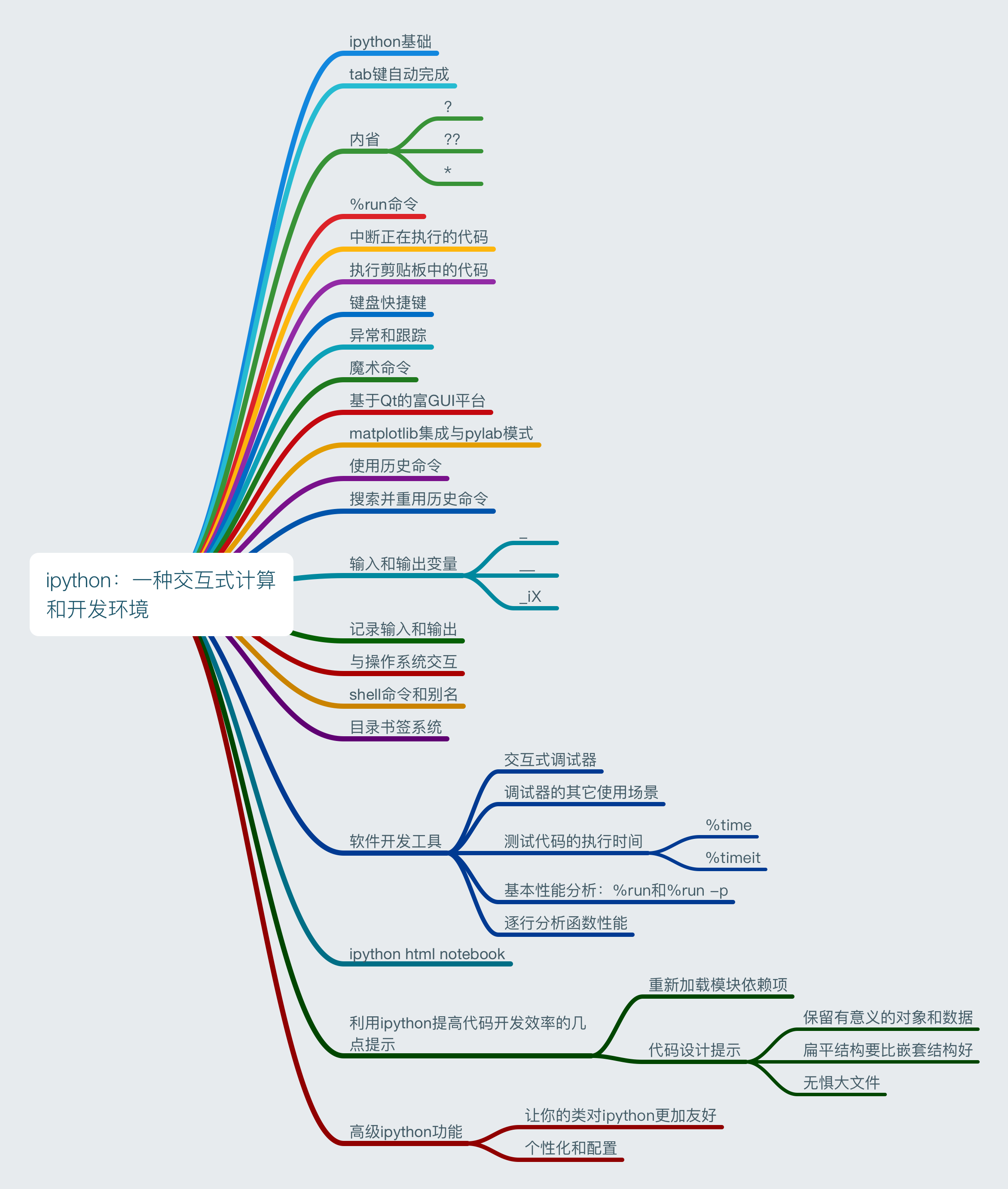
[IPython使用学习笔记](https://www.cnblogs.com/zzhzhao/p/5295476.html)

