

python内置函数大全	
<p>最近一直在看python的document，打算在基础方面重点看一下python的keyword、Build-in Function、Build-in Constants、Build-in Types、Build-in Exception这四个方面，其实在看的时候发现整个《The Python Standard Library》章节都是很不错的，其中描述了很多不错的主题。先把Build-in Function罗列一下吧，初学者的了解，分类可能不准确，一起交流。</p>	
一、数学运算类	
abs(x)	求绝对值 1、参数可以是整型，也可以是复数 2、若参数是复数，则返回复数的模
complex([real[, imag]])	创建一个复数
divmod(a, b)	分别取商和余数 注意：整型、浮点型都可以
float([x])	将一个字符串或数转换为浮点数。如果无参数将返回0.0
int([x[, base]])	将一个字符转换为int类型，base表示进制
long([x[, base]])	将一个字符转换为long类型
pow(x, y[, z])	返回x的y次幂
range([start, stop[, step]])	产生一个序列，默认从0开始
round(x[, n])	四舍五入
sum(iterable[, start])	对集合求和
oct(x)	将一个数字转化为8进制
hex(x)	将整数x转换为16进制字符串
chr(i)	返回整数i对应的ASCII字符
bin(x)	将整数x转换为二进制字符串
bool([x])	将x转换为Boolean类型
二、集合类操作	
basestring()	str和unicode的超类 不能直接调用，可以用作isinstance判断
format(value [, format_spec])	格式化输出字符串 格式化的参数顺序从0开始，如“I am {0},I like {1}”
unichr(i)	返回给定int类型的unicode
enumerate(sequence [, start = 0])	返回一个可枚举的对象,该对象的next()方法将返回一个tuple
iter(o[, sentinel])	生成一个对象的迭代器，第二个参数表示分隔符
max(iterable[, args...][key])	返回集合中的最大值
min(iterable[, args...][key])	返回集合中的最小值
dict([arg])	创建数据字典
list([iterable])	将一个集合类转换为另外一个集合类
set()	set对象实例化
frozenset([iterable])	产生一个不可变的set
str([object])	转换为string类型
sorted(iterable[, cmp[, key[, reverse]]])	对集合排序

<code>tuple([iterable])</code>	生成一个tuple类型
<code>xrange([start], stop[, step])</code>	<code>xrange()</code> 函数与 <code>range()</code> 类似，但 <code>xrange()</code> 并不创建列表，而是返回一个 <code>xrange</code> 对象，它的行为与列表相似，但是只在需要时才计算列表值，当列表很大时，这个特性能为我们节省内存
三、逻辑判断	
<code>all(iterable)</code>	1、集合中的元素都为真的时候为真 2、特别的，若为空串返回为True
<code>any(iterable)</code>	1、集合中的元素有一个为真的时候为真 2、特别的，若为空串返回为False
<code>cmp(x, y)</code>	如果 $x < y$ , 返回负数； $x == y$ , 返回0； $x > y$ , 返回正数
四、反射	
<code>callable(object)</code>	检查对象object是否可调用 1、类是可以被调用的 2、实例是不可以被调用的，除非类中声明了 <code>__call__</code> 方法
<code>classmethod()</code>	1、注解，用来说明这个方式是个类方法 2、类方法即可被类调用，也可以被实例调用 3、类方法类似于Java中的static方法 4、类方法中不需要有self参数
<code>compile(source, filename, mode[, flags[, dont_inherit]])</code>	将source编译为代码或者AST对象。代码对象能够通过exec语句来执行或者eval()进行求值。 1、参数source: 字符串或者AST（Abstract Syntax Trees）对象。 2、参数filename: 代码文件名称，如果不是从文件读取代码则传递一些可辨认的值。 3、参数model: 指定编译代码的种类。可以指定为'exec','eval','single'。 4、参数flag和dont_inherit: 这两个参数暂不介绍
<code>dir([object])</code>	1、不带参数时，返回当前范围内的变量、方法和定义的类型列表； 2、带参数时，返回参数的属性、方法列表。 3、如果参数包含方法 <code>__dir__()</code> ，该方法将被调用。当参数为实例时。 4、如果参数不包含 <code>__dir__()</code> ，该方法将最大限度地收集参数信息
<code>delattr(object, name)</code>	删除object对象名为name的属性
<code>eval(expression [, globals [, locals]])</code>	计算表达式expression的值
<code>execfile(filename [, globals [, locals]])</code>	用法类似exec()，不同的是execfile的参数filename为文件名，而exec的参数为字符串。
<code>filter(function, iterable)</code>	构造一个序列，等价于[ item for item in iterable if function(item)] 1、参数function: 返回值为True或False的函数，可以为None

