-x} dx$	伽马 (Gamma) 函数, 也叫欧拉第二积分函数
math.lgamma(x)	$\ln(\text{gamma}(x))$	伽马函数的自然对数

3.2 案例 3: 计算基本统计量

通常, 计算基本统计量需要完成以下两项工作。

- ① 学习基本统计量的计算公式;
- ② 编写基本统计量计算程序。

3.2.1 基本统计量的计算公式

计算基本统计量主要用到两个公式: 均值公式和标准差公式。



1. 均值公式

假设一组数据表示为 $S = s_0, s_1, \dots, s_{n-1}$, 均值公式参见式(3.1)。

$$\text{mean} = \left(\sum_{i=0}^{n-1} s_i \right) / n \quad (3.1)$$

2. 标准差公式

假设一组数据表示为 $S = s_0, s_1, \dots, s_{n-1}$, 标准差公式参见式(3.2)。

$$\text{dev} = \sqrt{\frac{\sum_{i=0}^{n-1} (s_i - \text{mean})^2}{n - 1}} \quad (3.2)$$

3.2.2 基本统计量计算程序

1. 程序说明/readme

程序运行后, 要求用户以逗号为分隔符输入 5 个数; 之后, 程序按照公式求均值和标准差; 最后, 输出最大值、最小值、均值和标准差 4 个值。

2. 源程序

```
# Input
a, b, c, d, e = eval(input("请输入用逗号分隔的 5 个数:"))

# Process
mean = (a+b+c+d+e) / 5
dev = (((a-mean)**2 + (b-mean)**2 + (c-mean)**2 + (d-mean)**2 + (e-mean)**2) / 4)**0.5

# Output
print("最大值为:", max(a, b, c, d, e))
print("最小值为:", min(a, b, c, d, e))
print("平均值为:", mean)
print("标准差为:", dev)
```

3. 运行示例

程序运行后, 屏幕显示如下。

请输入用逗号分隔的 5 个数: 1, 2, 3, 4, 5

最大值为: 5

最小值为: 1

平均值为: 3.0

标准差为: 1.5811388300841898

3.3 字符串类型及其操作

字符串类型及其操作主要包括以下 6 方面内容。

- ① 字符串类型的概念;
- ② 字符串类型的表示;



- ③ 字符串类型的运算操作符；
- ④ 内置的字符串处理函数；
- ⑤ 案例 4：查找化合物 ID 及水溶性值；
- ⑥ 内置的字符串处理方法。

3.3.1 字符串类型的概念

计算机由早期的主要用于数值计算,到现在更多地用于信息处理,字符串类型发挥了巨大的作用。Python 程序中,字符串类型的应用极其广泛,如药品的名称、种类、包装单位、中国药品电子监管码,以及化合物的序号、分类、InChIKey(化合物的身份证)、SMILES 码等,在 Python 中对应的数据类型都是字符串类型。

字符串类型中有一类特殊的字符串,其字符都是数字字符,这种字符串叫数字字符串。学号和大多数的身份证号就是常见的数字字符串。数字字符串不同于数字类型的数据,前者仅用于标识,不用于加、减、乘、除等数字类型的运算。注意区分数字字符串和数字类型的数据。

Python 字符串类型的数据是由一对引号作为定界符、一组字符按特定顺序构成的序列。字符串类型的数据有两个显著特征,具体如下。

① 字符串由字符组成,其中的字符既可以是中英文文字,也可以是数字字符、空格或其他特殊符号。例如:"生石膏 15~30g(先煎)、" " 01"和"A01\n"是 3 个 Python 合法的字符串类型的数据;

