1. 函数的调用：函数名称、参数
2. 函数的定义：函数名、参数
3. Module或Library与Package区别：

Module就是Python文件，而python file一般后缀为py，即xxx.py；

Library 库，也称为：库文件。之所以此处不说是Python中的library，那是因为，本身library这个词，一般都是针对其他的编译型语言，比如C，C#等语言来说的。常见的C/C#等语言中的library，一般指的就是：

静态的库文件：xxx.a 动态的库文件：xxx.dll

Package，可以理解为，一组的module，一堆（相关的）module组合而成的。

（1）、package是模块的集合，每一个Package的根目录下面都必须有一个\_\_init\_\_.py 文件。当解释器发现目录下有这个文件时，他就会认为这是一个Package，而不是一个普通的目录。\_\_init\_\_.py 文件定义了包的属性和方法，可以只是一个空文件，但是必须存在。

（2）、导入机制：

导入单个的module：import my\_module;

导入package一般是这样的：from my\_module.timing import function（从包中导入函数？？）

（3）、在不同层级目录中import模块的方法：

A、同级目录下的调用：

-- src

|-- mod1.py

|-- test1.py

若在程序test1.py中导入模块mod1, 则直接使用import mod1 或 from mod1 import \*；

B、调用子目录下的模块：

-- src

|-- mod1.py

|-- lib

| |-- mod2.py

|-- test1.py

这时看到test1.py和lib目录（即mod2.py的父级目录），如果想在程序test1.py中导入模块mod2.py ，可以在lib件夹中建立空文件\_\_init\_\_.py文件(也可以在该文件中自定义输出模块接口)，然后使用：

from lib.mod2 import \* 或 import lib.mod2

C、调用上级目录下的文件

-- src

|-- mod1.py

|-- lib

| |-- mod2.py

|-- sub

| |-- test2.py

这里想要实现test2.py调用mod1.py和mod2.py ，做法是我们先跳到src目录下面，直接可以调用mod1，然后在lib上当下建一个空文件\_\_init\_\_.py ，就可以像第二步调用子目录下的模块一样，通过import lib.mod2进行调用了。具体代码如下：

import sys

sys.path.append("..")

import mod1

import mod2.mod2

注意要点：C1、sys.path添加目录时注意是在windows还是在Linux下，windows下需要‘\\’否则会出错，即sys.path.append('C:\\test\\A\\C')；C2、可见，导入模块关键是能够根据sys.path环境变量的值，找到模块的绝对路径。

D、\_\_init\_\_.py补充：

\_\_init\_\_.py 文件定义了包的属性和方法，可以只是一个空文件，但是必须存在。

\_\_init\_\_.py 中有一个重要的变量，即\_\_all\_\_，用于控制将要引用被的模块。

1. 封装、继承、多态：

静态方法、动态方法：

1. 类class、实例instance、对象object：
2. Self、\_\_init\_\_：

self.valueName: 访问对应的变量（读取或者写入）; self.function():调用对应的函数时。

\_\_init\_\_:

1. 变量作用域variable scope

全局变量gName：

局部变量：