



# Objectives

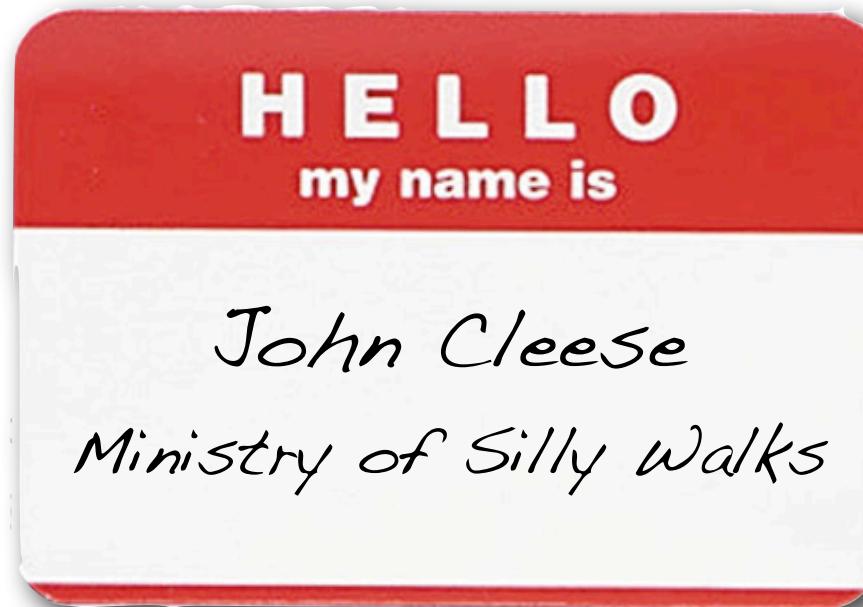
- Introduce you to the Python language
- Get you writing Python code. Build Python hacking muscle memory
- Convince you of its utility in your research life
- Instill good coding and curation practices
- We try not to proselytize, but sometimes it's too hard to resist

# Organization

- 3 days of modules (1-1.5 hr) lectures + demos  
<http://www.pythonbootcamp.info/lectures/>
- Breakout coding sessions (supervised) after each module
- Lunch + Caffeine provided
- Homework (small code projects)
- Blood, sweat, tears → a more productive you

# Connecting

In person



Wirelessly

## UC Berkeley AirBears Wireless Network Guest Account

Username

Password:

Use of this account denotes acceptance of the terms and policies set forth in the following websites:

<http://ist.berkeley.edu/airbears/fineprint> - AirBears Notice  
<http://technology.berkeley.edu/policy/> - Campus IT Policy  
<https://security.berkeley.edu/MinStdS/> - Minimum Standards for Networked Devices

Account valid: 08-23-2010 - 08-26-2010



**Something you should know about us...**

# Introduction

- What is Python?
- Why Python?
- Getting Started...

# What is Python?

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

<http://www.python.org/doc/essays/blurb/>

# What is Python?

<i>interpreted</i>	no need for a compiling stage
<i>object-oriented</i>	programming paradigm that uses objects (complex data structures with methods)
<i>high level</i>	abstraction from the way machine interprets & executes
<i>dynamic semantics</i>	can change meaning on-the-fly
<i>built in</i>	core language (not external)
<i>data structures</i>	ways of storing/manipulating data
<i>script/glue</i>	programs that control other programs
<i>typing</i>	the sort of variable (int, string)
<i>syntax</i>	grammar which defines the language
<i>library</i>	reusable collection of code
<i>binary</i>	a file that you can run/execute

# Development History

- started over the Christmas break 1989, by Guido van Rossum (now at Google)
- developed in the early 1990s
- name comes from **Monty Python's Flying Circus**
- Guido is the Benevolent Dictator for Life (BDFL), meaning that he continues to oversee Python's development.



# Development History

- Open-sourced development from the start (BSD licensed now)  
<http://www.opensource.org/licenses/bsd-license.php>
- Relies on large community input (bugs, patches) and 3rd party add-on software
- Version 2.0 (2000), 2.6 (2008), 2.7 (2010).  
We're using **2.7.2** in this class
- Version 3.X (2008) is not backward compatible with 1.X & 2.X. But 2.7 code is easily migrated to 3.X

# Why Python?

## Some of the Alternatives

### **C, C++, FORTRAN**

*Pros:* great performance, backbone of legacy scientific computing codes

*Cons:* syntax not optimized for causal programming, no interactive facilities, difficult visualization, text processing, etc.

### **Mathematics, Maple, Matlab, IDL**

*Pros:* interactive, great visuals, extensive libraries

*Cons:* costly, proprietary, unpleasant for large-scale programs and non-mathematical tasks.

**Perl:** <http://www.strombergers.com/python/>

# Why Python?

- ▶ Free (BSD license), highly portable (Linux, OSX, Windows, lots...)
- ▶ Interactive interpreter provided.
- ▶ Extremely readable syntax (“executable pseudo-code”).
- ▶ Simple: non-professional programmers can use it effectively
  - great documentation
  - total abstraction of memory management
- ▶ Clean object-oriented model, but not mandatory.
- ▶ Rich built-in types: lists, sets, dictionaries (hash tables), strings, ...
- ▶ Very comprehensive standard library (batteries included)
- ▶ Standard libraries for IDL/Matlab-like arrays (NumPy)
- ▶ Easy to wrap existing C, C++ and FORTRAN codes.

# Why Python?

## Amazingly Scalable

Interactive experimentation  
build small, self-contained scripts or million-lines projects.  
From occasional/novice to full-time use (try that with C++).

## The Kitchen Sink (in a good way)

really can do anything you want, with impressive simplicity

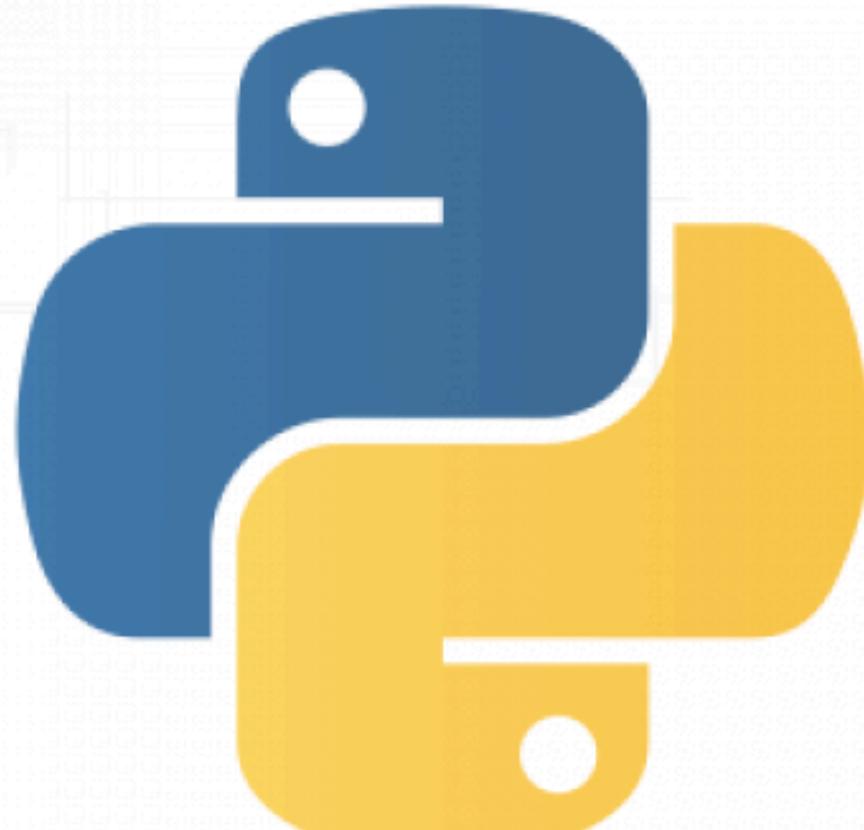
## Performance, if you need it

As an interpreted language, Python is slow.  
But...if you need speed you can do the heavy lifting in C or FORTRAN  
or you can use a Python compiler (e.g. Cython)

[Home](#) ›

## Readers' Choice Awards 2011

Dec 01, 2011 By Shawn Powers



### Best Programming Language

**Python**

*Runner-up: C++*

A three-time winner in our best programming category, Python continues to dominate. Close on its heels this year, however, is C++. In fact, a scant 6% separated the two. It's quite obvious, however, that our readers don't suffer from ophidiophobia in the least —hiss.

