Info 206: Software Prototyping for Data Science and Information Management

Niall C. Keleher

Fall 2017

E-mail: nkeleher@berkeley.edu
Office Hours: Th 1:00-3:00 P.M.
Office: 6A South Hall

Web: nkeleher.com/info206.html
Class Hours: T Th 11:00-12:30 P.M.
Class Room: 210 South Hall

Course Pages:

- http://www.nkeleher.com/info206.html
- Course Videos - https://learn.datascience.berkeley.edu
- Assignments and Grading - https://bcourses.berkeley.edu
- https://github.com/INFO206-Fall2017

Course Description

Info 206 is primarily designed for MIMS students in the School of Information. This course introduces software skills used in building prototype scripts for applications in data science and information management. The course gives an overview of procedural programming, object-oriented programming, and functional programming techniques in the Python scripting language, together with an overview of fundamental data structures, associated algorithms, and asymptotic performance analysis. Students will watch a set of instructional videos covering material and will have three hours of laboratory-style meetings each week.

Course Objectives

At the end of the course, a successful student should be able to:

- work collaboratively using source control
- understand and operationalize core Python objects
- create user-written functions in Python
- work with classes in Python
- clearly explain and execute good practices in software prototyping

Course Structure

The course is divided by in-class and out-of-class activities. You will be expected to watch instructional videos as well as read the relevant in advance of each in-class session. Classroom sessions are intended to review key information covered in the videos and readings, introduce new topics (e.g. algorithms and applications), and provide teams with time to work on their group projects. Each session, we will have approximately 10 minutes for a quiz, 40 minutes of review plus applications, and 30 minutes for group project collaboration.

Course Textbooks

- Mark Lutz. *Learning Python: Powerful Object-Oriented Programming*, 5th Edition, O'Reilly Media, Inc., 2013.
  - An online version is freely available at UCB through Safari publications: http://proquest.safaribooksonline.com/9780596513986
  - Code from the book is available through the book website: http://learning-python.com/about-lp5e.html
- Kent D. Lee and Steve Hubbard. *Data structures and algorithms with Python*, Springer, 2015.
  - A pdf and ebook version is freely available at UCB through Springer: https://link.springer.com/book/10.1007%2F978-3-319-13072-9
  - Code from the book is available through: http://knuth.luther.edu/~leekent/CS2Plus
- Scott Chacon and Ben Straub. *Pro Git*, 2nd Edition, Apress, 2014.
  - An online version is freely available at: https://git-scm.com/book/en/v2.

Course Project

Teams

You will be able to select one team member to work with on the course-long group project. Each pairing will be randomly assigned with one other pairing to form the complete project team.

Objectives

Throughout the duration of the course teams are expected to work together towards completion of a software development project. Teams are responsible for:

- identifying the problem that they aim to address
- working collaboratively and sharing code through github
- building and testing software
- presenting project updates twice to the class
- display final project results on the last day of the course (17 October)

Important Dates

- Project Status Update #1 - Problem statement (12 September)
- Project Status Update #2 - Codebase outline (05 October)
- Final Project Presentations (17 October)

Course Policy

Grading Policy

- **10%** of your grade will be determined by your **attendance** and participation in class. Generally, ask questions and answer them.
- **20%** of your grade will be determined by **quizzes**, which will occur at the beginning of every in-class meeting.
- **10%** of your grade will be determined by **individual assignments**.
- **10%** of your grade will be determined by **group and team assignments**.
- **50%** of your grade will be determined by a **group project** that culminates in a presentation at the end of the course.

Attendance Policy

Students are expected to attend all class meetings. If you are unable to attend any meetings, please inform me in advance of the class meeting.

E-mail, bcourses, and Github Policy

Class discussions are encouraged in-person and online. Please be sure to keep any sensitive course materials stored in private github repositories. For any questions or comments that are relevant to the wider group, you are encouraged to submit them through bcourses. For questions that are specific to you personally, you are encouraged to write me. Remember that we are all colleagues and part of a wider community thus respect for others is essential in all communications.

