1. **Python 双版本windows安装**

1.1安装python3及python27，各自添加环境变量 ---> 修改python3中python.exe为python3.exe, 🡪 修改python3中pip3.exe

* 1. 集成开发环境：

IDLE：标准python环境；

Pythonwin：面向windows环境

Eclipse: 流行灵活并开源的IDE

1. **Python基础**
   1. **序列**

Python包含6种内建序列：**列表、元组**、字符串、Unicode字符串、buffer对象、xrange对象

列表可修改序列，元组不能；所有序列都可以进行**索引、分片、加、乘、成员资格检查**

1. 索引： a=”hello world” 🡺 a[-1] = “d”; year = raw\_input()[2]

a = [1,2,3] 🡺 a[-1] = 3

1. 分片：通过：相隔的两个索引实现

Eg. a = ”hello world” 🡺 a[-2:] = ‘ld’

🡺 a[0:6:2] = “hlo”

3) 序列相加：[1,2,3]+[4,5,6] = [1,2,3,4,5,6]

4) 乘法： “hello”\*2 = “hellohello”; [10]\*2 = [10, 10]

5) 成员资格: a= “hollo” “h” in a: True

* 1. **列表**

1. Append 列表尾追加新的对象 list1.append(element)
2. Count 统计在列表中出现次数。 List1.count(element)
3. Extend 在列表尾一次性追加另个列表中多个值。 List1.extent(list2)
4. Index 列表中找出某值第一次匹配索引项位置 List1.index(element)
5. Insert : list1.insert(3, “a”)
6. Pop 移除一个元素 pop(), pop(2)
7. Remove remove(“be”)
8. Sort list1.sort(), list1.sort(cmp), list1.sort(reverse=True) , list1.sort(key=len),
9. list(“he”) = [“h”, ”e”]
   1. **元组**（不能修改）
10. 创建：3\*（40+2，）=（42, 42, 42）
11. 访问：x=1,2,3 🡺 x[0:2] = (1,2)
12. tuple([1,2,3]) = (1,2,3)
    1. **字符串**： 查看标准库
    2. **字典**
13. 创建： dict([(‘name’, ‘age’), (‘tom’, 50)]);

Dict(name=’tom’, age=50)

2) len(d)

3) d[‘k’], d.get(‘k’)

4) d[k] = v

5) 删掉一条del d[k]， 清空字典条目d.clear(), 删除字典del d

6) 返回所有键 d.keys(),

返回所有值d.values(),

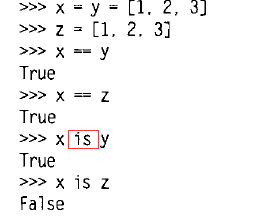
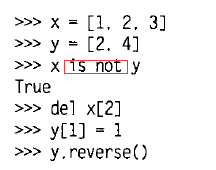
以列表返回可遍历的(键, 值) 元组数组[dict.items()](http://www.runoob.com/python/att-dictionary-items.html" \t "_blank)

把字典dict2的键/值对更新到dict里[dict.update(dict2)](http://www.runoob.com/python/att-dictionary-update.html" \t "_blank)

7) pop

2.3 条件与循环

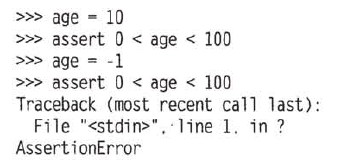
同一性运算符：

**2.4 布尔运算**

And or not三个运算符随意结合

* 1. **断言：assert**



* 1. **列表推导式**

[x\*x for x in range(10)] 🡺 [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

[x\*x for x in range(10) if x % 3 == 0] 🡺 [0, 9, 36, 81]

[(x,y) for x in range(2) for y in range (0,2)] 🡺 [(0,0), (0,1), (1, 0), (1,1)]

* 1. 执行和求和字符串exec和eval

Exec(“print ‘hello world’”) 🡺 hello world

eval(“2+4”) 🡺 6

* 1. 收集参数

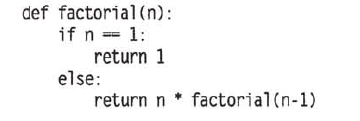
def print\_params(title, \*params):

