中国人民大学数据挖掘中心出品

Python 自学教程　　1.0

——Python 入门及其数据操作实例

我们的网站: http://rucdmc.net/

2014.11

前言

Why Python?

简单————Python是一种代表简单主义思想的语言。阅读一个良好的Python程序就感觉像是在读英语一样，尽管这个英语的要求非常严格！Python的这种伪代码本质是它最大的优点之一。它使你能够专注于解决问题而不是去搞明白语言本身。

易学————就如同你即将看到的一样，Python极其容易上手。前面已经提到了，Python有极其简单的语法。

免费、开源————Python是FLOSS（自由/开放源码软件）之一。简单地说，你可以自由地发布这个软件的拷贝、阅读它的源代码、对它做改动、把它的一部分用于新的自由软件中。FLOSS是基于一个团体分享知识的概念。这是为什么Python如此优秀的原因之一——它是由一群希望看到一个更加优秀的Python的人创造并经常改进着的。

高层语言————当你用Python语言编写程序的时候，你无需考虑诸如如何管理你的程序使用的内存一类的底层细节。

可移植性————由于它的开源本质，Python已经被移植在许多平台上（经过改动使它能够工作在不同平台上）。如果你小心地避免使用依赖于系统的特性，那么你的所有Python程序无需修改就可以在下述任何平台上面运行。这些平台包括Linux、Windows、FreeBSD、Macintosh、Solaris、OS/2、Amiga、AROS、AS/400、BeOS、OS/390、z/OS、Palm OS、QNX、VMS、Psion、Acom RISC OS、VxWorks、PlayStation、Sharp Zaurus、Windows CE甚至还有PocketPC、Symbian以及Google基于linux开发的android平台！

解释性————这一点需要一些解释。一个用编译性语言比如C或C++写的程序可以从源文件（即C或C++语言）转换到一个你的计算机使用的语言（二进制代码，即0和1）。这个过程通过编译器和不同的标记、选项完成。当你运行你的程序的时候，连接/转载器软件把你的程序从硬盘复制到内存中并且运行。而Python语言写的程序不需要编译成二进制代码。你可以直接从源代码 运行 程序。在计算机内部，Python解释器把源代码转换成称为字节码的中间形式，然后再把它翻译成计算机使用的机器语言并运行。事实上，由于你不再需要担心如何编译程序，如何确保连接转载正确的库等等，所有这一切使得使用Python更加简单。由于你只需要把你的Python程序拷贝到另外一台计算机上，它就可以工作了，这也使得你的Python程序更加易于移植。

可扩展性————如果你需要你的一段关键代码运行得更快或者希望某些算法不公开，你可以把你的部分程序用C或C++编写，然后在你的Python程序中使用它们。

可嵌入性————你可以把Python嵌入你的C/C++程序，从而向你的程序用户提供脚本功能。

丰富的库————Python标准库确实很庞大。它可以帮助你处理各种工作，包括正则表达式、文档生成、单元测试、线程、数据库、网页浏览器、CGI、FTP、电子邮件、XML、XML-RPC、HTML、WAV文件、密码系统、GUI（图形用户界面）、Tk和其他与系统有关的操作。记住，只要安装了Python，所有这些功能都是可用的。这被称作Python的“功能齐全”理念。除了标准库以外，还有许多其他高质量的库，如wxPython、Twisted和Python图像库等等。

