1.Python标识符

在 Python 里，标识符有字母、数字、下划线组成。

在 Python 中，所有标识符可以包括英文、数字以及下划线(\_)，但不能以数字开头。

Python 中的标识符是区分大小写的。

以下划线开头的标识符是有特殊意义的。以单下划线开头 \_foo 的代表不能直接访问的类属性，需通过类提供的接口进行访问，不能用 from xxx import \* 而导入；

以双下划线开头的 \_\_foo 代表类的私有成员；以双下划线开头和结尾的 \_\_foo\_\_ 代表 Python 里特殊方法专用的标识，如 \_\_init\_\_() 代表类的构造函数。

2.Python有五个标准的数据类型

* Numbers（数字）
* String（字符串）
* List（列表）
* Tuple（元组）
* Dictionary（字典）

Python支持四种不同的数字类型：

* int（有符号整型）
* long（长整型[也可以代表八进制和十六进制]）
* float（浮点型）
* complex（复数）

python的字串列表有2种取值顺序:

* 从左到右索引默认0开始的，最大范围是字符串长度少1
* 从右到左索引默认-1开始的，最大范围是字符串开头

List（列表） 是 Python 中使用最频繁的数据类型。

* 列表可以完成大多数集合类的数据结构实现。它支持字符，数字，字符串甚至可以包含列表（即嵌套）。
* 列表用 [ ] 标识，是 python 最通用的复合数据类型。
* 列表中值的切割也可以用到变量 [头下标:尾下标] ，就可以截取相应的列表，从左到右索引默认 0 开始，从右到左索引默认 -1 开始，下标可以为空表示取到头或尾。
* 加号 + 是列表连接运算符，星号 \* 是重复操作。

元组是另一个数据类型，类似于List（列表）。

* 元组用"()"标识。内部元素用逗号隔开。但是元组不能二次赋值，相当于只读列表。

字典(dictionary)是除列表以外python之中最灵活的内置数据结构类型。

* 列表是有序的对象结合，字典是无序的对象集合。两者之间的区别在于：字典当中的元素是通过键来存取的，而不是通过偏移存取。
* 字典用"{ }"标识。字典由索引(key)和它对应的值value组成。

3.Python数据类型转换

有时候，我们需要对数据内置的类型进行转换，数据类型的转换，你只需要将数据类型作为函数名即可。

以下几个内置的函数可以执行数据类型之间的转换。这些函数返回一个新的对象，表示转换的值。

|  |  |
| --- | --- |
| **函数** | **描述** |
| int(x [,base]) | 将x转换为一个整数 |
| long(x [,base] ) | 将x转换为一个长整数 |
| float(x) | 将x转换到一个浮点数 |
| complex(real [,imag]) | 创建一个复数 |
| str(x) | 将对象 x 转换为字符串 |
| repr(x) | 将对象 x 转换为表达式字符串 |
| eval(str) | 用来计算在字符串中的有效Python表达式,并返回一个对象 |
| tuple(s) | 将序列 s 转换为一个元组 |
| list(s) | 将序列 s 转换为一个列表 |
| set(s) | 转换为可变集合 |
| dict(d) | 创建一个字典。d 必须是一个序列 (key,value)元组。 |
| frozenset(s) | 转换为不可变集合 |
| chr(x) | 将一个整数转换为一个字符 |
| unichr(x) | 将一个整数转换为Unicode字符 |
| ord(x) | 将一个字符转换为它的整数值 |
| hex(x) | 将一个整数转换为一个十六进制字符串 |
| oct(x) | 将一个整数转换为一个八进制字符串 |

4.Python 运算符

* 算术运算符
* 比较（关系）运算符
* 赋值运算符
* 逻辑运算符
* 位运算符
* 成员运算符
* 身份运算符
* 运算符优先级

python算术运算符

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| + | 加 - 两个对象相加 | a + b 输出结果 30 |
| - | 减 - 得到负数或是一个数减去另一个数 | a - b 输出结果 -10 |
| \* | 乘 - 两个数相乘或是返回一个被重复若干次的字符串 | a \* b 输出结果 200 |
| / | 除 - x除以y | b / a 输出结果 2 |
| % | 取模 - 返回除法的余数 | b % a 输出结果 0 |
| \*\* | 幂 - 返回x的y次幂 | a\*\*b 为10的20次方， 输出结果 100000000000000000000 |
| // | 取整除 - 返回商的整数部分 | 9//2 输出结果 4 , 9.0//2.0 输出结果 4.0 |

①：python比较运算符

以下假设变量a为10，变量b为20：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| == | 等于 - 比较对象是否相等 | (a == b) 返回 False。 |
| != | 不等于 - 比较两个对象是否不相等 | (a != b) 返回 true. |
| <> | 不等于 - 比较两个对象是否不相等 | (a <> b) 返回 true。这个运算符类似 != 。 |
| > | 大于 - 返回x是否大于y | (a > b) 返回 False。 |
| < | 小于 - 返回x是否小于y。所有比较运算符返回1表示真，返回0表示假。这分别与特殊的变量True和False等价。注意，这些变量名的大写。 | (a < b) 返回 true。 |
| >= | 大于等于 - 返回x是否大于等于y。 | (a >= b) 返回 False。 |
| <= | 小于等于 - 返回x是否小于等于y。 | (a <= b) 返回 true。 |