Quiz and Assignment Policy

There will be no makeup quizzes. Quizzes are used to monitor attendance and cannot be made up, thus by missing a class you are deducted for lack of attendance and missing a quiz.

Assignments should be submitted by the due date indicated. In general, assignments are completed during class meetings - thus assignments are due at 12:30 PM. If you need additional time to complete the assignment, you can submit the assignment within 24 hours of the due date (e.g. by 12:30 PM on the day following a class meeting) with a 25% penalty. Assignments submitted between 24 and 48 hours of the due date will be docked a 50% penalty. Assignments submitted more than 48 hours after the due date will be docked a 75% penalty.

Academic Integrity

UC Berkeley Code of Student Conduct: http://sa.berkeley.edu/code-of-conduct

---

Course Calendar

Students must read the following before Tuesday's class session. Important: class readings are subject to change, contingent on mitigating circumstances and the progress we make as a class. Students are encouraged to attend lectures and check the course website for updates.

Videos can be found on ISVC (http://learn.datascience.berkeley.edu).

Required readings are denoted by (***).

Meeting 1 - 08/24: Introductions, Course overview, Command Line, Bash, and Git

Videos

NA

Readings

- (***) Command Line tutorial https://learnpythonthehardway.org/python3/appendixa.html
- (***) Chacon & Straub – Chapter 1 & 2
- (https://git-scm.com/book/en/v2)
- (***) Lutz Chapter 1: A Python Q&A Session
- (http://cdn.oreillystatic.com/oreilly/booksamplers/9781449355739_sampler.pdf)

Additional Resources:

- http://rogerdudler.github.io/git-guide/
- https://guides.github.com
- http://ohshitgit.com/

Meeting 2 - 08/29: Git Review, Teams, Python overview, Installing Python

Videos 1. Introducing Python (5 mins) 2. Programming Language Characteristics (10 mins) 3. Starting Python (4 mins) [optional] 4. The Jupyter Notebook Page (14 mins) [optional]

Readings

- (***) Lutz Chapter 2: How Python Runs Programs
- (***) Lutz Chapter 3: How You Run Programs

Meeting 3 - 08/31: Types, Polymorphism, Numerics, and Strings

Videos 1. Objects (4 mins) 2. Objects and Types (11 mins) 3. Representing Numbers (3 mins) 4. Numbers (7 mins) 5. Variables, Part 1 (9 mins) 6. Variables, Part 2 (3 mins) 7. Strings (19 mins)

Readings

- (***) Lutz Chapter 5: Numeric Types
- (***) Lutz Chapter 7: String Fundamentals

Meeting 4 - 09/05: Sequences, Lists, and Dictionaries

Videos 1. Sequences (11 mins) 2. Lists (12 mins) 3. Lists and Mutability (4 mins) 4. Mutability, Part 1 (8 mins) 5. Mutability, Part 2 (8 mins) 6. Tuples (7 mins) 7. Ranges (5 mins) 8. Dictionaries (21 mins) 9. Encoding Text (5 mins) 10. Unicode Strings (10 mins) 11. Encoding (16 mins)

Readings

- (***) Lutz Chapter 8: Lists and Dictionaries
- (***) Lutz Chapter 9: Tuples, Files, and Everything Else

Meeting 5 - 09/07: Statements and Syntax, Part 1

Videos N/A

Readings

- (***) Lutz Chapter 10: Introducing Python Statements
- (***) Lutz Chapter 11: Assignment, Expressions, and Prints

Meeting 6 - 09/12: Statements and Syntax, Part 2

Videos 1. Control (19 mins) 2. while Loops (19 mins) 3. for Loops (12 mins) 4. Fancy Loop Exits (23 mins) 5. Formatting (15 mins)

