

# PYTHON CRASH COURSE

<http://eddiema.ca/docs/pycrashcourse/>

LECTURE 1



```
def tell_me_if_i_am_crazy(i_see_a_python):  
    if i_see_a_python:  
        print 'Quite crazy.'  
    else:  
        print 'Less crazy.'
```

EDDIE MA  
JANUARY 2010  
<http://eddiema.ca>

# GETTING MORE HELP

From me

edoules

Skype

ed.yt.ma@gmail.com

G-Talk, E-Mail, MSN

# THE BIG IDEA

What I want you to get out of this course

- How to...
  - Recognize problems you can solve with Python
  - Solve such problems
  - Select the easiest programming solutions
- Absorb...
  - Just enough technical jargon to communicate

**BUT**

# INSTALLING PYTHON

Python is for everyone\*

- Everyone go to <http://python.org/download>
- Windows and Mac OS X users:
  - Download and Install Python 2.6.4
  - Takes roughly ten minutes total
- Linux users probably already have Python installed...

\*That brought a laptop today

# ANSWERS

For Windows and Mac

- Windows Installation
  - Just run the installer you downloaded and everything will be OK
- Mac Installation
  - Mount (open) the disk image you just downloaded
  - The installer is the icon labelled `Python.mpkg`
    - Run the installer
  - Unmount (eject) the disk image after you're done

# ANSWERS

For Linux

- Linux Users
  - Updating to the latest Python 2
    - At the time of this writing it's 2.6.2 for Ubuntu
  - *Update at home, this takes a while!*
    - *Just use what's already there for today*
    - Open a terminal, type:
      - `sudo apt-get install python-dev`
  - Linux gets interesting, contact me if you have any additional problems.

