# CS107/AC207

## SYSTEMS DEVELOPMENT FOR COMPUTATIONAL SCIENCE

## CLASS SYLLABUS

2022-07-29

Fabian Wermelinger

Harvard University

## *Objective:*

Writing correct code and developing software are a powerful skill that is highly sought in the professional market. Writing correct code is not a natural gift. It involves a set of tools, design principles and automation, just like a carpenter needs tools and blueprints to build a house. The primary goal of this course is to teach you how to develop effective software for scientific applications. In order to achieve this goal, there are several non-negotiable topics that must be included in the course. The course will be concerned with two primary thrusts: *System and Software Engineering* and *Language*. Moreover, the class aims to provide you with a suite of modern software development techniques and workflows widely used in academia and industry.

## *Learning Objectives:*

After successful completion of this course, the student will be able to:

- Use Python, including its advanced features to write scientific programs.
- Have a basic idea how the Python interpreter works.
- Understand the Python language execution model and how they impact the code you write.
- Write programs with good software engineering practices. These practices include: working on remote machines, version control, continuous integration, documentation and testing.
- Utilize data management techniques to store data, starting from a good understanding of data structures to databases.
- Be able to contribute software in academic or industrial environments.

## *Prerequisites:*

You should have some basic familiarity with programming (functions, variables, constants, differences between integer and floating point, etc.) at the level of CS50. Some comfort with a tool to edit text files is beneficial. Any text editor or IDE will suit this purpose. The student should have passed a basic calculus class. The project will require the use of the chain rule. The lectures will review the necessary fundamentals required to succeed with the class project. Besides this, you should have interest or investment in scientific computing.

## *Textbooks:*

The class does not follow a specific textbook. The following textbooks are suitable for additional reference:

- *"Fluent Python: Clear, Concise, and Effective Programming"*,
  L. Ramalho, O'Reilly 2015
- *"The Practice of Programming"*,
  B. Kernighan and R. Pike, Addison-Wesley 1999
- *"Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems"*,
  M. Kleppmann, O'Reilly 2017

## *Course Format:*

The course contains six main components:

1. ***Lectures:*** Deliver the main content of the class. Attendance is mandatory.
2. ***Quizzes:*** Intended to assess the learning progress.
3. ***Pair-programming:*** Offer practice on topics addressed in class and help support homework assignments.
4. ***Homeworks:*** Homework assignments deepen the lecture material and include coding exercises.
5. ***Projects:*** The class is accompanied by a project to practice the methods and tools learned in class on a real application.

## *Grading:*

| | |
|---|---|
| ***Homework:*** | 35% |
| ***Project:*** | 35% |
| ***Quizzes:*** | 15% |
| ***Pair-programming:*** | 10% |
| ***Communal Contributions:***<br>Via the class communication platforms. | 5% |

The class does not have standard midterm or final exams. The project work involves presentations.

## *Collaboration and Class Policies:*

You are welcome to discuss the course material and homework with others in order to better understand it, but the work you turn in must be your own (with exception of the project where collaborative work is permitted). Any work that is not your own, without properly citing the original author(s), is considered plagiarism. Failure to follow the academic integrity and dishonesty guidelines outlined in the Harvard Student Handbook will have an adverse effect on your final grade. This includes the removal of copyright notices in code. You may not submit the same or similar work to this course that you have submitted or will submit to another without permission.

The course related material will be distributed through `git` repositories on the https://code.harvard.edu/ platform. Membership in the CS107/AC207 organization on that platform is required to access the material.

| Wk | Tuesday | Thursday | Labs | Events |
|---|---|---|---|---|
| 1(35) | | ***Lecture 1:*** *2022-09-01*<br>• Class introduction/organization<br>• History of Bell Labs, Unix and Linux<br>• Command line introduction | ***Sign-up:***<br>Select one of the offered pair-programming lab session days according to your schedule<br>***PP1:*** ***(2022-09-02)***<br>Setup private class repository, `tmux` | ***Note:***<br>Handouts are typeset in green and deadlines in red. All deadlines are due 11:59 pm.<br>1. HW1 release (2022-09-01) |
| 2(36) | ***Lecture 2:*** *2022-09-06*<br>• Manual pages and help for Linux commands<br>• Unix philosophy<br>• Regular expressions and `grep`<br>• File attributes and permissions<br>• Short and unbiased journey into text editors | ***Lecture 3:*** *2022-09-08*<br>• Shell customization<br>• I/O redirection<br>• Process management<br>• Environment variables and shell scripting | ***PP2:*** ***(2022-09-09)***<br>Git workflow and `bash` scripting | 1. Select your preferred lab sections in my.harvard (2022-09-06) |
| 3(37) | ***Lecture 4:*** *2022-09-13*<br>• Introduction to version control systems (VCS)<br>• Centralized and distributed VCS<br>• Design and inner workings of Git | ***Lecture 5:*** *2022-09-15*<br>• Git blobs, trees and commits<br>• Git status of working tree and index<br>• Git remote repositories<br>• Git branches | ***PP3:*** ***(2022-09-16)***<br>Git branches and merge conflicts | 1. HW2 release (2022-09-13)<br>2. HW1 due (2022-09-13)<br>3. PP1 due (2022-09-16) |
| 4(38) | ***Lecture 6:*** *2022-09-20*<br>• Introduction to Python<br>• Nested environments and closures<br>• Decorators | ***Lecture 7:*** *2022-09-22*<br>• Object Oriented Programming (OOP)<br>• Classes in Python<br>• Inheritance and polymorphism<br>• ***Quiz 1*** | ***PP4:*** ***(2022-09-23)***<br>Fully connected neural networks and Python closures | 1. Project M1A due (2022-09-22)<br>2. PP2 due (2022-09-23)<br>3. C/C++ primer class 09/19 - 09/23 |
| 5(39) | ***Lecture 8:*** *2022-09-27*<br>• Duck typing<br>• The Python data model<br>• Special methods (dunder methods)<br>• *Aside:* software licenses and open source | ***Lecture 9:*** *2022-09-29*<br>• Python class methods, static methods, instance methods<br>• Python class attributes and instance attributes<br>• Python modules<br>• Python packages and PyPI | ***PP5:*** ***(2022-09-30)***<br>Python classes and dunder methods | 1. HW3 release (2022-09-27)<br>2. HW2 due (2022-09-27)<br>3. PP3 due (2022-09-30)<br>4. C/C++ primer class 09/26 - 09/30 |