Readings

- (***) Lutz Chapter 12: If Tests and Syntax Rules
- (***) Lutz Chapter 13: while and for Loops
- (***) Lutz Chapter 14: Iterations and Comprehensions

Exercises:

- File input and output
- Exhaustive Search
- Bisection Search

Meeting 7 - 09/14: Functions

Videos 1. Functions (10 mins) 2. Calling Functions (3 mins) 3. Nesting Functions (6 mins) 4. Functions and the Call Stack (2 mins) 5. The Stack Trace (3 mins) 6. Advantages of Functions (1 min) 7. Namespaces (6 mins) 8. Accessing Global Variables (12 mins) 9. Using Parameters (13 mins) 10. Functions are Objects (6 mins)

Readings

- (***) Lutz Chapter 16: Function Basics
- Lutz Chapter 17: Scope

- Lutz Chapter 18: Arguments

Exercises * Newton-Raphson Method * Heron's Method

Meeting 8 - 09/19: Recursion and List Comprehension

Videos 1. Recursions Basics (10 mins) 2. Traversing Nested Dictionaries with Recursion (13 mins) 3. Comprehensions (10 mins)

Readings

- (***) Lutz Chapter 20: Comprehensions and Generations

Exercises * Palindromes

Meeting 9 - 09/21: Modules, Part 1

Videos 1. Modules and Packages (4 mins) 2. Modules and the Import Statement (10 mins) 3. Packages (13 mins) 4. [optional] The Python Standard Library (25 mins) 5. Arrays (14 mins)

Exercises * Numpy

Readings

- (***) Lutz Chapter 22: Modules: The Big Picture
- (***) Lutz Chapter 23: Module Coding Basics

Meeting 10 - 09/26: Modules, Part 2

Videos N/A

Readings

- (***) Lutz Chapter 24: Module Packages

Exercises * Regular Expressions

Meeting 11 - 09/28: Object-Oriented Programming & Classes, Part 1

Videos 1. Object-Oriented Programming (5 mins) 2. Classes Introduction (3 mins) 3. Classes and Attributes (12 mins) 4. Using the Class Definition (12 mins) 5. Binding Methods (2 mins) 6. Initializing a Class (15 mins) 7. Controlling Access to Attributes (17 mins) 8. Class Odds and Ends (18 mins)

Readings

- (***) Lutz Chapter 26: OOP: The Big Picture
- (***) Lutz Chapter 27: Class Coding Basics

Meeting 12 - 10/03: Object-Oriented Programming & Classes, Part 2

Videos 1. Class Inheritance (4 mins) 2. Inheritance (21 mins) 3. More Class Inheritance (17 mins) 4. Using Polymorphism (15 mins)

Readings

- (***) Lutz Chapter 28: A More Realistic Example
- (***) Lutz Chapter 29: Class Coding Details

Exercises * Mortgage calculation

Meeting 13 - 10/05: Complexity

Videos 1. Measuring Execution Time (8 mins) 2. Big O Notation (10 mins) 3. Common Growth Functions 1 (6 mins) 4. Common Growth Functions 2 (8 mins) 5. Insertion Sort (6 mins) 6. Merge Sort (7 mins) 7. A Complexity Bound for Sorting (6 mins) 8. NP-Hard Problems: Conversation with Benjamin Johnson (17 mins) [optional]

Readings

- (***) Lee and Hubbard - Chapter 2: Computational Complexity

Exercises * Merge Sort

Meeting 14 - 10/10: Exceptions & Testing, Part 1

Videos 1. Exceptions (9 mins) 2. Test-Driven Development (7 mins) 3. Test-Drive Example (17 mins)

Readings * (***) Lutz Chapter 33: Exception Basics

Exercises * Hash Tables

Meeting 15 - 10/12: Exceptions & Testing, Part 2

Videos N/A

Readings * (***) Lutz Chapter 34: Exception Coding Details

Exercises * Coin-flipping Simulation

Meeting 16 - 10/17: Presentations