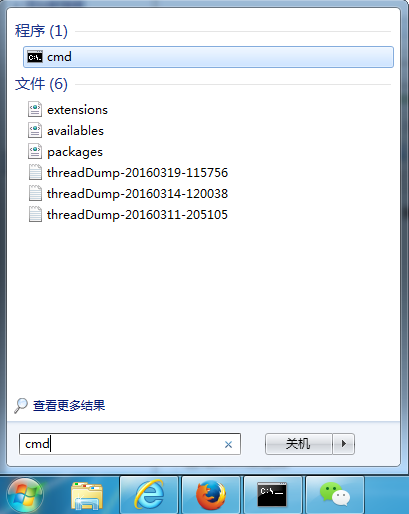
学习《利用python进行数据分析》第三章 IPython:一种交互式计算和开发环境的笔记，共享给大家，同时为自己作为备忘用。

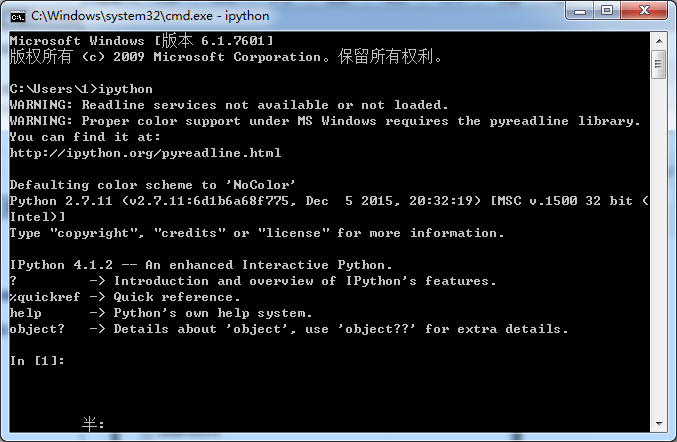


安装ipython用pip即可。ps.博主用的是win7系统，所以接下来的都是在windows系统下操作的。

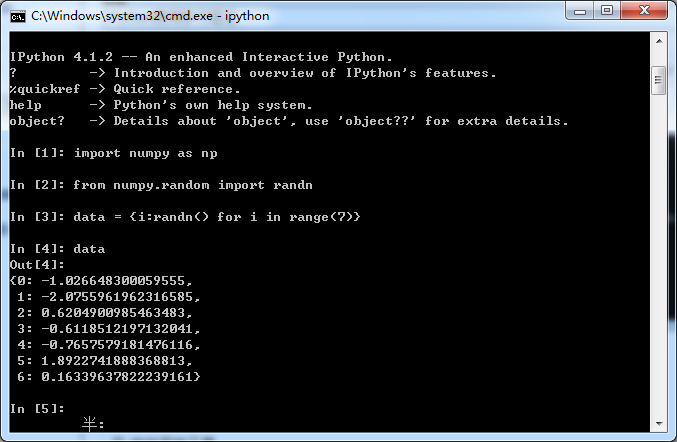
一.Ipython基础

启动：开始菜单-输入cmd-回车-输入ipython





初尝试

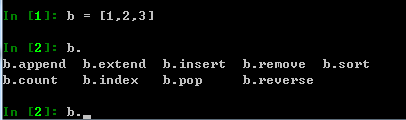


二.Tab键自动完成

在shell中输入表达式时，只要按下Tab键，当前命令控件中任何与输入的字符串相匹配的变量（对象、函数等）就会被找出来。

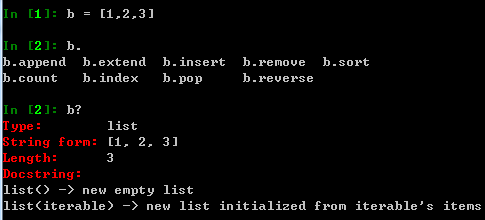
ps.之前我发现我的ipython没有Tab键自动补全功能，tab键的功能是缩进。最后 pip install pyreadline即可解决。

