**培训笔记**

1. 函数命名要求：动宾短语；培训期间用PEP8命名规范（附录1）。（单词的单复数也要考虑，准确命名，以免造成集合和个体混淆）

函数命名demo：load\_list\_from\_file

2.变量位置：定值循环外计算好，变值放循环内。

3.常量位置：放到（全局）函数外处理，节约资源。常量命名全大写。

4.全局变量：尽量不用全局变量。（绝大多数情况不要用）

5.函数命名单词简写：dst,pic,src等，pos=positon简写（附录2）适当用，函数命名表明目的，操作对象；函数形参的位置越靠前越重要。主要形参不加缺省值，辅助形参可添加缺省值。

6.函数返回值：无返回值无法使用级联用法：Foo(foo(x))（不推荐级联）, 因此函数一般加一个返回值

7.错误处理机制：抛出错误（异常），python为try except，不符合条件考虑抛出错误，（真实状况容易超出想象中的情况）

8.断言机制：assert，多用于调试功能

9.Kiss原则：keep it simple and stupid

10.涉及文件路径：与项目相关的资源放在子文件夹下。不要出现绝对路径，使用相对路径

11.索引问题：可以利用函数来辅助索引

12.利用常量定义，判断操作系统选取不同的常量（Enum）。Env.py (enviroment)

13.PEP8规范工具：black（可按照规范检查调整）pip install black（cmd中使用）

14.遇到不易理解的形参表达式用pos = 表达式代替便于阅读理解（代码可读性很重要）

15.看到固定的阿拉伯数字，考虑其理由是否充分，为何要用这个值（代码中尽量不用固定的阿拉伯数字，用的情况可能为固定的计算公式）

16．list补充知识：contaiver容器（将多个元素放到一起list属于容器的一种） array（数组）访问很快，增删较慢 队列形式。链表 访问较慢，增删较快

（数据结构，自学，此处不详解）

术语：我们对某一个容器进行遍历。注：遍历过程中容器是否可以被删节点。（不要说我用for咋咋咋。。。）

17.哈希表(hash)相关知识 hash key-value （自学，映射关系）

18．传递方式：传递容器为引用方式，即使利用新变量赋值仍会改变原数据。两种方式，按引用传递如容器，按值传递（改变不会影响原数据）。

19．面向对象编程：对象包含方法（函数）与属性

20．面对对象编程三大特性：封装（权限），继承、多态

21. 编程注意：相同功能代码尽量只有一份。继承少，接口多，高内聚，低耦合

22．考虑：职责划分，函数三要素是否齐全，面向对象时，类型名称（职责）、属性、功能是什么?

23．隐式条件（希望避免），紧耦合

24．面对对象编程时，策略（变化）与机制（不变）分离，策略-多态，机制-核心算法（功能）。

25. 检查公有的函数，参数，内外有别。

26. 测试代码：给输入看输出是否满足预期。目的：让自己对代码有100%的信心，能满足期望的目标。

单元测试：最基本单元为测试某一个类型。Unit\_test（测试框架）

测试案例：test\_case。测试代码要保证100%准确，简单直白逻辑清晰。每一个函数是一个case，遇到错误会标记错误而不会终止测试程序。

测试函数功能不希望在测试代码出现，希望是一个库。

测试框架过程（1）建立测试环境 （2）给输入 （3）检查输出

写测试案例明确测试目标（函数名清晰表示目的）

利用Mock（冒烟）模拟测试。实现对单一对象测试。

**附录1 PEP8命名规范**

1. Naming Conventions 命名规范  
   Python库的命名规范很乱，从来没能做到完全一致。但是目前有一些推荐的命名标准。新的模块和包（包括第三方框架）应该用这套标准，但当一个已有库采用了不同的风格，推荐保持内部一致性。