②：Python赋值运算符

以下假设变量a为10，变量b为20：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| = | 简单的赋值运算符 | c = a + b 将 a + b 的运算结果赋值为 c |
| += | 加法赋值运算符 | c += a 等效于 c = c + a |
| -= | 减法赋值运算符 | c -= a 等效于 c = c - a |
| \*= | 乘法赋值运算符 | c \*= a 等效于 c = c \* a |
| /= | 除法赋值运算符 | c /= a 等效于 c = c / a |
| %= | 取模赋值运算符 | c %= a 等效于 c = c % a |
| \*\*= | 幂赋值运算符 | c \*\*= a 等效于 c = c \*\* a |
| //= | 取整除赋值运算符 | c //= a 等效于 c = c // a |

③：Python位运算符

下表中变量 a 为 60，b 为 13，二进制格式如下：

a = 0011 1100

b = 0000 1101-----------------

a&b = 0000 1100

a|b = 0011 1101

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| & | 按位与运算符：参与运算的两个值,如果两个相应位都为1,则该位的结果为1,否则为0 | (a & b) 输出结果 12 ，二进制解释： 0000 1100 |
| | | 按位或运算符：只要对应的二个二进位有一个为1时，结果位就为1。 | (a | b) 输出结果 61 ，二进制解释： 0011 1101 |
| ^ | 按位异或运算符：当两对应的二进位相异时，结果为1 | (a ^ b) 输出结果 49 ，二进制解释： 0011 0001 |
| ~ | 按位取反运算符：对数据的每个二进制位取反,即把1变为0,把0变为1 | (~a ) 输出结果 -61 ，二进制解释： 1100 0011， 在一个有符号二进制数的补码形式。 |
| << | 左移动运算符：运算数的各二进位全部左移若干位，由"<<"右边的数指定移动的位数，高位丢弃，低位补0。 | a << 2 输出结果 240 ，二进制解释： 1111 0000 |
| >> | 右移动运算符：把">>"左边的运算数的各二进位全部右移若干位，">>"右边的数指定移动的位数 | a >> 2 输出结果 15 ，二进制解释： 0000 1111 |

a^b = 0011 0001~a = 1100 0011

④：Python逻辑运算符

Python语言支持逻辑运算符，以下假设变量 a 为 10, b为 20:

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **逻辑表达式** | **描述** | **实例** |
| and | x and y | 布尔"与" - 如果 x 为 False，x and y 返回 False，否则它返回 y 的计算值。 | (a and b) 返回 20。 |
| or | x or y | 布尔"或" - 如果 x 是非 0，它返回 x 的值，否则它返回 y 的计算值。 | (a or b) 返回 10。 |
| not | not x | 布尔"非" - 如果 x 为 True，返回 False 。如果 x 为 False，它返回 True。 | not(a and b) 返回 False |

⑤：Python成员运算符

除了以上的一些运算符之外，Python还支持成员运算符，测试实例中包含了一系列的成员，包括字符串，列表或元组。

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| in | 如果在指定的序列中找到值返回 True，否则返回 False。 | x 在 y 序列中 , 如果 x 在 y 序列中返回 True。 |
| not in | 如果在指定的序列中没有找到值返回 True，否则返回 False。 | x 不在 y 序列中 , 如果 x 不在 y 序列中返回 True。 |

⑥：Python身份运算符

身份运算符用于比较两个对象的存储单元

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| is | is 是判断两个标识符是不是引用自一个对象 | x is y, 类似 id(x) == id(y) , 如果引用的是同一个对象则返回 True，否则返回 False |
| is not | is not 是判断两个标识符是不是引用自不同对象 | x is not y ， 类似 id(a) != id(b)。如果引用的不是同一个对象则返回结果 True，否则返回 False。 |

注：is 与 == 区别：

     is 用于判断两个变量引用对象是否为同一个， == 用于判断引用变量的值是否相等。

5：Python 循环语句

Python提供了for循环和while循环（在Python中没有do..while循环）:

|  |  |
| --- | --- |
| **循环类型** | **描述** |
| [while 循环](http://www.runoob.com/python/python-while-loop.html) | 在给定的判断条件为 true 时执行循环体，否则退出循环体。 |
| [for 循环](http://www.runoob.com/python/python-for-loop.html) | 重复执行语句 |
| [嵌套循环](http://www.runoob.com/python/python-nested-loops.html) | 你可以在while循环体中嵌套for循环 |

循环控制语句可以更改语句执行的顺序。Python支持以下循环控制语句：

|  |  |
| --- | --- |
| **控制语句** | **描述** |
| [break 语句](http://www.runoob.com/python/python-break-statement.html) | 在语句块执行过程中终止循环，并且跳出整个循环 |
| [continue 语句](http://www.runoob.com/python/python-continue-statement.html) | 在语句块执行过程中终止当前循环，跳出该次循环，执行下一次循环。 |
| [pass 语句](http://www.runoob.com/python/python-pass-statement.html) | pass是空语句，是为了保持程序结构的完整性。 |

