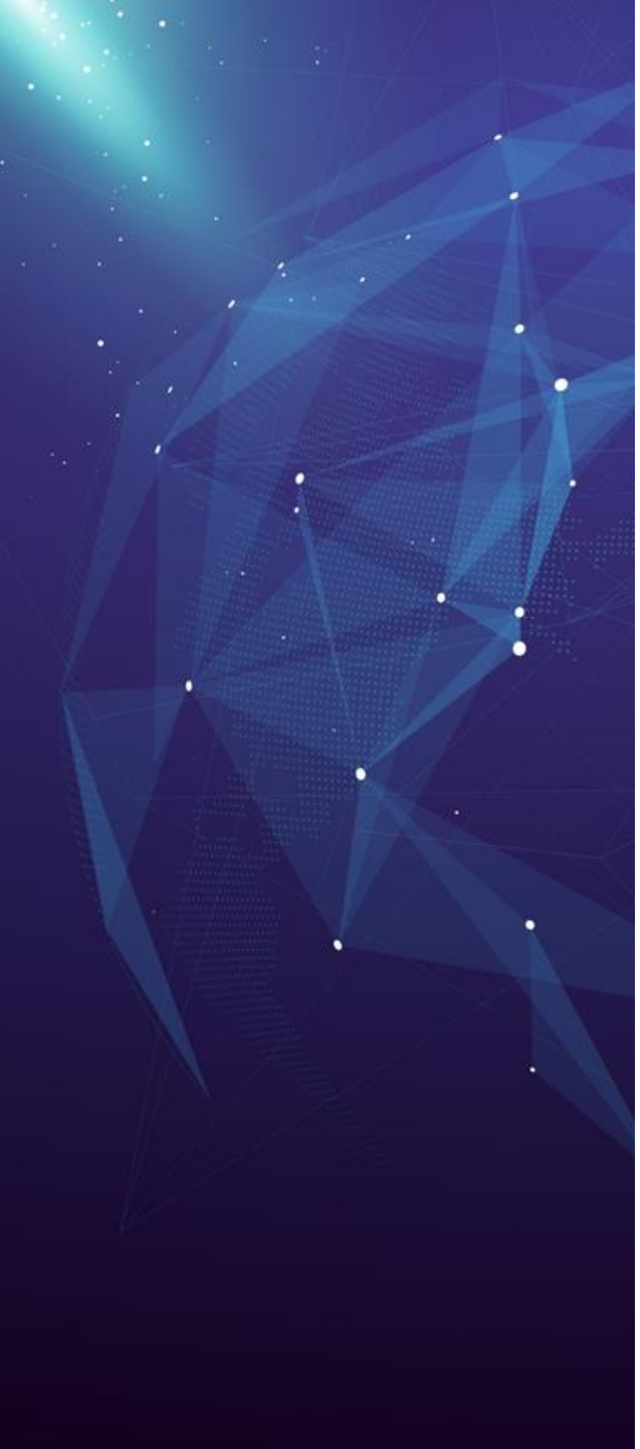
Python基础（一）

蓝鲸智云讲师



课程内容

- Python 简介
- Hello world 程序
- Python 语法
- 基础数据类型
- 类、动态特性
- 程序结构
- 内置模块
- 第三方模块

Python 简介

Python 简介

1989年的圣诞节期间，吉多·范罗苏姆开发了一个新的脚本解释器，并命名为 Python，作为 ABC 语言的一种继承。

ABC 是专门为非专业程序员设计的一种教学语言。

特点

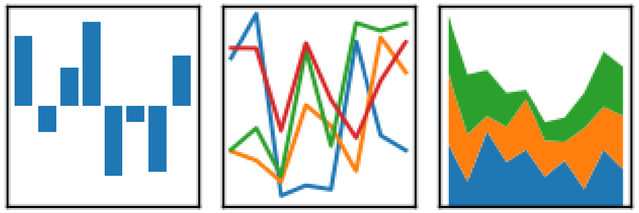
- 解释型语言：开发过程中没有编译这个环节
- 交互式语言：可以在一个 Python Shell 里，直接互动执行写你的程序
- 面向对象语言：支持面向对象的风格或代码封装在对象的编程技术
- 初学者的语言：易于学习，有丰富的标准库/模块