下面这个例子，输入b.后按下<tab>键即可得到下图的结果。



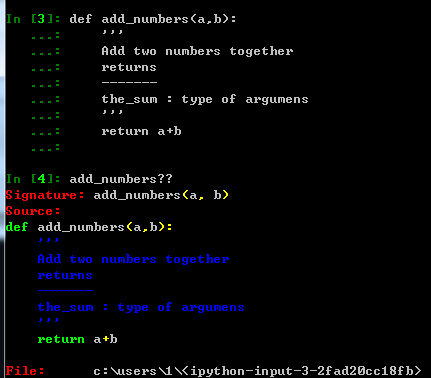
三.内省

在变量的前面或后面加上一个问号(?)就可以将有关该对象的一些通用信息显示出来。这就叫做对象的内省。



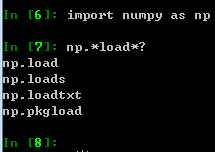
如果对象是一个函数或实例方法，则其docstring也会被现实出来。

使用??还将显示出该函数的源代码。



一些字符串配以通配符(\*)即可显示出所有与该通配符表达式相匹配的名称。

例如，我们可以列出Numpy顶级命名空间中含有“load”的所有函数



简直太棒了有木有！！！！！！！

四.%run命令

在ipython会话环境中，所有文件都可以通过%run命令当做Python程序来运行。

输入 %run 路径+python文件名称即可。

ps.《集体智慧编程》里面代码的执行可以用这个

五.中断正在执行的代码（急性子必备23333）

按下Ctrl-C就会引发一个KeyboardInterrupt。除了一些非常特殊的情况下，绝大部分python程序都会立即停止执行

六.执行剪贴板中的代码

书上说使用ctrl+shift+v将剪贴板的代码片段粘贴出来，然而windows系统似乎不可行，所以鼠标右键粘贴好了。

%paste可以承载剪贴板中的一切文本，并在shell中以整体形式执行。

%cpaste跟%paste差不多，只不过它多出了一个用于粘贴代码的特殊提示符而已。如果发现粘贴的代码有错，只需要按下ctrl+c即可终止%cpaste提示如。

七.键盘快捷键

其实不到真正使用我也记不住23333，在此粘贴给大家作为备忘。

1. Ctrl-P    或上箭头键 后向搜索命令历史中以当前输入的文本开头的命令
2. Ctrl-N   或下箭头键 前向搜索命令历史中以当前输入的文本开头的命令
3. Ctrl-R   按行读取的反向历史搜索（部分匹配）
4. Ctrl-Shift-v   从剪贴板粘贴文本
5. Ctrl-C   中止当前正在执行的代码
6. Ctrl-A   将光标移动到行首
7. Ctrl-E   将光标移动到行尾
8. Ctrl-K   删除从光标开始至行尾的文本
9. Ctrl-U   清除当前行的所有文本译注12
10. Ctrl-F   将光标向前移动一个字符
11. Ctrl-b   将光标向后移动一个字符
12. Ctrl-L   清屏

八.异常和跟踪

九.魔术命令

粘贴在此以作备忘

命令 说明  
%quickref 显示IPython的快速参考  
%magic 显示所有魔术命令的详细文档  
%debug 从最新的异常跟踪的底部进入交互式调试器  
%hist 打印命令的输入（可选输出）历史  
%pdb 在异常发生后自动进入调试器  
%paste 执行剪贴板中的Python代码  
%cpaste 打开一个特殊提示符以便手工粘贴待执行的Python代码  
%reset 删除interactive命名空间中的全部变量/名称  
%page OBJECT 通过分页器打印输出OBJECT  
%run script.py 在IPython中执行一个Python脚本文件  
%prun statement 通过cProfile执行statement，并打印分析器的输出结果  
%time statement 报告statement的执行时间  
%timeit statement 多次执行statement以计算系综平均执行时间。对那些执行时  间非常小的代码很有用  
%who、%who\_ls、%whos 显示interactive命名空间中定义的变量，信息级别/冗余度可变  
%xdel variable 删除variable，并尝试清除其在IPython中的对象上的一切引用

