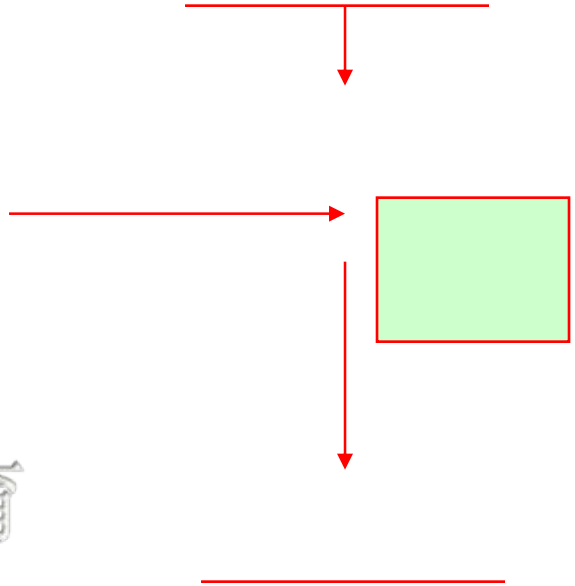
马哥教育

主讲：马永亮(马哥)
QQ:113228115
客服QQ：2813150558，1661815153
http://www.magedu.com
http://mageedu.blog.51cto.com

- 编程语言
  - 用户：
    - 问题空间
  - 计算机：解决问题
    - 解空间
    - 抽象
    - 机器代码 → 微码编程 → 高级语言



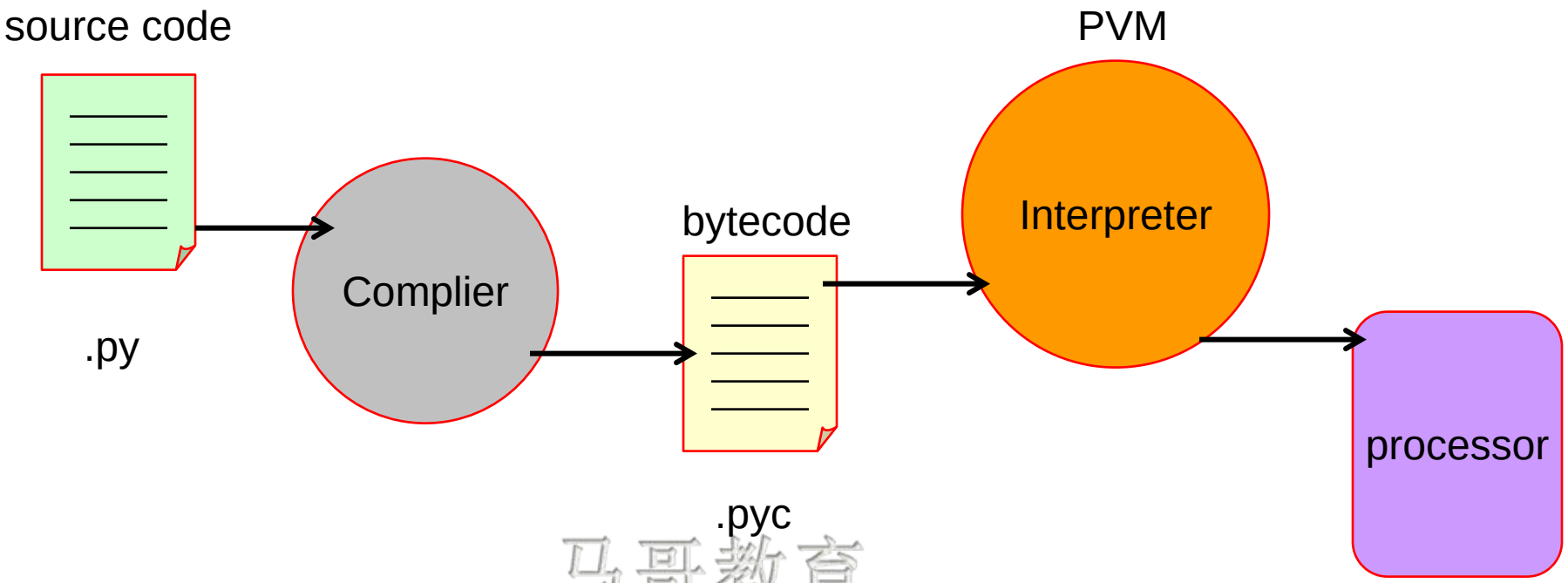
马哥教育

Python编程基础

主讲：马永亮(马哥)
QQ:113228115
客服QQ：2813150558，1661815153
http://www.magedu.com
http://mageedu.blog.51cto.com