7.1 Overriding Principle 最重要的原则  
那些暴露给用户的API接口的命名，应该遵循反映使用场景而不是实现的原则。

7.2 Descriptive: Naming Styles 描述：命名风格  
有许多不同的命名风格。这里能够帮助大家识别正在使用什么样的命名风格，而不考虑他们为什么使用。  
以下是常见的命名方式：

b（单个小写字母）  
B（单个大写字母）  
lowercase 小写字母  
lower\_case\_with\_underscores 使用下划线分隔的小写字母  
UPPERCASE 大写字母  
UPPER\_CASE\_WITH\_UNDERSCORES 使用下划线分隔的大写字母  
CapitalizedWords（或者叫 CapWords，或者叫CamelCase 驼峰命名法 —— 这么命名是因为字母看上去有起伏的外观5）。有时候也被称为StudlyCaps。  
注意：当在首字母大写的风格中用到缩写时，所有缩写的字母用大写，因此，HTTPServerError 比 HttpServerError 好。  
mixedCase（不同于首字母大写，第一个单词的首字母小写）  
Capitalized\_Words\_With\_Underscores（巨丑无比！）  
也有用唯一的短前缀把相关命名组织在一起的方法。这在Python中不常用，但还是提一下。比如，os.stat()函数中包含类似以st\_mode，st\_size，st\_mtime这种传统命名方式命名的变量。（这么做是为了与 POSIX 系统的调用一致，以帮助程序员熟悉它。）  
X11库的所有公共函数都加了前缀X。在Python里面没必要这么做，因为属性和方法在调用的时候都会用类名做前缀，函数名用模块名做前缀。  
另外，下面这种用前缀或结尾下划线的特殊格式是被认可的（通常和一些约定相结合）：

*single\_leading\_underscore：（单下划线开头）弱“内部使用”指示器。比如 from M import \* 是不会导入以下划线开始的对象的。  
single\_trailing\_underscore*：（单下划线结尾）这是避免和Python内部关键词冲突的一种约定，比如：Tkinter.Toplevel(master, class\_=’ClassName’)  
\_\_double\_leading\_underscore：（双下划线开头）当这样命名一个类的属性时，调用它的时候名字会做矫正（在类FooBar中，\_\_boo变成了\_FooBar\_\_boo；见下文）。  
double\_leading\_and\_trailing\_underscore：（双下划线开头，双下划线结尾）“magic”对象或者存在于用户控制的命名空间内的属性，例如：init,import\_\_或者\_\_file。除了作为文档之外，永远不要命这样的名。  
7.3 Prescriptive: Naming Conventions 约定俗成：命名约定  
7.3.1 Names to Avoid 应避免的名字  
永远不要使用字母‘l’（小写的L），‘O’（大写的O），或者‘I’（大写的I）作为单字符变量名。  
在有些字体里，这些字符无法和数字0和1区分，如果想用‘l’，用‘L’代替。

7.3.2 Package and Module Names 包名和模块名  
模块应该用简短全小写的名字，如果为了提升可读性，下划线也是可以用的。Python包名也应该使用简短全小写的名字，但不建议用下划线。  
当使用C或者C++编写了一个依赖于提供高级（更面向对象）接口的Python模块的扩展模块，这个C/C++模块需要一个下划线前缀（例如：\_socket）

7.3.3 Class Names 类名  
类名一般使用首字母大写的约定。  
在接口被文档化并且主要被用于调用的情况下，可以使用函数的命名风格代替。  
注意，对于内置的变量命名有一个单独的约定：大部分内置变量是单个单词（或者两个单词连接在一起），首字母大写的命名法只用于异常名或者内部的常量。

7.3.4 Exception Names 异常名  
因为异常一般都是类，所有类的命名方法在这里也适用。然而，你需要在异常名后面加上“Error”后缀（如果异常确实是一个错误）。

7.3.5 Global Variable Names 全局变量名  
（我们希望这一类变量只在模块内部使用。）约定和函数命名规则一样。  
通过 from M import \* 导入的模块应该使用all机制去防止内部的接口对外暴露，或者使用在全局变量前加下划线的方式（表明这些全局变量是模块内非公有）。

7.3.6 Function Names 函数名  
函数名应该小写，如果想提高可读性可以用下划线分隔。  
大小写混合仅在为了兼容原来主要以大小写混合风格的情况下使用（比如 threading.py），保持向后兼容性。

7.3.7 Function and method arguments 函数和方法参数  
始终要将 self 作为实例方法的的第一个参数。  
始终要将 cls 作为类静态方法的第一个参数。 如果函数的参数名和已有的关键词冲突，在最后加单一下划线比缩写或随意拼写更好。因此 class\_ 比 clss 更好。（也许最好用同义词来避免这种冲突）

7.3.8 Method Names and Instance Variables 方法名和实例变量  
遵循这样的函数命名规则：使用下划线分隔小写单词以提高可读性。  
在非共有方法和实例变量前使用单下划线。  
通过双下划线前缀触发Python的命名转换规则来避免和子类的命名冲突。  
Python通过类名对这些命名进行转换：如果类 Foo 有一个叫 \_\_a 的成员变量， 它无法通过 Foo.\_\_a 访问。（执着的用户可以通过 Foo.\_Foo\_\_a 访问。）一般来说，前缀双下划线用来避免类中的属性命名与子类冲突的情况。  
注意：关于\_\_names的用法存在争论（见下文）。

