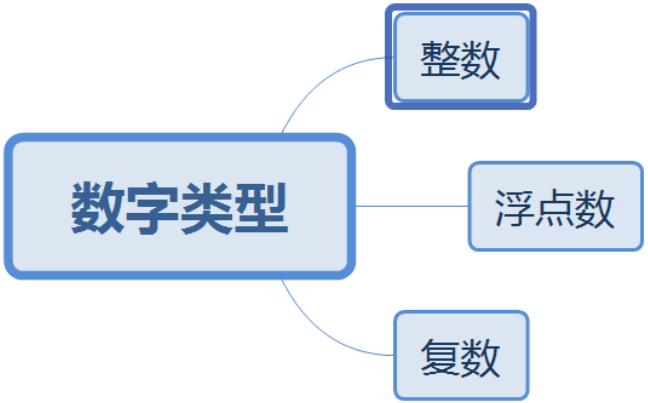
数字类型



1.整数

整数类型概念与数学中的概念一致，理论上的取值范围是[-∞,+∞]。实际上Python支持任意大的数字，只受计算机内存大小限制。整数类型有4种进制表示：十进制、二进制、八进制和十六进制。

进制种类	引导符号	描述
二进制	0b或0B	由字符0和1组成，例：0b1010
八进制	0o或0O	由字符0到7组成，例0o1010
十进制	无	默认情况，例：1010
十六进制	0x或0X	由字符0~9，a~f或A~F组成，例：0x1010

2.浮点数

浮点数类型与数学中实数的概念一致，表示带有小数的数值。Python语言中要求浮点数类型必须带有小数部分，小数部分可以是0。

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

受限于计算机表示浮点数使用的存储宽度，计算的二进制数并不是0.1和0.2而是计算机内部最接近0.1和0.2的二进制数。求得的数反映到十进制表示上，就会产生一个不确定尾数，至于尾数是多少，计算机内部会根据二进制运算确定产生。从用户的角度来看，尾数是不确定的，故称为 ‘不确定尾数’ 。

3.复数

复数类型表示数学中的复数。复数有一个基本单位元素j，被定义为j=$\sqrt{-1}$。含有虚数单位的数被称为复数。例如：

3+4j　　-2.5+4j　　11.2e+3+26j

Python中，复数被看为二元有序实数对(a,b)，表示a+bj，虚部通过j或J表示。

复数类型中实部和虚部都是浮点类型，对于复数z，可以用z.real和z.imag分别获得实数部分和虚数部分。例如：

```
>>> (3+4j).real
3.0
>>> (3+4j).imag
4.0
```

数值运算操作符

操作符	描述
x+y	求两数之和
x-y	求两数之差
x*y	求两数之积
x/y	求两数之商，结果为浮点数
x//y	求两数的整数商
x%y	两数的余数
x**y	x的y次幂，即x^y
-x	一个数的负数
+x	一个数本身

基本规则：

- 整数和浮点数混合运算，输出结果浮点数；
- 整数之间运算，产生结果类型与操作符相关；
- 整数或浮点数与复数运算，输出结果是复数。

增强操作运算符

操作符	描述
x += y	等价于：x = x + y
x -= y	等价于：x = x - y
x *= y	等价于：x = x * y
x /= y	等价于：x = x / y
x //= y	等价于：x = x // y
x %= y	等价于：x = x % y
x **= y	等价于：x = x ** y

数值运算函数

函数	描述
abs(x)	x的绝对值
divmod(x,y)	(x//y,x%y)，输出为二元组形式
pow(x,y)或pow(x,y,z)	x**y或(x**y)%z，幂运算
round(x)或round(x,d)	对x四舍五入，保留d位小数，无参则返回x的整数
max(x_1,x_2,…,x_n)	任意数量的最大值
min(x_1,x_2, …,x_n)	任意数量的最小值

（1）abs(x)