- Python
  - 英国发音：/ˈpaɪθən/
  - 美国发音：/ˈpaɪθɑːn/

Python程序的执行



Python的实现

- CPython
  - 原始、标准的实现方式
- Jython
  - 用于与Java语言集成的实现
- IronPython
  - 用于与.NET框架集成的实现

Python性能优化工具

- Psyco
  - Python 语言的一个扩展模块，可以即时对程序代码进行专业的算法优化，可以在一定程度上提高程序的执行速度，尤其是在程序中有大量循环操作时
  - 目前开发工作已经停止，由PyPy所接替

- PyPy
  - PyPy是用Python实现的Python解释器
  - Python语言的动态编译器，是Psyco的后继项目
  - 可以运行在Linux的32位和64位、MacOSX和Windows的32位平台中

- Shed Skin
  - Python编译器，能够将python代码转换成优化的C++代码

编写、执行Python代码

- 交互式解释器
  - 直接启动python，其显示信息取决于程序版本及操作系统等

```
[root@www.magedu.com ~]# python
Python 2.7.6 (default, Feb 19 2014, 12:02:31)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-4)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print "Hello World!"
Hello World!
>>>
```

- python程序文件
  - 交互式模式下的程序执行完成后难以再次运行
  - 将编写的程序保存至文件(.py)中方便多次运行
    - python的此类包含了一系列预编写好的语句的程序文件称作"模块"
    - 能够直接运行的模块文件通常称作脚本(即程序的顶层文件)

python程序文件

- python源程序文件通常以.py为扩展名
  - 例如，新建一个名为firstpycode.py的文件，内容如下所示

```
#!/usr/bin/python
import platform
print platform.uname()
```

  - 第一行为shebang，即执行脚本时通知内容要启动的解释器
  - 第二行通过import导入一个python模块platform
  - 第三行打印platform模块的uname方法的执行结果
- 给予此脚本以执行权限，并执行即可

```
[root@www.magedu.com ~]# chmod +x firstpycode.py
[root@www.magedu.com ~]# ./firstpycode.py
('Linux', 'www.magedu.com', '2.6.32-431.el6.x86_64', '#1 SMP Fri Nov 22 03:15:
09 UTC 2013', 'x86_64', 'x86_64')
```

python程序文件

- Python程序可以分解成模块、语句、表达式和对象
  - 程序由模块构成
  - 模块包含语句
  - 语句包含表达式
  - 表达式建立并处理对象
    - 表达式是"某事"，而语句是"做某事(即指令)"；
      - 例如，"3+4"是某事，而"print 3+4"则是做某事；
    - 语句的特性：它们改变了事物，例如，赋值语句改变了变量，print语句改变了屏幕输出等；

Python IDE

- IDLE
  - 标准python环境提供
- Eclipse和PyDev
- PythonWin
- Komodo
- Wingware
- PyCharm

马哥教育

Python过程型程序设计快速入门

主讲：马永亮(马哥)
QQ:113228115
客服QQ：2813150558，1661815153
http://www.magedu.com
http://mageedu.blog.51cto.com

数据结构

- 数据结构
  - 通过某种方式(例如对元素进行编号)组织在一起的数据元素的集合，这些数据元素可以是数字或者字符，甚至可以是其它的数据结构
  - Python的最基本数据结构是序列
  - 序列中的每个元素被分配一个序号——即元素的位置，也称为索引；索引从0开始编号
  - Python包含6种内建的数据序列：列表、元组、字符串、Unicode字符串、buffer对象和xrange对象