print title

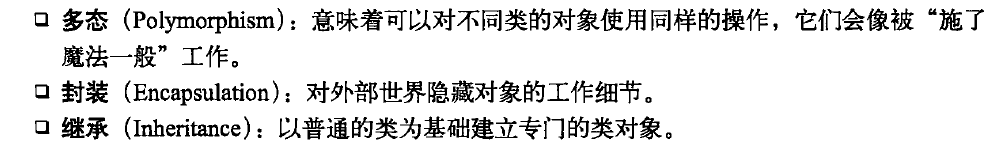
print params

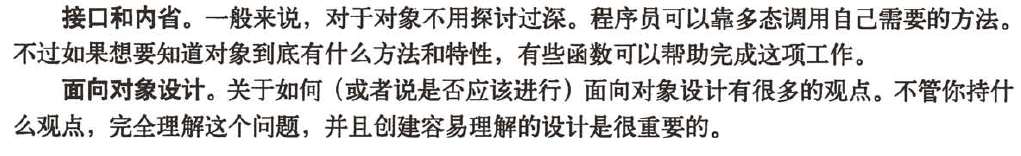
* 1. 递归

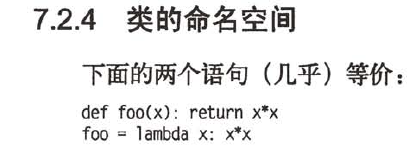


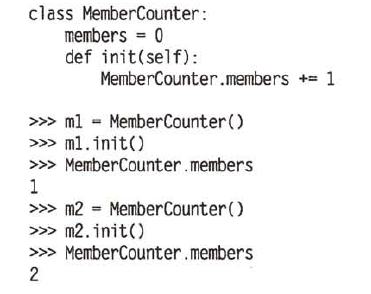


* 1. 多态

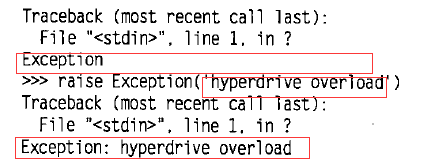


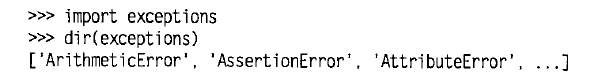


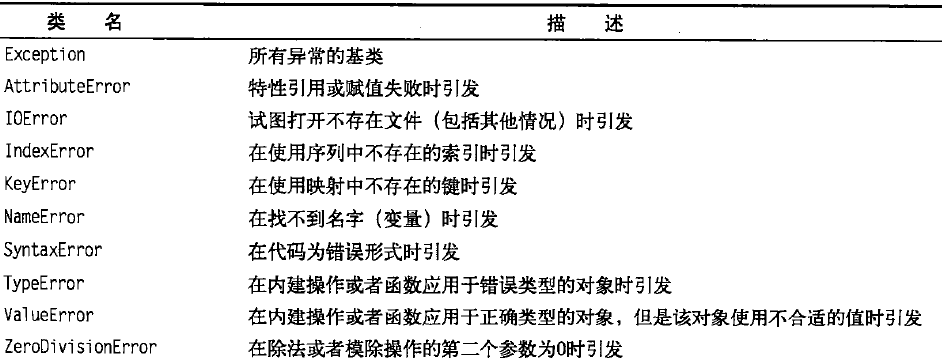




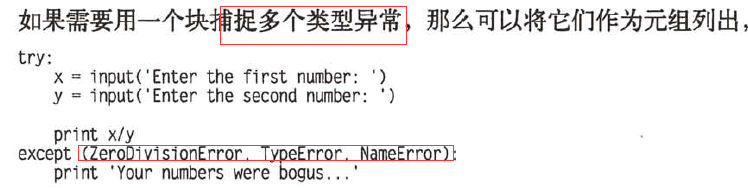
* 1. 异常

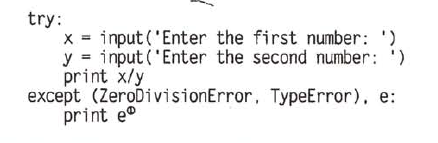


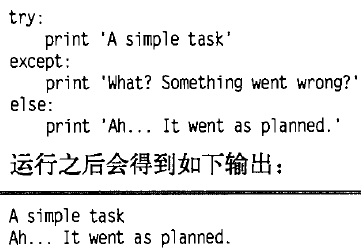


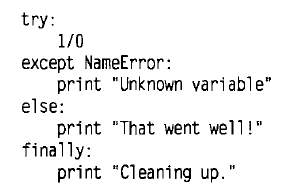


捕捉异常：　try/except

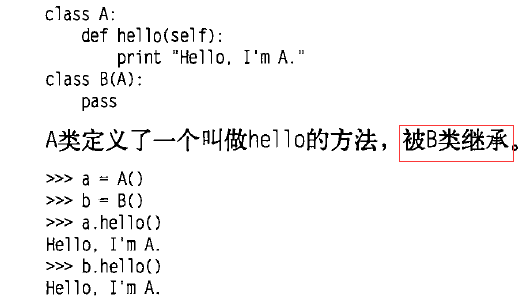


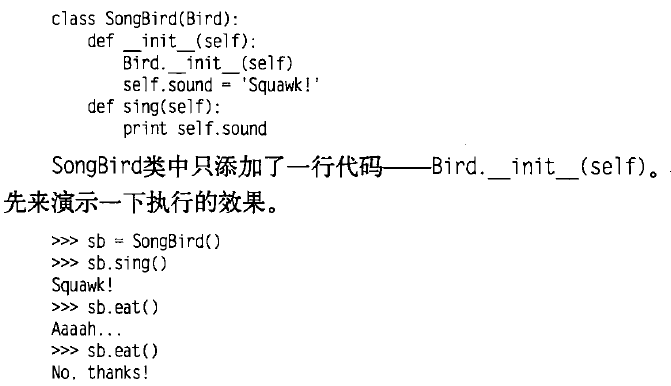


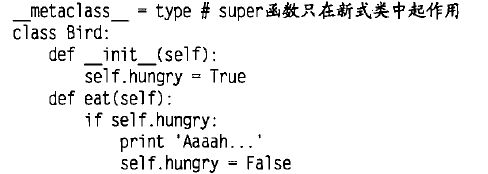


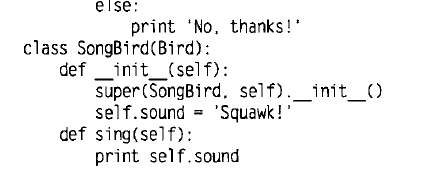


* 1. 继承

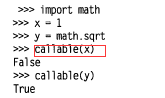




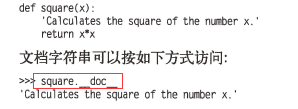




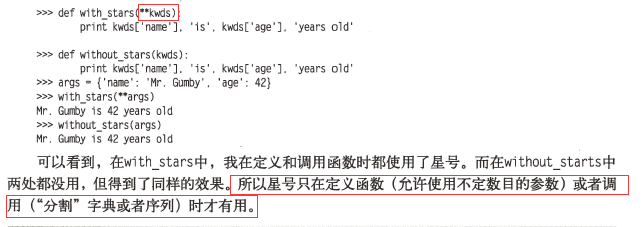
**创建函数**



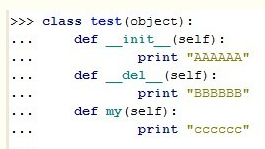
**记录函数**



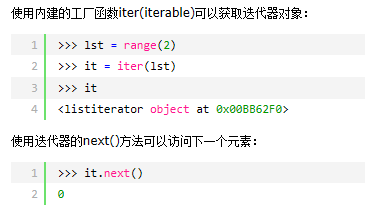
**反转过程**



**构造函数和析构函数**



**迭代器：**



**1. 标准库核心模块**

o **1.2. \_ \_builtin\_ \_ 模块**：这个模块包含 Python 中使用的内建函数. 一般不用手动导入这个模块;