6：Python Number(数字)

Python 支持四种不同的数值类型：

* 整型(Int) - 通常被称为是整型或整数，是正或负整数，不带小数点。
* 长整型(long integers) - 无限大小的整数，整数最后是一个大写或小写的L。
* 浮点型(floating point real values) - 浮点型由整数部分与小数部分组成，浮点型也可以使用科学计数法表示（2.5e2 = 2.5 x 102 = 250）
* 复数(complex numbers) - 复数由实数部分和虚数部分构成，可以用a + bj,或者complex(a,b)表示， 复数的实部a和虚部b都是浮点型。

    注：长整型也可以使用小写"L"，但是还是建议您使用大写"L"，避免与数字"1"混淆。Python使用"L"来显示长整型。

* Python还支持复数，复数由实数部分和虚数部分构成，可以用a + bj,或者complex(a,b)表示， 复数的实部a和虚部b都是浮点型

Python Number类型转换：

int(x [,base ]) 将x转换为一个整数

long(x [,base ]) 将x转换为一个长整数

float(x ) 将x转换到一个浮点数

complex(real [,imag ]) 创建一个复数

str(x ) 将对象 x 转换为字符串

repr(x ) 将对象 x 转换为表达式字符串

eval(str ) 用来计算在字符串中的有效Python表达式,并返回一个对象

tuple(s ) 将序列 s 转换为一个元组

list(s ) 将序列 s 转换为一个列表

chr(x ) 将一个整数转换为一个字符

unichr(x ) 将一个整数转换为Unicode字符

ord(x ) 将一个字符转换为它的整数值

hex(x ) 将一个整数转换为一个十六进制字符串

oct(x ) 将一个整数转换为一个八进制字符串

python数学函数：

|  |  |
| --- | --- |
| **函数** | **返回值 ( 描述 )** |
| [abs(x)](http://www.runoob.com/python/func-number-abs.html) | 返回数字的绝对值，如abs(-10) 返回 10 |
| [ceil(x)](http://www.runoob.com/python/func-number-ceil.html) | 返回数字的上入整数，如math.ceil(4.1) 返回 5 |
| [cmp(x, y)](http://www.runoob.com/python/func-number-cmp.html) | 如果 x < y 返回 -1, 如果 x == y 返回 0, 如果 x > y 返回 1 |
| [exp(x)](http://www.runoob.com/python/func-number-exp.html) | 返回e的x次幂(ex),如math.exp(1) 返回2.718281828459045 |
| [fabs(x)](http://www.runoob.com/python/func-number-fabs.html) | 返回数字的绝对值，如math.fabs(-10) 返回10.0 |
| [floor(x)](http://www.runoob.com/python/func-number-floor.html) | 返回数字的下舍整数，如math.floor(4.9)返回 4 |
| [log(x)](http://www.runoob.com/python/func-number-log.html) | 如math.log(math.e)返回1.0,math.log(100,10)返回2.0 |
| [log10(x)](http://www.runoob.com/python/func-number-log10.html) | 返回以10为基数的x的对数，如math.log10(100)返回 2.0 |
| [max(x1, x2,...)](http://www.runoob.com/python/func-number-max.html) | 返回给定参数的最大值，参数可以为序列。 |
| [min(x1, x2,...)](http://www.runoob.com/python/func-number-min.html) | 返回给定参数的最小值，参数可以为序列。 |
| [modf(x)](http://www.runoob.com/python/func-number-modf.html) | 返回x的整数部分与小数部分，两部分的数值符号与x相同，整数部分以浮点型表示。 |
| [pow(x, y)](http://www.runoob.com/python/func-number-pow.html) | x\*\*y 运算后的值。 |
| [round(x [,n])](http://www.runoob.com/python/func-number-round.html) | 返回浮点数x的四舍五入值，如给出n值，则代表舍入到小数点后的位数。 |
| [sqrt(x)](http://www.runoob.com/python/func-number-sqrt.html) | 返回数字x的平方根，数字可以为负数，返回类型为实数，如math.sqrt(4)返回 2+0j |

Python随机函数：

随机数可以用于数学，游戏，安全等领域中，还经常被嵌入到算法中，用以提高算法效率，并提高程序的安全性。

Python包含以下常用随机数函数：

|  |  |
| --- | --- |
| **函数** | **描述** |
| [choice(seq)](http://www.runoob.com/python/func-number-choice.html) | 从序列的元素中随机挑选一个元素，比如random.choice(range(10))，从0到9中随机挑选一个整数。 |
| [randrange ([start,] stop [,step])](http://www.runoob.com/python/func-number-randrange.html) | 从指定范围内，按指定基数递增的集合中获取一个随机数，基数缺省值为1 |
| [random()](http://www.runoob.com/python/func-number-random.html) | 随机生成下一个实数，它在[0,1)范围内。 |
| [seed([x])](http://www.runoob.com/python/func-number-seed.html) | 改变随机数生成器的种子seed。如果你不了解其原理，你不必特别去设定seed，Python会帮你选择seed。 |
| [shuffle(lst)](http://www.runoob.com/python/func-number-shuffle.html) | 将序列的所有元素随机排序 |
| [uniform(x, y)](http://www.runoob.com/python/func-number-uniform.html) | 随机生成下一个实数，它在[x,y]范围内。 |