概括————Python确实是一种十分精彩又强大的语言。它合理地结合了高性能与使得编写程序简单有趣的特色。

规范的代码————Python采用强制缩进的方式使得代码具有极佳的可读性。

目录

Chap.1 快速安装Python

1. 在Windows上安装
2. 在Mac OS X上安装
3. 在Linux上安装

参考

Chap.2 基础知识入门

一、基本概念
二、运算符与表达式
三、函数使用
四、输入输出

Chap.3 数据结构

一、 序列

1. 通用操作
2. 列表
3. 元组
4. 字符

二、 映射

1. 字典

参考

Chap.4 流程控制

1. 辅助
2. if语句&while语句&for语句

参考

Chap.5 模块与包

一、模块是什么
二、使用模块有什么好处
三、模块的导入
四、搜索模块
五、查看、弹出已加载模块

六、重新加载
七、包
八、高级模块话题

参考

Chap.6 后续

1. 类
2. 异常处理
3. 迭代器和生成器
4. 常用模块

练习

Chap.1 快速安装Python

1. **在Windows上安装**

   到Python官网下载最新版本的Python安装包。安装过程和其它基于Windows的软件类似
   对于Windows 2000、XP、2003，点击控制面板---系统---高级---环境变量。在“系统变量”中点击PATH，选择编辑，然后在已有内容的最后部分添加;C:Python33（请核实存在该文件夹，对于较新版本Python来说，文件夹的名字可能不同）。当然，要使用正确的目录名。
   对于Windows Vista：
   点击“开始”，选择“控制面板”。点击“系统”，在右侧您将看到“查看计算机基本信息”。
   左侧是一个任务列表，其最后一项是“高级系统设置”，点击它。显示“系统属性”对话框“高级”选项卡。点击右下角的“环境变量”按钮。在下方标题为“系统变量”框中，滚动“Path”，点击“编辑”按钮。
   按需修改路径。
   重启系统。除非重启，Vista不会意识到系统路径变量的修改。
   对于Windows 7：
   在桌面上右击“计算机”，选择“属性”；或点击“开始”，选择“控制面板”---“系统和安全”---“系统”，点击左侧的“高级系统设置”，然后选择“高级”选项卡。点击底部的“环境变量”按钮，在下方的“系统变量”中找到PATH变量，选中它点击“编辑”。
   在变量值的最后，追加;C:Python33。
   如果这个值是%SystemRoot%system32;，它将变成%SystemRoot%system32;C:Python33。
   点击“确定”完成。不需要重启。
   在Windows命令行上运行Python：
   要打开Windows终端，点击开始按钮，点击“运行”。在对话框输入cmd，按下回车键。然后输入python -V，会输出Python的版本号，确保没有错误。

2. **在Mac OS X上安装**
   Mac OS X用户来说，通过按Command+Space键打开终端（打开Spotlight搜索），输入Terminal然后回车。
   安装Homebrew时运行：
   ruby -e "$(curl -fsSkL raw.github.com/mxcl/homebrew/go)"
   然后安装Python 3 使用
   brew install python3
   现在，运行python3 -V，确保没有错误。
3. **在Linux上安装**
   对于Linux用户，通过打开Terminal应用程序打开终端，或者按下Alt + F2，然后输入gnome-terminal。如果不成功，请参考文档或你所用Linux发行版的论坛。
   下一步，我们需要安装python3包。例如，在Ubuntu上，可以使用sudo apt-get install python3。
   一旦你完成安装，在shell运行python3 -V，在屏幕上你应该能够看到Python版本：
   $ python3 -V
   Python 3.3.0

参考：
http://www.cnblogs.com/hongten/p/hongten_python_install.html
http://see.xidian.edu.cn/cpp/html/1801.html

Chap.2 基础知识入门

一、基本概念

了解python中数的四种类型，字符串的定义编码和转移，类型转化函数，缩进语句块。

http://www.cnblogs.com/renzo/archive/2011/08/08/2131146.html
http://blog.csdn.net/porcupinefinal/article/details/623539
http://www.cnblogs.com/Peter-Zhang/archive/2011/12/25/2300187.html

二、运算符与表达式

了解运算符与其用法以及运算符优先级（从低到高）。

http://blog.163.com/zhulp0372@yeah/blog/static/1158944792011712434843 5/
http://www.w3cschool.cc/python/python-operators.html

三、函数使用

了解函数的定义和好处，如何定义函数，参数的原理以及使用，函数的语法结构以及返回值，函数的调用。

http://www.cnblogs.com/hongten/p/hongten_python_method.html
http://www.cnblogs.com/tqsummer/archive/2011/01/25/1944416.html
http://developer.51cto.com/art/200809/88052_2.htm
http://developer.51cto.com/art/200809/88052_2.htm
http://blog.sina.com.cn/s/blog_4513dde60100o6w4.html
http://www.cnblogs.com/xudong-bupt/p/3833933.html
http://www.jb51.net/article/47991.htm