# My group uses it for....

## Running a robotic telescope

- interfacing with legacy hardware device drivers
- talking over serial & parallel lines to telescope control hardware
- oversee functioning of all sub-systems (themselves written in Python)
- sending email and pager alerts when distressed
- writing real-time web pages (for data display, weather)
- moving image data over the network
- interacting with databases

<http://pairitel.org>

# **My group uses it for....**

## **Data reduction & Analysis**

- processing FITS images quickly
- wrapping around 3rd party software

## **A Handy & Quick Calculator**

## **Prototyping new algorithms/ideas**

## **Making plots for papers**

## **Making fast, parallel, and efficient webservices**

<http://dotastro.org>

# My group uses it for....

Writing Google-hosted (cloud-based) websites  
that we use for research (& collaboration)

The screenshot shows the homepage of PTFInfoBot. The header features the logo "PTFInfoBot" in blue. Below the header is a navigation bar with links: Home (Main page), Preferences (Your info & preferences), PTF Sources (Look up a source by PTF name), Keywords (View all current keywords), Contact, and Links.

The main content area has two columns. The left column contains a list of recent activity entries:

- 2010-08-02 05:28:37.701861: Source [10pvh](#) added. 1 email(s) regarding this source.
- 2010-08-02 05:28:36.022048: Source [10oet](#) added. 1 email(s) regarding this source.
- 2010-08-02 05:28:35.270651: Source [10qfo](#) added. 2 email(s) regarding this source.
- 2010-08-02 05:28:35.032813: Source [10mwb](#) added. 1 email(s) regarding this source.

The right column contains several sections of text:

- Welcome to PTFInfoBot**: A brief introduction stating PTFInfoBot is an effort to assimilate and organize communications regarding PTF sources and other subjects of interest to the PTF/CDI community.
- What PTFInfoBot Does**: A description of how the app processes emails addressed to `string@ptfinfobot.appspotmail.com`, where string is any valid string whitespace, such as 'test' (Note: `feedback@ptfinfobot.appspotmail.com` is reserved for user feedback and comments, and emails sent to this address will NOT be processed by the app).
- Important note:** PTF source names must be of the form 10abc, without a space (i.e. NO abc), or PTF10abc, in order for them to be properly recognized.
- Some examples of what you can do with PTFInfoBot**: A list of two items:
  - View all emails that mention a given PTF source
  - Add PTF sources or any keyword to your watchlist, and receive email or XMPP

# Many Others Use it Too

Google

Honeywell



devis

You Tube  
Broadcast Yourself

EVE  
ONLINE

USA  
*United Space Alliance*

INDUSTRIAL  
LIGHT & MAGIC

AstraZeneca

Quora

Pollenation Internet.

PHILIPS

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Verity™

GravityZoo



Dropbox

your  
grandmother



# Interactive Notebooks

Untitled (Sage)

http://localhost:8000/home/admin/1/ Google

Apple ▾ Amazon eBay Yahoo! News ▾

```
f = u.function(x, y); f
(x,y) ↪ log(2-x/y+5)

x = var("x")
y = x^2
dy = derivative(y, x)
z = integral(sqrt(1 + dy^2), x, 0, 2)
print z
```

$$\frac{\operatorname{arcsinh}(4) + 4 \sqrt{17}}{4}$$

```
I = CDF.0
show(line([zeta(1/2 + k*I/6) for k in xrange(180)],
rgbcolor=(3/4, 1/2, 5/8)))
```

jsMath

Sage

## IP[y]: Notebook

Untitled0

[Save](#)[QuickHelp](#)

## Notebook

## Actions

[New](#) [Open](#)  
[Download](#) [ipynb](#) [Print](#)

## Cell

## Actions

[Delete](#)  
Format [Code](#) [Markdown](#)  
Output [Toggle](#) [ClearAll](#)  
Insert [Above](#) [Below](#)  
Move [Up](#) [Down](#)  
Run [Selected](#) [All](#)  
Autoindent: 

## Kernel

## Actions

[Interrupt](#) [Restart](#)  
Kill kernel upon exit: 

## Help

## Links

[Python](#) [IPython](#)  
[NumPy](#) [SciPy](#)  
[MPL](#) [SymPy](#)

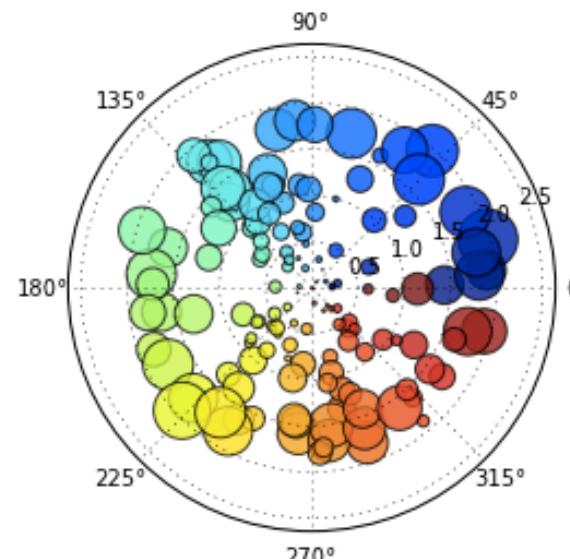
Shift-Enter : run selected cell

Ctrl-Enter : run selected cell in-place

Ctrl-m h : show keyboard shortcuts

In [1]:

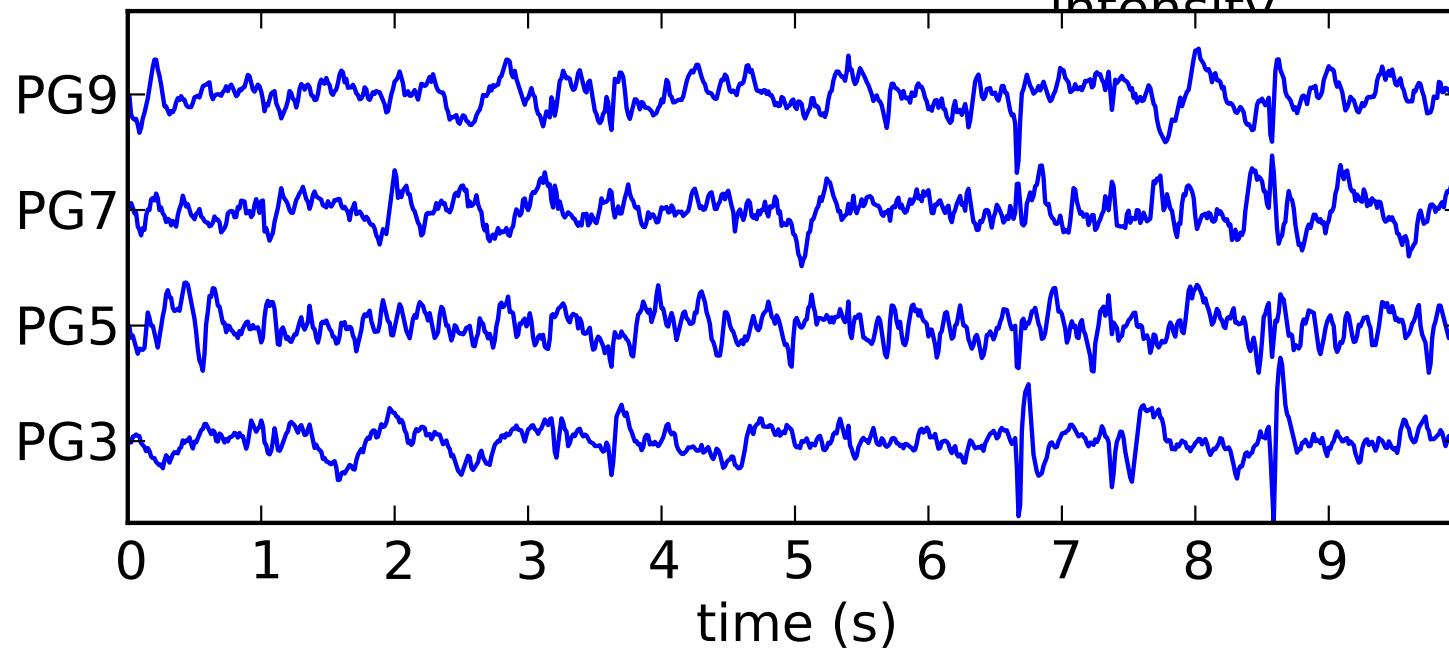
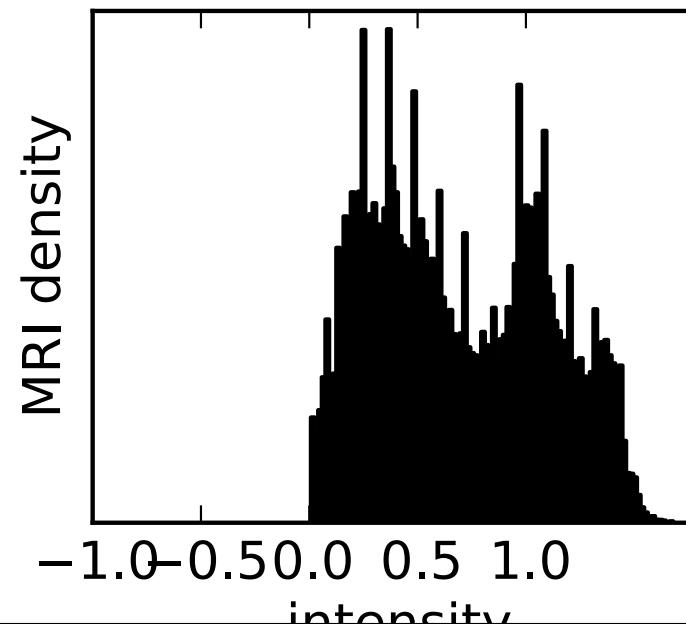
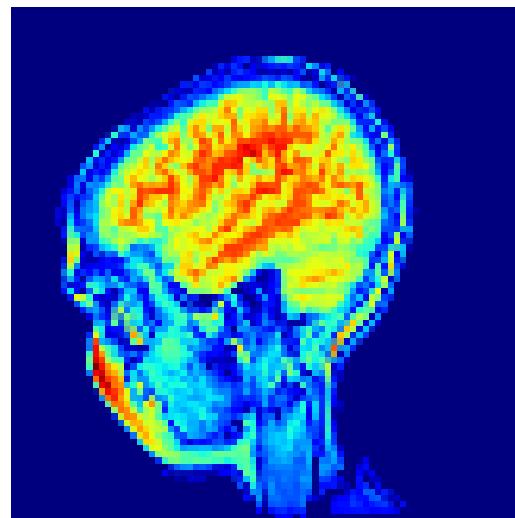
```
num_bootcampers=142
# note: rand not Ron
r = 2*rand(num_bootcampers)
theta = 2*pi*rand(num_bootcampers)
area = 200*r**2*rand(num_bootcampers)
colors = theta
ax = subplot(111, polar=True)
c = scatter(theta, r, c=colors, s=area)
c.set_alpha(0.75)
```



In [ ]:

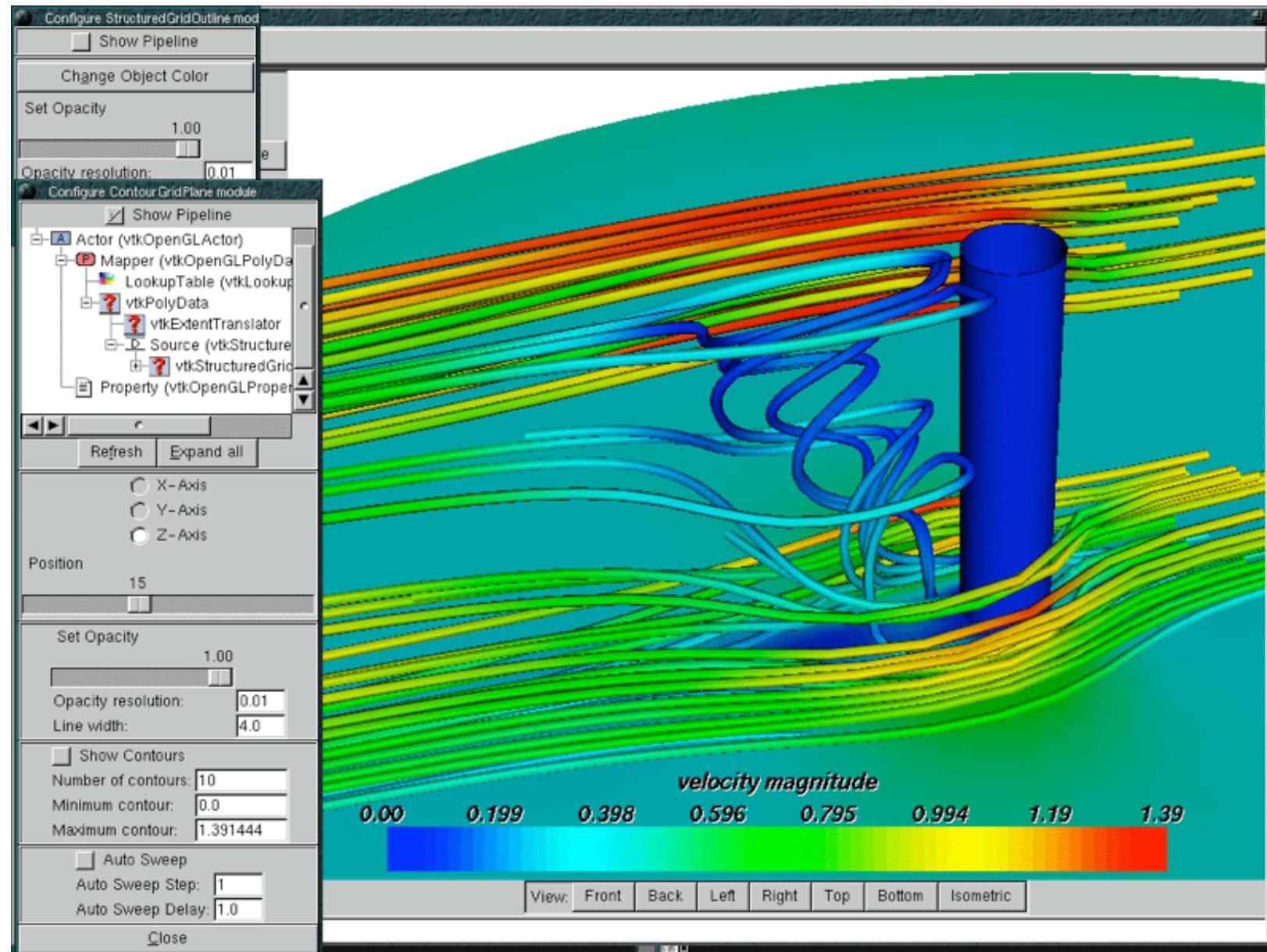
iPython notebook

# Visualization



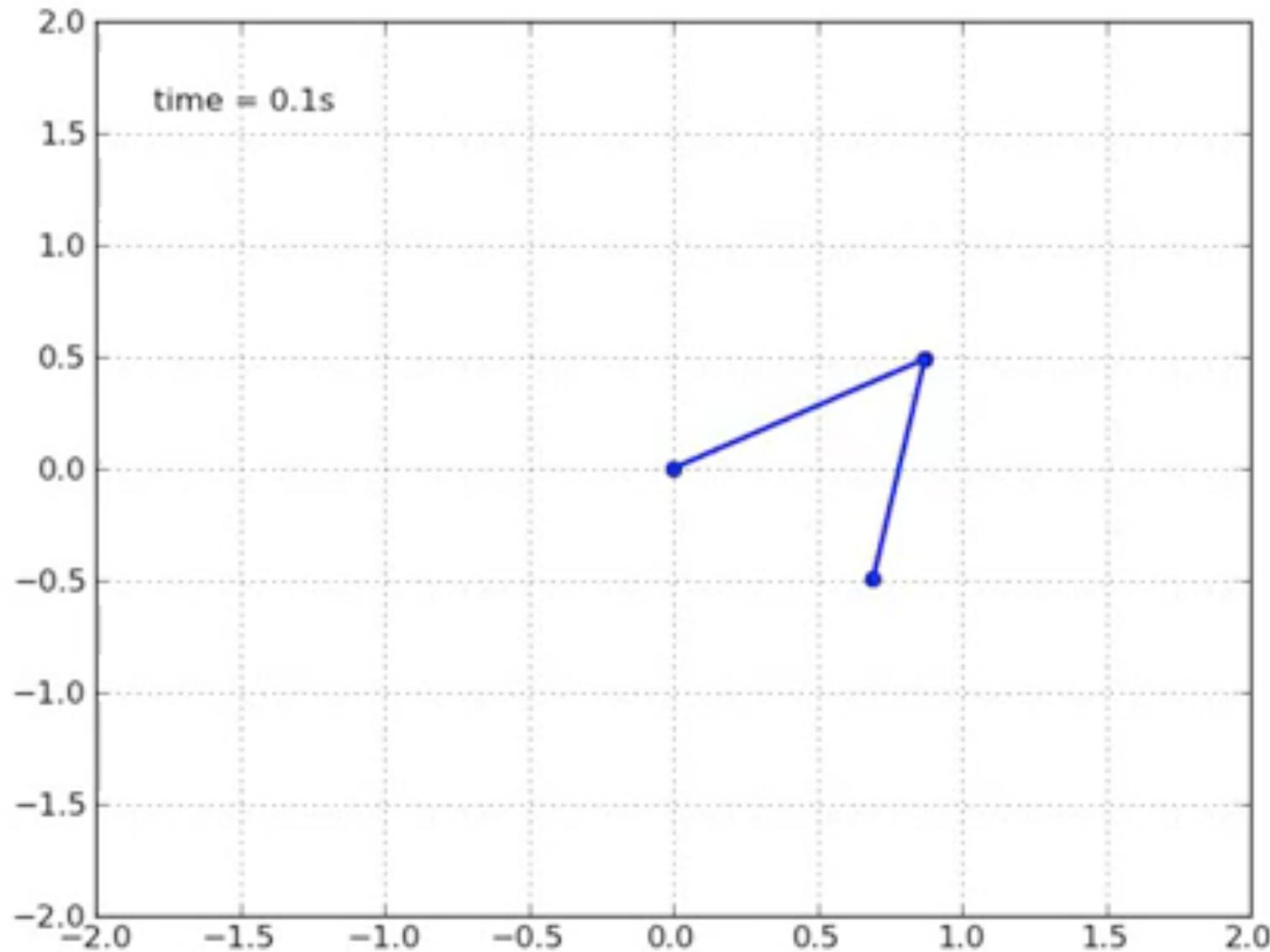
matplotlib

# Visualization



Mayavi

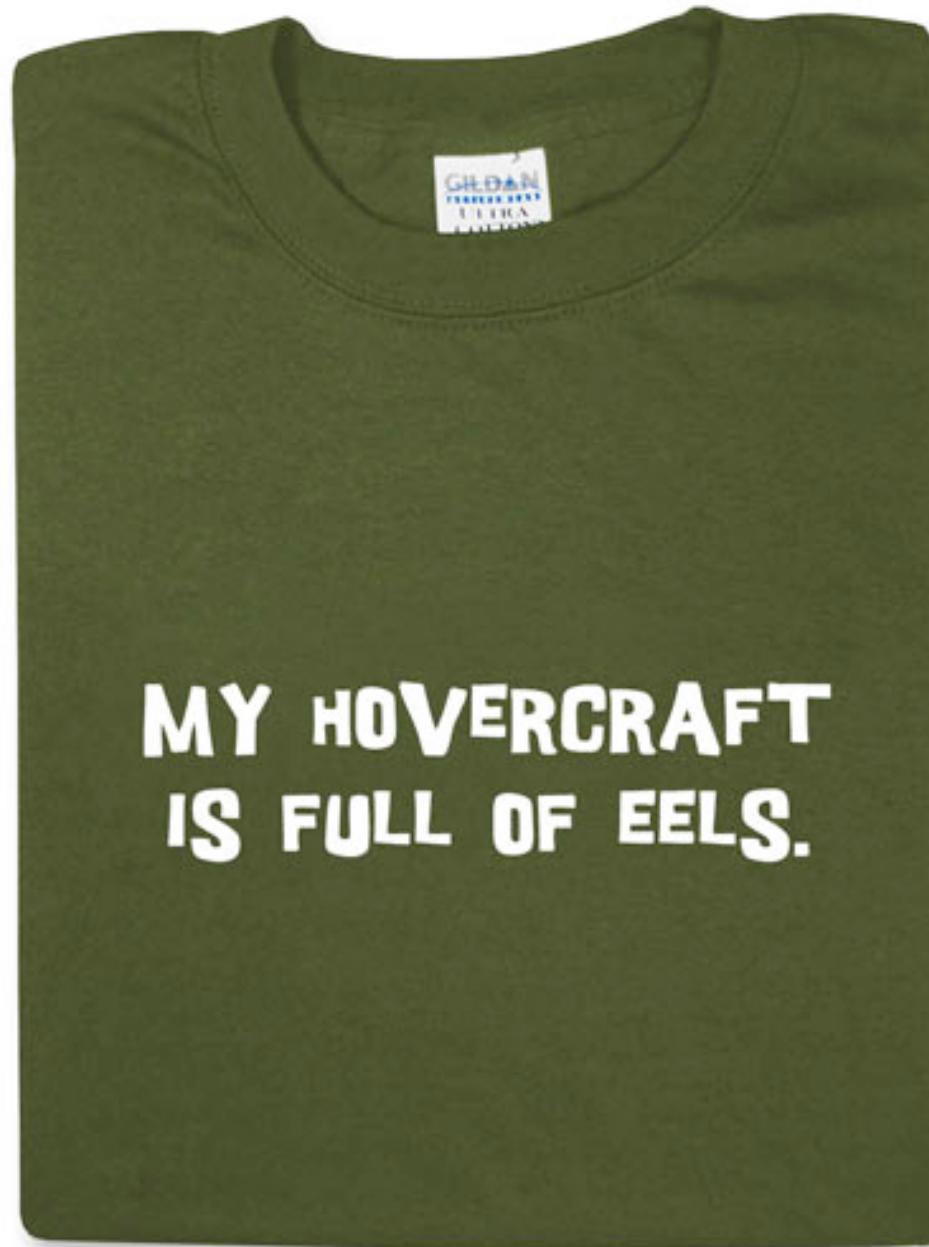
# Animation



[http://matplotlib.sourceforge.net/examples/animation/double\\_pendulum\\_animated.html](http://matplotlib.sourceforge.net/examples/animation/double_pendulum_animated.html)