| Wk | Tuesday | Thursday | Labs | Events |
|---|---|---|---|---|
| 6(40) | ***Lecture 10:*** *2022-10-04*<br>- Preliminary automatic differentiation (AD)<br>- Derivatives and the Jacobian<br>- Newton's method<br>- Numerical approximation of derivatives (finite-difference method) | ***Lecture 11:*** *2022-10-06*<br>- Review of chain rule<br>- Evaluation trace of a function and computational graph<br>- Forward mode AD<br>- Forward mode in higher dimensions | ***PP6:*** ***(2022-10-07)***<br>Forward mode automatic differentiation | 1. Project M1B due (2022-10-04)<br>2. PP4 due (2022-10-07) |
| 7(41) | ***Lecture 12:*** *2022-10-11*<br>- Dual numbers and complex numbers<br>- Implementation of forward mode AD: operator overloading | ***Lecture 13:*** *2022-10-13*<br>- Reverse mode AD<br>- Some examples AD applications and extensions<br>- ***Quiz 2*** | ***PP7:*** ***(2022-10-14)***<br>Forward mode AD, Jacobian, seed vector | 1. HW4 release (2022-10-11)<br>2. HW3 due (2022-10-11)<br>3. PP5 due (2022-10-14) |
| 8(42) | ***Lecture 14:*** *2022-10-18*<br>- Continuous integration (CI)<br>- Testing Python code (`pytest` and `unittest`)<br>- Code coverage<br>- Documenting code (docstrings, shpinx, readthedocs) | ***Lecture 15:*** *2022-10-20*<br>- Python virtual environments<br>- Introduction to docker containers and `Dockerfiles` | ***PP8:*** ***(2022-10-21)***<br>Python virtual environments and deploying Python packages | 1. Project M1 due (2022-10-20)<br>2. PP6 due (2022-10-21) |
| 9(43) | ***Lecture 16:*** *2022-10-25*<br>- Building your own docker containers<br>- Integration of custom containers in CI workflows | ***Lecture 17:*** *2022-10-27*<br>- Abstract data types<br>- Linked lists<br>- Iterators | ***PP9:*** ***(2022-10-28)***<br>Binary trees, binary search trees | 1. HW5 release (2022-10-25)<br>2. HW4 due (2022-10-25)<br>3. PP7 due (2022-10-28) |
| 10(44) | ***Lecture 18:*** *2022-11-01*<br>- Trees and binary trees<br>- Binary search trees (BST)<br>- Priority queues and heaps | ***Lecture 19:*** *2022-11-03*<br>- Python generators<br>- Python coroutines<br>- Introduction to Python internals | ***PP10:*** ***(2022-11-04)***<br>Python generators | 1. Project M2A due (2022-11-01)<br>2. PP8 due (2022-11-04) |
| 11(45) | ***Lecture 20:*** *2022-11-08*<br>- Code objects and Python bytecode<br>- The Python interpreter and the evaluation loop<br>- Memory allocation in Python<br>- Performance of pure Python lists and NumPy arrays | ***Lecture 21:*** *2022-11-10*<br>- Data models and databases<br>- Structured query language (SQL)<br>- SQLite in Python<br>- ***Quiz 3*** | ***PP11:*** ***(2022-11-11)***<br>Databases | 1. HW6 release (2022-11-08)<br>2. HW5 due (2022-11-08)<br>3. Project M2B due (2022-11-10)<br>4. PP9 due (2022-11-11) |

| Wk | Tuesday | Thursday | Labs | Events |
|---|---|---|---|---|
| 12(46) | ***Lecture 22:*** *2022-11-15*<br>• Databases<br>• In-class exercise with SQL and Python SQLite | ***Lecture 23:*** *2022-11-17*<br>• Databases<br>• In-class exercise with SQL and Python SQLite<br>• Table joins<br>• Pandas | ***PP12:*** ***(2022-11-18)***<br>Finish in-class database exercises | 1. Project M2 due (2022-11-17)<br>2. PP10 due (2022-11-18) |
| 13(47) | ***Lecture 24:*** *2022-11-22*<br>• Debugging in Python<br>• Generating profiles for performance analysis<br>• Bytecode instructions and performance | ***Thanksgiving break:*** *2022-11-24* | | 1. HW7 release (2022-11-22)<br>2. HW6 due (2022-11-22)<br>3. PP11 due (2022-11-25) |
| 14(48) | ***Lecture 25:*** *2022-11-29*<br>• Project work<br>• ***Quiz 4*** | ***Lecture 26:*** *2022-12-01*<br>• Project work | | 1. PP12 due (2022-12-02)<br>2. HW7 due (2022-12-04) |
| 15(49) | ***Reading period:*** *2022-12-06* | ***Exam period:*** *2022-12-08* | | 1. Project final milestone due (2022-12-10) |
| 16(50) | ***Exam period:*** *2022-12-13* | ***Exam period:*** *2022-12-15* | | |