CS 2: Introduction to Programming Methods — Winter 2024

Course Syllabus

Information At-A-Glance

Instructor	
Name:	**Adam Blank**
E-mail:	blank@caltech.edu
Office:	ANB 115

Course Website
https://debuggi.ng

Lecture
BCK Institute Auditorium on MWF 02:00 PM – 02:55 PM

Course Overview

This course is intended as a continuation of CS 1. We focus on abstraction in programming–both in design of programs and data storage. We discuss and implement fundamental data structures and algorithms via a series of labs and projects. We will grade on correctness and efficiency of our programs. This course uses Java as an implementation language, but we do not expect any prior Java programming experience.

Assessments

In this course, there are three types of assessments: labs, diagnostics, and projects. The course will have approximately **nine** labs, **seven** programming projects, and **thirty** diagnostic opportunities. Additionally, in the last week, you will be allowed to redo a **single** lab or project (but not both) for full credit.

Labs

During labs, we will practice pair programming, testing, debugging, and implementing skills in a supervised setting. Labs are intended to be a bridge between lecture and the projects; as such, we expect the labs to significantly decrease the amount of time necessary to complete the projects. If you do not finish the lab during class, you and your partner may *collaboratively* finish the lab outside of class. Final submissions for lab assignments will be due at *11:30 PM on Thursdays*. Due to all of these features of lab, *to receive any credit, you must attend your assigned lab section*. Students that are more than *three minutes late* will receive no credit. In exceptional circumstances, students might be allowed to get credit for the lab without attending at the discretion of the instructor.

Diagnostics

In the last five minutes of most lectures, we will hand out a "diagnostic". Each diagnostic will have a single question on the material from that day's lecture. During the remaining time in lecture, you will be able to make a first attempt at writing a solution to the best of your ability.

Before you leave, we will ask you to turn in your diagnostic attempt on paper to a member of course staff. **If you do not attend the full lecture, we reserve the right to not give you credit for having completed the diagnostic.**

While we will "grade" the diagnostic as you submit it, you will be allowed a second attempt (for full credit)–and you will only need to pass **ten** of them, in total. (This works out to approximately $\frac{1}{3}$ of the diagnostics.)

Each of the two attempts have slightly different grading schemes and rules which are outlined here:

- Diagnostics will be graded on a $-$, ✓, $+$ grade scheme.
- For the first attempt from lecture, we will only grade the diagnostics as either a $-$ or a ✓. If you earn a ✓, we will consider you to have "passed" the diagnostic. If you earn a $-$ and you still want credit for the

diagnostic, you will be allowed a second attempt (outside of lecture).

- For the second attempt, you can collaborate with students and ask course staff for help. Additionally, we will try our best to provide a list of some common errors we saw while grading the first attempts. Your second attempt will be graded on a slightly higher standard (since you are untimed and can collaborate) and can only receive a $-$ or a $+$; on this scale, a $+$ will be considered a "pass". The second attempt will be due at the same time as the next project on the following Monday.

Note that if you do not make a first attempt at a diagnostic in lecture, you will not be allowed to turn in a second attempt for that diagnostic.

Projects

The projects are the heart and soul of this course. We prefer the term *project* to *set* because all the individual parts of the assignment will come together to create a single finished product that we hope you will be proud of. The projects will be exclusively programming assignments. They will be auto-graded via sets of unit and integration tests we have designed to catch most of the common mistakes. Because this is an introductory course, we will provide you with the source code of our tests for all projects. In addition to passing the tests, we have several static analysis tools which test basic stylistic components of your code. To receive any credit, you must be passing these tools. Projects will be due at *11:30 PM on Mondays*.

Course Standards

Ultimately, when a student passes a course, the course staff is providing a certification that (a) that student is ready to move on to any courses that have this course as a prerequisite, and (b) that student has *at least* a satisfactory understanding of most of the course material.

As such, in this course, we use a form of "alternative grading" called "standards-based grading" which, rather than using *points*, uses *standards* to evaluate student understanding.