用于计算整数或浮点数x的绝对值，结果为非负值。该函数也可以计算复数的绝对值。例如：

```
>>> abs(-22)
22
>>> abs(-15+11j)
18.601075237738275
>>>
```

（2）divmod(x,y)

用于计算x和y的除余结果，返回两个值，分别是x和y的整数除，即x//y，以及x与y的余数，即x%y。例如：

```
>>> divmod(10,3)
(3, 1)
>>>
```

（3）pow(x,y)

用于计算x的y次幂。pow(x,y,z)则用来计算x^y%z，模运算与幂运算同时进行，速度更快。例如：

```
>>> pow(10, 2)
100
>>> pow(2, 5, 3)
2
>>> pow(2, 5)%3
2
```

（4）round(x)

对整数或浮点数x进行四舍五入运算。采用“奇进偶不进”的方式运算。

```
>>> round(1, 3)
1
>>> round(0.5)
0
>>> round(1.5)
2
```

(5) max(x_1,x_2,...,x_n)

对任意多个数字进行最大值比较，并输出结果。例如：

```
>>> max(0.2, 5, 4, 47, 22, 11.5, 88)
88
```

(6) min(x_1,x_2,...,x_n)

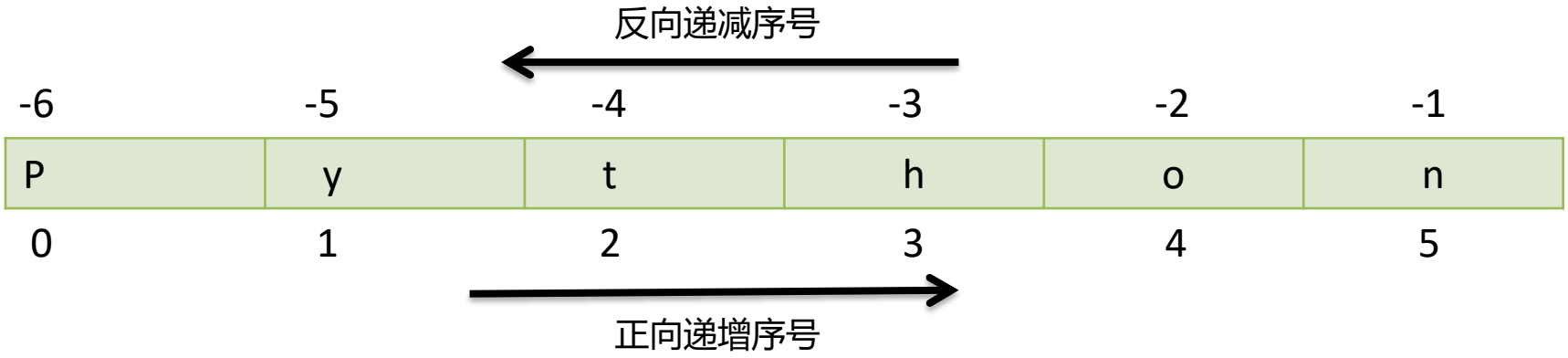
对任意多个数字进行最小值比较，并输出结果。例如：

```
>>> min(0.2, 44, 56, 0.0002)
0.0002
```

字符串类型

用一对双引号 “” 或者一对单引号 ‘’ 括起来。字符串包括两种序列体系：正向递增和反向递减。

多行字符串利用三对双引号”” “ ”” ”” 或者三对单引号’’’ ’ ’’’ 表示。



字符串索引

字符串利用索引方式可以找到其中某个字符。

索引格式：<字符串或字符串变量>[N]

Python中索引有两种访问方式：

- 从前往后的正向索引，n个字符串，索引值从0到n-1；
- 从后往前的反向索引，n个字符串，索引值从-1到-n。

字符串切片

在Python中，可以使用切片从字符串中提取子串，切片适用于字符串、列表、元组、range对象等类型。

切片格式：<字符串或字符串变量>[N:M:step]

