Introduction to Python

Jeffrey R. de Wet

August, 2015

This is an outline of what we will be covering in class on the Python programming language. We will not be looking at a slide deck. Instead, each student will run an iPython notebook which will allow them to try the code snippets for themselves, experiment and change values as they like, and end up with a record of what they did that can be revisited and rerun at any time using the iPython notebook system. There are also repositories of iPython notebooks on the internet that can be downloaded and used as a guide to certain topics or experimented with.

A. Python is a very high level object oriented interpreted programming language
    a. Code is not compiled ahead of time
    b. Read and execute by the interpreter at run time
    c. Python takes care of memory management for you
    d. python.org and documentation
    e. Start an ipython notebook
        i. What you can do with an iPython notebook
    f. What's an object?
        i. Everything you work with in Python is an object
        ii. It allows a close association between data and methods (functions) that operate on that data in some function
        iii. Example with a str (string) object
            1. dir function, console documentation
            2. help function, console documentation
            3. Accessing methods with the . (dot) operator
            4. Two important methods: strip() and split()
    g. Loading a module
        i. Python standard modules
        ii. Large number of specialized data analysis modules added by anaconda
        iii. Example of standard math module
        iv. Accessing module functions with the . (dot) operator
    h. Importing some Python 3 behavior from __future__
        i. Printing and integer division
            1. from __future__ import division, print_function
B. Python built-in data and structure objects
    a. Python variables are references – names for things (objects) we want to be able to use, but a reference has no type or properties, it is just a name

i. Reference name must start with an alphabetic character or _, but may contain numerical digits.

b. = is the assignment operator
    i. Try some simple examples
c. Two types of built-in objects: immutable and mutable
    i. Immutable
        1. int - integer, arbitrarily long
        2. float – double precision, 64 bit floats
            a. standard math operators
                i. +, -, *, /, ** (power)
            b. In-place math operators
                i. +=, -=, *=, /=,
        3. str – character strings, iterable (see for loop), can be sliced
            a. May use ‘ or “, must be paired
            b. “””Triple quoted string
               May extend over multiple lines”””
            c. Triple quoted string can also be used to add help strings to your code that are accessible to the Python console help system
        4. bool – True or False
            a. logical operators
            b. comparison operators
        5. None – special null object
        6. tuple – similar to a list in that it is an ordered collection of objects; once made, its contents can’t be changed; iterable (see for loop), can be sliced
        7. Conversion of types
    ii. Mutable – think of them as containers that can hold other objects
        1. list – Contents accessed by position (order) , iterable (see for loop), can be sliced
        2. set – Unordered collection of unique items that can perform set operations on the contents, iterable (see for loop)
        3. dict – Dictionary, a mapping type that stores key : value pairs
            a. Very fast look-ups
            b. Iterable (see for loop)
        4. len() function – how big is my collection?
    iii. More data structures are available in the collections module

C. Loops, Logic and Branching – program flow control

    a. for loop
    b. while loop
    c. if elif else
    d. logical operators
        i. Boolean operators: and, or, not
        ii. Comparison operators: ==, >, <, >=, <=, !=

D. Opening and reading files
    a. The file object is iterable – when placed in a for loop, it gives one line each pass through the loop
    b. Can also read the entire file as one string
    c. Can get a list of all of the lines in a file
    d. Can move around in a file by byte locations

E. Functions – modularity and grouping of related segments of code
    a. Increase readability
    b. Reusability of code
    c. Easier debugging
    d. Rule of thumb – all of the code in a function should be viewable without scrolling
    e. Note: a collection of functions can be saved in a file and imported into a program just as you import a standard, built-in module

F. Putting it together – writing a simple program
    a. sys.argv – getting an argument into your program, such as a file name, from the command line

Resources

https://www.python.org/ Python home, source and documentation

http://www.continuum.io/ Home of the Anaconda distribution of Python and several innovative data analysis and visualization packages

http://scipy.org/ Home of SciPy, advanced Python scientific computing software and resources

http://pandas.pydata.org/ Home of Pandas, a Python data analysis library