四、输入输出

熟悉print和input,raw_input的使用以及区别。

http://www.cnblogs.com/plwang1990/p/3757549.html
http://blog.csdn.net/carolzhang8406/article/details/6093537
http://bbs.csdn.net/topics/390277547?page=1
http://www.jb51.net/article/55768.htm
http://blog.csdn.net/sruru/article/details/7790436
http://blog.csdn.net/wusuopubupt/article/details/23680491
http://www.cnblogs.com/way_testlife/archive/2011/03/29/1999283.html

Chap.3 数据结构

数据结构是通过某种方式组织在一起的数据元素的集合。
Python中的数据结构主要有两种：序列（字符串、列表、元组）和映射（字典）

首先介绍序列：（细节操作详见参考）

1. 序列的通用操作：索引、分片、加、乘
    - 索引 seq[i]：序列中的元素是有编号的，pyhton中是从0开始从左向右递增，递增至最后一位（序列长度n-1）。
    - 分片 seq[k:m:l]: [开始索引值：结束索引值：步长]
      如果要同时一定范围内的元素则需要分片操作（通过两个索引来实现）。通过两个索引为边界[k:m]，范围为索引为k至m-1的元素（第一个索引的元素包括，但是左后一个元素的索引不包括）
    - 步长：在分片的格式上多加一个参数，即[开始索引值：结束索引值：步长] 。可以将步长理解为跳出值，即每隔几个元素访问一次。
    - 加法：运用‘+’可以将两个类型相同的序列进行连接
    - 乘法：用‘*’是生成新的序列，将原来的序列重复n次
2. 列表：列表是最通用的Python复合数据类型。列表中包含以逗号分隔，并在方括号（[]）包含的项目，所有属于一个列表中的项目可以是不同的数据类型的，且列表是可以修改的。
    - 生成/转化：list（）
    - 追加新值：list.append(obj)/list.extend(seq)
    - 插入新值：list.insert(index, obj)/list[index:index]=seq
    - 删除元素：list.remove(element)/del list[index]/list.pop(obj=list[-1])
    - 获得索引: list.index(element)
    - 排序：list.reverse()/list.sort([func])/sorted(list,key=func,reverse=TRUE)
3. 元组：
   元组是类似于列表的序列数据类型。一个元组由数个逗号分隔的值用圆括号括起来组成。元组可以被认为是只读列表，即不可修改。

4. 字符串：
   在Python中的字符串被确定为一组在引号之间的连续字符。 Python允许在任何字符串用单引号或双引号（或三引号）。
   字符串还有一些特殊的函数和操作：
    - 格式化输出：print 格式标记字符串 % 要输出的值
    - 查找字符串：string.find()
    - 连接：‘sep’.join(seq)
    - 大小写转换：string.lower()/string.upper()
    - 替换： string.replace(old, new)
    - 分割：string.split('sep')
      -去除两端字符：string.strip(‘sep’)

接下来介绍映射：（细节操作详见参考）
Python中唯一内建的映射是字典，Python的字典是一种哈希表型。他们像关联数组或哈希在Perl中一样，由键 - 值对组成。字典键几乎可以是任何Python类型，但通常是数字或字符串。值可以是任意Python的对象。字典是由花括号包含（{}），可分配值，并用方括号（[]）访问，它的元素是无序的。

- 创建字典： 直接赋值，用dict()函数处理键值对的序列，用fromkeys()函数快速生成
- 访问字典中的值 ： 用for遍历访问key或者dict[key]或者用dict.items()，用dict[key]来直接索引访问，用dict.items()得到（key，value)的列表，并与for结合，用dict.keys()和dict.values()得到所有的键和值
- 更新字典：覆盖更新，update()：将一个字典的内容添加到另外一个字典，删除字典元素和字典del dict['name']，dict.clear()，del dict，dict.pop('name'): 删除并返回键为“name”的条目
- dict.setdefault(key,value):如果字典的键中存在参数key，则返回key所对应的值；如果字典的键中不存在参数key，则在字典中创建键值对key：value。

参考:

Magnus Lie Hetland. Beginning Python: From Novice to Professional, Second Edition. 人民邮电出版社．2010.7