② 字符串中的字符是有序的。Python 支持两种字符串索引:正向索引和反向索引,如图 3.1 所示。

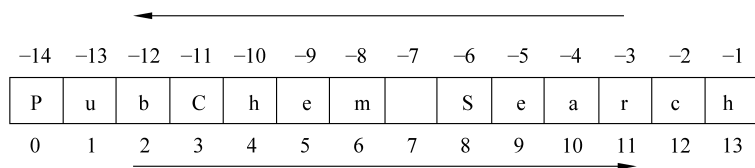


图 3.1 Python 的字符串索引

其中,正向索引从 0 开始编号,图 3.1 所示字符串共 14 个字符,最后一个字符的索引号为 13;反向索引从 -1 开始编号,最后一个字符的索引号为 -14。每个字符都有两个索引号:正向索引号和反向索引号,如图 3.1 中的“空格”字符的索引号是 7 和 -7。

3.3.2 字符串类型的表示

在 Python 中书写字符串类型的数据时,需要用一对单引号'或者一对双引号",或者一对三引号(三个单引号或三个双引号构成 1 个三引号)作为定界符,括起字符串中的所有字符。其中,单引号和双引号的作用相同,三引号字符串常用作多行注释语句。

例如,一个程序中包含以下 6 行代码,6 行代码总体构成 1 个多行注释语句:

```
''' 清肺排毒汤处方组成:  
麻黄 9g、炙甘草 6g、杏仁 9g、生石膏 15~ 30g (先煎)、
```



```
桂枝 9g、泽泻 9g、猪苓 9g、白术 9g、茯苓 15g、  
柴胡 16g、黄芩 6g、姜半夏 9g、生姜 9g、紫菀 9g、  
冬花 9g、射干 9g、细辛 6g、山药 12g、枳实 6g、陈皮 6g、藿香 9g  
'''
```

字符串类型的数据可以用于创建变量并为变量赋值,也可以作为1个数据项出现在表达式中,这一点和数字类型的数据相同。

例如,运行以下语句后创建变量s,并为s赋一个字符串类型的数据。

```
s = "麻黄 9g、炙甘草 6g、杏仁 9g、生石膏 15~30g(先煎)、桂枝 9g、泽泻 9g、\n猪苓 9g、白术 9g、茯苓 15g、柴胡 16g、黄芩 6g、姜半夏 9g、生姜 9g、紫菀 9g、\n冬花 9g、射干 9g、细辛 6g、山药 12g、枳实 6g、陈皮 6g、藿香 9g"
```

注意: 由于这个字符串是处方信息,包含的字符个数较多,书写时采用Python续行符“\”把一条长语句分写在3行,这里的续行符“\”并不是s字符串中的字符。

输出字符串类型的数据时需要注意以下3个问题。

① 调用print()函数输出字符串类型的数据时,作为定界符的引号不显示在屏幕上。例如,print("hi")语句运行后,屏幕上只显示“h”和“i”两个字符,引号不会出现在屏幕上。

② 书写内含引号的字符串时,单引号、双引号和三引号要配合使用,以确保作为定界符的引号和字符串内含的引号不同。例如,拟在屏幕上输出字符串“She said “Hi, Jacky”, then she smiled.”,正确的Python语句为

```
print('She said "Hi, Jacky", then she smiled.')
```

或者

```
print(''''She said "Hi, Jacky", then she smiled.'''')
```

其中,由于双引号是字符串内含的字符,因此定界符引号只能采用单引号或三引号。

③ 转义符问题。

Python中,反斜杠“\”用作转义符,即反斜杠“\”用于和特定的字符组合构成新的字符,这些新字符往往具有特殊的含义。例如:“\t”是1个字符,表示制表符;“\n”表示换行符,“\r”表示将光标定位到当前行的行首,“\b”表示将光标回退1位,“\0”表示其后续的字符不输出等。

如果字符串数据内含反斜杠作为正常字符,而不是用作转义符,就需要书写两个反斜杠表示1个反斜杠字符。

例如:以下代码中,x字符串中的反斜杠用作转义符;y字符串中的反斜杠是正常字符。

```
>>> # \t 表示制表符  
>>> x = "c:\tese"  
>>> print(x)  
c:  ese  
>>> # \\表示 1 个反斜杠字符  
>>> y = "c:\\tese"  
>>> print(y)  
c:\tese
```



