
Scripting for Scientists

BMI 507/607

Fall, 2009–2.0 credit hours

Tuesdays & Fridays, from 14:00-15:00, in BICC 124

Course Description: This course is designed to equip research scientists with the skills necessary for identifying and solving manual labor-intensive tasks with utility scripting. The course is divided into five primary sub-topics, each of which deals with a commonly-encountered roadblock for automating components of one’s workflow. Techniques for identifying problems conducive to automated solutions will be discussed at length, as will methods articulating efficient and generalizable solutions. More specifically, we will focus on three general classes of scripting problems: numeric problems, natural language and text problems, and network and database communication problems. It is not expected that students have formal computer programming experience, as scripting languages will be described on a need-to-know basis for the particular problems at hand. However, class lectures will be supplemented by frequent in-class demonstrations and lab sessions in which students should expect to “get their hands dirty.”

Prerequisites: Familiarity the use of computers in a scientific research setting, and minimal exposure to the Python, R, or any other scripting language.

Class Time/Location: Tuesdays & Fridays, from 14:00-15:00, in BICC 124.

Note: on the 6th, 9th and 30th of October, we will convene at the same times, but in BICC 513.

Instructors: Steven Bedrick & Kyle H. Ambert (mentored by Dr. Aaron Cohen & Dr. Judy Logan)

Required Text: Campbell, Gries, Montojo, and Wilson. “Practical Programming: An Introduction to Computer Science Using Python.” The Pragmatic Bookshelf, 2009.

Supplemental Reading:

- Martelli, Ravenscroft, & Ascher. “Python Cookbook.” O’Reilly Media, Inc, 2005.
- Wilson. “Data Crunching: Solve Everyday Problems Using Java, Python, and More.” The Pragmatic Bookshelf, 2005.
- Additional supplemental reading materials will be provided by the instructors on an as-needed basis.

Evaluation: Students will be evaluated based on class participation, weekly assignments, and a course project: Participation = 30%, Homework = 70%. Homework will be distributed at the end of each week, and due the following meeting. There will be no traditional Final Exam for this course.

Grading Scale: Grading will follow the standard OHSU grading range.

Learning Competencies:

- [1] **Working with Problems Algorithmically:** Be able to identify "scriptable" tasks in your workflow, and able to describe its procedural solution in pseudocode.
- [2] **Working with Language:** Be able to obtain textual data from an input source, perform operations and match patterns, and output the results.
- [3] **Working with Files:** Be able to identify file management situations that scripting may be able to help in. Understand several file formats including CSV and XML, and be able to write a script to read and write both.
- [4] **Working with Numbers:** Be able to recognize situations in which scripting can be used to augment a traditional spreadsheet-and-analysis-software workflow. Be able to load numerical data into a script, perform analyses or calculations on that data, and output the results. Be able to use a scripting language to generate visualizations of numerical data.
- [5] **Working with Networks and Databases:** Be able to obtain data from a network source and use it from within a script. Understand the concepts behind several different types of databases, and identify situations for which each would be appropriate. Be able to write a script to obtain data from a database, perform operations on it, and output the results.

Schedule

Python, & Working with Problems Algorithmically

[1] Python I

READING: Chapters 1 & 2 (Campbell)

[1a] The Motivation for Scientific Scripting, & an Introduction to the Scripter's Toolbox.

[1b] The Python Programming Language.

[2] Python II

READING: Chapters 6, 7, & 13 (Campbell)

[2a] Controlling Flow and Objects in Python.

[2b] Thinking Like a Software Engineer.

Working with Files

[3] All things File.

READING: Chapter 8 (Campbell),

[3a] Scripting with Standard Text Files, & Getting Helpful Feedback From Your Script.

[3b] Using Alternate File Formats, & Scaling File-based Operations.

Working with Numbers

[4] Numerical Scripting with Python and R.

READING: Python Cookbook (Chp. 18)*; Chapters 1, 3, & 6 in *icebreakR* (Robinson, 2008)*

[4a] The Basics of Numerical Scripting with Python.

[4b] Using R without Losing your Mind.

-
- [5] Data visualization scripting.
READING: Murrell, P. *Static Graphics* (In *Handbook of Data Visualization*, 2008)*
[5a] The Principles of Data Visualization.
[5b] Using R to Make the Right Graphic.

Working with Language

- [6] Text processing with Python.
READING: Chapters 3 (Campbell), Python Cookbook (Chapter 1)*, python online *codecs* module documentation
[6a] Operating on strings.
[6b] Effective Text Manipulation.
- [7] Pattern Matching by Scripting.
READING: python online *re* module documentation
[7a] Pattern Matching in Free and Structured Text.
[7b] Information Extraction from Common File Formats (HL7, GenBank, PDB, Medline, etc.).
- [8] Markup: HTML & XML
READING: Python Cookbook (Chapter 12)
[8a] An Introduction to the HTML and XML Markup Languages.
[8b] Parsing Structured Text with Python.

Working with Networks and Databases

- [9] Networks
READING: Python Cookbook (Chapter 13)
[9a] Basic Concepts in Networking.
[9b] Scripting for Network-based Information Extraction.
- [10] Databases
READING: Chapter 15 (Campbell)
[10a] Databases & You.
[10b] Interfacing with Databases Using Python & R.

*Denotes a reading that will be distributed in class or via sakai.