Python三角函数：

Python包括以下三角函数：

|  |  |
| --- | --- |
| **函数** | **描述** |
| [acos(x)](http://www.runoob.com/python/func-number-acos.html) | 返回x的反余弦弧度值。 |
| [asin(x)](http://www.runoob.com/python/func-number-asin.html) | 返回x的反正弦弧度值。 |
| [atan(x)](http://www.runoob.com/python/func-number-atan.html) | 返回x的反正切弧度值。 |
| [atan2(y, x)](http://www.runoob.com/python/func-number-atan2.html) | 返回给定的 X 及 Y 坐标值的反正切值。 |
| [cos(x)](http://www.runoob.com/python/func-number-cos.html) | 返回x的弧度的余弦值。 |
| [hypot(x, y)](http://www.runoob.com/python/func-number-hypot.html) | 返回欧几里德范数 sqrt(x\*x + y\*y)。 |
| [sin(x)](http://www.runoob.com/python/func-number-sin.html) | 返回的x弧度的正弦值。 |
| [tan(x)](http://www.runoob.com/python/func-number-tan.html) | 返回x弧度的正切值。 |
| [degrees(x)](http://www.runoob.com/python/func-number-degrees.html) | 将弧度转换为角度,如degrees(math.pi/2) ， 返回90.0 |
| [radians(x)](http://www.runoob.com/python/func-number-radians.html) | 将角度转换为弧度 |

Python数学常量：

|  |  |
| --- | --- |
| **常量** | **描述** |
| pi | 数学常量 pi（圆周率，一般以π来表示） |
| e | 数学常量 e，e即自然常数（自然常数）。 |

7：Python字符串

Python转义字符：

在需要在字符中使用特殊字符时，python用反斜杠(\)转义字符。如下表：

|  |  |
| --- | --- |
| \(在行尾时) | 续行符 |
| \\ | 反斜杠符号 |
| \' | 单引号 |
| \" | 双引号 |
| \a | 响铃 |
| \b | 退格(Backspace) |
| \e | 转义 |
| \000 | 空 |
| \n | 换行 |
| \v | 纵向制表符 |
| \t | 横向制表符 |
| \r | 回车 |
| \f | 换页 |
| \oyy | 八进制数，yy代表的字符，例如：\o12代表换行 |
| \xyy | 十六进制数，yy代表的字符，例如：\x0a代表换行 |
| \other | 其它的字符以普通格式输出 |

Python字符串运算符：

下表实例变量 a 值为字符串 "Hello"，b 变量值为 "Python"：

|  |  |  |
| --- | --- | --- |
| **操作符** | **描述** | **实例** |
| + | 字符串连接 | >>>a + b 'HelloPython' |
| \* | 重复输出字符串 | >>>a \* 2 'HelloHello' |
| [] | 通过索引获取字符串中字符 | >>>a[1] 'e' |
| [ : ] | 截取字符串中的一部分 | >>>a[1:4] 'ell' |
| in | 成员运算符 - 如果字符串中包含给定的字符返回 True | >>>"H" in a True |
| not in | 成员运算符 - 如果字符串中不包含给定的字符返回 True | >>>"M" not in a True |
| r/R | 原始字符串 - 原始字符串：所有的字符串都是直接按照字面的意思来使用，没有转义特殊或不能打印的字符。 原始字符串除在字符串的第一个引号前加上字母"r"（可以大小写）以外，与普通字符串有着几乎完全相同的语法。 | >>>print r'\n' \n >>> print R'\n' \n |
| % | 格式字符串 | 请看下一章节 |

Python字符串格式化：

Python 支持格式化字符串的输出 。尽管这样可能会用到非常复杂的表达式，但最基本的用法是将一个值插入到一个有字符串格式符 %s 的字符串中。

在 Python 中，字符串格式化使用与 C 中 sprintf 函数一样的语法。

如下实例：

# !/usr/bin/pythonprint "My name is %s and weight is %d kg!" % ('Zara', 21)以上实例输出结果：

My name is Zara and weight is 21 kg!python字符串格式化符号：

|  |  |
| --- | --- |
| **符   号** | **描述** |
| %c | 格式化字符及其ASCII码 |
| %s | 格式化字符串 |
| %d | 格式化整数 |
| %u | 格式化无符号整型 |
| %o | 格式化无符号八进制数 |
| %x | 格式化无符号十六进制数 |
| %X | 格式化无符号十六进制数（大写） |
| %f | 格式化浮点数字，可指定小数点后的精度 |
| %e | 用科学计数法格式化浮点数 |
| %E | 作用同%e，用科学计数法格式化浮点数 |
| %g | %f和%e的简写 |
| %G | %f 和 %E 的简写 |
| %p | 用十六进制数格式化变量的地址 |