Mark Lutz. Learning Python, Fourth Edition. 机械工业出版社. 2011-4

http://www.jb51.net/article/47972.htm

http://www.cnblogs.com/SunWentao/archive/2008/06/19/1225690.html

http://www.cnblogs.com/linjiqin/p/3608541.html

http://www.cnblogs.com/huangcong/archive/2011/08/29/2158268.html

http://blog.sina.com.cn/s/blog_63c7b0330100iwpq.html

http://sebug.net/paper/books/dive-into-python3/strings.html

http://www.jb51.net/article/47956.htm

http://www.cnblogs.com/dolphin0520/archive/2013/03/08/2950223.html

http://blog.csdn.net/facevoid/article/details/5338048

http://www.cnblogs.com/skyhacker/archive/2012/02/04/2337682.html

http://blog.csdn.net/zimohuakai/article/details/7315958

http://www.2cto.com/kf/201206/137662.html

http://www.cnblogs.com/mguo/archive/2013/02/25/2931894.html

http://blog.csdn.net/bolike/article/details/20402077

http://blog.csdn.net/moodytong/article/details/7647684

http://blog.csdn.net/nrs12345/article/details/4869272

http://www.cnblogs.com/rubylouvre/archive/2011/06/19/2084739.html

http://www.douban.com/group/topic/21801800/

http://www.linuxidc.com/Linux/2012-01/51638.htm

http://www.cnblogs.com/kaituorensheng/archive/2012/11/21/2781738.html

http://blog.csdn.net/bolike/article/details/19996667

Chap.4 流程控制

掌握if语句、for语句和while语句。
了解range函数、break&continue以及pass语句，并配合前三个语句使用：

- range():
  range（start， end， scan):
  start:计数从start开始。默认是从0开始。
  end:计数到end结束，但不包括end。
  scan：每次跳跃的间距，默认为1。
  http://www.cnblogs.com/buro79xxd/archive/2011/05/23/2054493.html
  http://www.2cto.com/kf/201402/277634.html
  http://www.cnblogs.com/hongten/p/hongten_python_range.html
  http://www.jb51.net/article/35014.htm

  break&continue:
  break :跳出break语句所在的最近的循环，即停止执行一个循环语句，即使循环条件还没有成为False或序列的项目没有被完全遍历且只跳出最近的一个循环。
  continue: 来到continue语句所在的最近的循环的首行，即跳过当前循环块中其余剩下的语句，继续循环的下一次迭代。
  http://see.xidian.edu.cn/cpp/html/1823.html
  http://www.qttc.net/201302272.html

  pass:
  不进行任何运算，即空占位。
  http://www.w3cschool.cc/python/python-pass-statement.html

  循环中的else语句：只有当循环正常离开时才会执行（正常离开循环即没有碰到break语句）。

- if， while, for
  - if语句
    用来检查一个条件：如果条件为真(true)，我们运行一个语句块（称为if块），否则(else)，我们执行另一个语句块（称为else块）。else子语句是可选的。
  - while语句：
    python会一直顶端测试，若为真，则执行循环主体内的语句，然后再返回到语句的顶端再测试，直到测试返回假值为止，然后执行while块后的语句。即顶端测试为真则重复执行一个语句块，为假则执行后面的语句块。
  - for 语句
    for循环是Python中通用的一个序列迭代器，可以遍历任何有序的序列和一些无序对象（如字典，文件等）。首行定义了赋值目标以及遍历对象，运行时会逐个将遍历对象的元素赋值给目标，然后为目标执行循环主体。

关于流程控制的详细资料和实际运用可以参考如下：
Magnus Lie Hetland. Beginning Python: From Novice to Professional, Second Edition. 人民邮电出版社. 2010.7
Mark Lutz. Learning Python, Fourth Edition. 机械工业出版社. 2011-4

http://www.php100.com/manual/Python/ch06s04.html
http://www.cnblogs.com/GarfieldTom/archive/2013/01/14/2860206.html
http://sebug.net/paper/python/ch06s04.html
http://www.w3cschool.cc/python/python-for-loop.html
http://www.jb51.net/article/45864.htm
http://www.cnblogs.com/way_testlife/archive/2010/06/14/1758276.html
http://developer.51cto.com/art/200808/84690.htm
http://woodpecker.org.cn/abyteofpython_cn/chinese/ch06s03.html
http://www.w3cschool.cc/python/python-while-loop.html
http://ipseek.blog.51cto.com/1041109/793031
http://www.open-open.com/lib/view/open1346511811678.html
http://www.cnblogs.com/liubin0509/archive/2011/11/27/2265091.html
http://see.xidian.edu.cn/cpp/html/1820.html
http://blog.csdn.net/lynn_yan/article/details/5464911
http://www.2cto.com/kf/201106/92691.html