参数N是切片的起始索引序号；参数M是切片的结束索引序号；参数step是切片的步长（可省略）。

转义字符

反斜杠（\）是一个特殊字符，表示”转义”。

如：\n表示换行、\\表示反斜杠、\’表示单引号字符、\t表示制表符。

```
>>> print("这是一个\n换行")
这是一个
换行
>>> print("这是一个反斜杠\\")
这是一个反斜杠\
>>> print("将双引号\"作为普通字符输出来")
将双引号"作为普通字符输出来
>>> print("制\t表\t符")
制      表      符
```

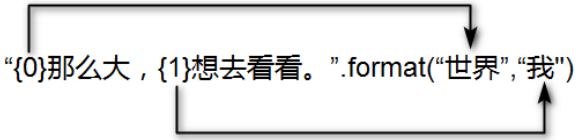
format方法基本使用

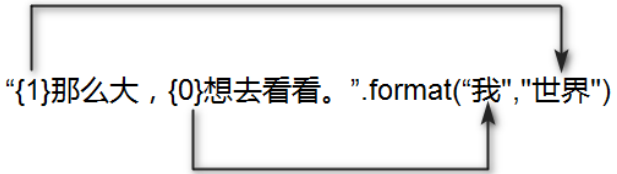
字符串使用方式：<模板字符串>.format(<逗号分隔的参数>)

其中模板字符串是一个由字符串和槽组成的字符串，用来控制字符串和变量的显示效果，例如：

```
>>> "{}那么大，{}想去看看。".format("世界","我")
'世界那么大，我想去看看。'
```

format()方法参数的使用顺序：





format方法格式控制

format()的槽不仅包括参数序号还包括格式控制信息，语法格式为：

{<参数序号>:<格式控制标记>}

:	<填充>	<对齐>	<宽度>	<,>	<.精度>	<类型>
引导符号	用于填充的单个字符	<左对齐 >右对齐 ^居中对齐	槽的设定输出宽度	数字的千位分隔符适用于整数和浮点数	浮点数小数部分的精度或字符串的最大输出长度	整数类型b,c,d,o,x,X 浮点数类型e,E,f,%

输出整数和浮点数类型的格式规则

整数：

b:输出整数的二进制方式

c:输出整数对应的unicode字符

d :输出整数的十进制方式

o: 输出整数的八进制方式

x: 输出整数的十六进制方式

X:输出整数的大写十六进制方式

浮点数：

e: 输出浮点数对应的小写字母e的指数形式

E：输出浮点数对应的大写字母E的指数形式

f :输出浮点数的标准浮点型形式

% :输出浮点数的百分比形式

字符串类型的操作

str1 + str2　：连接两个字符串

str * n 或 n * str　：复制n次字符str

s in str　：判断s字符串是否在str中

数字和字符串函数

函数	描述
len(x)	返回字符串x的长度或者是其他组合类型的元素个数
chr(x)	返回Unicode编码对应的单字符
ord(x)	返回单个字符对应的Unicode编码
bin(x)	返回整数x对应的二进制的小写形式
oct(x)	返回整数x对应的八进制的小写形式
hex(x)	返回整数x对应的十六进制的小写形式
str(x)	将x转换为字符串
int(x)	将x转换为整数
float(x)	将x转换为浮点数

字符串处理方法

方法	描述
str.lower()	以小写的方式全部返回str的副本
str.upper()	以大写的方式全部返回str的副本
str.split(sep=None)	返回一个列表，以sep作为分隔点，sep默认为空格
str.count(sub)	返回sub子串出现的次数
str.replace(old,new)	返回字符串str的副本，所有old子串被替换为new
str.center(width,fillchar)	字符串居中函数，fillchar参数可选
str.strip(chars)	从字符串str中去掉在其左侧和右侧chars中列出的字符
str.join(iter)	将iter变量的每一个元素后面增加一个str字符串