十.基于Qt的富GUI控制台

Ipython团队开发了一个基于Qt框架（其目的是为终端应用程序提供诸如内嵌图片、多行编辑、语法高亮之类的富文本剪辑功能）的GUI控制台。

使用

ipython qtconsole --pylab=inline

来启动的话可为其添加绘图功能。

ps.这一步刚开始无法执行，我的解决方法是：

pip install qtconsole，就可以运行了- -..

十一.matplotlib集成与pylab模式

通常我们通过在启动Ipython时加上--pylab标记来集成matplotlib

注意空格啊~是

ipython --pylab

十二.使用历史命令

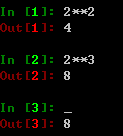
十三.搜索并重用历史命令

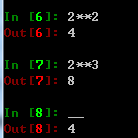
历史命令用上下箭头就好啦，ctrl+p和ctrl+n 太麻烦啦。

ctrl+r用于实现部分增量搜索，按下ctrl+r并输入你想搜索的行中的几个字符。按下ctrl+r将会循环搜索历史命令中每一条与输入相符的行。

十四.输入和输出变量

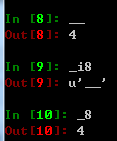
ipython将最近的两个输出结果保存在\_(一个下划线)和\_\_(两个下划线)变量中





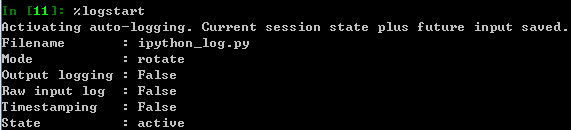
输入的文本被保存在名为\_iX的变量中，其中X是输入行的行号。比如说，在输入完27行后，就会产生两个新变量\_27(输出变量)和\_i27(输入变量)