Chap.5 模块与包

一、模块是什么（详细定义见参考）

形象地说，模块是一个工具箱，里面放着封装好的可供使用的工具。

具体的说，模块就是一个.py文件，里面放着编好的Python代码，包括变量与函数。

二、模块的导入

import: 使客户端（导入者）以一个整体获取一个模块

from:容许客户端从一个模块文件中获取特定的变量名

reload:在不中止Python程序的情况下，提供了一个重新载入模块文件代码的方法。

基础语句介绍:

- import module：导入整个模块
  module是模块名,使用该语句导入后，对该模块中任一对象x的使用均要在前面加上module.， 如module.x。
- from module import a,b,c ： 导入该模块中的对象a,b,c
- from module import a as e,b as f,c as g: 在使用中用e,f,g来代替a,b,c
- from module import * ： 取得模块顶层所有赋了值的变量名的拷贝，类似于import module

四、搜索模块

一个Python程序包括了多个含有Python语句的文件。程序是作为一个主体的，顶层的文件来构造的，配合有零个或多个支持文件，在Python中这些文件称作模块。

标准模块：python自带了200多个使用的模块、成为标准连接库

Python搜索模块的路径:

1. 程序的主目录
2. PTYHONPATH目录（如果已经进行了设置）
3. 标准连接库目录（一般在/usr/local/lib/python2.X/）
4. 任何的.pth文件的内容（如果存在的话）.新功能，允许用户把有效果的目录添加到模块搜索路径中去.pth后缀的文本文件中一行一行的地列出目录。

五、查看、弹出已加载模块

查看已加载模块：dir()

弹出已加载模块：del 模块名

六、重新加载

模块程序代码默认值对每个过程执行一次，要强制使模块代码重新载入并重新运算需要使用reload内置函数。reload()之前需得import过一次.

注意：reload（）仅对 `import module` 语句有效，如果是 `from module import a,b,c` 需要对a，b，c进行重新加载，那么只需要重新执行 `from module import a,b,c` 一遍即可。

七、包

除模块名以外，导入也可以指定目录路径，Python代码的目录就是称为包, 因此这类导入就称为包导入。

形象地说，是存放工具箱的房间，或者说是个更大号的工具箱，里面可以放小工具箱。

具体地说，是存放在模块与小包的文件夹。（其实文件夹也是一种特殊的文件，所以包的本质和模块类似）。

八、高级模块话题

1. 混合用法模式：__name__和__main__
   这是一个特殊的与模块相关的技巧，可以把文件作为模块导入，并以独立式程序的形式运行。每个模块都有个名__name__的内置属性。Python会自动设置该属性：
   如果文件是以顶层程序文件执行，在启动时，__name__就会被设置为字符串__main__
   如果文件被导入，__name__就会改设成客户端所了解模块名
   结果就是模块可以检测自己的__name__，来确定它是执行还是在导入。

2. 修改模块搜索路径
   可以通过PYTHONPATH以及可能的.pth路径文件进行定制。
   Python程序本身是修改sys.path的内置列表。sys.path在程序启动时就进行初始化，但那之后也可以随意对其元素进行删除，附加和重设

3. Python也在sys.modules字典中导出所有已经加载的模块,并提供一个内置函数getattrr，让我们以字符串名来取出属性

参考:

Magnus Lie Hetland. Beginning Python: From Novice to Professional, Second Edition. 人民邮电出版社. 2010.7
Mark Lutz. Learning Python, Fourth Edition. 机械工业出版社. 2011-4
http://blog.chinaunix.net/uid-10328574-id-2951038.html
http://wenku.baidu.com/link?url=zhkTmkHlRgjGDtOsgjtsZrWN0MANxIWqvnGfWwLc-NMBZZ4n4DbyX_ywMOJ5UBi9fRcRhuxDBm_2U8DEcXqPlygITwRYF7uK7prAWZL3TCm
http://blog.csdn.net/lcyangcss/article/details/7249961
http://www.cnblogs.com/captain_jack/archive/2011/01/11/1933366.html
http://www.iteye.com/news/2639
http://www.cnblogs.com/dolphin0520/archive/2013/03/19/2969152.html
http://blog.csdn.net/eoe2005/article/details/1550389
http://blog.sina.com.cn/s/blog_4b5039210100ennq.html
http://blog.csdn.net/meteor1113/article/details/4350224
http://www.cnblogs.com/coser/p/3551285.html