Python 简介 - 应用领域

- 网站后台
- 网络爬虫
- 科学计算
- 数据挖掘
- 机器学习
- 系统运维

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Hello world 程序

Python 之 “Hello world”

有两种方式可以执行 Python 代码

- 交互式命令行

直接输入 Python 进入交互模式，每输入一行就执行一行

```
In [3]: print 'Hello world'
Hello world

In [4]:
```

- Py 文件

用文本编辑器（Sublime、Atom，不要使用记事本）写 Python程序，然后保存为后缀为.py的文件，就可以用P ython 直接运行这个程序了

```
shaowenchen@SHAOWENCHEN-PC1 C:\Users\shaowenchen\Desktop
$ cat hello.py
print 'Hello world'
shaowenchen@SHAOWENCHEN-PC1 C:\Users\shaowenchen\Desktop
$ python hello.py
Hello world
```

Python 的交互式 Shell

Ipython

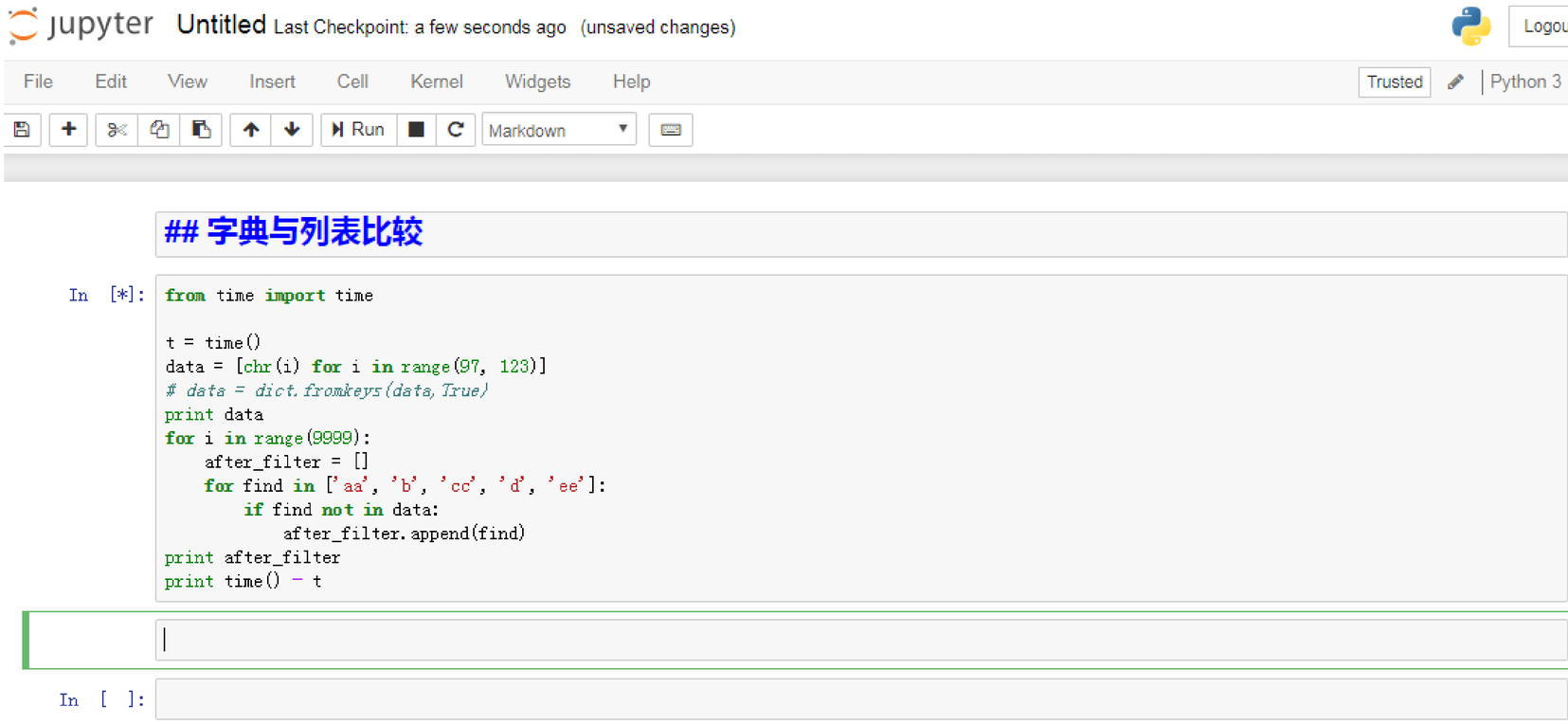
一个增强的交互式 Python shell，tab补全，对象自省，强大的历史机制，内嵌的源代码编辑，集成Python调试器，%run机制，宏，创建多个环境以及调用系统shell的能力