我这儿就拿第八行举例子吧2333

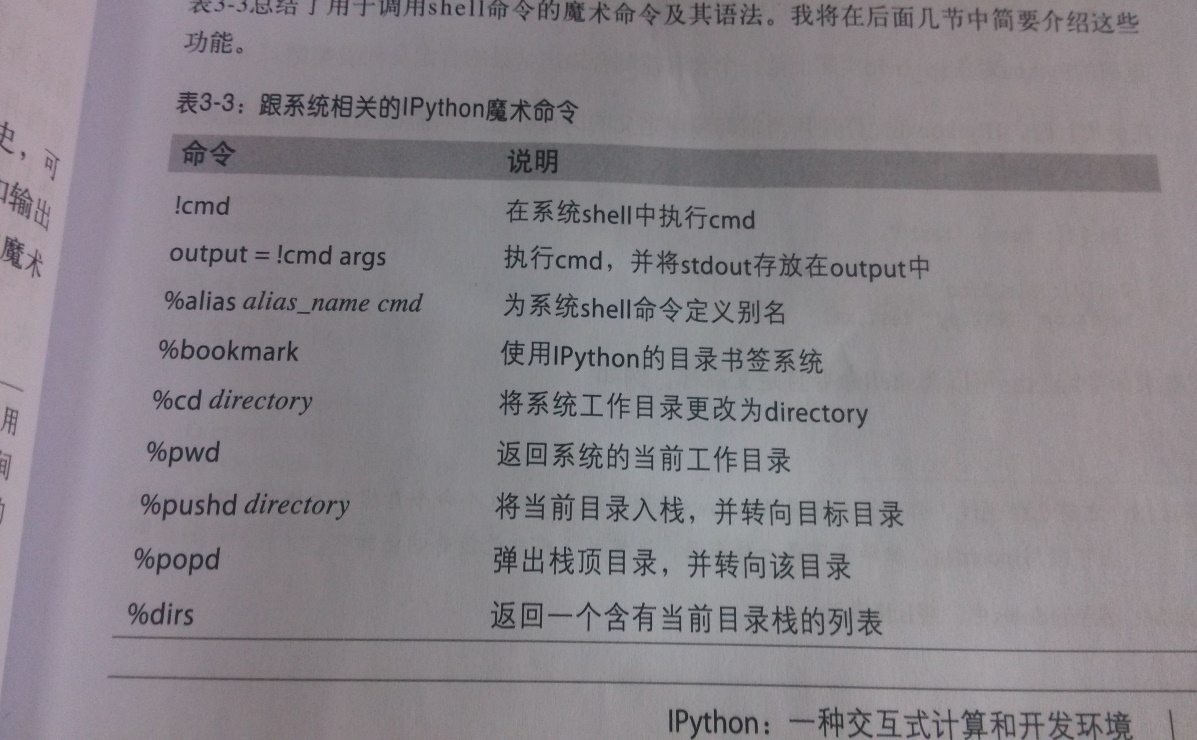


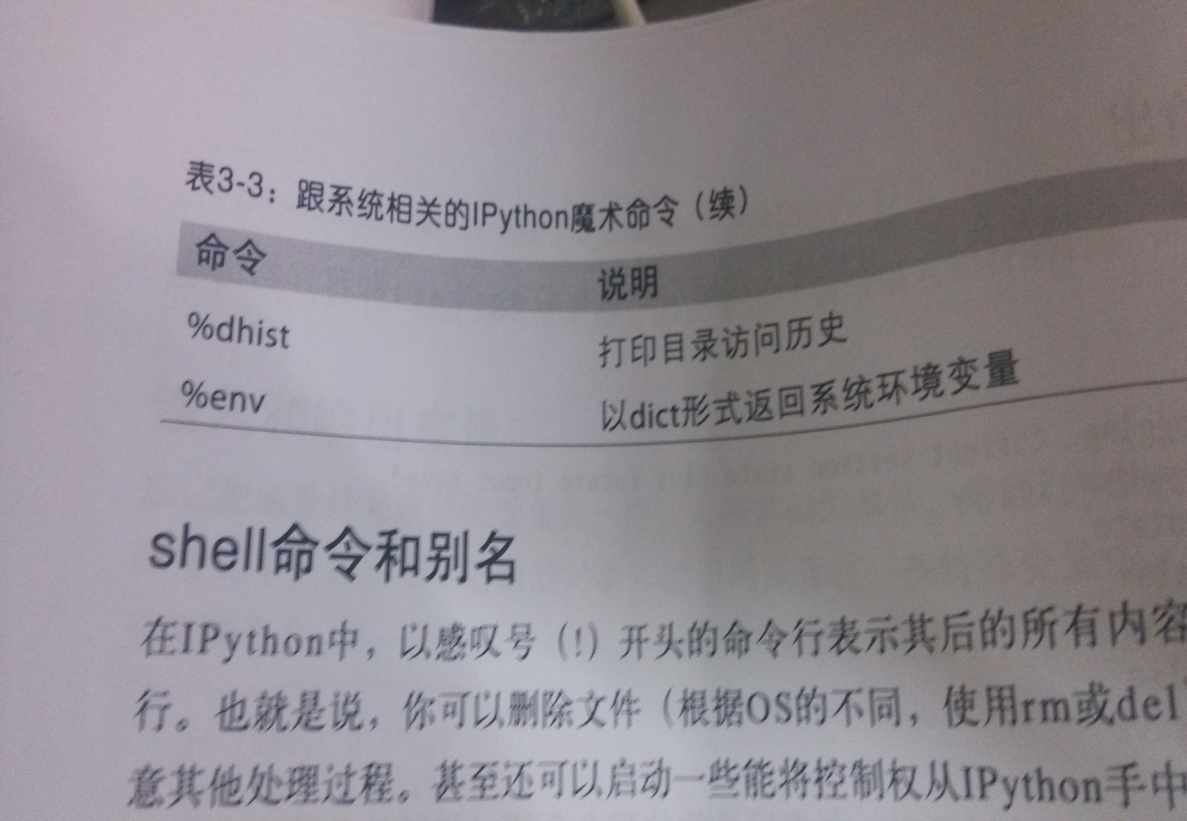
十五.记录输入和输出

执行%logstart既可开始记录日志



 十六.与操作系统交互

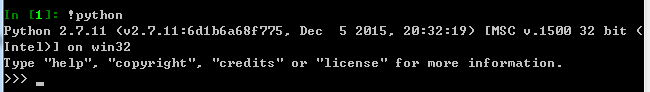




十七.shell命令和别名（这儿我好多代码执行不出来，待解决。）

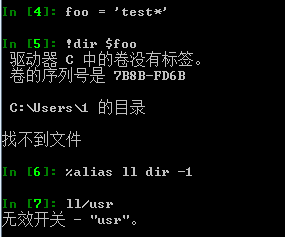
在ipython中，以感叹号(!)开头的命令行表示其后的所有内容需要在系统shell中执行。

##再启动一个python解释器



在使用！时，ipython还允许使用当前环境中定义的python值。只需要在变量名前面加上$即可