Chap.6 后续

由于篇幅限制，部分内容未放入本版本中，将在后续版本中跟进。现提前将未涉及内容的参考资料发布如下：

1. 类：
   Magnus Lie Hetland. Beginning Python: From Novice to Professional, Second Edition. 人民邮电出版社. 2010.7　P89-P126
   Mark Lutz. Learning Python, Fourth Edition. 机械工业出版社. 2011-4　P631-P821
   http://blog.csdn.net/ccat/article/details/8364
   http://www.jb51.net/article/42623.htm
   http://bdxnote.blog.163.com/blog/static/844423520134134257721/
   http://www.cnblogs.com/lixinjie/archive/2013/02/25/python-classes.html
2. 异常处理：
   Magnus Lie Hetland. Beginning Python: From Novice to Professional, Second Edition. 人民邮电出版社. 2010.7　P127-P138
   Mark Lutz. Learning Python, Fourth Edition. 机械工业出版社. 2011-4　P825-P888
3. 迭代器和生成器
   Magnus Lie Hetland. Beginning Python: From Novice to Professional, Second Edition. 人民邮电出版社. 2010.7　P139-P164
   Mark Lutz. Learning Python, Fourth Edition. 机械工业出版社. 2011-4　P360-P381
4. 常用模块：
   安装：
   http://blog.csdn.net/hengcai001/article/details/4166996
   标准库总览：
   python标准库中文版
   re:
   Python正则表达式指南
   os.path:
   http://www.2cto.com/kf/201110/108206.html
   xlrd&xlrt:
   http://blog.sina.com.cn/s/blog_629c53bd0100zo8v.html
   http://zhaojing366.blog.163.com/blog/static/952384020124311071392/
   pandas：
   http://pda.readthedocs.org/en/latest/chp5.html
   http://www.open-open.com/lib/view/open1402477162868.html
   scipy：
   http://sebug.net/paper/books/scipydoc/scipy_intro.html#id4
   math：
   http://www.jb51.net/article/48433.htm
   http://blog.csdn.net/iamaiearner/article/details/9381347
   random:
   http://www.cnblogs.com/yd1227/archive/2011/03/18/1988015.html

http://blog.csdn.net/javasus/article/details/8529801
http://www.linuxidc.com/Linux/2012-12/77059.htm
http://my.oschina.net/cuffica/blog/33336
time&datetime:
http://blog.csdn.net/kiki113/article/details/4033017
http://www.cnblogs.com/xupeizhi/archive/2012/11/05/2755550.html
http://www.2cto.com/kf/201110/108205.html
http://blog.csdn.net/JGood/article/details/5457284
http://huaxia524151.iteye.com/blog/1409943
http://huaxia524151.iteye.com/blog/1409943
http://blog.csdn.net/five3/article/details/8772410

练习

数据文件dur.txt，是对手机用户的APP使用情况的监测记录，共有10001行。
其中列从左到右分别对应着uid、APPid、起始日、起始时间、结束日、结束时间。
不同uid对应不同用户，每个用户的uid相同。
不同APPid对应不同用户，每个APP的APPid相同。
起始日为用户开始使用该APP的日期，用距离基准天的天数表示。
结束日为用户停止使用该APP的日期，用距离基准日天的天数表示。（其中结束日和起始日的基准日为同一天）
起始时间是用户开始使用该APP的当天的开始使用的时刻，格式为时：分：秒。
结束时间是用户停止使用该APP的当天的停止使用的时刻，格式为时：分：秒。

操作要求：

1. 计算每个用户的每次使用时长和相邻两次使用的时间间隔（对所有APP，不必每个APP均统计）
2. 统计单次使用时长所对应的用户数，构造频数分布。
3. 统计相邻两次使用的时间间隔所对应的用户数，构造频数分布。
4. 每隔一千行记录创造一个新文件，即将原来的文件进行按行数分割成子文件。
5. 按照每个用户分割文件，即对每个用户的所有记录单独创建一个新文件。

