Instructor: Walker White

Comparison of CS1110 and CS1112: Both introduce computing concepts. The courses emphasize techniques of problem analysis and the development of algorithms and programs.

CS 1110 (White): Computing using Python

CS 1112 (Fan): Computing using Matlab

CS 1110 and 1112 do not require previous programming experience. CS1112 requires 1 semester of calculus. For more information, see www.cs.cornell.edu/ugrad/FirstCSCourse/index.htm.

Course webpage (see top of page). Look at it several times a week: It is a major communication medium for the course. If you miss a handout, download it from the website.

Piazza www.piazza.com/cornell/fall2015/cs1110. Piazza is highly catered to getting you help fast and efficiently from classmates, the TAs, and instructors. We encourage you to post your questions on Piazza rather than use email to instructors.

CMS cms.csuglab.cornell.edu. This is our course management system for handling assignments, grades, etc.

Course material (see course web page for more info)

(1) *Think Python: How to Think Like a Computer Scientist to Programming Using Java* by Allen Downey. Available as a free PDF from

http://www.greenteapress.com/thinkpython/

Print copies available at the Cornell bookstore.

(2) Active Python, a free Python environment.

(3) iClicker. We will be using iClickers, and every student is expected to bring their iClicker to every class. You may either buy iClicker at the bookstore, or download the iClicker App for your mobile device.

Homework will consist of 5 computer projects, which you can do with one partner, and some written assignments. Computer projects will be submitted electronically using our CMS (see above).

We do our best to make the programming assignments interesting! Graphics, manipulating files, games, string manipulation — we will show you what you can do with Python. We might even see how to program for mobile devices (Android, iPhone).

Tests: Two prelims and a final. To find out when they are, visit the course home page, scroll to the bottom of the page, and click the link. We give make-ups only in special circumstances.

Surveys: We are constantly improving this course based on student feedback. We will post surveys periodically on CMS and announce them to class. You are expected to complete them as part of your participation grade for the course.

Recitations-Sections-labs: Sections/labs are either in the ACCEL laboratory in Carpenter library, or Phillips 318, depending on your section. Each lab will ask you to do something on the computer, either to reinforce what is being taught in lecture or to introduce new topics to you. At the end, show what you did to the lab instructor to get credit (if you can't finish in time, show it to the instructor the next week).

Attendance will be taken. Miss three of them without valid excuses (given to us ahead of time) and your letter grade *may* decrease (e.g. B to B–).

Syllabus: A lecture-by-lecture topic list is on the course website. See "Lectures".

Academic integrity. This course is not a case of student against faculty. It is not about grades. It is about all of us working together to teach you as much about programming as possible in as efficient a manner as possible. The staff knows that you have other courses and strives to make your workload in this course reasonable. We are ready to help you in any way we can. On your side, we expect you to be honest. We expect you to come to us early if problems arise, so that we can solve them together. Do not wait four or five weeks, because then you may be too far behind.

Read the academic integrity statement on the course website and complete the Integrity Quiz on CMS.

Fix your PCs. To reduce chances of errors later, fix your PCs so that extensions (e.g. .py and .doc) always appear. To do this: Open an explorer window. Click menu item *Tools / Folder Options*. Click the view tab. Uncheck the box "Hide extensions for known file types". You may have to do something different depending on what Windows OS you use.

Practice, practice, practice. Learning to program well takes practice. The more time you spend on the computer, trying things out, getting acquainted with programming features and techniques, the better you will do in this course and later. Therefore, practice, practice, practice.

It is better to practice every day or every other day for ½ hour than it is to do nothing for a week or two and then spend 4 hours. Steady progress is best. In many ways, it is like learning a spoken language; if you stop, you will start to forget things.