o **1.3. exceptions 模块：**提供了标准异常的层次结构. Python 启动的时候会自动导入这个模块, 并且将它加入到 \_ \_builtin\_ \_ 模块中

o **1.4. os 模块：**大部分函数通过对应平台相关模块实现, 比如 posix 和 nt. os模块会在第一次导入的时候自动加载合适的执行模块

**o 1.5. os.path 模块：**包含了各种处理长文件名(路径名)的函数. 先导入 (import) os模块, 然后就可以以 os.path 访问该模块.

o **1.6. stat 模块：**这个模块包含了一些 os.stat函数中可用的常量和测试函数.

o **1.7. string 模块**：提供了一些用于处理字符串类型的函数

o **1.8. re 模块**：提供了一系列功能强大的正则表达式 (regular expression) 工具, 它们允许你快速检查给定字符串是否与给定的模式匹配 (使用 match 函数), 或者包含这个模式 (使用 search 函数)

o **1.9. math 模块：**math 模块实现了许多对浮点数的数学运算函数. 这些函数一般是对平台 C 库中同名函数的简单封装, 所以一般情况下, 不同平台下计算的结果可能稍微地有所不同, 有时候甚至有很大出入

o **1.10. cmath 模块：**包含了一些用于复数运算的函数.

o **1.11. operator 模块：**为 Python 提供了一个 "功能性" 的标准操作符接口. 当使用map 以及 filter 一类的函数的时候, operator 模块中的函数可以替换一些lambda 函式.

o **1.12. copy 模块**：模块包含两个函数, 用来拷贝对象,

o **1.13. sys 模块**：提供了许多函数和变量来处理 Python 运行时环境的不同部分

o **1.14. atexit 模块：**允许你注册一个或多个终止函数(暂且这么叫), 这些函数将在解释器终止前被自动调用.

**o 1.15. time 模块：**提供了一些处理日期和一天内时间的函数. 它是建立在 C 运行时库的简单封装.

o **1.16. types 模块：**包含了标准解释器定义的所有类型的类型对象

o **1.17. gc 模块：**提供了到内建循环垃圾收集器的接口. Python 使用引用记数来跟踪什么时候销毁一个对象; 一个对象的最后一个引用一旦消失, 这个对象就会被销毁.