8.Python列表：

Python包含以下函数:

|  |  |
| --- | --- |
| **序号** | **函数** |
| 1 | [cmp(list1, list2)](http://www.runoob.com/python/att-list-cmp.html)  比较两个列表的元素 |
| 2 | [len(list)](http://www.runoob.com/python/att-list-len.html)  列表元素个数 |
| 3 | [max(list)](http://www.runoob.com/python/att-list-max.html)  返回列表元素最大值 |
| 4 | [min(list)](http://www.runoob.com/python/att-list-min.html)  返回列表元素最小值 |
| 5 | [list(seq)](http://www.runoob.com/python/att-list-list.html)  将元组转换为列表 |

Python包含以下方法:

|  |  |
| --- | --- |
| **序号** | **方法** |
| 1 | [list.append(obj)](http://www.runoob.com/python/att-list-append.html)  在列表末尾添加新的对象 |
| 2 | [list.count(obj)](http://www.runoob.com/python/att-list-count.html)  统计某个元素在列表中出现的次数 |
| 3 | [list.extend(seq)](http://www.runoob.com/python/att-list-extend.html)  在列表末尾一次性追加另一个序列中的多个值（用新列表扩展原来的列表） |
| 4 | [list.index(obj)](http://www.runoob.com/python/att-list-index.html)  从列表中找出某个值第一个匹配项的索引位置 |
| 5 | [list.insert(index, obj)](http://www.runoob.com/python/att-list-insert.html)  将对象插入列表 |
| 6 | [list.pop(obj=list[-1])](http://www.runoob.com/python/att-list-pop.html)  移除列表中的一个元素（默认最后一个元素），并且返回该元素的值 |
| 7 | [list.remove(obj)](http://www.runoob.com/python/att-list-remove.html)  移除列表中某个值的第一个匹配项 |
| 8 | [list.reverse()](http://www.runoob.com/python/att-list-reverse.html)  反向列表中元素 |
| 9 | [list.sort([func])](http://www.runoob.com/python/att-list-sort.html)  对原列表进行排 |

9.Python元组

Python的元组(tuple)与列表类似，不同之处在于元组的元素不能修改。

元组使用小括号，列表使用方括号。

元组内置函数：

Python元组包含了以下内置函数

|  |  |
| --- | --- |
| **序号** | **方法及描述** |
| 1 | [cmp(tuple1, tuple2)](http://www.runoob.com/python/att-tuple-cmp.html)  比较两个元组元素。 |
| 2 | [len(tuple)](http://www.runoob.com/python/att-tuple-len.html)  计算元组元素个数。 |
| 3 | [max(tuple)](http://www.runoob.com/python/att-tuple-max.html)  返回元组中元素最大值。 |
| 4 | [min(tuple)](http://www.runoob.com/python/att-tuple-min.html)  返回元组中元素最小值。 |
| 5 | [tuple(seq)](http://www.runoob.com/python/att-tuple-tuple.html)  将列表转换为元组。 |

10.Python字典（dictionary）

字典是另一种可变容器模型，且可存储任意类型对象。

字典的每个键值(key=>value)对用冒号(:)分割，每个对之间用逗号(,)分割，整个字典包括在花括号({})中

字典内置函数及方法：

Python字典包含了以下内置函数：

|  |  |
| --- | --- |
| **序号** | **函数及描述** |
| 1 | [cmp(dict1, dict2)](http://www.runoob.com/python/att-dictionary-cmp.html)  比较两个字典元素。 |
| 2 | [len(dict)](http://www.runoob.com/python/att-dictionary-len.html)  计算字典元素个数，即键的总数。 |
| 3 | [str(dict)](http://www.runoob.com/python/att-dictionary-str.html)  输出字典可打印的字符串表示。 |
| 4 | [type(variable)](http://www.runoob.com/python/att-dictionary-type.html)  返回输入的变量类型，如果变量是字典就返回字典类型。 |

Python字典包含了以下内置方法：

|  |  |
| --- | --- |
| **序号** | **函数及描述** |
| 1 | [dict.clear()](http://www.runoob.com/python/att-dictionary-clear.html)  删除字典内所有元素 |
| 2 | [dict.copy()](http://www.runoob.com/python/att-dictionary-copy.html)  返回一个字典的浅复制 |
| 3 | [dict.fromkeys(seq[, val]))](http://www.runoob.com/python/att-dictionary-fromkeys.html)  创建一个新字典，以序列 seq 中元素做字典的键，val 为字典所有键对应的初始值 |
| 4 | [dict.get(key, default=None)](http://www.runoob.com/python/att-dictionary-get.html)  返回指定键的值，如果值不在字典中返回default值 |
| 5 | [dict.has\_key(key)](http://www.runoob.com/python/att-dictionary-has_key.html)  如果键在字典dict里返回true，否则返回false |
| 6 | [dict.items()](http://www.runoob.com/python/att-dictionary-items.html)  以列表返回可遍历的(键, 值) 元组数组 |
| 7 | [dict.keys()](http://www.runoob.com/python/att-dictionary-keys.html)  以列表返回一个字典所有的键 |
| 8 | [dict.setdefault(key, default=None)](http://www.runoob.com/python/att-dictionary-setdefault.html)  和get()类似, 但如果键不存在于字典中，将会添加键并将值设为default |
| 9 | [dict.update(dict2)](http://www.runoob.com/python/att-dictionary-update.html)  把字典dict2的键/值对更新到dict里 |
| 10 | [dict.values()](http://www.runoob.com/python/att-dictionary-values.html)  以列表返回字典中的所有值 |
| 11 | [pop(key[,default])](http://www.runoob.com/python/python-att-dictionary-pop.html)  删除字典给定键 key 所对应的值，返回值为被删除的值。key值必须给出。 否则，返回default值。 |
| 12 | [popitem()](http://www.runoob.com/python/python-att-dictionary-popitem.html)  随机返回并删除字典中的一对键和值。 |