7.3.9 Constants 常量  
常量通常定义在模块级，通过下划线分隔的全大写字母命名。例如： MAX\_OVERFLOW 和 TOTAL。

7.3.10 Designing for inheritance 继承的设计  
始终要考虑到一个类的方法和实例变量（统称：属性）应该是共有还是非共有。如果存在疑问，那就选非共有；因为将一个非共有变量转为共有比反过来更容易。  
公共属性是那些与类无关的客户使用的属性，并承诺避免向后不兼容的更改。非共有属性是那些不打算让第三方使用的属性；你不需要承诺非共有属性不会被修改或被删除。  
我们不使用“私有（private）”这个说法，是因为在Python中目前还没有真正的私有属性（为了避免大量不必要的常规工作）。  
另一种属性作为子类API的一部分（在其他语言中通常被称为“protected”）。有些类是专为继承设计的，用来扩展或者修改类的一部分行为。当设计这样的类时，要谨慎决定哪些属性时公开的，哪些是作为子类的API，哪些只能在基类中使用。  
贯彻这样的思想，一下是一些让代码Pythonic的准则：

公共属性不应该有前缀下划线。  
如果公共属性名和关键字冲突，在属性名之后增加一个下划线。这比缩写和随意拼写好很多。（然而，尽管有这样的规则，在作为参数或者变量时，‘cls’是表示‘类’最好的选择，特别是作为类方法的第一个参数。）  
　　注意1：参考之前的类方法参数命名建议

对于单一的共有属性数据，最好直接对外暴露它的变量名，而不是通过负责的 存取器（accessor）/突变（mutator） 方法。请记住，如果你发现一个简单的属性需要成长为一个功能行为，那么Python为这种将来会出现的扩展提供了一个简单的途径。在这种情况下，使用属性去隐藏属性数据访问背后的逻辑。  
　　　注意1：属性只在new-style类中起作用。  
　　　注意2：尽管功能方法对于类似缓存的负面影响比较小，但还是要尽量避免。  
　　　注意3：属性标记会让调用者认为开销（相当的）小，避免用属性做开销大的计算。

如果你的类打算用来继承的话，并且这个类里有不希望子类使用的属性，就要考虑使用双下划线前缀并且没有后缀下划线的命名方式。这会调用Python的命名转换算法，将类的名字加入到属性名里。这样做可以帮助避免在子类中不小心包含了相同的属性名而产生的冲突。  
　　　注意1：只有类名才会整合进属性名，如果子类的属性名和类名和父类都相同，那么你还是会有命名冲突的问题。  
　　　注意2：命名转换会在某些场景使用起来不太方便，例如调试，**getattr**()。然而命名转换的算法有很好的文档说明并且很好操作。  
　　　注意3：不是所有人都喜欢命名转换。尽量避免意外的名字冲突和潜在的高级调用。

7.4 Public and internal interfaces 公共和内部的接口  
任何向后兼容保证只适用于公共接口，因此，用户清晰地区分公共接口和内部接口非常重要。  
文档化的接口被认为是公开的，除非文档明确声明它们是临时或内部接口，不受通常的向后兼容性保证。所有未记录的接口都应该是内部的。  
为了更好地支持内省（introspection），模块应该使用\_\_all\_\_属性显式地在它们的公共API中声明名称。将\_\_all\_\_设置为空列表表示模块没有公共API。  
即使通过\_\_all\_\_设置过，内部接口（包，模块，类，方法，属性或其他名字）依然需要单个下划线前缀。  
如果一个命名空间（包，模块，类）被认为是内部的，那么包含它的接口也应该被认为是内部的。  
导入的名称应该始终被视作是一个实现的细节。其他模块必须不能间接访问这样的名称，除非它是包含它的模块中有明确的文档说明的API，例如 os.path 或者是一个包里从子模块公开函数接口的**init**模块。

**附录2 常用编程缩写**

A  
absolute -> abs  
address -> addr  
application -> app  
argument -> arg  
arguments -> args  
array -> arr  
assemble -> asm  
asynchronize -> async  
attribute -> attr  
attributes -> attrs  
average -> avg

B  
background -> bg  
button -> btn  
buffer -> buf

C  
calculate -> calc  
configure -> conf  
control -> ctrl  
count -> cnt  
current -> curr

D  
database -> db  
delete -> del  
description -> desc  
destination -> dest  
document -> doc

E  
error -> err  
escape -> esc

F  
function -> fn

I  
image -> img  
increment -> inc  
index -> idx  
information -> info  
initialize -> init  
insert -> ins

