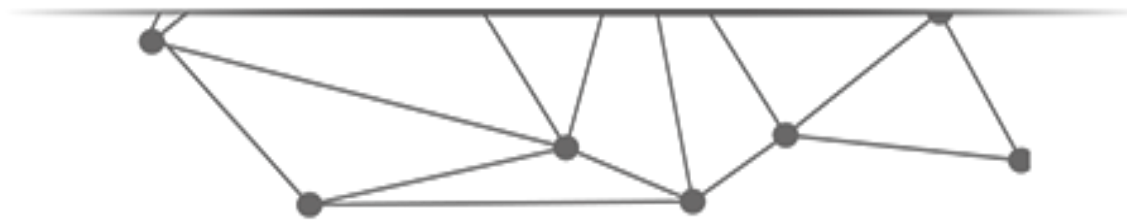




3.1 数字类型（整型、浮点型、复数类型）





整数类型

与数学中整数的概念一致

- 可正可负，没有取值范围限制
- `pow(x,y)`函数：计算 x^y ，想算多大算多大

```
>>> pow(2,100)
```

```
126765060022822940149670320  
5376
```

```
>>> pow(2,pow(2,15))
```

```
141546103104495478900155  
3.....
```



整数类型

4种进制表示形式

- 十进制: 1010, 99, -217 **int()**
- 二进制, 以0b或0B开头: 0b010, -0B101 **bin()_binary**
- 八进制, 以0o或0O开头: 0o123, -0O456 **oct()_Octal number system**
- 十六进制, 以0x或0X开头: 0x9a, -0X89 **hex()_Hexadecimal**

英 [ˌheksəˈdesɪml]



关于Python整数，就需要知道这些。

- 整数无限制 `pow()`
- 4种进制表示形式



浮点数类型

与数学中实数的概念一致

- **带有小数点及小数的数字**
- **浮点数取值范围和小数精度都存在限制，但常规计算可忽略**
- **取值范围数量级约 -10^{307} 至 10^{308} ，精度数量级 10^{-16}**



浮点数类型

- 浮点数间运算存在不确定尾数，不是bug

- `>>> 0.1 + 0.3`

- `0.4`

- `>>> 0.1 + 0.2`

- `0.30000000000000004`

- 不确定尾数



浮点数类型

浮点数间运算存在不确定尾数，不是bug

0.1

53位二进制表示小数部分，约 10^{-16}

0.0001100110011001100110011001100110011001100110011010 (二进制表示)

0.1000000000000000055511151231257827021181583404541015625 (十进制表示)

二进制表示小数，可以无限接近，但不完全相同

0.1 + 0.2

结果无限接近0.3，但可能存在尾数



浮点数类型

浮点数间运算存在不确定尾数

```
>>> 0.1 + 0.2 == 0.3
```

```
False
```

```
>>> round(0.1+0.2, 1) == 0.3
```

```
True
```




浮点数类型

浮点数间运算存在不确定尾数

- **round(x, d): 对x四舍五入, d是小数截取位数**
- **浮点数间运算与比较用round()函数辅助**
- **不确定尾数一般发生在 10^{-16} 左右, round()十分有效**



浮点数类型

浮点数可以采用科学计数法表示

- 使用字母e或E作为幂的符号，以10为基数，格式如下：

$\langle a \rangle e \langle b \rangle$ 表示 $a * 10^b$

- 例如： $4.3e-3$ 值为0.0043 $9.6E5$ 值为960000.0



关于Python浮点数，需要知道这些。

- 取值范围和精度基本无限制
- 运算存在不确定尾数 `round()`
- 科学计数法表示



复数类型

与数学中复数的概念一致

如果 $x^2 = -1$ ，那么 x 的值是什么？

- 定义 $j = \sqrt{-1}$ 以此为基础，构建数学体系
- $a + bj$ 被称为复数，其中， a 是实部， b 是虚部



复数类型

复数实例

$$z = \underline{1.23e-4} + \underline{5.6e+89}j$$

- 实部是什么? `z.real` 获得实部 返回值默认保留一位小数
- 虚部是什么? `z.imag` 获得虚部



数值运算操作符

操作符是完成运算的一种符号体系

操作符及使用	描述
$x + y$	加, x与y之和
$x - y$	减, x与y之差
$x * y$	乘, x与y之积
x / y	除, x与y之商 10/3结果是 3.3333333333333333335
$x // y$	整数除, x与y之整数商 10//3结果是3



数值运算操作符

操作符是完成运算的一种符号体系

操作符及使用	描述
$+ x$	x本身
$- y$	y的负值
$x \% y$	余数，模运算 10%3结果是1
$x ** y$	幂运算，x的y次幂， x^y
	当y是小数时，开方运算10**0.5结果是 $\sqrt{10}$



数值运算操作符

二元操作符有对应的增强赋值操作符

增强操作符及使用	描述
x op = y	即 $x = x \text{ op } y$, 其中, op 为二元操作符
	$x += y$ $x -= y$ $x *= y$ $x /= y$
	$x //= y$ $x \% = y$ $x ** = y$
	<pre>>>> x = 3.1415 >>> x **= 3 # 与 x = x **3 等价 31.006276662836743</pre>



数字类型的关系

类型间可进行混合运算，生成结果为“最宽”类型

- 三种类型存在一种逐渐“扩展”或“变宽”的关系：

整数 -> 浮点数 -> 复数

- 例如：123 + 4.0 = 127.0 （整数+浮点数 = 浮点数）

如果有复数参加运算的时候，返回值会带一个括号



数值运算函数

一些以函数形式提供的数值运算功能

函数及使用	描述
<code>abs(x)</code>	绝对值, x的绝对值 <code>abs(-10.01)</code> 结果为 10.01
<code>divmod(x,y)</code>	商余, $(x//y, x\%y)$, 同时输出商和余数 <code>divmod(10, 3)</code> 结果为 (3, 1)
<code>pow(x, y[, z])</code>	幂余, $(x**y)\%z$, [...]表示参数z可省略 <code>pow(3, pow(3, 99), 10000)</code> 结果为 4587
<code>round(x[, d])</code>	四舍五入, d是保留小数位数, 默认值为0 <code>round(-10.123, 2)</code> 结果为 -10.12
<code>max(x₁,x₂, ... ,x_n)</code>	最大值, 返回 x_1, x_2, \dots, x_n 中的最大值, n不限 <code>max(1, 9, 5, 4, 3)</code> 结果为 9
<code>min(x₁,x₂, ... ,x_n)</code>	最小值, 返回 x_1, x_2, \dots, x_n 中的最小值, n不限 <code>min(1, 9, 5, 4, 3)</code> 结果为 1

一些以函数形式提供的数值运算功能

函数及使用	描述
<code>int(x)</code>	将x变成整数，舍弃小数部分 <code>int(123.45)</code> 结果为123; <code>int("123")</code> 结果为123
<code>float(x)</code>	将x变成浮点数，增加小数部分 <code>float(12)</code> 结果为12.0; <code>float("1.23")</code> 结果为1.23
<code>complex(x)</code>	将x变成复数，增加虚数部分 <code>complex(4)</code> 结果为 (4 + 0j)