COMS 4995-1: Intro Social Networks Thursday, September 6th, 2012

A Basic Introduction To Python

Chris J. Riederer (TA) A. Chaintreau (instructor)

1 Introduction

Python is an easy-to-use and very powerful language. We'll be using it to help us analyze social networks.

2 Getting Help

Since Python has many users, it has a many online resources. Below are some sites that can help you quickly learn python.

http://www.python.org/
http://www.python.org/getit/
http://www.codecademy.com/tracks/python
http://www.trypython.org/
http://docs.python.org/tutorial/
The "help()" command in Python. Just start running Python, and type "help()".
That whole "Google" thing.

Because Python is an interpreted language, it's really easy and quick to experiment with. You can run just a few lines of code to see results and quickly experiment or try out different commands to see what works.

3 Hello, world!

```
print "Hello, world!"
```

No need for semicolons. The end of the line will be the end of the statement.

4 Data types

Unlike Java or C, Python does not require you to declare data types.

```
x = 1337
name = 'Chaintreau'
myFraction = 0.25
```

You can change variables to other types. Floats and integers are often interchangeable. The result will usually be a float.

```
x = "I'm a string!"
```

```
x = 1 + 3.4
x = 7 / 14.5
```

5 A Quick Comment

Python has single-line and multi-line comments. Single-line comments start with a # symbol. Anything after the # will be ignored.

```
x = 5 # I just set x to be 5 here.
```

Multiline comments are created by repeating a quotation mark three times.

```
"""
This is a multi-line comment.
I'm just a verbose commentor, I guess.
"""
```

6 Advanced Math: Importing a Module

Python keeps a lot of code in modules. This is equivalent to a library or header file in other languages. Suppose we want to find the square roots of some numbers.

```
sqrt(4) # gives an error
import math
math.sqrt(4) # returns 2
import math as m # you can shorten/change module names.
m.sqrt(4) # return 2
m.floor(4.3) # returns 4
from math import sqrt # import just one function
sqrt(4) # return 2
floor(4.3) # gives an error, we didn't import it
from math import * # import everything
floor(4.3) # returns 4
```

7 Conditionals

Like C, Java, and other common programming languages, Python has conditional statements. Unlike C or Java, which uses { } to designate what statements are conditionally executed, Python uses indentation. Indentation can be either spaces or tabs, but shouldn't change in length or type from line to line. See under caveats for more about indentation.

```
x = True
if x:
    print "This will print"
else:
    print "this will not print :("
x = 7
if x == 8:
```

```
    print "python uses standard logic operators, except for !"
elif x <= 7:
    print "this will print!"
x = 12
if x:
    print "Python treats non-zero integers as True"
if number is 12:
    print "You can use 'is'"
if number is not 11:
    print "you can also use 'is not', which will work like you'd expect"
```

8 Lists

Lists are easy to make and use.

```
x = [1,2,3,4,5]
print "This should be 1:", x[0]
```

Some functions return lists.

```
x = range(5)
```

You can add to them with methods.

```
x = []
x.append(1)
x.append(2)
x.append([3,4,5])
```

What do you think the list `x` looks like at the end of these commands?

9 For Loops

Python also has for loops. Instead of assigning a variable which always increases, Python iterates over every item in a list (or iterable). You can do this with any item that Python might think of as a list or iterable, such as strings.

```
for x in range(5):
    print x
for x in [25,16,9,4,1]:
    print Math.sqrt(x)
for myLetterVariable in "Not actually a sentence":
    print "The next letter is", myLetterVariable
```

10 Using Built-in String Methods

Python has a large number of built-in methods and functions. Here are some examples of string methods.

```
haiku = """An old silent pond
A frog jumps into the pond,
Splash!  Silence again"""
print haiku.upper() # prints all uppercase haiku
haiku.upper()
print haiku # prints the original haiku, calling methods returns a NEW string.
# It doesn't modify the old.
print haiku.split() # returns a list.
# The elements of the list will be the words of the haiku.
# split() breaks apart strings at whitespace.
print haiku.split('\n') # returns a list.  The elements of the list are the lines of the
haiku.
# Giving a parameter to split() will break apart a string only at that parameter.
```

11 While loops

While loops are pretty simple and should be about what you'd expect, given what you've seen from the if and for loops.

```
x = 7
while x > 0:
      print x
      x -= 1
```

The keywords `continue` and `break` do the same thing as in other languages. They go to the next loop variable or break out of the loop, respectively.

12 Functions

Functions are created with the "def" keyword. Note the indentation after the function's name.

```
def Hypotenuse(a, b):
    return Math.sqrt(a**2 + b**2)

def PrintHello():
    print "Hello():"
    return
```

Functions don't always need a return.

```
def NoReturn():
    print "no return!"
```

Note that indentation gets deeper inside other indents.

```
def IsItTheWeekend(dayInitial):
    if dayInitial.upper() in "MWTRF":
        print "NO. Get back to work".upper()
    else:
        print "Yes.  Woo!"
    return
```

13 Dictionaries

It's really easy to make and use dictionaries (aka hashtables) in Python.

```
players = {"rodgers":  "QB", "finley":  "TE"}
prints "Aaron Rodgers is the Green Bay Packers's", players["finley"]
```

You can also initialize dictionaries and add to them or modify them. Additionally, you can use different data types as the keys of dictionaries. `where = {}`

```
where["apple"] = "augustin's desk"
where[5] = "the set of all integers"
where["apple"] = "student's hand"
```

14 Crazier Stuff: List Comprehensions

List comprehensions are a "pythonic" way of doing things. I often find them to be a little hard to read and they can be less efficient. However, they can quickly do the work of a for loop and can sometimes be fun to write!

Suppose I want a list of the first 10 squares. Here's a way to do it with list comprehensions.

```
integers = range(1,11) # creates a list of integers 1-10
squares = [i**2 for i in integers]
```

For every item in integers, we output that item squared into a new list. This is just scratching the surface of list comprehensions.

15 Even More Stuff

This is just a basic sampling. Python has support for object-oriented programming, which you can find more about online or by contacting Chris. Python has support for functional-style programming through use of lambda expressions. Python has a huge array of built-in functions and libraries, which can quickly and easily accomplish many tasks. The BeautifulSoup library is one of many projects built for Python. It makes parsing websites incredibly easy. Try to install Python or play with an interpreter online, write some programs, and have some fun.

16 Caveats

It's easy to mess up your indentation, especially if you open your code in multiple browsers. Whenever you get an indentation error, make sure you're using the same type of indentation everywhere, like tabs versus spaces. To avoid this, you can try to open your code in the same text editor, or use one with

built-in support for Python. Notepad++ is a great editor, and KomodoEdit lets you run code in the window. Emacs and Vi are classic options that still work very well, and gedit is great for Linux users.

Note that the lack of type-checking can often cause you some difficulty. Any number

Methods as data

Last caveat: Python was named after Monty Python, so if you don't like jokes about spam, you may find some of the documentation annoying.

Remember, it's easy to isolate and experiment with code. You don't need to compile anything, you don't need to run a debugger, you can just copy and paste your code into the interpreter and inspect what the resultant values are.