$ pip install ipython

Jupyter Notebook

$ pip install jupyter

$ jupyter notebook



__main__ 函数

- if __name__ == "__main__"

当 Python 解释器读取一个源文件时，它会执行所有的代码。在执行代码前,会定义一些特殊的变量。

__name__ 就是其中之一。

如果解释器运行的模块（源文件）作为主程序，它将会把 __name__ 设置成“__main__”。如果只是引入其他的模块，__name__将会设置成模块的名字。

```
$ cat main_test.py
if __name__ == '__main__':
    print 'I am in main'
else:
    print 'I am not in main'
print __name__
shaowenchen@SHAOWENCHEN-PC1 C:\Users\shaowenchen\Desktop
$ python main_test.py
I am in main
__main__

shaowenchen@SHAOWENCHEN-PC1 C:\Users\shaowenchen\Desktop
$ ipython
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (Intel)]
Type "copyright", "credits" or "license" for more information.

IPython 5.4.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: import main_test
I am not in main
main_test
```

Python 语法

Python 语法

- Python 代码块通过缩进对齐表达代码逻辑，而不是使用大括号。

```
class TicketBackend(ModelBackend):

    def authenticate(self, ticket=None, request=None):
        account = AccountFactory.getAccountObj()
        status, user = account.check_backend_login_status(request)

        if not status or not user:
            return None
        return user
```

Python 语法 – 变量名

- 变量由字母、数字、下划线组成，区分大小写
- 保留字符 Python 2.7 有 31 个

```
In [7]: import keyword

In [8]: len(keyword.kwlist)
Out[8]: 31

In [9]: print keyword.kwlist
['and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'e
xec', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'not', 'or', 'pa
ss', 'print', 'raise', 'return', 'try', 'while', 'with', 'yield']

In [10]:
```

Python 语法 – 多行、字符串、注释

- 使用斜杠（ \ ）将一行的语句分为多行显示

```
total = item_one + \
        item_two + \
        item_three
```

- 使用引号(')、双引号(")、三引号(''' 或 """) 来表示字符串

```
word = ‘字符串’
sentence = “这是一个句子。”
paragraph = “““这是一个段落，
可以由多行组成”””
```

- 单行注释采用 # 开头，多行注释使用三个单引号(''')或三个双引号(""")。

```
# 第一个注释
print ‘Hello, Python!’
‘‘‘
多行注释
多行注释
’’’
“““
多行注释
多行注释
”””
```

Python 语法 – 输入、输出

- 输入

1. raw_input(“请输入一个整数”)

- 输出

```
x = ‘a’
y = ‘b’
# 换行输出
print x
print y

print ‘------’
# 不换行输出
print x,
print y,

# 不换行输出
print x, y
```

基础数据类型

Python 基础数据类型

在 Python 程序中，每个数据都是对像，每个对像都有自己的一个类型。不同类型有不同的操作方法，使用内置数据类型独有的操作方法，可以更快的完成很多工作。

- 数值 – int、float等
- 字符串 - str
- 布尔型 – bool
- 列表 – list
- 元组 - tuple
- 字典 - dict
- 集合 - set