J  
javascript -> js

L  
length -> len  
library -> lib

M  
manager -> mgr  
message -> msg

N  
number -> num

O  
object -> obj  
option -> opt  
options -> opts

P  
parameter-> param  
password -> pwd  
picture -> pic  
point -> pt  
position -> pos  
previous -> prev  
property -> prop

S  
source -> src  
string -> str

T  
temporary -> tmp

V  
value -> val

W  
window -> wnd

命名缩写  
通用    缩写    翻译    控件    缩写    翻译  
address    addr    地址    calendar    cdr    日历  
application    app    应用程序    messageDialog    msgdlg    消息框  
asynchronization    asyn    异步    drawer    drw    抽屉  
average    avg    平均数    buttonGroup    btngrp    按钮分组  
bitmap    bmp    位图    checkBox    chk    复选框  
buffer    buf    缓冲区    container    cntr    容器  
character    char    字符    button    btn    按钮  
color    clr    颜色    comboBox    cmb    下拉框  
command    cmd    命令    lable    lbl    标签  
delete    del    删除    progressBar    prg    进度条  
destination    dest/dst    目的地    pageIndicator    pgindic    Tab分页控件  
decrease    dec    减少    radioButton    rdo    单选框  
device    dev    设备    rangeSlider    rngsld    滑块  
different    diff    不同的    scrollView    svw    视图滚动  
directory    dir    目录    scrollBar    vsb(垂直)/hsb(水平)    滚动条  
document    doc    文档    slider    sld    滑动器  
dynamic    dyna    动态的    spinBox    spn    旋转框  
environment    env    环境    splitView    spltvw    分割视图  
error    err    错误    stackView    stackvw    堆栈视图  
execute    exec    执行    statusBar    statbr    状态条  
group    grp    组    swipeView    swpvw    滑动控件  
image    img    图像    switch    sw    开关  
information    info    信息    tabBar    tbbr    tab栏  
initialize    init    初始化    textField    txtfld    文本输入框  
library    lib    库    toolBar    tlbr    工具栏  
maximum    max    最大值    toolTip    tltp    提示框  
minimum    min    最小值    toolButton    tlbtn    工具按钮  
message    msg    消息    treeView    trvw    树形视图  
number    num    数字，数，数量    layout    lyt    布局  
object    obj    对象    mainMenu    mmnu    菜单栏  
parameter    Param    参数(形参)    groupBox    gbx    组合框  
package    pkg    打包    pictureBox    pic    图片框  
position    pos    位置    listView    lvw    列表视图  
previous    pre    前一个    window    win(wnd)    窗口  
process/Procedure    proc    进程/过程    form    frm    表单  
pointer    ptr    指针    textBox    txt    文本框  
password    pwd    密码    linkLabel    llbl    超链接标签框  
public    pub    公共的    dialog    dlg    对话框  
reference    ref    引用    scrollIndicator    scrlindic    滚动指示器  
source    src    源              
string    str    字符串              
summation    sum    和              
synchronization    sync    同步              
system    sys    系统              
table    tbl    表格              
temporary    tmp    临时              
text    txt    文本              
variable    var    变量              
addition    add    加法/添加              
argument    arg    实参              
array    arr    数组              
calculate    calc    计算              
configuration    config    配置              
column    col    列              
control    ctrl    控制              
hexadecimal    hex    十六进制              
original    orig    原件              
return    rtn    返回              
repeat    rpt    重复              
signal    sig    信号              
status    stat    状态              
stack    stk    栈区              
standard    std    标准              
trigger    trig    触发              
background    bg    背景              
escape    esc    退出              
user    usr    用户              
list    lst    列表              
server    srv    服务              
manager    mgr    管理者              
insert    ins    插入              
point    pt    点              
function    func    函数              
back    bk    返回              
break    brk    间断              
data    dat    日期              
edit    edt    编辑              
flag    flg    标志              
grid    grd    网格              
increment    inc    增量              
length    len    长度              
oracle    ora    甲骨文数据库              
panorama    pano    全景              
print    prn    打印              
program    prg    程序              
statistic    stat    统计              
administrator    adm    管理员              
database    db    数据库              
change    chg    改变              
click    clk    点击              
compare    cmp    比较              
coordinates    coord    坐标              
copy    cpy    复制              
current    cur    当前的              
dictionary    dict    字典              
display    disp    显示              
driver    drv    驱动              
extend    ex/ext    扩展              
frame    frm    框架/帧              
index    idx / ndx    索引              
horizontal    horz    水平              
instance    ins    实例              
increase    inc    增加              
link    lnk    链接              
middle    mid    中间              
multiply    mul    乘              
resource    res    资源              
select    sel    选择              
test    tst    测试              
vertical    vert    垂直