3.3.3 字符串类型的运算操作符

Python 提供了 5 个字符串类型的运算操作符(见表 3.10),分别是连接、复制、成员运算、按索引号取字符和切片。

表 3.10 字符串类型的运算操作符

运算表达式	描 述
<code>x+y</code>	连接两个字符串 <code>x</code> 与 <code>y</code> ,返回得到的新字符串
<code>x * n</code> 或 <code>n * x</code>	复制 <code>n</code> 次字符串 <code>x</code> ,返回得到的新字符串
<code>x in s</code>	如果 <code>x</code> 是 <code>s</code> 的子串,则返回 <code>True</code> ,否则返回 <code>False</code>
<code>str[i]</code>	按索引号取字符,返回索引号为 <code>i</code> 的字符
<code>str[start:end[:step]]</code>	字符串切片

1. 连接

Python 的连接运算符是“+”,连接运算返回将两个字符串连接而成的新字符串。

例如: 以下代码将 3 个字符串进行连接,返回由 3 个字符串连接而成的新字符串。

```
>>> "Xu" + "chang" + "qing"
'Xuchangqing'
```

2. 复制

Python 的复制运算符是“*”,复制运算返回字符串重复连接形成的新字符串。

例如: 以下代码返回一个字符串复制 3 次形成的新字符串。

```
>>> '紫苑' * 3
'紫苑紫苑紫苑'
```

3. 成员运算

Python 的成员运算符是“in”,用来判断一个子串是否在另一个字符串中,若在,则返回 `True`,否则返回 `False`。

例如: 以下代码使用成员运算判别“桂枝”和“人参”是否在字符串 `formula` 中。

```
>>> formula = "麻黄 9g、炙甘草 6g、杏仁 9g、桂枝 9g、泽泻 9g、\
猪苓 9g、白术 9g、茯苓 15g、柴胡 16g、黄芩 6g、姜半夏 9g、\
冬花 9g、射干 9g、细辛 6g、山药 12g、枳实 6g、陈皮 6g、藿香 9g"
>>> "桂枝" in formula
True
>>> "人参" in formula
False
```

4. 按索引号取字符

Python 采用 `str[i]` 实现按索引号取字符,其中,`str` 是字符串常量或变量,`i` 是索引号,



返回取得的字符。

例如：以下代码分别求字符串'PubChem Search'中索引号为5和索引号为-7的字符。

```
>>> 'PubChem Search'[5]
'e'
>>> 'PubChem Search'[-7]
''
```

5. 字符串切片

字符串切片通过两个索引号确定一个取值范围,并按照一定的步长在确定的范围内取字符,最后返回取得的所有字符构成的子串。

字符串切片的语法为

```
<string>[[start]:[end]:[step]]
```

其中,start和end对应两个索引号,用于确定取值范围;step是步长值,是非0整数。

字符串切片的运行过程主要包括两步:方向一致性判别和取字符。

- 方向一致性判别

切片操作首先根据step的值确定第1个方向,确定方法是:如果step大于0,方向为从左到右,反之方向为从右到左;然后,从start对应的字符开始扫描字符串,直到end对应的字符结束扫描,扫描行进的方向为第2个方向。若两个方向不一致,则切片运算返回空串;否则进入第二步,即取字符。

- 取字符

取字符的具体过程为:以start的值为索引号取第1个字符,以start+step的值为索引号取第2个字符,……,以此类推,以start+(i-1)*step的值为索引号取第i个字符,当索引号对应的字符达到或越过end对应的字符时,停止取字符,end对应的字符不取,返回之前取得的所有字符组成的字符串。