Grading "Bundles"

Each lab and project will "bundle" together many tests into "grade levels" spanning some combination of "A+", "A", "B", "C", and "D". These bundles *roughly correspond to the following:*

- "A+": significantly above and beyond
- "A": very solid understanding of topic with no large holes
- "B": solid understanding of topic with at most one major hole
- "C": satisfactory understanding of topic, potentially with several large holes
- "D": cursory understanding of some of the core ideas of the topic, potentially with some major misconceptions

Note that these bundles are not independent: to get credit for a bundle, you must also achieve all the bundles below it. (For example, to earn a "B" bundle, you must pass all of the "D", "C", and "B" bundles that exist.)

How to Pass CS 2

Our goal is for you to meet our course objectives. To encode this into a *standards-based grading policy*, we have created standards-based requirements on each part of the course which you must meet to pass. **Since there are quite a lot of details here, we've written a "CS 2 Grades App" which will detail the individual project and lab scores you have earned so far.**
The standards to pass the course are the following:

- For labs, you must show at least a solid understanding ("B" level) for **six** labs, at least a satisfactory understanding ("C" level) for **two** labs, and at least a cursory understanding ("D" level) for **one** lab. Note that all topics listed in this section must be disjoint.
- For projects, you must satisfy two requirements:
  - You must show **at least** a satisfactory understanding ("C" level) on P3, P5, and P7. That is, getting below a C on any of these will cause you to fail the course.
  - You must show at least a very solid understanding ("A" level) for **one** project, at least a solid understanding ("B" level) for **three** projects, at least a satisfactory understanding ("C" level) for **two** projects, and at least a cursory understanding ("D" level) for **one** project. Again, note that these projects all must be disjoint and can only include P1-P7.
- For diagnostics, you must achieve a ✓or a + on at least **ten** distinct diagnostics (either on the first or second attempt).

If you find yourself in a sticky situation, please please come talk to Professor Blank before dropping the course. There's often a way to work things out!

Late Policy

Throughout the term, we will keep track of a "count" for you. The "count" represents how much grace you'll have in turning in projects and labs late. Diagnostics must always be turned in on time or not at all. Your grace count starts at zero, but turning in assessments early or late can change it in the following ways:

- For any *project* (but not lab) that is last submitted early (with *at least up to the B tests passing*), your count increases by `Math.floor(`$\frac{\text{hours early}}{24}$`)`.
- For any *project* or *lab* that is turned in on time, your count remains unchanged.
- For any *project* or *lab* that is submitted after the due date, your count decreases by `Math.ceil(`$\frac{\text{hours late}}{24}$`)`.
- For a partner project or lab, both students increment or decrement their counts together.
- If your count reaches seven, you can spend all your grace to receive a Hopper Token from Prof. Blank.

Your grace count is subject to the following additional rules:

- Under **normal circumstances**, the count cannot go below 0. This means that if turning in an assignment would make your count go negative, you cannot turn it in.
- If you have **extenuating circumstances** or CASS accommodations that might entitle you to go below 0 and still turn in an assignment, you must request the extension **at least 24 hours before the normal due date** by e-mailing Prof. Blank. TAs (including the Head TAs) are not allowed to grant extensions. Note that extensions **do** affect your count, as you are turning in an assignment late.
- Under **absolutely no circumstances** can the count go lower than -5 or larger than 7 during the course. This includes all circumstances, even illness and family emergency.

  Note that in extreme cases such as deaths in the family, you will still be limited to the restrictions above if you choose to complete the course normally. However, we find in such circumstances, it's often the case that I grades are appropriate instead, which we are happy to work with you on. To potentially receive an I grade, you should contact the Deans's Office!

Note that, regardless of your grace count, office hours help will **only** be available for the **current** project/lab.

Collaboration & Academic Integrity

See our "collaboration table" on the website. We reserve the right to modify or clarify this policy as needed. Notably, you may not, under any circumstances, look at another student's/group's code or write pseudocode with another group.