魔术命令%alias可以为shell命令自定义简称。



这是怎么回事儿~！

十八.目录书签系统

ipython有一个简单的目录书签系统，它使你能保存常用目录的别名以实现快速跳转。

（这儿我电脑同样执行不出来，待解决。）

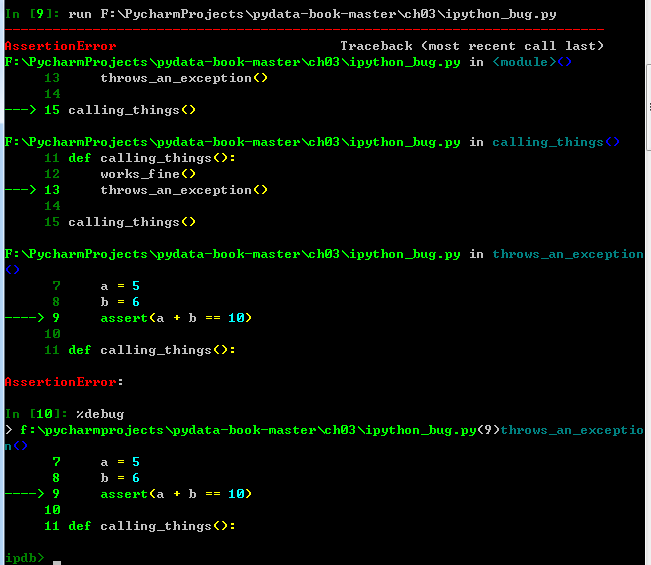
十九.软件开发工具

(一)交互式调试器

调试代码的最佳时期之一就是错误刚刚发生那会儿。

%debug命令（在发生异常之后马上输入）将会调用哪个“事后调试器”，并直接跳转到引发异常的那个栈帧（stack frame）

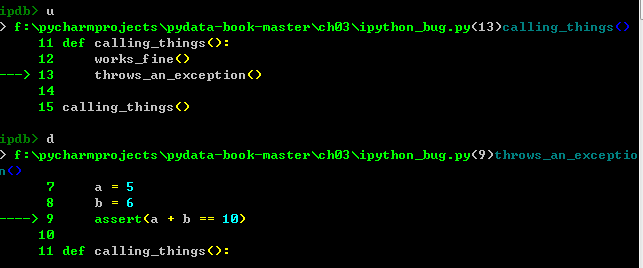
这儿需要打开ipython\_bug.py文件，需要的同学可以问我要，路径还需要自己改一改。