注意: start缺省表示开始取字符没有限制;end缺省表示结束取字符没有限制;step的默认值为1。

例如:以下代码展示了3次字符串切片运算的结果。

```
>>> s = 'PubChem Search'
>>> #以下 step 为 1,第 1 个方向为左-->右,字符 r 到字符 c 的方向为右-->左,结果为空串
>>> s[-3:3]
''
>>> #以下 step 为 1,第 1 个方向为左-->右;字符 c 到字符 r 的方向为左-->右,两个方向一致
>>> s[3:-3]
'Chem Sea'
>>> #step 为 -1,第 1 个方向为右-->左;开始、结束没限制,从右到左无限制取字符
>>> s[::-1]
'hcraeS mehCbuP'
```

其中,最后一次切片运算是常用的求字符串的逆序串的方法。



3.3.4 内置的字符串处理函数

Python 解释器主要提供了 4 个内置的字符串处理函数,参见表 3.11。

表 3.11 内置的字符串处理函数

函 数	描 述
len(x)	返回字符串 x 的字符个数
str(x)	返回任意类型 x 所对应的字符串形式
chr(x)	返回 Unicode 编码 x 对应的单字符
ord(x)	返回单字符对应的 Unicode 编码

其中,第 1 个函数 len(x)求字符串 x 的字符个数时,由于 Python 3 以 Unicode 字符为计数单位,因此英文字符和中文字符都算 1 个字符。另外,“\t”“\n”“\b”“\r”“\0”“\123”等特殊字符也算 1 个字符。

3.3.5 案例 4: 查找化合物 ID 及水溶性值

1. 程序说明/readme

从数据文件中读出的水溶性数据是 1 个长字符串,调用 print()函数展示在屏幕上如图 3.2所示。

其中,第 1 列是化合物的序号,第 2 列是化合物的 ID,第 3 列是化合物的水溶性值。本程序的功能为:用户从键盘输入 1 个化合物的序号,程序查找并输出相应化合物的 ID 及其水溶性值。

```
>>> print(solubilityTxt)
SN, IdentityNo, Solubility
01, LT-615-348, 0.019714
02, LT-771-215, 0.03072346
03, LT-771-216, 0.03174257
04, LT-323-560, 0.06074634
05, LT-619-512, 0.10491267
```

图 3.2 化合物水溶性数据展示

2. 源程序

```
# Input:
# 使用创建变量并赋值的语句替代读文件操作输入 solubilityTxt 的值
solubilityTxt = "SN, IdentityNo, Solubility\n01, LT-615-348, 0.019714\n\
02, LT-771-215, 0.03072346\n03, LT-771-216, 0.03174257\n\
04, LT-323-560, 0.06074634\n05, LT-619-512, 0.10491267"
# 创建变量 sn, 从键盘输入 1 个序号赋值给变量 sn
sn = input("Please input the serial number(01-05):")

# Process
# 序号为 int(sn) 的化合物的信息字符串, 其起始索引号为 pos=int(sn) * 25
pos = int(sn) * 25
id = solubilityTxt[pos+3:pos+13]
sValue = float(solubilityTxt[pos+14:pos+24])

# Output
# 输出变量 id 和 sValue 的值
print("the compound ID is:", id)
```



```
print("the solubility value is:",sValue)
```

3. 运行示例

程序运行后,屏幕显示如下。

```
Please input the serial number(01-05): 02
the compound ID is: LT-771-215
the solubility value is: 0.03072346
```

3.3.6 内置的字符串处理方法

所谓方法,通俗地讲就是打包在类定义中的函数,其调用语法和函数的调用语法有一点不同:即调用方法时,方法名前必须有“.”,“.”之前必须有类名、对象数据或对象变量名。内置的方法指的是定义语句集成在 Python 解释器里,Python 使用者可以直接调用的方法。