Python 基础数据类型- Numbers

Python 的数值类型

- int（有符号整型）10
- long（长整型）11111111111111L
- float（浮点型）

  >>> 0.1 + 0.1 + 0.1 - 0.3

  5.55111512312578 3e-17

  误差来自十进制转换为计算机内部二进制时，精度丢失。

  使用decimal进行十进制浮点运算

  >>> Decimal('0.1') +Decimal('0.1') +Decimal('0.1') -Decimal('0.3')

  Decimal('0.0')

- complex（复数）

  4.53-7j

Python 基础数据类型- String

s = ‘ilovepython’ #str(basestring)

s2 = u’ 你好’ #Unicode(basestring)

→ step>0

0	1	2	3	4	5	6	7	8	9	10
i	l	o	v	e	p	y	t	h	o	n
-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

← step<0

➢ 切片

s[start : end : step]

正向索引：从左到右，默认0向上加，范围是0到字符串长度-1

反向索引：从右到左，默认-1向下减，范围是-1到字符串长度*-1

```
s[0]        # I
s[1:5]          # love    正向索引
s[-6:]      #python反向索引
s[0:10] != s[0:]
s[::-1]         #nohtypevoli      # 等价于s[-1::-1]        # step为负数，从右开始
```

➢ 字符串常用方法

startswith()/endswith()/count()/find()/index()/ join()/replace()/split()/strip()

Python 基础数据类型- List

seq = ['physics', 'chemistry', 1997, 2000]

- 访问列表中的值：

  seq[1]

  seq[0:1]

- 列表操作

  [1, 2, 3][0:1] #[1]

  [1] + [2] #[1, 2]

  [1] * 2 #[1, 1]

  ‘a’ in [‘a’ , ‘b’] # True

- 常用列表方法

  append()

  extend() # a.extend(b) 等价于 a+=b

  insert()/pop()/sort()/count()/index()

Python 基础数据类型- Tuple

seq = ('physics', 'chemistry', 1997, 2000)

seq = 'physics', 'chemistry', 1997, 2000　任意无括号的对象，以逗号隔开，默认为元组

- 访问元组

  参考列表(list)

- 元组不可变

  元组中的元素值不允许修改，元组是不可变的，或者说只读。

  元组操作依然和列表一样是可用的

  ```
  t = (1, 2)
  t = t + (3,)          # t: (1, 2, 3)
  ```

- 元组方法

  count()

  index()

Python 基础数据类型- Dictionary

dict2 = { 'abc': 123, 98.6: 37, (1,2): 200 }

字典的每个键值（key=>value）对用冒号:分割，每个对之间用逗号,分割，整个字典包括在花括号{}中

字典基本操作

```
dict['abc']          #取值
dict['abc'] = 321 #修改
del dict['abc']     #删除
```

字典特性

不允许同一个键出现两次。创建时如果同一个键被赋值两次，后一个值会将原值替换。

键必须不可变（hashable），所以可以用数字，字符串或元组充当，所以不能使用列表。

常用字典方法

clear() pop() fromkeys() get() has_key() items() setdefault() update() keys() values()

Python 基础数据类型- Set

set1 = {1, 2, 3}

set2 = {2, 3, 4} 和列表类似，使用大括号取代列表的中括号

集合基本操作

```
2 in set2
set1 | set2          #并集          {1,2,3,4}
set1 & set2          #交集          {2,3}
set1 - set2          #差集          {1}
set1 ^ set2          #对称差集      {1, 4} 等价于 (set1-set2) | (set2-set1)
```

集合特性

元素是无序的

元素是不可变的（hashable）

元素是不重复的

常用集合方法

add()/remove()/discard()

类、动态特性

Python 类

类(Class)和实例(Instance)是面向对象最重要的概念。Python 中如何定义一个类呢？

```
1. class Student(object):
2.     def __int__(self, name, score):
3.         self.name = name
4.         self.score = score
```

- 定义

关键字 class 后面跟着类名，类名通常是大写字母开头的单词

- 继承

紧接着是(object)，表示该类是从哪个类继承下来的

- 方法

__init__ 方法的第一个参数永远都是 self，表示创建实例本身

Python -动态特性

- 动态修改属性

  获取属性

  getattr(object，name [，default])