# DID IT ALL GO OKAY?

Checking if it was installed correctly

- Open up a terminal window
  - Windows Vista: Click on start and type `cmd`, enter
  - Good Windows: Go to start, then run and enter `cmd`
  - Mac OS: Go to spotlight, then type `terminal`
- Type `python`
- You should see something like the following

```
129-97-185-218:~/zincsvn/PyCrashCourse eddiema$ python
Python 2.6.4 (r264:75821M, Oct 27 2009, 19:48:32)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

# GREAT, WE'RE READY.

## Python in Console Mode

- You're now running Python in console mode
  - Also called interactive mode because it's interpreting you, a living being instead of a script
- Console mode...
  - Is useful for trying out code fragments
  - Makes Python your calculator

# CALCULATE STUFF!

Try these expressions, note down how Python interpreted them.

- $3+17+1$
- $13*2*2$
- $4-8-1$
- $4-(8-1)$
- $8/2$
- $9/2$
- $9/2.0$
- $9.0/2$
- $0\%3$
- $1\%3$
- $2\%3$
- $3\%3$
- $4\%3$
- $5\%3$
- $4**2$
- $9**0.5$
- $9**(1/2)$
- $9**(1/2.0)$
- $9**(1.0/2)$
- $9**1/2.0$

I'll take questions about the math after the next two slides.

# WHAT HAPPENED?

How Python interprets mathematical expressions

- *Normal order of operations* applies along with left-to-right precedence when operators are tied
- ADDITION “+”, SUBTRACTION “-” and MULTIPLICATION “\*” are straight forward
- DIVISION “/” drops off the fractional part of the answer if it only received integers to work with.
  - We force DIVISION to give us real numbers by explicitly saying “2.0” instead of “2”
- BRACKETS “(”, “)” modify order of operations as expected

# AND THE REST

But you've probably figured them out

- MODULUS “%” gives us the remainder of DIVISION
- EXPONENTIATION “\*\*” is straight forward too
  - Remember, a fractional power is that power's reciprocal's root ( $a^{1/3} = \sqrt[3]{a}$ )
  - And three rights make a left

# Take a Breath!

Five minute brain break

This is the 3/4 checkpoint of the lesson.

There's only two more small itty bitty items to learn.

I promise your brain can take it.

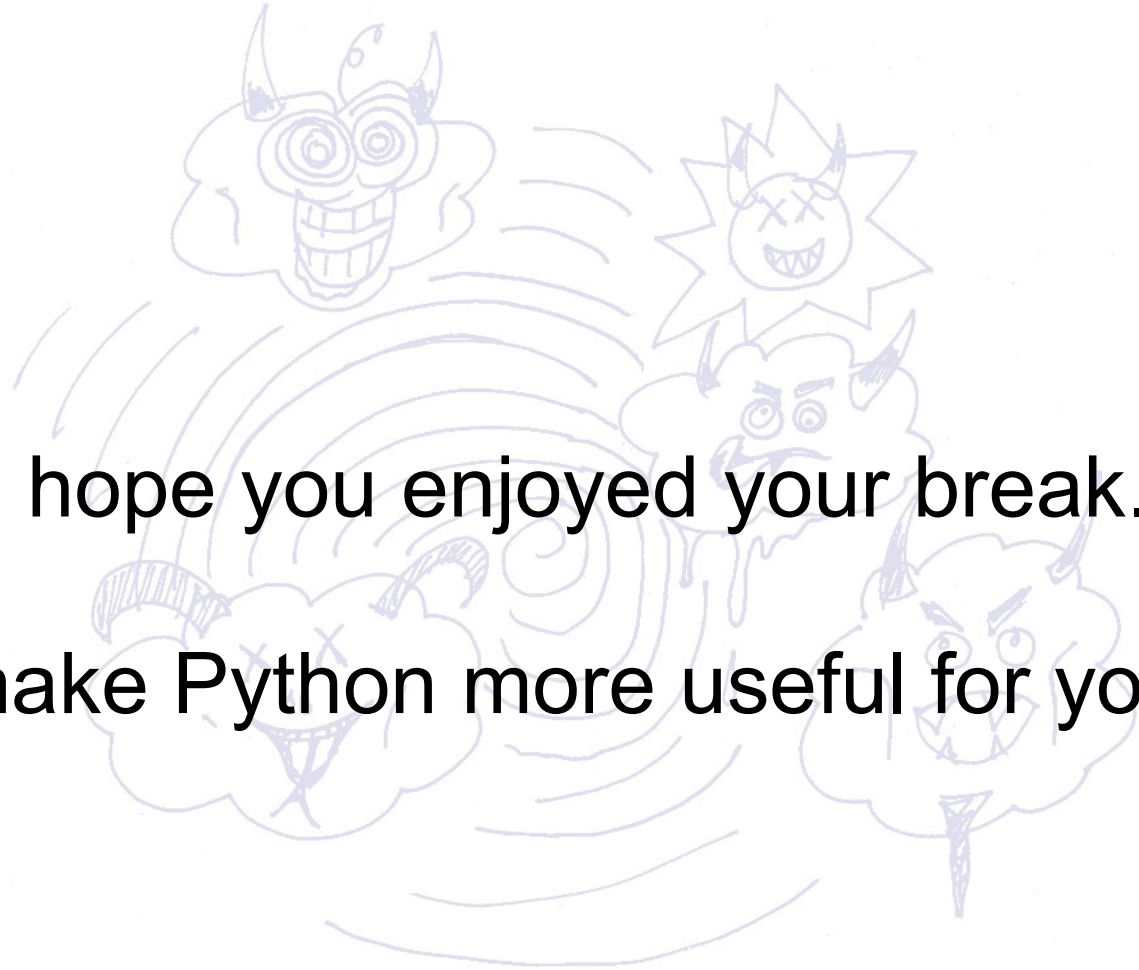
Have some Youtube.

# Welcome Back

Thanks for coming back to the swirling madness

I hope you enjoyed your break.

Let's make Python more useful for you now.



# TWO FINAL ITEMS

Get out your terminal windows...

- These items are...
  - The `print` statement
  - Variables

# THE `print` STATEMENT

It prints stuff

- Start Python in interactive mode again.
- Try each of the following commands and observe what Python does
  - `print 3`
  - `print 3+5*2`
  - `print '3+5*2'`
  - `print 'the'`
  - `print the`

I'll take questions about these examples after the next three slides.  
(Yes, including the scary looking error...)

# WHAT HAPPENED THIS TIME?

Why did Python do what it did?

- The first two examples, “`print 3`” and “`print 3+5*2`” were pretty straight forward.
  - It didn't do anything different than when we didn't say `print`
  - In interactive mode, the result of many statements that Python evaluates (runs) is already printed to the screen
  - `print` is used to force Python to output
  - This is especially useful when Python runs a script since Python wouldn't show you things interactively

# AND THE NEXT TWO?

What's the deal with the quoted items?

- The examples `print '3+5*2'` and `print 'the'` use things called **STRINGS**.
  - A **STRING** is a sequence of characters
  - A character is a little glyph like `"7"` or `"b"` that you can type on the keyboard
  - A **STRING** in python is specified with matched single quotes or double quotes.
  - It doesn't matter which kinds of quotes we use for today.
  - A print statement just prints **STRINGS** literally.

# WHAT ABOUT THE SCARY ERROR?

Don't worry, that's normal!

- Did Python say something like this?