Python 解释器总计内置了 43 个字符串类型的方法。下面分别从常用的字符串方法和 str.format()方法两方面介绍。

1. 常用的字符串方法

常用的字符串方法有 17 个,见表 3.12。

表 3.12 常用的字符串方法

方 法	描 述
str.lower()	返回字符串 str 的副本,副本中的字符全部采用小写字母
str.upper()	返回字符串 str 的副本,副本中的字符全部大写
str.islower()	判别字符串 str 中的字符是否都是小写,若是,则返回 True;否则返回 False
str.isprintable()	判别字符串 str 中的字符是否都是可打印字符,若是,则返回 True;否则返回 False
str. isnumeric()	判别字符串 str 中的字符是否都是数字字符(包括中文的数字零、一、二、三等),若是,则返回 True;否则返回 False
str.isspace()	判别字符串 str 中的字符是否都是空白,若是,则返回 True;否则返回 False。 注意: Python 中的空白包括空格和"\r""\t""\n"等特殊字符
str.endswith(suffix[,start[,end]])	判别字符串切片 str[start: end] 是否以 suffix 的值结尾,若是,则返回 True;否则返回 False
str.startswith(prefix[,start[, end]])	判别字符串切片 str[start: end] 是否以 prefix 的值开头,若是,则返回 True,否则返回 False
str.split(sep=None, maxsplit=-1)	以 sep 的值为分隔符切分 str,得到若干子串,返回由这些子串组成的列表
str.count(sub[,start[,end]])	返回切片 str[start: end]中 sub 的值出现的次数
str.replace(old, new[, count])	返回 str 的副本,其中前 count 个 old 值的子串被替换为 new 值的子串。如果 count 缺省,则替换所有 old 值的子串



续表

方 法	描 述
<code>str.center(width[, fillchar])</code>	返回 <code>str</code> 值居中、总宽度为 <code>width</code> 值、两侧用 <code>fillchar</code> 值填充的新字符串
<code>str.strip([chars])</code>	返回 <code>str</code> 的副本,其中左右两侧的 <code>chars</code> 值的字符被删除
<code>str.zfill(width)</code>	返回 <code>str</code> 值左边补 0 后总宽度为 <code>width</code> 值的新字符串
<code>str.join(iterable)</code>	返回由 <code>str</code> 值连接 <code>iterable</code> 值中所有元素形成的新字符串
<code>str.find(sub[, start[, end]])</code>	在切片 <code>str[start: end]</code> 中查找 <code>sub</code> 值的子串,若找到,则返回第一次找到的 <code>sub</code> 值第 1 个字符的索引;若找不到,则返回 -1
<code>str.index(sub[, start[, end]])</code>	在切片 <code>str[start: end]</code> 中查找 <code>sub</code> 值的子串,若找到,则返回第一次找到的 <code>sub</code> 值第 1 个字符的索引;若找不到,则终止程序并报错

其中,相对常用的 7 个字符串方法应用的例子如下。

- `str.lower()` 方法的例子

```
>>> s1 = "Drug"
>>> s1.lower()
'drug'
```

- `str.isspace()` 方法的例子

```
>>> #Python 中的空白包括空格、\r、\t、\n 等特殊字符
>>> "\r \t \n".isspace()
True
>>> #空串不是空格串,空串中不包含任何字符
>>> "".isspace()
False
```

- `str.split()` 方法的例子

```
>>> tanreqing = "黄芩、熊胆粉、山羊角、金银花、连翘"
>>> herbs = tanreqing.split(",")
>>> herbs
['黄芩', '熊胆粉', '山羊角', '金银花', '连翘']
```

- `str.join(iterable)` 方法的例子

```
>>> #iterable 的元素必须是字符串类型
>>> herbs=['黄芩', '熊胆粉', '山羊角', '金银花', '连翘']
>>> ",".join(herbs)
'黄芩、熊胆粉、山羊角、金银花、连翘'
```



- str.replace()方法的例子

```
>>> tanreqing = "黄芩、熊胆粉、山羊角、金银花、连翘"
>>> #str.replace()方法返回新的字符串,原字符串不变
>>> tanreqing.replace("、", ",")
'黄芩,熊胆粉,山羊角,金银花,连翘'
>>> tanreqing.replace(", ", "")
'黄芩熊胆粉山羊角金银花连翘'
>>> tanreqing
'黄芩、熊胆粉、山羊角、金银花、连翘'
```