Python的关键要素

- 基本数据类型
- 对象引用
- 组合数据类型
- 逻辑操作符
- 控制流语句
- 算术操作符
- 输入/输出
- 函数的创建与调用

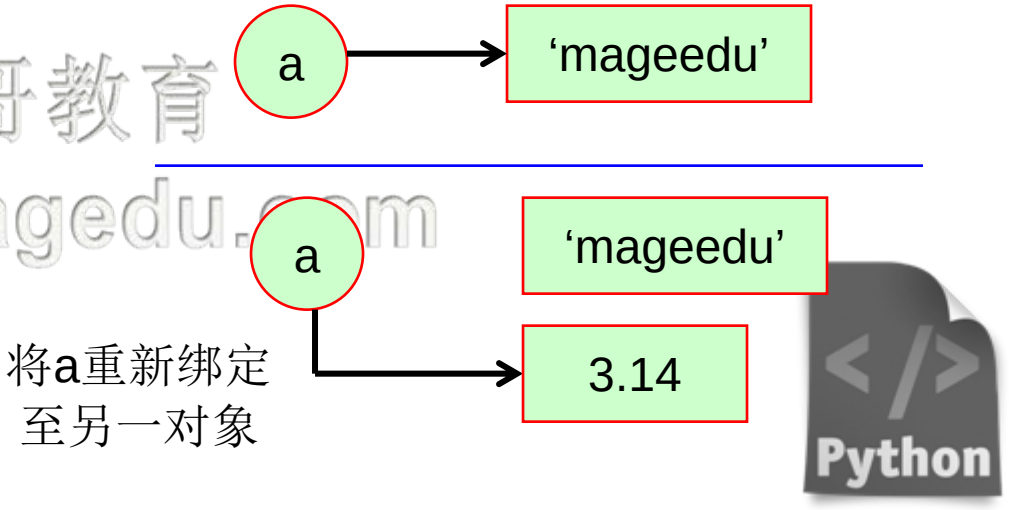
要素# 1：基本数据类型

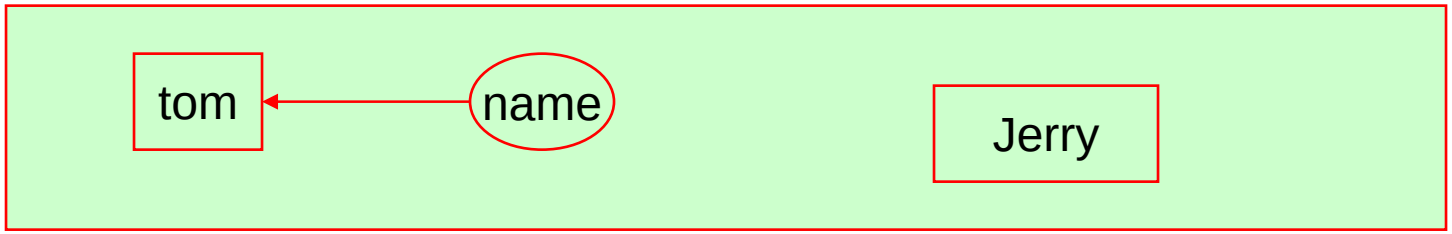
- 任何程序语言都必须能够表示基本数据项
- Python中的基本数据类型有
  - Integral类型
    - 整型：不可变类型
      - -257, 20162458333711437339583 6
    - 布尔型
      - True, False
  - 浮点类型
    - 浮点数
      - 3.141592
    - 复数
      - 3+6j
    - 十进制数字
  - 字符串
    - 'GNU is Not Unix', "hello", "'world'"

要素# 2：对象引用(变量)

- Python将所有数据存为内存对象
- python中，变量事实上是指向内存对象的引用
- 动态类型：在任何时刻，只要需要，某个对象引用都可以重新引用一个不同的对象(可以是不同的数据类型)
- 内建函数type()用于返回给定数据项的数据类型
- "="用于将变量名与内存中的某对象绑定：如果对象事先存在，就直接进行绑定；否则，则由"="创建引用的对象

```
>>> a='mageedu'
>>> type(a)
<type 'str'>
>>> a=3.14
>>> type(a)
<type 'float'>
```



要素# 2：对象引用(变量)

- 变量命令规则
  - 只能包含字母、数字和下划线，且不能以数字开头
  - 区分字母大小写
  - 禁止使用保留字
    - Python2与Python3的保留字有所不同
- 命令惯例
  - 以单一下划线开头变量名(_x)不会被from module import *语句导入
  - 前后有下划线的变量名(__x__)是系统定义的变量名，对python解释器有特殊意义
  - 以两个下划线开头但结尾没有下划线的变量名(__x)是类的本地变量
  - 交互式模式下，变量名"_"用于保存最后表达式的结果
- 注意：变量名没有类型，对象才有

要素#3：组合数据类型

- 数据结构：通过某种方式(例如对元素进行编号)组织在一起的数据元素的集合
- Python常用的组合数据类型
  - 序列类型
    - 列表：使用[]创建，如['Call','me', 'Ishmeal','.']
    - 元组：使用()创建，如('one', 'two')
    - 字符串也属于序列类型
  - 集合类型
    - 集合
  - 映射类型
    - 字典