# Tackling Python: Let's Get Started





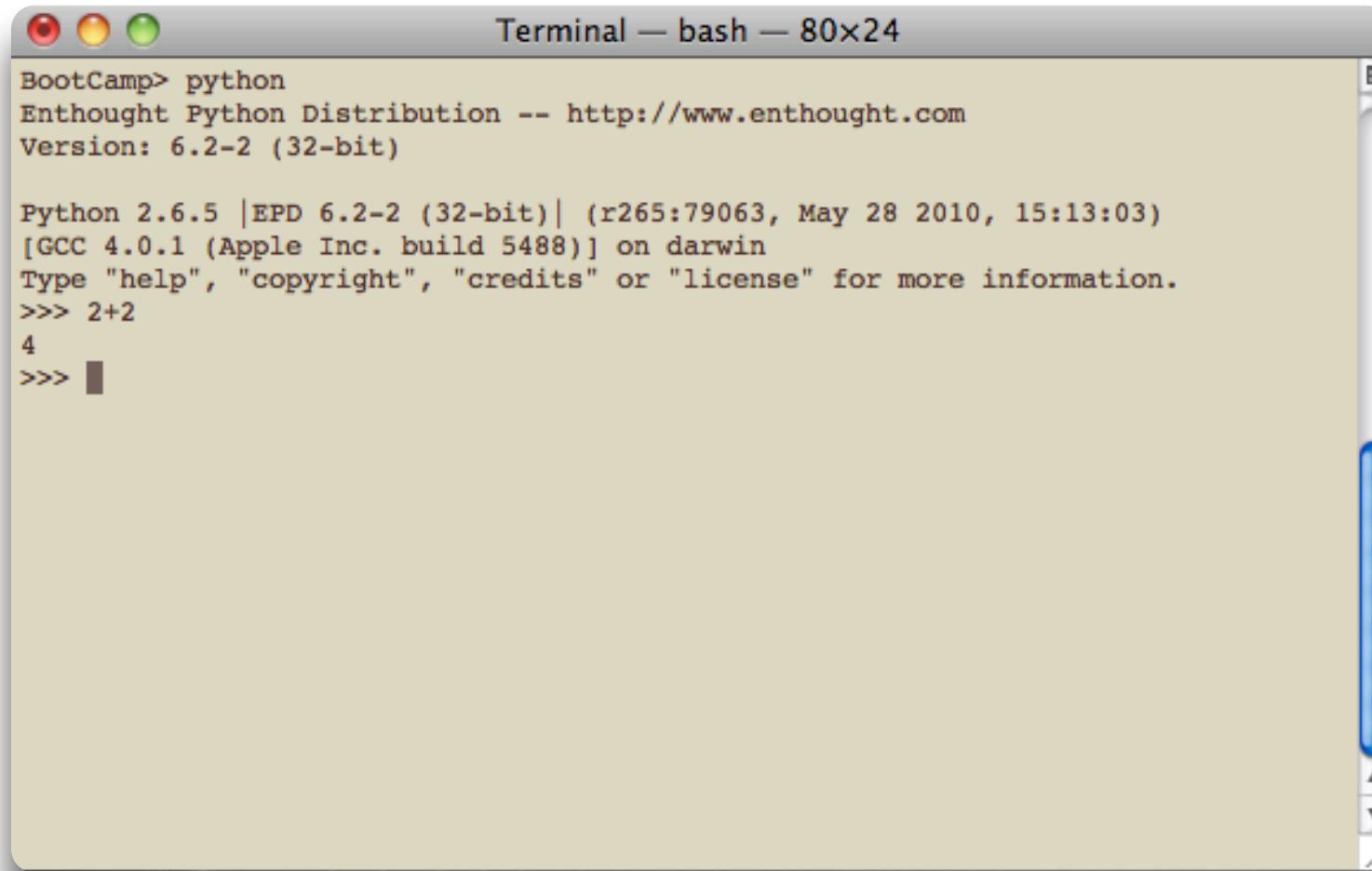
## **Asking Questions during Module presentations**

- Speak up
- Raise your hand

## **Getting Help at Any Time**

- Raise your hand and make eye contact with a counselor
- Join the IRC chat (#pyboot)
- Send email [ucbpythonclass+bootcamp@gmail.com](mailto:ucbpythonclass+bootcamp@gmail.com)