在这个调试器中，可以执行任意python代码并查看各个栈帧中的一切对象和数据（也就是解释还“留了条生路”的那些）。

默认从最低级开始的（即错误发生的地方）。

输入u（up）和d（down）即可在栈跟踪的个级别之间的切换



执行%Pdb命令可以让ipython在出现异常之后自动调用调试器。

此外，调试器还可以为代码开发工作提供帮助，尤其是当你想要设置断点或对函数、脚本进行单步调试以查看各条语句的执行情况时。

实现这个目的的方式有几个：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*待完善

(二)调试器的其它使用场景

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*待完善

(三)测试代码的执行时间：%time和%timeit

有时候你可能会希望了解某个复杂计算过程中到底是哪些函数占用的时间最多。

ipython专门提供了两个魔术函数（%time和%timeit）以便自动完成该过程。

%time一次执行一条语句，然后报告总体执行时间。

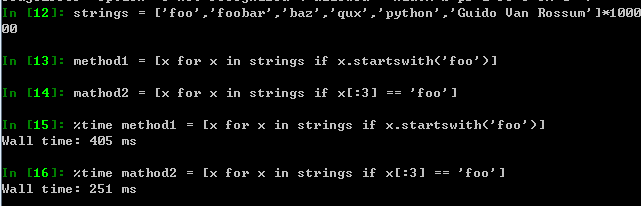
例子：下面是一个拥有60万字符串的数组，以及两个不同的“能够选出其中以foo开头的字符串”的方法：

#一个非常大的字符串数组

strings = ['foo','foobar','baz','qux','python','Guido Van Rossum']\*100000

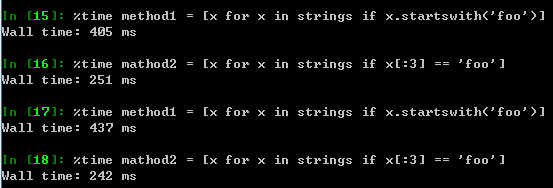
method1 = [x for x in strings if x.startswith('foo')]

mathod2 = [x for x in strings if x[:3] == 'foo']

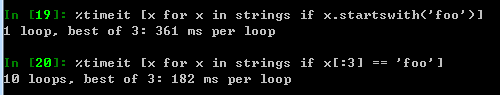
通过%time来测试：  


啦啦啦 差距出来了。

但是这并不是一个非常精确的结果，如果你对相同的语句多次执行%time的话，就会发现其结果是会变的。（亲测真的会变233看下图）



为了得到更为精确的结果，需要使用魔术函数%timeit，对于任意语句，它会自动多次执行以产生一个非常精确的平均执行时间。



(四)基本性能分析：%prun和%run -p

(五)逐行分析函数性能

二十.Ipython HTML Notebook

IPython Notebook使用浏览器作为界面，向后台的IPython服务器发送请求，并显示结果。在浏览器的界面中使用单元(Cell)保存各种信息。Cell有多种类型，经常使用的有表示格式化文本的[Markdown](http://daringfireball.net/projects/markdown/)单元，和表示代码的Code单元。

每个代码单元都有一个输出区域，在Code单元中输入代码，按 Shift-Enter 将运行此代码，代码中最后一个表达式的值将输出区域显示。如果希望屏蔽输出，可以在最后一条语句之后添加一个分号：”;”。此外，代码中还可以使用 print语句在输出区域中显示信息。

在Markdown单元中还可以直接使用Html和Javascript。

\*\*\*\*\*\*\*\*\*\*\*我的电脑进不去！！待解决

二十一.利用ipython提高代码开发效率的几点提示

(一)重新加载模块依赖项

(二)代码设计提示

1.保留有意义的对象和数据

2.扁平结构要比嵌套结构好

3.无惧大文件

二十一.高级ipython功能

(一)让你的类对ipython更加友好

(二)个性化和配置