练习参考代码

目标：熟悉处理相应问题的编码流程，再根据实际情况调整

- 计数和加和

```
#encoding=utf8
import sys
reload(sys)
sys.setdefaultencoding('utf-8')

f=open(‘file1’,’r’)
g=open(‘file2‘,’w+')      #w+可以添加，a              #  a不用close    w要close?
dic={}                              #设定空字典
for i in open(‘file1’,’r’):
  line=f.readline()              #字符串
  line=line.split('\t')           #列表
  for  j in range(1,len(line)):
    a=line[j].strip()          #遍历，读取元素
    a=int(a)                  #将变量转换为int型，否则后面sorted排序时按照字符串规则排序
    dic.setdefault(a,0)
    #利用setdefault进行频数统计
    dic[a]+=1

sort=sorted(dic.items(),key=lambda e:e[0], reverse=False)   #排序时考虑类型  int&
str的规则不同

for i in sort:
# i被复制元组，此处可以考虑用解包赋值
  a=[str(i[0]),str(i[1])]
  # 下行的join要求是字符串
  a='\t'.join(a)
  print >>g, a.strip()
  #strip后只有print产生的一个\t

f.close()
g.close()
```

- 按大小分割文件

```
def split(fromfile,todir,chunksize=1024*1024*200):
     if not os.path.exists(todir):            #todir是否已经存在
```

```
        os.mkdir(todir)
    else:
        for fname in os.listdir(todir):         #若已存在，则清空：显示目录中的目录名
            os.remove(os.path.join(todir,fname))        #删除目录（dir+name）
    partnum = 0
    inputfile = open(fromfile,'rb')             #打开原文件，‘rb’
    while True:
        chunk = inputfile.read(chunksize)               #存在指针，自动移动
        if not chunk:                                 #len为0或者空列表[ ]（if中的Fal
se)
            break
        partnum += 1
        filename = os.path.join(todir, 'part%04d' % partnum)
        fileobj = open(filename,'wb')
       #利用os.path.join创建路径名，然后open生成文件
        fileobj.write(chunk)
 #利用write一次写入，要多次最好用print >>  ,
        fileobj.close()
    return partnum
if __name__=='__main__':
       fromfile  = input('File to be split?')
        todir     = input ('Directory to store part files?')
        chunksize = int(input('Chunksize to be split?'))
        absfrom,absto = map(os.path.abspath,[fromfile,todir])
   #map迭代函数，[ ] None 并行 [x for x in seq]
        print('Splitting',absfrom,'to',absto,'by',chunksize)
        try:
            parts = split(fromfile,todir,chunksize)
        except:
            print('Error during split:')
            print(sys.exc_info()[0],sys.exc_info()[1])
        else:
           print('split finished:',parts,'parts are in',absto)


#设置todir时要小心，尽量不在原目录中，我之前执行时手误把自己的资料全部删除了

 #不能放在同一个目录下，from和in分开放
```

- 合并文件

```
import sys,os

def joinfile(fromdir,filename,todir):
```

```
    if not os.path.exists(todir):
         os.mkdir(todir)
    if not os.path.exists(fromdir):
         print('Wrong directory')

     outfile = open(os.path.join(todir,filename),'wb')
     files = os.listdir(fromdir)
     files.sort()

     for file in files:
        filepath = os.path.join(fromdir,file)
        infile = open(filepath,'rb')
        data = infile.read()
        outfile.write(data)
        infile.close()

    outfile.close()

if __name__=='__main__':
        fromdir = input('Directory containing part files?')
        filename = input('Name of file to be recreated?')
        todir   = input('Directory to store recreated file?')

        try:
            joinfile(fromdir,filename,todir)
        except:
            print('Error joining files:')
            print(sys.exc_info()[0],sys.exc_info()[1])

 #不能放在同一个目录下，from和in分开放，否则in会被读取
```