11.匿名函数lambda

python 使用 lambda 来创建匿名函数。

* lambda只是一个表达式，函数体比def简单很多。
* lambda的主体是一个表达式，而不是一个代码块。仅仅能在lambda表达式中封装有限的逻辑进去。
* lambda函数拥有自己的命名空间，且不能访问自有参数列表之外或全局命名空间里的参数。
* 虽然lambda函数看起来只能写一行，却不等同于C或C++的内联函数，后者的目的是调用小函数时不占用栈内存从而增加运行效率。

如：

sum = lambda arg1, arg2: arg1 + arg2;print "相加后的值为 : ", sum( 10, 20 ) //输出30

12.python import语句

* From...import语句

　　　　Python 的 from 语句让你从模块中导入一个指定的部分到当前命名空间中。语法如下：

　　　　from modname import name1[, name2[, ... nameN]]　　　　例如，要导入模块 fib 的 fibonacci 函数，使用如下语句：

　　　　from fib import fibonacci　　　　这个声明不会把整个 fib 模块导入到当前的命名空间中，它只会将 fib 里的 fibonacci 单个引入到执行这个声明的模块的全局符号表。

* From...import\*语句

　　　　把一个模块的所有内容全都导入到当前的命名空间也是可行的，只需使用如下声明：

　　　　from modname import \*　　　　这提供了一个简单的方法来导入一个模块中的所有项目。然而这种声明不该被过多地使用。

　　　　例如我们想一次性引入 math 模块中所有的东西，语句如下：

　　　　from math import \*

* 13.Python文件操作打开和关闭文件

现在，您已经可以向标准输入和输出进行读写。现在，来看看怎么读写实际的数据文件。

Python 提供了必要的函数和方法进行默认情况下的文件基本操作。你可以用 file 对象做大部分的文件操作。

* open函数

你必须先用Python内置的open()函数打开一个文件，创建一个file对象，相关的方法才可以调用它进行读写。

语法：

file object = open(file\_name [, access\_mode][, buffering])各个参数的细节如下：

* file\_name：file\_name变量是一个包含了你要访问的文件名称的字符串值。
* access\_mode：access\_mode决定了打开文件的模式：只读，写入，追加等。所有可取值见如下的完全列表。这个参数是非强制的，默认文件访问模式为只读(r)。
* buffering:如果buffering的值被设为0，就不会有寄存。如果buffering的值取1，访问文件时会寄存行。如果将buffering的值设为大于1的整数，表明了这就是的寄存区的缓冲大小。如果取负值，寄存区的缓冲大小则为系统默认。

不同模式打开文件的完全列表：

|  |  |
| --- | --- |
| **模式** | **描述** |
| r | 以只读方式打开文件。文件的指针将会放在文件的开头。这是默认模式。 |
| rb | 以二进制格式打开一个文件用于只读。文件指针将会放在文件的开头。这是默认模式。 |
| r+ | 打开一个文件用于读写。文件指针将会放在文件的开头。 |
| rb+ | 以二进制格式打开一个文件用于读写。文件指针将会放在文件的开头。 |
| w | 打开一个文件只用于写入。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| wb | 以二进制格式打开一个文件只用于写入。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| w+ | 打开一个文件用于读写。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| wb+ | 以二进制格式打开一个文件用于读写。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。 |
| a | 打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。 |
| ab | 以二进制格式打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。 |
| a+ | 打开一个文件用于读写。如果该文件已存在，文件指针将会放在文件的结尾。文件打开时会是追加模式。如果该文件不存在，创建新文件用于读写。 |
| ab+ | 以二进制格式打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。如果该文件不存在，创建新文件用于读写。 |

* File对象的属性

一个文件被打开后，你有一个file对象，你可以得到有关该文件的各种信息。

以下是和file对象相关的所有属性的列表：

|  |  |
| --- | --- |
| **属性** | **描述** |
| file.closed | 返回true如果文件已被关闭，否则返回false。 |
| file.mode | 返回被打开文件的访问模式。 |
| file.name | 返回文件的名称。 |
| file.softspace | 如果用print输出后，必须跟一个空格符，则返回false。否则返回true。 |

