

# EE 200: Accelerated programming for graduate students (a.k.a. “Machine-centric approach prog.”)

Fall 2019  
M/W 4:30-5:45pm, Halligan 108

Welcome to EE 200! This course is a fast-paced introduction to programming in C and C++, as well as data structures and algorithms analysis. The aim is for you to become a competent programmer, not merely for you to learn C++ syntax. As a result, we will spend parts of the course discussing more general themes in programming languages and software development, and we will use a set of development tools widely employed in embedded systems contexts.

After successfully completing this course, you will be able to:

- Read, write, and debug code written in C and C++, using all of the core features of both languages.
- Describe the central ideas of object-oriented design, and write software using object-oriented principles.
- Describe the interface for abstract data structures such as vectors, trees, stacks, queues, and graphs, and give example implementations for these.
- Compare algorithms using Big-O analysis.
- Use standard Unix/Linux development software tools, including SSH, Vim or Emacs, GDB, Git, Make and Valgrind.

## Communication:

Steven Bell    [sbell@ece.tufts.edu](mailto:sbell@ece.tufts.edu)

Halligan Hall 202C

Office hours:

- Mondays 10:30-12:30pm
- Tuesdays 3:00-5:00pm
- Thursdays 3:00-4:00pm (walking office hours – see my website for details)
- I’m also available other times by appointment, or just drop by my office if my door is open.

To minimize distraction, I generally only check email a few times a day. However, I will make a strong effort to answer all messages within 24 hours on weekdays.

All materials will be posted on the course website: <http://www.ece.tufts.edu/ee/200/>

If you have a general question about the course content or course logistics, please post on Piazza rather than emailing me. That way anyone can answer the question, and everyone benefits from the response. Sign up for this course on Piazza here: <http://piazza.com/tufts/fall2019/ee200>.

## Prerequisites:

There are no prerequisites for the course except for graduate standing. In particular, we will not assume any programming experience in any language.

## Textbook:

We will use the textbook “All of Programming” by Andrew Hilton and Anne Bracy. Information about the book is online at <http://aop.cs.cornell.edu/index.html>. The text is only available as an e-book, which you can download from [Google Play](#) for the bargain price of \$10. Once you’ve bought the book, it’s possible to download it as a PDF or transfer it to various e-readers.

**Class sessions:**

Class sessions are on Mondays and Wednesdays, 4:30-5:45pm in Halligan Hall 108.

Programming is a bit like playing a sport: you improve by practicing, both on your own and under the direction of a coach. You don't improve much by watching professionals or by listening to hour-long lectures about your technique. As a result, this course is structured to let you *practice* programming, rather than listen to lectures about programming.

**Before class:** You will read the assigned sections in *All of Programming*, and complete a short quiz with questions from the reading. Each quiz will also contain a space for you to ask questions or identify material that was particularly confusing. I will use these responses to focus my time in class.

**In class:** We will begin with a brief lecture on the topics that came up on the quiz, and on any relevant tools or examples that were not covered by the book.

The rest of the class time will be devoted to working on the day's assignments individually and in groups. Sometimes we'll pause to have a group discussion about particular concepts or problems as you work through the assignments.

**After class:** You will work to complete the assignments, and submit them by the following week.

**Assignments:**

Assignments will be made available nearly every class session. Unless otherwise specified, Monday assignments will be due Fridays at 5pm (so they don't ruin your evening), and Wednesday assignments will be due the following Monday at class time.

Each assignment has two grading components: *correctness* and *style*. Correctness will be assessed with a series of automated tests which execute your function or program and ensure that it produces the correct result. Style will be manually graded, and includes your algorithm design, coding style (variable naming, indentation, braces, etc.), and documentation (comments and READMEs).

Because everything in this course builds on previous material, it is critical to have a solid understanding of each concept or technique before moving on. To encourage this, you will have a chance to revise and resubmit each assignment and get up to 75% credit for the correctness points you missed.

**Late work:**

Work submitted after the deadline but before we release the autograder results will be accepted for 90% credit. Work submitted after the autograder results are released will count for partial credit as if the original submission was a zero (i.e., it will count for 75% credit).

**Exams:**

There will be a total of three exams including the final, one after each main section of the course material. The course content is cumulative, and so the exams will be as well. However, each exam will be weighted toward the most recent material.

**Grading:**

Grades will be assigned on an absolute scale (not curved), with the following components:

Pre-class quizzes: 10%

Homework: 45%

Exams: 45%

**Plagiarism:**

Collaboration is essential in real-world software development, so you are welcome (and encouraged) to discuss approaches and solutions with other students, and to help each other debug code. Likewise, it is expected that you will use online resources such as [cplusplus.com](https://cplusplus.com) and [Stack Overflow](https://stackoverflow.com) as you develop and debug your code.

However, as with learning a sport, the person who puts in the effort gets the gains, and therefore it is essential that every student put in their own effort to learn. Concretely, this means that each student must write all of their own code. It is not acceptable to copy code from another student or from the Internet, whether you are copy-pasting or just typing from another screen.

In your submission, you must document anyone you worked with and any sources you used to develop your code (which is good practice in any software development context). Homework submissions will be run through MOSS, a tool widely used in academia to detect plagiarism.

**Schedule:**

The schedule of class topics and assigned readings is on the course website. It may be adjusted slightly to meet the needs and pace of the class.