  增加属性

  setattr(object, name, value)

  判定

  hasattr (object，name)

  删除属性

  delattr(object, name)

```
1. class A(object):
2.     def __int__(self):
3.         self.name = ‘My Name’
4.
5.     def method(self):
6.         print ‘My Method, self is %s’ % self.name
7.
8. a = A()
9. setattr(a, ‘age’, 28)
10. print a.age
```

- 元类

  type('Foo', (), {'bar':True})

```
1. class Foo():
2.     bar = True
```

程序结构

Python 程序结构

- 模块

  一个 Python 文件，以 .py 结尾，包含了 Python 对象定义和 Python 语句

- 函数

  函数是组织好的，可重复使用的，用来实现单一，或相关联功能的代码段。

- 顺序

  顺序结构就是一条一条地从上到下执行语句

- 条件

  条件结构就是根据条件来判断执行哪些语句

- 循环

  循环结构就是在满足指定条件时，重复执行某些语句

Python 程序结构 – 函数

- 定义一个函数

```
def function_name(*args, **kwargs):
    #do_something
    #return result
    pass
```

- 匿名函数

```
sum = lambda a, b: a + b
```

- 必备参数

定义　　def func(p)

- 缺省参数

定义　　def func(a, b=1, c=2)

- 可变参数

定义　　def func(a, *args, **kwargs)

Python 程序结构 – 函数参数

- 必备参数

  定义　　def func(p)

  调用　　func(1)　func(p=1)

- 缺省参数

  定义　　def func(a, b=1, c=2)

  调用　　func(1, 1, 1)　func(1, c=1)　　func(1)

  参数值　a=1,b=1,c=1　　a=1,b=1,c=1　　a=1,b=1,c=2

Python 程序结构 – 条件

- 条件

关键字，if 、elif 、else

基本形式：

```
if condition:
    block
elif condition:
    block
...
else:
    block
```

```
i = 5
if i % 2 == 1:
    print u"奇数"
else:
    print u"偶数"
```

```
i = 5
print u"奇数" if i % 2 == 1 else u"偶数"
```

Python 程序结构 – 循环

循环

提供两种循环，for 和 while

- for
- while
- continue 关键字
- break 关键字

```
sum = 0
for var in range(1, 101):
    sum += var
print sum
```

```
count = 2
while count > 0:
    print ‘I love Python’
    count = count - 1
```

内置模块

Python 的内置模块

- datetime是Python处理日期和时间的标准库。
- collections是Python内建的一个集合模块，提供了许多有用的集合类。
- Base64是一种用64个字符来表示任意二进制数据的方法。
- hashlib提供了常见的摘要算法，如MD5，SHA1。
- itertools提供了非常有用的用于操作迭代对象的函数。
- urllib提供了一系列用于操作URL的功能
- HTMLParser来非常方便地解析HTML

Python 的内置模块 - os

- os模块，用于提供系统级别的操作

- os.getcwd()　　获取当前工作目录，即当前python脚本工作的目录路径
- os.chdir("dirname")　　改变当前脚本工作目录；相当于shell下cd
- os.curdir　　返回当前目录: ('.')
- os.pardir　　获取当前目录的父目录字符串名：('..')
- os.removedirs('dirname1') 若目录为空，则删除，并递归到上一级目录，如若也为空，则删除，依此类推
- os.mkdir('dirname')　　生成单级目录；相当于shell中mkdir dirname
- os.rmdir('dirname')　　删除单级空目录，若目录不为空则无法删除，报错；相当于shell中rmdir dirname
- os.remove() 删除一个文件
- os.rename("oldname","newname") 重命名文件/目录
- os.sep　　输出操作系统特定的路径分隔符，win下为"\\",Linux下为"/"
- os.linesep　　输出当前平台使用的行终止符，win下为"\t\n",Linux下为"\n"