### 

* close()方法

File 对象的 close（）方法刷新缓冲区里任何还没写入的信息，并关闭该文件，这之后便不能再进行写入。

当一个文件对象的引用被重新指定给另一个文件时，Python 会关闭之前的文件。用 close（）方法关闭文件是一个很好的习惯。

语法：fileObject.close();

* write()方法

write()方法可将任何字符串写入一个打开的文件。需要重点注意的是，Python字符串可以是二进制数据，而不是仅仅是文字。

write()方法不会在字符串的结尾添加换行符('\n')：

语法：

* fileObject.write(string);read()方法

read（）方法从一个打开的文件中读取一个字符串。需要重点注意的是，Python字符串可以是二进制数据，而不是仅仅是文字。

语法：

fileObject.read([count]);

* 文件定位

tell()方法告诉你文件内的当前位置；换句话说，下一次的读写会发生在文件开头这么多字节之后。

seek（offset [,from]）方法改变当前文件的位置。Offset变量表示要移动的字节数。From变量指定开始移动字节的参考位置。

如果from被设为0，这意味着将文件的开头作为移动字节的参考位置。如果设为1，则使用当前的位置作为参考位置。如果它被设为2，那么该文件的末尾将作为参考位置。

* 重命名和删除文件

Python的os模块提供了帮你执行文件处理操作的方法，比如重命名和删除文件。

要使用这个模块，你必须先导入它，然后才可以调用相关的各种功能

remove方法

你可以用remove()方法删除文件，需要提供要删除的文件名作为参数。

* Python里的目录

所有文件都包含在各个不同的目录下，不过Python也能轻松处理。os模块有许多方法能帮你创建，删除和更改目录。

1.mkdir()方法

可以使用os模块的mkdir()方法在当前目录下创建新的目录们。你需要提供一个包含了要创建的目录名称的参数。

语法：

os.mkdir("newdir")

2.chdir()方法

可以用chdir()方法来改变当前的目录。chdir()方法需要的一个参数是你想设成当前目录的目录名称。

语法：

os.chdir("newdir")

3.rmdir()方法

rmdir()方法删除目录，目录名称以参数传递。

在删除这个目录之前，它的所有内容应该先被清除。

语法：

* os.rmdir('dirname')文件、目录相关方法

三个重要的方法来源能对Windows和Unix操作系统上的文件及目录进行一个广泛且实用的处理及操控，如下：

[File 对象方法](http://www.runoob.com/python/file-methods.html): file对象提供了操作文件的一系列方法。

[OS 对象方法](http://www.runoob.com/python/os-file-methods.html): 提供了处理文件及目录的一系列方法。

14.Python File(文件)方法

file 对象使用 open 函数来创建，下表列出了 file 对象常用的函数：

|  |  |
| --- | --- |
| **序号** | **方法及描述** |
| 1 | [file.close()](http://www.runoob.com/python/file-close.html)  关闭文件。关闭后文件不能再进行读写操作。 |
| 2 | [file.flush()](http://www.runoob.com/python/file-flush.html)  刷新文件内部缓冲，直接把内部缓冲区的数据立刻写入文件, 而不是被动的等待输出缓冲区写入。 |
| 3 | [file.fileno()](http://www.runoob.com/python/file-fileno.html)  返回一个整型的文件描述符(file descriptor FD 整型), 可以用在如os模块的read方法等一些底层操作上。 |
| 4 | [file.isatty()](http://www.runoob.com/python/file-isatty.html)  如果文件连接到一个终端设备返回 True，否则返回 False。 |
| 5 | [file.next()](http://www.runoob.com/python/file-next.html)  返回文件下一行。 |
| 6 | [file.read([size])](http://www.runoob.com/python/python-file-read.html)  从文件读取指定的字节数，如果未给定或为负则读取所有。 |
| 7 | [file.readline([size])](http://www.runoob.com/python/file-readline.html)  读取整行，包括 "\n" 字符。 |
| 8 | [file.readlines([sizehint])](http://www.runoob.com/python/file-readlines.html)  读取所有行并返回列表，若给定sizeint>0，返回总和大约为sizeint字节的行, 实际读取值可能比sizhint较大, 因为需要填充缓冲区。 |
| 9 | [file.seek(offset[, whence])](http://www.runoob.com/python/file-seek.html)  设置文件当前位置 |
| 10 | [file.tell()](http://www.runoob.com/python/file-tell.html)  返回文件当前位置。 |
| 11 | [file.truncate([size])](http://www.runoob.com/python/file-truncate.html)  截取文件，截取的字节通过size指定，默认为当前文件位置。 |
| 12 | [file.write(str)](http://www.runoob.com/python/python-file-write.html)  将字符串写入文件，没有返回值。 |
| 13 | [file.writelines(sequence)](http://www.runoob.com/python/file-writelines.html)  向文件写入一个序列字符串列表，如果需要换行则要自己加入每行的换行符。 |