```
>>> print the
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'the' is not defined
>>>
```

- That's an EXCEPTION
- For now, EXCEPTIONS just tell us that Python couldn't finish evaluating the most recent statement
- We'll return to this particular “NameError” in just a moment...

**NOW, ONTO OUR LAST**

# VARIABLES

Just like in Algebra! Only different...

- VARIABLES are names we use to refer to OBJECTS
- OBJECTS in Python include the simple INTEGERS, FLOATS and STRINGS we have already encountered
- A VARIABLE in Python doesn't have a particular TYPE, so it can point at any kind of OBJECT.
- A variable's name is just a word
- Variables may only be named using [a-z], [A-Z], [0-9] and “\_” and cannot start with [0-9].
  - e.g.: “radius\_degrees”, “\_angle”, “densityCount”, “protein75”, “\_id\_”

# TRY THESE STATEMENTS!

Observe how Python treats variables.

- Round things...

```
r=13.0
```

```
d=2*r
```

```
pi=3.14
```

```
a=pi*r**2.0
```

```
c=2.0*pi*r
```

```
v=4.0/3.0*pi*r**3.0
```

```
s=4.0*pi*r**2.0
```

```
print d,a,c,v,s
```

- A little more on Strings...

- CAREFUL: Watch the spaces after words

```
prep = "away "
```

```
part = "jumping "
```

```
noun = "cat "
```

```
verb = "is "
```

```
print noun + verb + part + prep
```

```
print part + prep + verb + noun
```

```
print part + 3*prep
```

```
print "not a " + part + noun
```

```
print len(part)
```

```
print len(noun)
```

```
print len("cat")
```

```
print len("cat ")
```

# HOW ABOUT THE `NameError`?

Exception explained!

- If you'll recall, I asked you to type `print 'the'` followed by `print the`.
  - In the first, a string is printed
  - But in the second, a variable is printed
- The `NameError` is Python's way of telling us that we never defined a variable named `the`!
- Don't worry, you'll see `NameError` again among many many other exceptions!

# HOMework

Do things with stuff you just learned!

- Use Python to help you calculate the volume, area of one face and surface area of a 3D cube where one edge is 22.576 units long
- Use Python to figure out how long this sentence is in characters (hint: starting from and including the word Use and ending with the final close bracket-- more hints: string, len() function)
- Answer these – hint – try and find out!
  - What kind of error does Python give you when you divide by zero?
  - What kind of error does Python give you when you try to divide a string by a number?

# THE END

Next week...

- Writing and running your own scripts
- Arguing with the command line
  - (Command line arguments)
- Functions
- Conditional branching and looping
  - Using `if`, `elif`, `else`, `while`
  - The amount of truthiness something contains
    - (Boolean conditions: `true` and `false`)

# SNEAK PREVIEW

Instructions about the code at the bottom of this page...



- Copy the code into Python in interactive mode
- The whitespaces on the left side are tabs
- Press enter an extra time to finish
- Congrats! You've just defined your first function
- Now try  
`tell_me_if_i_am_crazy(True)`
- and try  
`tell_me_if_i_am_crazy(False)`

```
def tell_me_if_i_am_crazy(i_see_a_python):  
    if i_see_a_python:  
        print 'Quite crazy.'  
    else:  
        print 'Less crazy.'
```