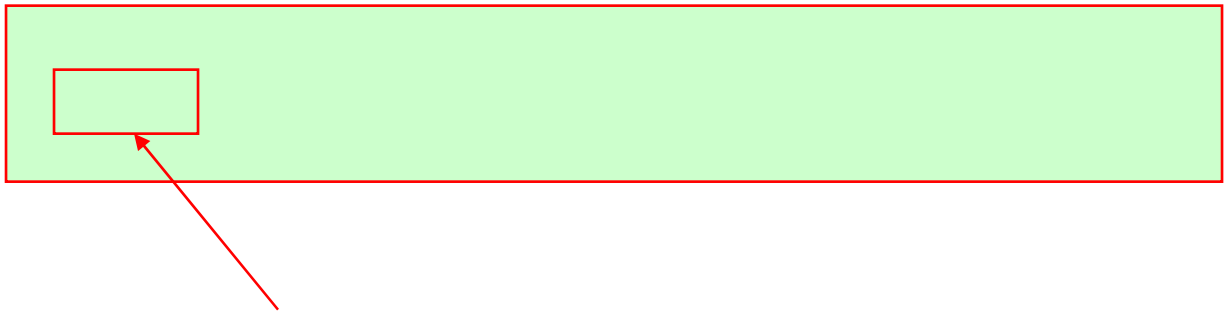
0	Call
1	me
2	Ishmael
3	.

要素#3：组合数据类型

- 列表是可变序列，元组是不可变序列
- Python中，组合数据类型也是对象，因此其可以嵌套
  - ['hello','world', [1,2,3]]
- 实质上，列表和元组并不真正存储数据，而是存放对象引用
- Python对象可以具有其可以被调用的特定"方法（函数）"
- 元组、列表以及字符串等数据类型是"有大小的"，也即，其长度可使用内置函数len()测量；

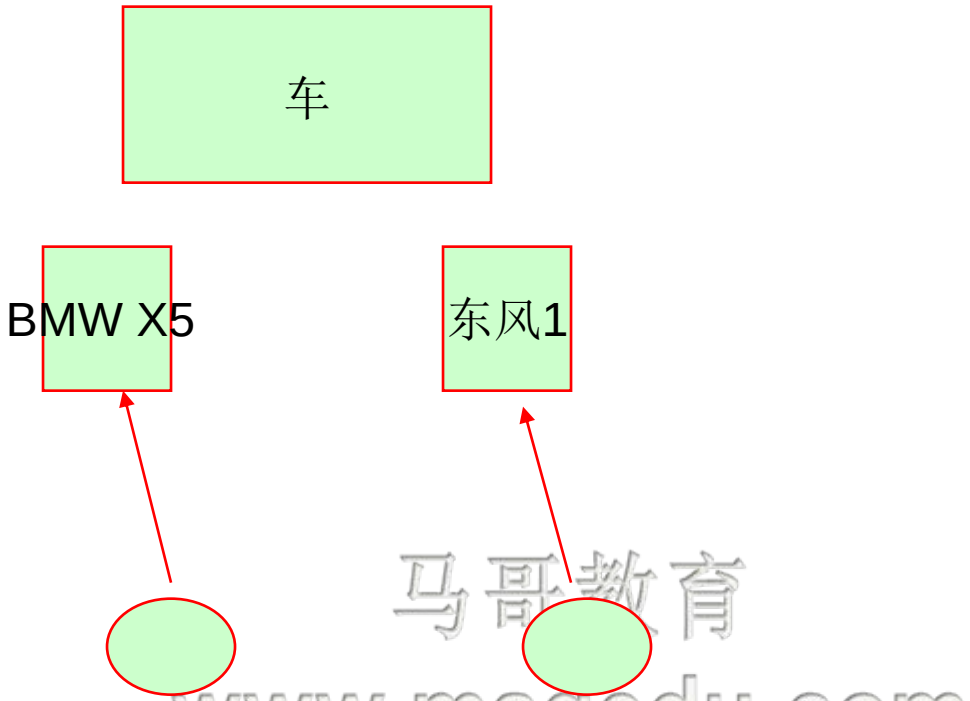
```
>>> myList=['one','two']
>>> type(myList)
<type 'list'>
>>> len(myList)
2
>>> myList.append('three')
>>> print myList
['one', 'two', 'three']
```

列表对象的append()方法可用于为其补充新元素



要素#4：逻辑操作符

- 逻辑运算是任何程序设计语言的基本功能
- Python提供了4组逻辑运算
  - 身份操作符
    - is：判定左端对象引用是否相同于右端对象引用；也可以与None进行；
  - 比较操作符
    - <，>，<=，>=，!=，==
  - 成员操作符
    - in或not in：测试成员关系
  - 逻辑运算符
    - and，or，not