15.Python内置函数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [abs()](http://www.runoob.com/python/func-number-abs.html) | [divmod()](http://www.runoob.com/python/pytho-func-divmod.html) | [input()](http://www.runoob.com/python/python-func-input.html) | [open()](http://www.runoob.com/python/pytho-func-open.html) | [staticmethod()](http://www.runoob.com/python/pytho-func-staticmethod.html) |
| [all()](http://www.runoob.com/python/python-func-all.html) | [enumerate()](http://www.runoob.com/python/python-func-enumerate.html) | [int()](http://www.runoob.com/python/python-func-int.html) | [ord()](http://www.runoob.com/python/python-func-ord.html) | [str()](http://www.runoob.com/python/python-func-str.html) |
| [any()](http://www.runoob.com/python/python-func-any.html) | [eval()](http://www.runoob.com/python/python-func-eval.html) | [isinstance()](http://www.runoob.com/python/python-func-isinstance.html) | [pow()](http://www.runoob.com/python/func-number-pow.html) | [sum()](http://www.runoob.com/python/python-func-sum.html) |
| [basestring()](http://www.runoob.com/python/python-func-basestring.html) | [execfile()](http://www.runoob.com/python/python-func-execfile.html) | [issubclass()](http://www.runoob.com/python/python-func-issubclass.html) | [print()](http://www.runoob.com/python/python-func-print.html) | [super()](http://www.runoob.com/python/python-func-super.html) |
| [bin()](http://www.runoob.com/python/python-func-bin.html) | [file()](http://www.runoob.com/python/python-func-file.html) | [iter()](http://www.runoob.com/python/python-func-iter.html) | [property()](http://www.runoob.com/python/python-func-property.html) | [tuple()](http://www.runoob.com/python/att-tuple-tuple.html) |
| [bool()](http://www.runoob.com/python/python-func-bool.html) | [filter()](http://www.runoob.com/python/python-func-filter.html) | [len()](http://www.runoob.com/python/att-string-len.html) | [range()](http://www.runoob.com/python/pytho-func-range.html) | [type()](http://www.runoob.com/python/python-func-type.html) |
| [bytearray()](http://www.runoob.com/python/python-func-bytearray.html) | [float()](http://www.runoob.com/python/python-func-float.html) | [list()](http://www.runoob.com/python/att-list-list.html) | [raw\_input()](http://www.runoob.com/python/pytho-func-raw_input.html) | [unichr()](http://www.runoob.com/python/python-func-unichr.html) |
| [callable()](http://www.runoob.com/python/python-func-callable.html) | [format()](http://www.runoob.com/python/att-string-format.html) | [locals()](http://www.runoob.com/python/python-func-locals.html) | [reduce()](http://www.runoob.com/python/python-func-reduce.html) | unicode() |
| [chr()](http://www.runoob.com/python/python-func-chr.html) | [frozenset()](http://www.runoob.com/python/python-func-frozenset.html) | [long()](http://www.runoob.com/python/python-func-long.html) | [reload()](http://www.runoob.com/python/python-func-reload.html) | [vars()](http://www.runoob.com/python/python-func-vars.html) |
| [classmethod()](http://www.runoob.com/python/python-func-classmethod.html) | [getattr()](http://www.runoob.com/python/python-func-getatt.html) | [map()](http://www.runoob.com/python/python-func-map.html) | [repr()](http://www.runoob.com/python/python-func-repr.html) | [xrange()](http://www.runoob.com/python/python-func-xrange.html) |
| [cmp()](http://www.runoob.com/python/func-number-cmp.html) | [globals()](http://www.runoob.com/python/python-func-globals.html) | [max()](http://www.runoob.com/python/func-number-max.html) | [reversed()](http://www.runoob.com/python/att-list-reverse.html) | [zip()](http://www.runoob.com/python/python-func-zip.html) |
| [compile()](http://www.runoob.com/python/python-func-compile.html) | [hasattr()](http://www.runoob.com/python/python-func-hasattr.html) | [memoryview()](http://www.runoob.com/python/python-func-memoryview.html) | [round()](http://www.runoob.com/python/func-number-round.html) | [\_\_import\_\_()](http://www.runoob.com/python/python-func-__import__.html) |
| [complex()](http://www.runoob.com/python/python-func-complex.html) | [hash()](http://www.runoob.com/python/python-func-hash.html) | [min()](http://www.runoob.com/python/func-number-min.html) | [set()](http://www.runoob.com/python/python-func-set.html) |  |
| [delattr()](http://www.runoob.com/python/python-func-delattr.html) | [help()](http://www.runoob.com/python/python-func-help.html) | [next()](http://www.runoob.com/python/python-func-next.html) | [setattr()](http://www.runoob.com/python/python-func-setattr.html) |  |
| [dict()](http://www.runoob.com/python/python-func-dict.html) | [hex()](http://www.runoob.com/python/python-func-hex.html) | object() | [slice()](http://www.runoob.com/python/python-func-slice.html) |  |
| [dir()](http://www.runoob.com/python/python-func-dir.html) | [id()](http://www.runoob.com/python/python-func-id.html) | [oct()](http://www.runoob.com/python/python-func-oct.html) | [sorted()](http://www.runoob.com/python/python-func-sorted.html) | [exec 内置表达式](http://www.runoob.com/python/python-func-exec.html) |