# Firing up the Interpreter

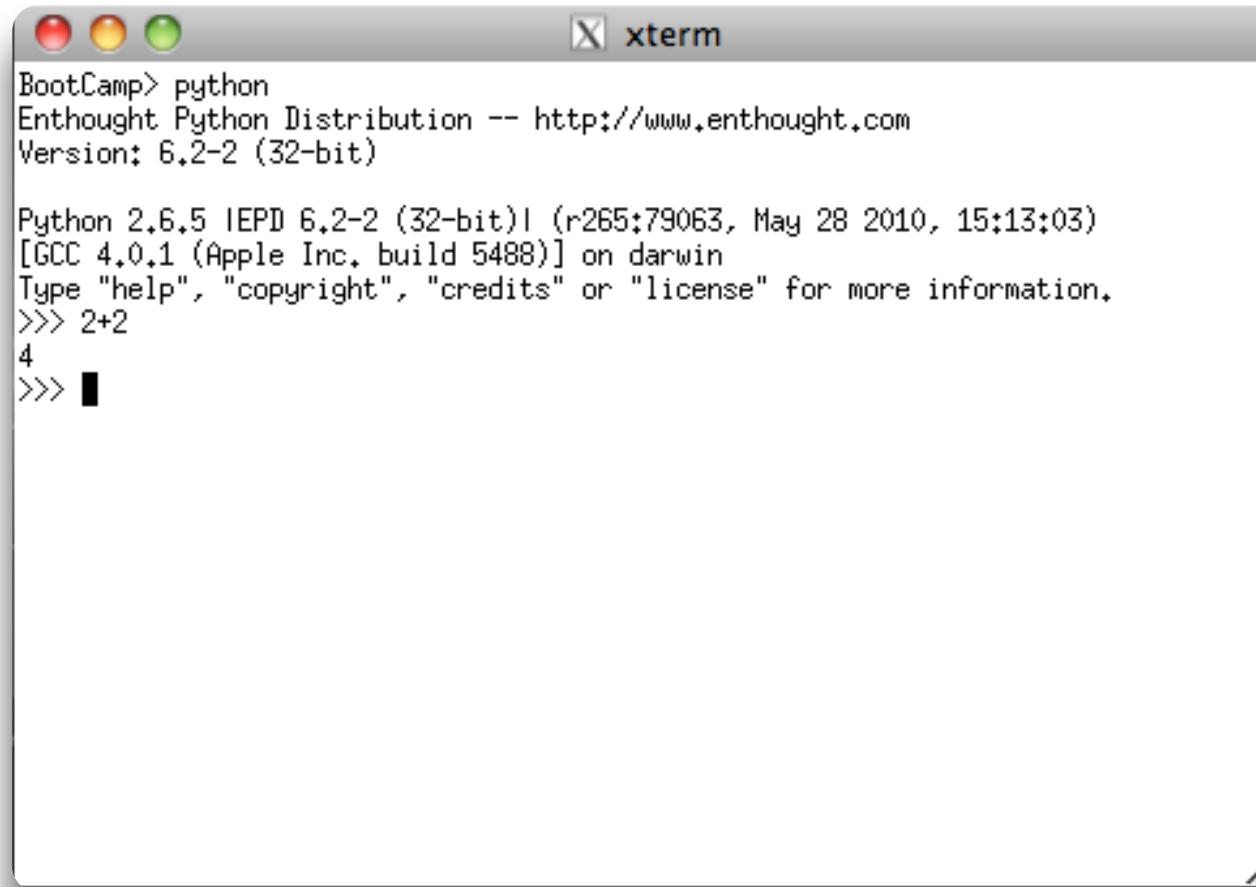


```
BootCamp> python
Enthought Python Distribution -- http://www.enthought.com
Version: 6.2-2 (32-bit)

Python 2.6.5 |EPD 6.2-2 (32-bit)| (r265:79063, May 28 2010, 15:13:03)
[GCC 4.0.1 (Apple Inc. build 5488)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> █
```

Mac OS X (Terminal)

# Firing up the Interpreter



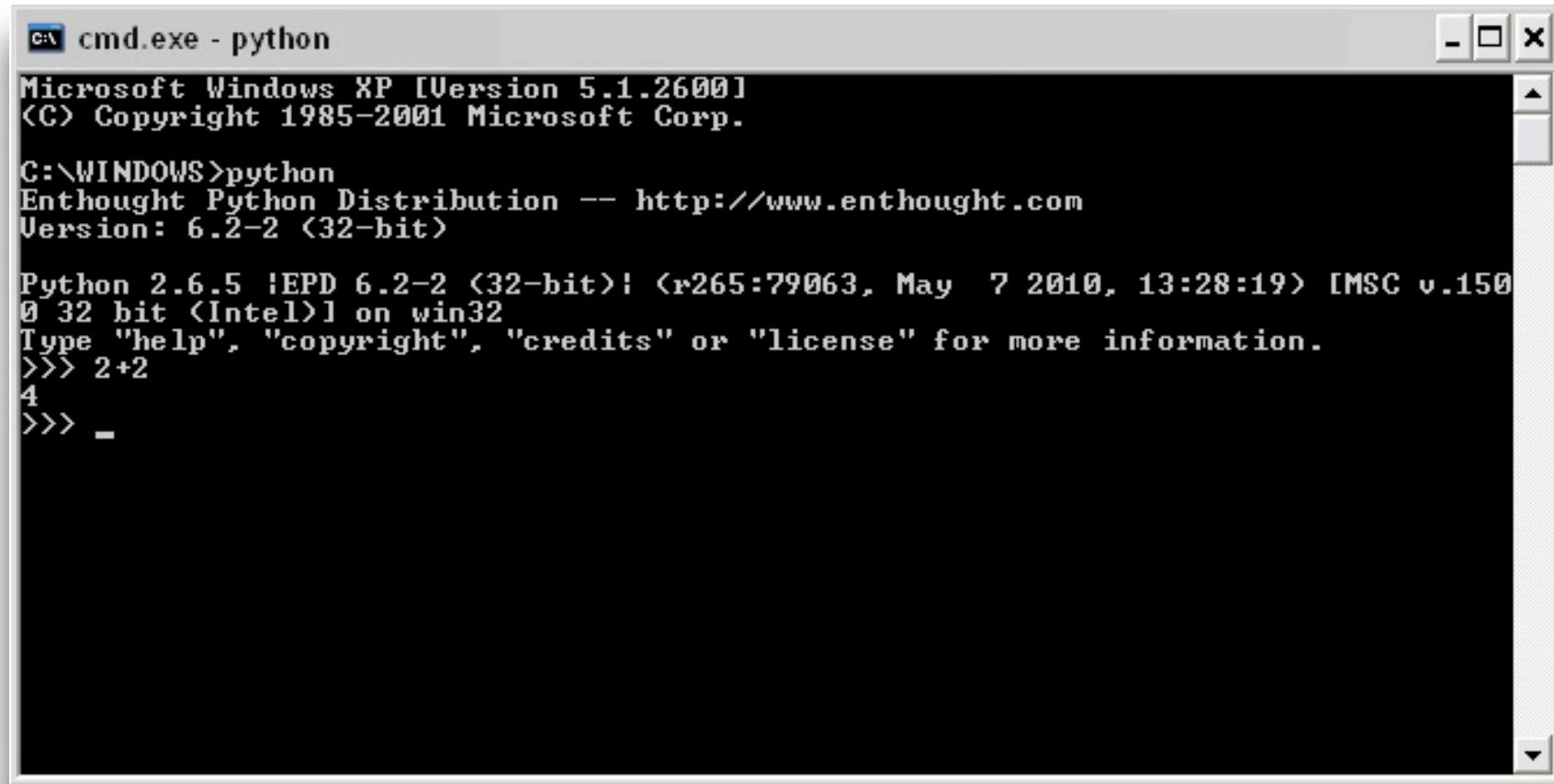
The image shows a screenshot of an xterm window titled "xterm". The window has a Mac OS X style title bar with red, yellow, and green buttons. The terminal displays the following text:

```
BootCamp> python
Enthought Python Distribution -- http://www.enthought.com
Version: 6.2-2 (32-bit)

Python 2.6.5 |EPD 6.2-2 (32-bit)| (r265:79063, May 28 2010, 15:13:03)
[GCC 4.0.1 (Apple Inc. build 5488)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> █
```

Linux/UNIX/Mac OS X (X11/Xterm)

# Firing up the Interpreter



A screenshot of a Microsoft Windows XP desktop environment. In the foreground, there is a command-line window titled "cmd.exe - python". The window displays the following text:

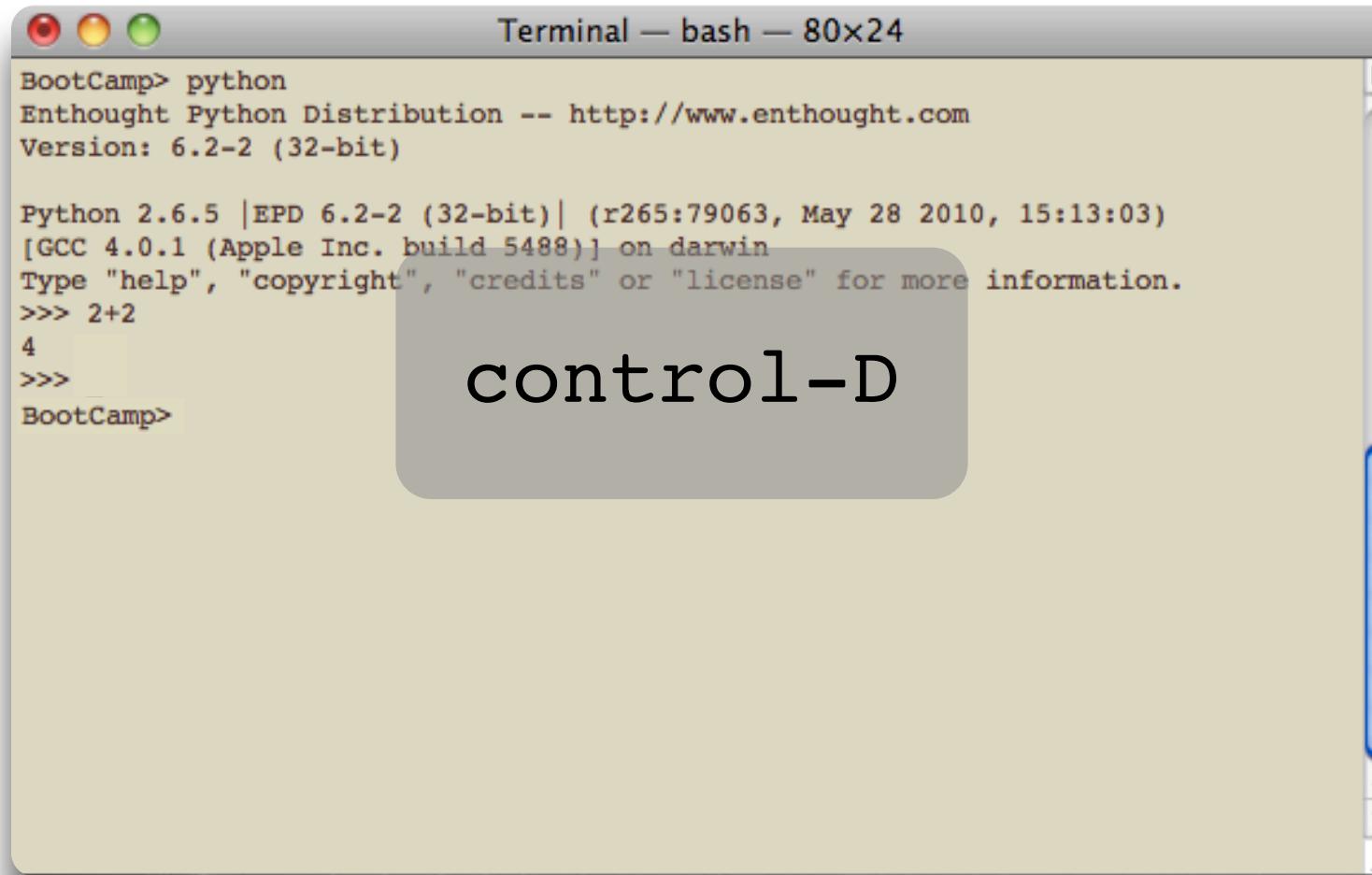
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS>python
Enthought Python Distribution -- http://www.enthought.com
Version: 6.2-2 (32-bit)

Python 2.6.5 |EPD 6.2-2 (32-bit)| (r265:79063, May  7 2010, 13:28:19) [MSC v.150
0 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> -
```

Windows

# Firing up the Interpreter



```
BootCamp> python
Enthought Python Distribution -- http://www.enthought.com
Version: 6.2-2 (32-bit)

Python 2.6.5 |EPD 6.2-2 (32-bit)| (r265:79063, May 28 2010, 15:13:03)
[GCC 4.0.1 (Apple Inc. build 5488)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>>
BootCamp>
```

control-D

to exit: either **control-D** or **exit()**

# Firing up the Interpreter

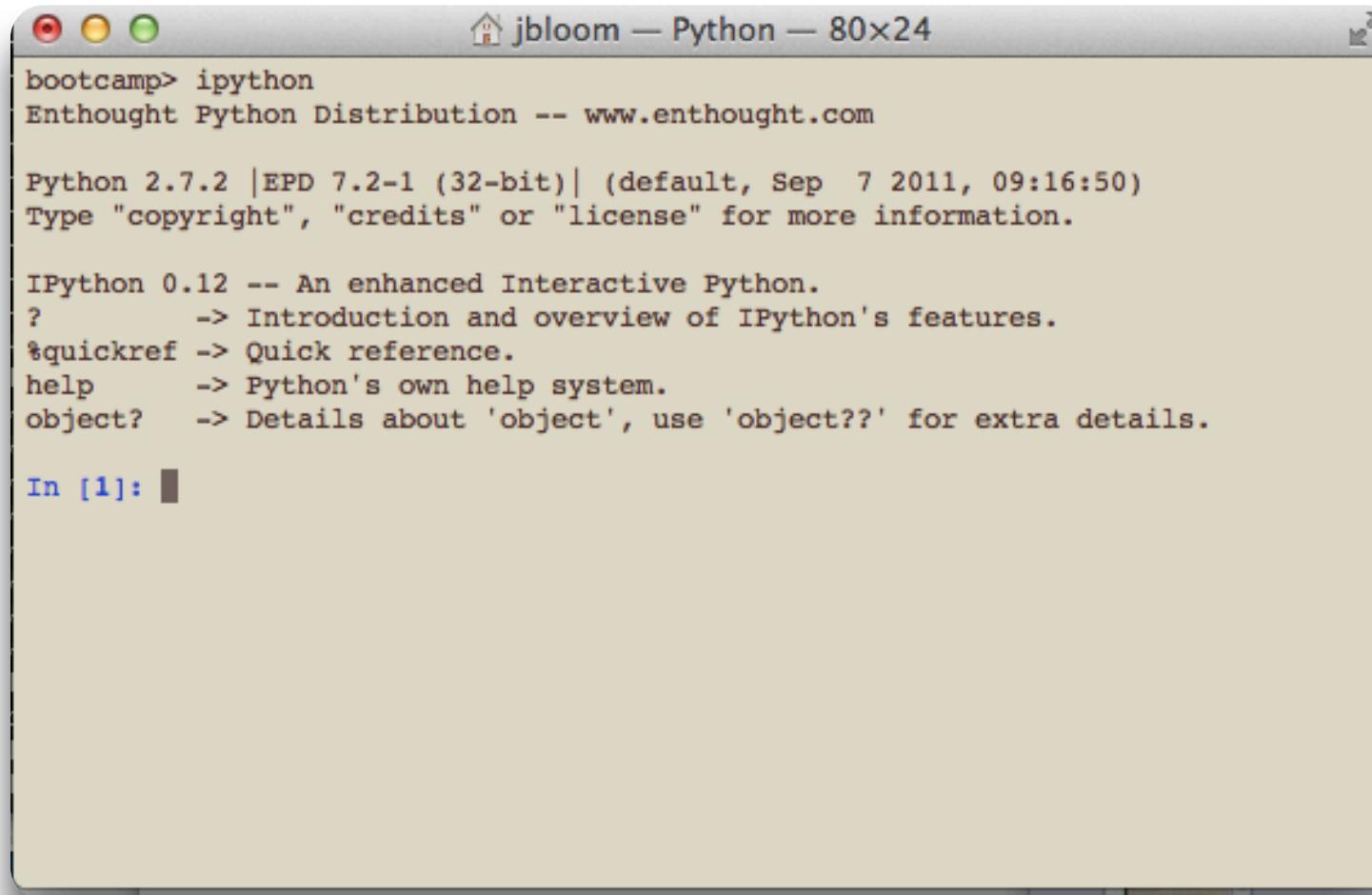
```
BootCamp> python
Enthought Python Distribution -- http://www.enthought.com
Version: 6.2-2 (32-bit)