	2、参数 <b>iterable</b> : 序列或可迭代对象
getattr(object, name [, default])	获取一个类的属性
globals()	返回一个描述当前全局符号表的字典
hasattr(object, name)	判断对象 <b>object</b> 是否包含名为 <b>name</b> 的特性
hash(object)	如果对象 <b>object</b> 为哈希表类型, 返回对象 <b>object</b> 的哈希值
id(object)	返回对象的唯一标识
isinstance(object, classinfo)	判断 <b>object</b> 是否是 <b>class</b> 的实例
issubclass(class, classinfo)	判断是否是子类
len(s)	返回集合长度
locals()	返回当前的变量列表
map(function, iterable, ...)	遍历每个元素, 执行 <b>function</b> 操作
memoryview(obj)	返回一个内存镜像类型的对象
next(iterator[, default])	类似于 <b>iterator.next()</b>
object()	基类
property([fget[, fset[, fdel[, doc]]]])	属性访问的包装类, 设置后可以通过 <b>c.x=value</b> 等来访问 <b>setter</b> 和 <b>getter</b>
reduce(function, iterable[, initializer])	合并操作, 从第一个开始是前两个参数, 然后是前两个的结果与第三个合并进行处理, 以此类推
reload(module)	重新加载模块
setattr(object, name, value)	设置属性值
repr(object)	将一个对象变幻为可打印的格式
slice ( )	
staticmethod	声明静态方法, 是个注解
super(type[, object-or-type])	引用父类
type(object)	返回该 <b>object</b> 的类型
vars([object])	返回对象的变量, 若无参数与 <b>dict()</b> 方法类似
	返回一个 <b>byte</b> 数组
	1、如果 <b>source</b> 为整数, 则返回一个长度为 <b>source</b> 的初始化数组;
	2、如果 <b>source</b> 为字符串, 则按照指定的 <b>encoding</b> 将字符串转换为字节序列;
bytearray([source [, encoding [, errors]]])	3、如果 <b>source</b> 为可迭代类型, 则元素必须为[0,255]中的整数;
	4、如果 <b>source</b> 为与 <b>buffer</b> 接口一致的对象, 则此对象也可以被用于初始化 <b>bytearray</b> .
zip([iterable, ...])	实在是没有看懂, 只是看到了矩阵的变幻方面
五、IO操作	
	<b>file</b> 类型的构造函数, 作用为打开一个文件, 如果文件不存在且 <b>mode</b> 为写或追加时, 文件将被创建。添加 <b>'b'</b> 到 <b>mode</b> 参数中, 将对文件以二进制形式操作。添加 <b>'+'</b> 到 <b>mode</b> 参数中, 将允许对文件同时进行读写操作
file(filename [, mode [, bufsize]])	1、参数 <b>filename</b> : 文件名称。

	2、参数mode: 'r'（读）、'w'（写）、'a'（追加）。
	3、参数bufsize: 如果为0表示不进行缓冲，如果为1表示进行行缓冲，如果是一个大于1的数表示缓冲区的大小。
input([prompt])	获取用户输入 推荐使用raw_input，因为该函数将不会捕获用户的错误输入
open(name[, mode[, buffering]])	打开文件 与file有什么不同？推荐使用open
print	打印函数
raw_input([prompt])	设置输入，输入都是作为字符串处理
六、其他	
help()--帮助信息	
__import__()--没太看明白了，看到了那句“Direct use of __import__() is rare”之后就没心看下去了	
apply()、buffer()、coerce()、intern()---这些是过期的内置函数，故不说明	
七、后记	
内置函数，一般都是因为使用频率比较频繁或是元操作，所以通过内置函数的形式提供出来，通过对python的内置函数分类分析可以看出来：基本的数据操作基本都是一些数学运算（当然除了加减乘除）、逻辑操作、集合操作、基本IO操作，然后就是对于语言自身的反射操作，还有就是字符串操作，也是比较常用的，尤其需要注意的是反射操作。	
将整理出来的Excel表格作为附件。	
<a href="#">python内建函数.zip (14.7 KB)</a>	
下载次数: 30	