要素#5：控制流语句

- 控制流语句是过程式编程语言的基本控制机制
- Python的常见控制流语句
  - if
  - while
  - for...in
  - try

```
if boolean_expression1:
    suite1
elif boolean_expression2:
    suite2
...
else:
    else_suite
```

```
while boolean_expression:
    suite
```

```
for variable in iterable:
    suite
```


要素#6：算术操作符

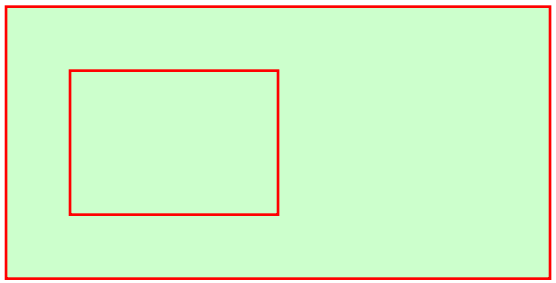
- Python提供了完整的算术操作集
- 很多的Python数据类型也可以使用增强的赋值操作符，如+=、-=等；
- 同样的功能，使用增强型赋值操作符的性能较好；
- Python的int类型是不可变的，因此，增强型赋值的实际过程是创建了一个新的对象来存储结果后将变量名执行了重新绑定

要素#7：输入/输出

- 现实中，具有实际功能的程序必须能够读取输入(如从键盘或文件中)，以及产生输出，并写到终端或文件中；
- Python的输入/输出
  - 输出
    - Python3：print()函数
    - Python2：print语句
  - 输入
    - input()
    - raw_input()

要素#7：输入/输出

- Python解释器提供了3种标准文件对象，分别为标准输入、标准输出和标准错误，它们在sys模块中分别以sys.stdin、sys.stdout和sys.stderr形式提供
- Python的print语句实现打印——一个对程序员友好的标准输出流接口
- 从技术角度来讲，print是把一个或多个对象转换为其文本表达形式，然后发送给标准输出或另一个类似文件的流
  - 在Python中，打印与文件和流的概念联系紧密
    - 文件写入方法是把字符串写入到任意文件
    - print默认把对象打印到stdout流，并添加了一些自动的格式化
  - 实质上，print语句只是Python的人性化特性的具体实现，它提供了sys.stdout.write()的简单接口，再加一上些默认的格式设置
  - print接受一个逗号分隔的对象列表，并为行尾自动添加一个换行符，如果不需要，则在最后个元素后添加逗号



要素#7：输入/输出

- print "String %format1 %format2 ..." %(variable1, varialbe2, ...)

字　符	输出格式
d, i	十进制整数或长整数
u	无符号整数或长整数
o	八进制整数或长整数
x	十六进制整数或长整数
X	十六进制整数（大写字母）
f	浮点数，如[-]*m.dddddd*
e	浮点数，如[-]*m.ddddddexx*
E	浮点数，如[-]*m.ddddddExx*
g, G	指数小于-4或更高精度时使用%e或%E，否则使用%f
s	字符串或任意对象。格式化代码使用str()生成字符串
r	同repr()生成的字符串
c	单个字符
%	字面量%

要素#7：输入/输出

- %后面可以使用的修饰符，(如果有，则只能按如下顺序)
  - %[(name)][flags][width][.precision]typecode
    - 位于括号中的一个属于后面的字典的键名，用于选出一个具体项
    - 下面标志中的一个或多个
      - -：表示左对齐，默认为右对齐
      - +：表示包含数字符号，正数也会带"+"
      - 0：表示一个零填充
    - 一个指定最小宽度的数字
    - 一个小数点，用于按照精度分割字段的宽度
    - 一个数字，指定要打印字符串中的最大字符个数，浮点数中小数点之后的位数，或者整数的最小位数;
    - 例子：
      - d={'x':32, 'y':27.490325, 'z':65}
      - print "%(x)-10d %(y)0.3g" % d

要素#8：函数的创建与调用

- 函数是实现模块化编程的基本组件
- Python使用def语句定义函数

```
def functionName(arguments):
    suite
```