- str.strip()方法的例子

```
>>> #从键盘输入数据时,前后各输入一个空格
>>> CpId = input("please input a compound ID: ")
please input a compound ID: LT-615-348
>>> CpId
'LT-615-348 '
>>> #调用 CpId.strip()方法去除字符串前后的空格,用返回值新串再次创建 CpId 并赋值新串
>>> CpId = CpId.strip()
>>> CpId
'LT-615-348'
```

- str.find()方法的例子

```
>>> formula = "麻黄 9g、炙甘草 6g、杏仁 9g、生石膏 15~30g"
>>> formula.find(",")
4
```

2. str.format()方法

str.format(paras)方法首先按照 str 中槽的规定格式化参数 paras 得到子串,然后用子串替换 str 中对应的槽,返回得到的 str 的副本。

该方法的调用格式为:

<模板字符串 str>.format(<逗号分隔的参数 paras>)

其中,<模板字符串>包括了需原样返回的多个字符和若干槽,槽规定如何格式化参数。

下面重点介绍槽的构造方法,并举例说明 str.format()的使用方法。

- 槽的构造方法

槽的内容具体为:

{<参数序号>:<格式控制标记>}

其中,大括号是槽的定界符;<参数序号>可以省略,表示槽与参数按序号一一对应;冒号“:”是引导符号,后续只要有格式控制标记,冒号便不可以省略;<格式控制标记>规定



如何格式化相应参数,依次包括“填充”“对齐”“宽度”“,”“精度”和“类型”6个字段的值。

注意:书写格式控制标记时,6个字段都是可选项,可以组合6个字段中的若干字段使用,但必须遵循表 3.13 中字段的先后顺序。

表 3.13 格式控制标记的字段

填充	对 齐	宽 度	,	.精度	类 型
默认用空格填充	<左对齐 >右对齐 ^居中对齐 默认数字类型右对齐;字符串类型左对齐	槽的设定输出宽度,对应格式化参数得到的子串的字符总数	数字类型的千位分隔符,适用于整数和浮点数	有效数字位数或小数位数或字符串的最大长度	整数类型: b,c,d,o,x,X; 浮点数类型: e,E,f,%

其中,整数的 6 种格式的含义如下。

- ① b: 二进制整数形式;
- ② c: unicode 码为该整数的字符;
- ③ d: 十进制整数形式;
- ④ o: 八进制整数形式;
- ⑤ x: 十六进制小写形式;
- ⑥ X: 十六进制大写形式。

浮点数的 4 种格式的含义如下。

- ① e: 类科学计数法表示形式,其中的“e”小写;
- ② E: 类科学计数法表示形式,其中的“E”大写;
- ③ f: 浮点数的十进制表示形式;
- ④ %: 浮点数的百分数形式。

- str.format()调用的例子

以下是 6 次调用 str.format()方法的例子,注意观察模板字符串和槽的设置与返回值之间的对应关系。

```
>>> #1) 参数序号省略,表示槽与参数按序号一一对应
>>> #".2"表示保留 2 位小数,"%"表示浮点数的百分数形式
>>> "计算化合物分子量时{}的占用率为{:.2%}".format("CPU", 0.1)
'计算化合物分子量时 CPU 的占用率为 10.00%'
```

```
>>> from math import pi
>>> pi
3.141592653589793
>>> #2) 填充符号为空格、右对齐、宽度为 15 个字符、有效数字位数为 5
>>> "{: >15.5}".format(pi)
'          3.1416'
```



```
>>> #3) 填充符号为 *、右对齐、宽度为 15 个字符、有效数字位数为 5
>>> "{: * >15.5}".format(pi)
'*****3.1416'

>>> #4) 填充符号为 *、右对齐、宽度为 15 个字符、有效数字位数为变量 effD 的值
>>> effD = 5
>>> "{0: * >15.{1}}".format(pi, effD)
'*****3.1416'

>>> #5) 填充符号为 *、右对齐、宽度为 15 个字符、小数位数为 5
>>> "{: * >15.5f}".format(pi)
'*****3.14159'

>>> strPi = str(pi)
>>> strPi
'3.141592653589793'
>>> #6) 填充符号为 *、右对齐、宽度为 15 个字符、字符串最大长度为 5
>>> "{: * >15.5}".format(strPi)
'*****3.141'
```