Python 的内置模块 - sys

- sys模块提供了一系列有关Python运行环境的变量和函数

- sys.argv　　命令行参数List，第一个元素是程序本身路径
- sys.exit(n)　　退出程序，正常退出时exit(0)
- sys.version　　获取Python解释程序的版本信息
- sys.maxint　　最大的Int值
- sys.path　　返回模块的搜索路径，初始化时使用PYTHONPATH环境变量的值
- sys.platform　　返回操作系统平台名称
- sys.stdout.write('please:')
- val = sys.stdin.readline()[:-1]

第三方模块

常用第三方 Python 模块

- PIL ，python图像处理
- Paramiko ， ssh python 库
- Numpy，科学计算
- Matplotlib，画图
- Scrapy，爬虫
- Selenium，浏览器自动化测试工具selenium的python 接口
- Gevent，高并发的网络性能库
- twisted，基于事件驱动的网络引擎框架
- sh，强大的系统系统管理神器
- Jinja2，模板引擎

常用第三方 Python 模块 - requests

- 简介

一个 HTTP 客户端库，跟 urllib，urllib2类似

- 用途

发送请求

- 安装

$ pip install requests

- 使用

```
import requests

r = requests.get('https://github.com/timeline.json')
r = requests.put("http://httpbin.org/put")
r = requests.delete("http://httpbin.org/delete")
r = requests.head("http://httpbin.org/get")
r = requests.options("http://httpbin.org/get")
```

- 参考链接

http://docs.python-requests.org/zh_CN/latest/user/quickstart.html

常用第三方 Python 模块 - xlrd

- 简介

  Xlrd 是读 excel，xlwt 是写 excel

- 安装

  $ pip install xlrd

- 使用

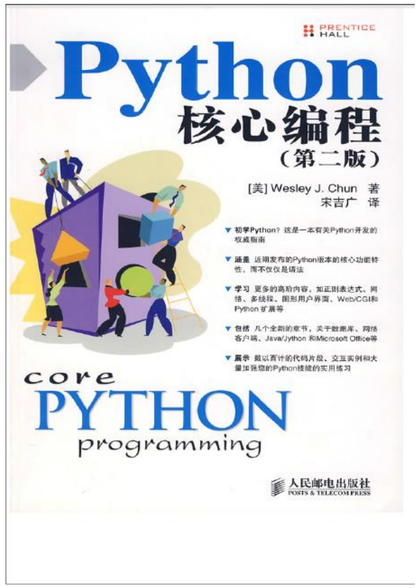
- 参考链接

  http://www.python-excel.org/

```python
# coding=utf-8
import xlrd
book = xlrd.open_workbook("myfile.xls")

# 获取sheets数目
print book.nsheets
```

Python 学习书籍



Python 核心编程

Python Cookbook

作业（1）

- **题目**：输入一个整数 n，输出 0-n 之间的所有素数

- **题目**：倒叙输出一个九九乘法表。

- **题目**：写一个交互程序，实现摄氏和华氏温度的相互转换。摄氏温度转华氏温度的公式为：celsius * 1.8 = fahrenheit – 32。
- **程序分析**：input 函数介绍用户的输入，然后通过输入的最后一个字符是 c/C，还是 f/F 来区分用户输入的是摄氏温度，还是华氏温度。

作业（2）

- **题目**：有四个数字：1、2、3、4，能组成多少个互不相同且无重复数字的三位数？各是多少？
- **程序分析**：可填在百位、十位、个位的数字都是1、2、3、4。组成所有的排列后再去 掉不满足条件的排列。

- **题目**：斐波那契数列。
- **程序分析**：斐波那契数列（Fibonacci sequence），又称黄金分割数列，指的是这样一个数列：0、1、1、2、3、5、8、13、21、34、……

  在数学上，费波那契数列是以递归的方法来定义：

  F0 = 0 (n=0) F1 = 1 (n=1) Fn = F[n-1]+ F[n-2](n=>2)

  输出第10个斐波那契数

下集预告

Python 基础（二）

- 异常处理
- Python函数
- 函数式编程
- 装饰器函数
- 编码规范

蓝鲸智云

腾讯云

蓝鲸智云公众号

蓝鲸高校版交流群