- 时间数据处理

```
#encoding=utf8
import sys
reload(sys)
sys.setdefaultencoding('utf-8')
import datetime
 # date和datetime模块用于解决时间数据处理
import time


f=open('file1','r')
```

```
g=open('file2‘,’w+')
#  三段常用代码：f=open()  for i in open()  line=f.readline()
dic={}
for i in open('file1','r'):
  line=f.readline()          #得到字符串
  line=line.split('\t')                          #得到列表，但是要考虑有无空白符或者‘\t’在元素中（用strip())
  uid=line[0]                                    #用strip（）最好输出几行看看实际数据的输出情况
  uid=int(uid)
  dic.setdefault(uid,[]) #计数的经典函数

  etim=line[5].strip()        #对元素strip()
  stim=line[3].strip()
                                                 #先转换为time格式并在转换为datetime格式才能加减
  etim=time.strptime(etim,'%H:%M:%S')           #将元素装换为time格式并定义
  stim=time.strptime(stim,'%H:%M:%S')
#转换为datetime需要定义年月日，否则使用默认年月日（最好自己加上）（索引为0，1，2）
  etim=datetime.datetime(2014,8,10,etim[3],etim[4],etim[5])          #将元素转换为datetime格式并定义+datetime.timedelta(days=int(line[4].strip()))
#并加减计算
  stim=datetime.datetime(2014,8,10,stim[3],stim[4],stim[5])+datetime.timedelta(days=int(line[2].strip()))           #计算时间差时，日期上不同算入timedelta(days)
#如果存在不同天的话相加减前一定要加上天数（timedelta），否则存在默认加一天的情况
  standtim=datetime.datetime(2014,8,10,0,0,0)
  spoint=(stim-standtim).seconds+24*60*60*(stim-standtim).days
 #利用datetime相减计算时间差要考虑差值大时会输出days
#seconds只显示小时分钟秒的秒数（一天内），不显示超过一天的天数的秒数，故天数的时间要单独算
  #输出天和秒，分别计算（不会有其他），作为排序依据

  epoint=(etim-standtim).seconds+24*60*60*(etim-standtim).days

  dic[uid].append(spoint)                        #把秒数输入
  dic[uid].append(epoint)
#计算时间差时，日期算入days
for i in dic:
  a=dic[i]
  b=sorted(a)
 # print b              #看看输出情况
  i=str(i)             #后面用join需为str
  user=[i]
  if len(a)==4 :                #后面有range(2,len(a)/2) 如果len=4, 则range(2,2)输出空列表
```

```
    c=b[2]-b[1]
    c=str(c)
    user.append(c)
  else:
    for k in range(2,len(a)/2):
      c=b[2*k-2]-b[2*k-3]
      c=str(c)    #用join函数前考虑是否是字符
      user.append(c)
  user='\t'.join(user)
  #用'\t' 连接成字符后再输出
  print >>g, user.strip()
  #有strip则不用加print最后不用加,

f.close()
g.close()
```

- 按关键字分割文件

```
f=open("filename",'r')

a=f.readline()   #截去表头，然后再for

for i in open("filename",'r'):
    #open这里代表行数
    i=f.readline()
    #从第二行开始读
    b=i.split('\t')

    if b[0]!='':

        #判断是否已经读完，否则会报错
      #按照b[12]为关键字，对文件分割，读入两个文件
        output=open("filename",'a')
        ##注意这里的a表示续写！！！     #输入到两个文件     #用'a'不断续写
        output.write(i)
        ##output=open("filename",'w')
        ##注意这里的a表示续写！！！
        ##print>>output,i,

#a续写，不要close，用w则需要close

f=open("filename",'r')
a=f.readline()
c={}
```

```
d={}

for i in open("filename",'r'):
    i=f.readline()
    b=i.split('\t')

    if b[0]!='':
#统计里要判断，否则报错  计算机里 while True:  break
        if(b[12] in c): #has_key方法已经被废除                    #判断是否已经在  in
   T/F
            c[b[12]]=c[b[12]]+float(b[4])
            d[b[12]]=d[b[12]]+1
    #相当于dict.setdefault()
        else:
            c[b[12]]=float(b[4])
            d[b[12]] = 1

output=open("filename",'a')
for (i,j) in c.items():
    print("%s;%f"%(i,j/d[i]))                                    #for 字典时 用 di
ct.items()
    #print>>file=fl,"%s;%f"%(i,j/d[i])
    print("%s;%.3f"%(i,j/d[i]),file=output)
    #k=i+";"+str(j/d[i])+'\n'
    #output.write(k)
```