CS 3: Introduction to Software Design — Spring 2019

Course Syllabus

Information At-A-Glance

Instructor	
Name:	**Adam Blank**
E-mail:	blank@caltech.edu
Office:	ANB 115
Office Hours:	Mon: 4:00pm – 5:00pm Wed: 3:00pm – 5:00pm Thu: 7:00pm – 9:00pm Or by private meeting.

Course Website
http://courses.cms.caltech.edu/cs3
Visit early. Visit often.

Lecture
ANB 105 on M
03:00 PM – 03:55 PM

Course Overview

This course covers large program design in the C programming language. Prerequisite: CS 2 or equivalent. CS 3 is a practical introduction to designing large programs in a low level language. Heavy emphasis is placed on documentation, testing, and software architecture. Students will work in teams in two 5-week long projects. In the first half of the course, teams will focus on testing and extensibility. In the second half of the course, teams will use POSIX APIs, as well as their own code from the first five weeks, to develop a large software deliverable. Software engineering topics covered include code reviews, testing and testability, code readability, API design, refactoring, and documentation.

Grading Policy

Your grade will be composed of several categories with varying weights: 4 labs (5% each), project0 (5%), a physics engine (30%), a game design document and proposal (5%), a game or simulation (30%), and participation (10%).

- Labs will be graded on a completion basis. We will check that you've completed lab n at the beginning of lab $n+1$. If you have completed it, you'll get full credit; otherwise, you will get no credit.
- project0, the physics engine, and the game will be graded based on three dimensions: functionality, testing, and design. The percentages of these categories will vary drastically for each project, but we will tell you explicitly what they are. You will only receive testing and design points for functionality that is implemented correctly. In other words, we will multiply the design and testing components by the fraction of functionality points you received.

  Functionality points will be assigned based on gitlab tests and demos during code review. Testing and design points will be assigned on a $+/\checkmark/-/0$ scale after code reviews. To convert these grades to points we will use the following scale: $+ \mapsto 100$, $\checkmark \mapsto 80$, $- \mapsto 60$, $0 \mapsto 0$.

  We expect you to improve your testing and design scores on most assignments by implementing the feedback provided by your code reviewer. This will allow your grade in each section to move one rank up on the scale.
- Your game design document and proposal will be graded on a rubric we will provide in the assignment write-up.
- Participation points are somewhat less defined. Anything that indicates that you're "involved" in the course and trying to learn will get you points for this section. Examples include (but are not limited to) completing diagnostics in lecture, going to office hours, helping other students as a learning assistant, making appointments with Adam, and providing useful, constructive feedback for the course. **Additionally, your relative contribution to the team projects will be factored into your participation score.** We

expect you to be on time to the code reviews and reserve the right to factor this into your participation grade. Our goal for the participation score is to make your grade reflect the *effort you are putting in*.

Late Policy

You will receive *two* "late tokens", each of which will allow you to extend deadline by a weekend. You may use these tokens at your discretion subject to the following rules:

- You may not use more than one token per week
- Each token gets you an entire extra weekend, but you must set up a Code Review on the following Monday with Adam
- If you are in a group, everyone in that group must use a late token

Getting Help

Please don't be afraid to ask for help if you don't understand something. Adam holds *at least four* office hours a week, and he gets lonely and bored if you don't show up! He also shows up early to lecture and is happy to answer any questions you might have before or after lecture.

At office hours, you can ask for clarification on a lecture (or for a *repetition* of the lecture!). You can ask for help with a frustrating part of the homework. You can even show up just to tell us you're frustrated and vent.

Here's some first steps on how to get help:

- Come to office hours
- Ask someone on course staff questions before/after lecture, before/after lab, etc.
- Post on Piazza asking a question

Collaboration & Academic Integrity

You may not look at other students' code outside of your group. No one except TAs may look at your code outside of your group. You may use the Internet, but you may NOT look at online code repositories that were not okayed by course staff. You may discuss design decisions outside of your group. You may NOT discuss code or pseudocode outside of your group. You may discuss the way C works with anyone. You may NOT discuss the way C works in the context of the project outside of your group. We reserve the right to modify or clarify this policy as needed.