Python 2.6.5 |EPD 6.2-2 (32-bit)| (r265:79063, May 28 2010, 15:13:03)
[GCC 4.0.1 (Apple Inc. build 5488)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>>
BootCamp>
BootCamp> python
Enthought Python Distribution -- http://www.enthought.com
Version: 6.2-2 (32-bit)

Python 2.6.5 |EPD 6.2-2 (32-bit)| (r265:79063, May 28 2010, 15:13:03)
[GCC 4.0.1 (Apple Inc. build 5488)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> exit()
BootCamp>
```

to exit: either **control-D** or **exit()**

# Firing up the Interpreter



A screenshot of a terminal window titled "jbloom — Python — 80x24". The window shows the startup of the IPython interpreter. The text output is as follows:

```
bootcamp> ipython
Enthought Python Distribution -- www.enthought.com

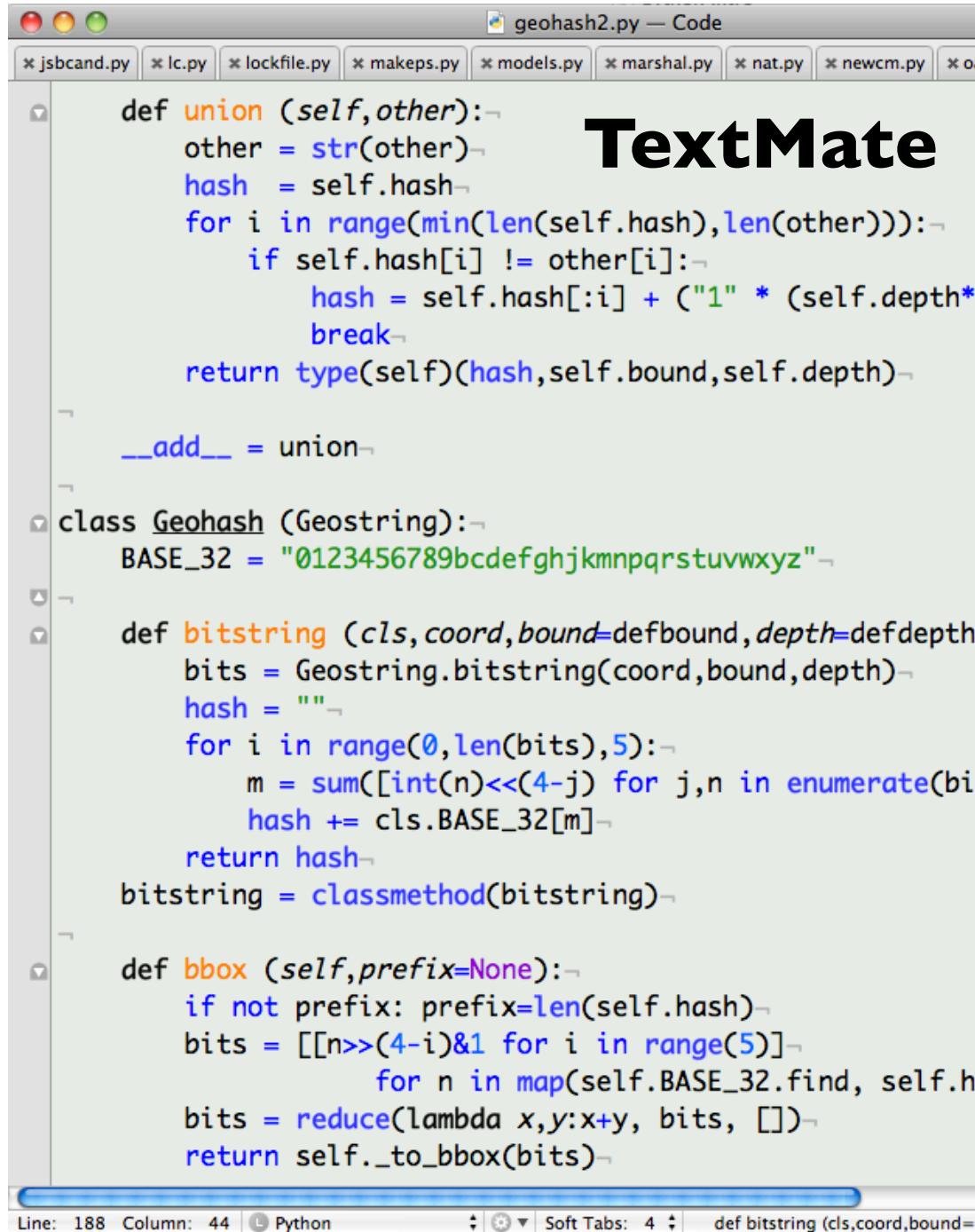
Python 2.7.2 |EPD 7.2-1 (32-bit)| (default, Sep  7 2011, 09:16:50)
Type "copyright", "credits" or "license" for more information.

IPython 0.12 -- 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]:
```

ipython

# Editing Python Files



A screenshot of the TextMate code editor. The title bar says "geohash2.py — Code". The editor shows Python code for a Geohash class. The code includes methods like `union`, `bitstring`, and `bbox`. The TextMate logo is overlaid in the center of the editor window.

```
def union (self,other):-
    other = str(other)
    hash = self.hash
    for i in range(min(len(self.hash),len(other))):-
        if self.hash[i] != other[i]:-
            hash = self.hash[:i] + ("1" * (self.depth*2))
            break
    return type(self)(hash,self.bound,self.depth)

__add__ = union

class Geohash (Geostring):-
    BASE_32 = "0123456789bcdefghjklmnpqrstuvwxyz"

    def bitstring (cls,coord,bound=defbound,depth=defdepth):-
        bits = Geostring.bitstring(coord,bound,depth)
        hash = ""
        for i in range(0,len(bits),5):-
            m = sum([int(n)<<(4-j) for j,n in enumerate(bits[i:i+5])])
            hash += cls.BASE_32[m]
        return hash
    bitstring = classmethod(bitstring)

    def bbox (self,prefix=None):-
        if not prefix: prefix=len(self.hash)
        bits = [[n>>(4-i)&1 for i in range(5)]-
                for n in map(self.BASE_32.find, self.hash)]
        bits = reduce(lambda x,y:x+y, bits, [])
        return self._to_bbox(bits)
```

Line: 188 Column: 44 Python Soft Tabs: 4 def bitstring (cls,coord,bound=defbound,depth=defdepth):-



usually we name  
python files with  
a `.py` suffix

- **snazzy GUI-based editors:**  
BBEdit, TextWrangler (Mac);  
NotePad++, SublimeText (Windows);  
KWrite, Scribes, eggys (linux)

- **old/powerful editors:**  
vim, emacs, nano, ...

<http://bit.ly/ucb-textmate>

<http://wiki.python.org/moin/PythonEditors>