3.4 案例 5：清肺排毒汤处方展示

通常,格式化展示清肺排毒汤的处方需要完成以下 3 项工作。

- ① 明确清肺排毒汤处方展示要求;
- ② 设计清肺排毒汤处方展示算法;
- ③ 编写清肺排毒汤处方展示程序。

3.4.1 清肺排毒汤处方展示要求

清肺排毒汤是由我国汉代张仲景所著《伤寒杂病论》中的多个经典方剂优化组合而成的,在防治新冠肺炎方面发挥了重大作用。清肺排毒汤的处方展示样例,如图 3.3 所示。

清肺排毒汤			
麻黄9g	炙甘草6g	杏仁9g	生石膏15~30g(先煎)
桂枝9g	泽泻9g	猪苓9g	白术9g
茯苓15g	柴胡16g	黄芩6g	姜半夏9g
生姜9g	紫菀9g	冬花9g	射干9g
细辛6g	山药12g	枳实6g	陈皮6g
藿香9g			

图 3.3 清肺排毒汤处方展示样例



由图 3.3 可见,这种展示清肺排毒汤处方的方式清晰明了,具有以下 4 个特点。

- ① 处方名称字符串“清肺排毒汤”居中显示在第一行。
- ② 每行显示 4 味草药,每味草药的信息占 13 个字符宽度。
- ③ 每行后面有 1 个空行。
- ④ 最后 1 味草药的信息单独显示。

3.4.2 清肺排毒汤处方展示算法

实现图 3.3 所示的清肺排毒汤处方格式化展示的算法步骤共 7 步。

- ① 通过创建变量并赋值的方法,输入处方名称字符串-->fname。
- ② 通过创建变量并赋值的方法,输入处方内容字符串-->f。
- ③ 处方名称字符串 fname 居中输出,然后输出 1 个空行。
- ④ 计算顿号个数-->n。
- ⑤ 对变量 re(remain 的简写)赋初值 f。
- ⑥ 对前 n 味草药,统一采用如下方法处理。
 - 查找“、”的位置-->pos。
 - 以 pos 为界,切片得到 re 字符串中第 1 味草药的字符串-->herb。
 - 调用 format()方法格式化 herb 为 13 个字符宽度、左对齐的字符串。
 - 输出格式化后的 herb 的信息。
 - 判断已输出草药的序号 i 是不是 4 的倍数,如果是,则输出 1 个空行。
 - 从 re 中去掉已输出草药的信息及其后的一个顿号,用剩余字符组成的字符串再次创建 re,并为 re 赋值剩余字符组成的字符串。
- ⑦ 输出最后 1 味草药的信息(即 re 的值)。

3.4.3 清肺排毒汤处方展示程序

1. 程序说明/readme

本程序的功能为:按照图 3.3 所示,实现清肺排毒汤处方的格式化展示。

2. 源程序

```
# Input 部分
fname = "清肺排毒汤"
f = "麻黄 9g、炙甘草 6g、杏仁 9g、生石膏 15~30g(先煎)、\
    桂枝 9g、泽泻 9g、猪苓 9g、白术 9g、茯苓 15g、柴胡 16g、黄芩 6g、\
    姜半夏 9g、生姜 9g、紫菀 9g、冬花 9g、射干 9g、细辛 6g、山药 12g、\
    枳实 6g、陈皮 6g、藿香 9g"

# Process & Output 部分
print(fname.center(52), "\n")
n = f.count(",")
re = f
for i in range(1, n + 1):
    pos = re.find(",")
```