- 函数可以参数化，通过传递不同的参数来调用
- 每个Python函数都有一个返回值，默认为None，也可以使用"return value"明确定义返回值
- def语句会创建一个函数对象，并同时创建一个指向函数的对象引用
  - 函数也是对象，可以存储在组合数据类型中，也可以作为参数传递给其它函数
  - callable()可用于测试函数是否可调用


```
>>> def testFunc(arg1):
...     print arg1
...
>>> callable(testFunc)
True
>>> testFunc('hello world!')
hello world!
```

要素#8：函数的创建与调用

- Python有众多内置函数
- Python标准库拥有众多内置模块，这些模块拥有大量函数
  - Python模块实际上就是包含Python代码的.py文件，其拥有自定义的函数与类及变量等
  - 导入模块使用import语句进行，后跟模块名称(不能指定模块文件名的后缀.py)
  - 导入一个模块后，可以访问其内部包含的任意函数、类及变量

```
>>> import random
>>> x=random.choice(['a','b','c','d','e','f','g'])
>>> print x
d
```


马哥教育

Python编程基础及编程风格

主讲：马永亮(马哥)
QQ:113228115
客服QQ：2813150558，1661815153
http://www.magedu.com
http://mageedu.blog.51cto.com

- 语句和语法
- 标识符
- 基本编程风格

语句和语法

- 注释
  - #：可以从一行的任何地方开始
- 续行
  - \：
  - '''：闭合操作符，单一语句跨多行
- 代码组
  - 缩进相同的一组语句构成的一个代码块
  - 首行以关键字开始，如if、while等，以冒号结束
  - Python使用缩进来分隔代码组，同一代码组的代码行必须严格左对齐，否则会造成语法错误

语句和语法

- 同一行放置多个语句
  - ;:以分号做为分隔符
- 模块
  - 每一个Python脚本文件都可以被当成是一个模块
  - 模块里的代码可以是一段直接执行的脚本，也可以是一些类似库函数的代码从而可由别的模块执行导入(import)

标识符

- 标识符是计算机语言中允许作为名字的有效字符串集合
  - 其中有一部分是关键字，它们是语言的标识符，因此是保留字，不能用于其它用途
  - Python还有称为"内建"的标识符集合，虽不是保留字，仍不推荐使用这些特别的名字
- Python标识符
  - 第一个字符只能使用字母或下划线
  - 余下的字符可以使用字母、数字或下划线
  - 区分字符大小写

Python 3的关键字

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Python基本编程风格

- 注释
  - 既不能缺少注释，亦要避免过渡注释
- 文档
  - Python允许通过__doc__动态获得文档字串
- 缩进
  - 统一缩进4个字串
- 标识符名称
  - 见名知义
- Python风格指南
  - https://code.google.com/p/soc/wiki/PythonStyleGuide
  - 译文：http://www.elias.cn/Python/PythonStyleGuide

Python命名惯例

- 以单一下划线开头的变量名(_x)不会被from module import *语句导入
- 前后有下划线的变量名(__x__)是系统变量名，对解释器有特殊意义
- 以两个下划线开头、但结尾没有下划线的变量名(__x)是类的本地变量
- 交互式模式下，只有单个下划线的变量名(_)用于保存最后表达式的结果

Python文件结构

```
#/usr/bin/env python
```
(1) 起始行

```
"this is a test module"
```
(2) 模块文档（文档字符串）

```
import sys
import os
```
(3) 模块导入

```
debug = True
```
(4)（全局）变量定义

```
class FooClass (object):
    "Foo class"
     pass
```
(5) 类定义（若有）

```
def test():
    "test function"
    foo = FooClass()
    if debug:
       print 'ran test()'
```
(6) 函数定义（若有）

```
if __name__ == '__main__':
    test()
```
(7) 主程序

Python文件主程序

- 主程序
  - 无论当前模块是被别的模块导入还是作为脚本直接执行，都会执行这部分代码
- 注意：所有的模块都有能力执行代码
  - 最高级别的Python语句(没有缩进的)在模块被导入时就会执行，无论是否真的需要执行
  - 妥当的做法：除了那些真正需要执行的代码以外，所有的功能代码都通过函数建立，因此
    - 仅在主程序模块中编写大量的顶级可执行代码
    - 用于被导入的模块只应该存在较少的顶级执行代码
- __name__指示模块应该如何被加载
  - 如果模块是被导入，__name__的值是模块名字
  - 如果模块是直接执行，__name__的值是'__main__'

关于马哥教育

- 博客：http://mageedu.blog.51cto.com
- 主页：http://www.magedu.com
- QQ：2813150558，1661815153，113228115
- QQ群：203585050，279599283

马哥教育

Thank You!