CS 11: Introduction to Computer Science

Spring 2024

https://www.cs.tufts.edu/comp/11/

Class Sessions	Tues, Thurs; Joyce Cummings Center, Room 270
	Section 1: 10:30am-11:45am
	Section 2: 12:00pm-1:15pm
Lab Sessions	Wed; Joyce Cummings Center, Rooms 240 and 235
Instructor	Richard Townsend, richard@cs.tufts.edu
Richard's Office Hours (Cummings 440A)	Tuesdays 3:00pm-4:00pm
	Wednesdays 2:30pm-3:30pm
	(also available by appointment, just email me!)
Midterm Exam	Thursday, March 14 (Normal class time)
Final Exam	Monday, May 6
	Section 1: 3:30pm-5:30pm
	Section 2: 12:00pm-2:00pm

Contents

Course Overview

This course serves as an introduction to computer science via the programming language C++. You will learn how to devise precise procedures for solving problems, and how to specify these procedures using the C++ programming language. Along the way, you will strengthen your computational thinking skills (helpful for any form of problem solving) and begin to form a mental model of how a computer operates.

CS 11 does not assume any prior programming experience. Even if you have never programmed before, this course is for you! That being said, CS 11 is a fast-paced, challenging course that may require more of your time and effort than what you've experienced in other courses; if you have no prior experience, **expect to budget 10-12 hours per week to complete the homework**

assignments. By staying aware of the course policies, following the advice of the course staff, and taking advantage of the provided resources, you will set yourself up for success in this course regardless of your background.

Course Goals

By the end of this course, students should be able to

1. Demonstrate computational problem solving with basic programming constructs.
2. Interpret a sequence of English instructions (an algorithm) as a C++ program and vice versa.
3. Apply computational thinking to solve problems.
4. Assess a C++ program's aesthetic value and functional correctness.

Course Expectations

These expectations are designed to prevent students from missing important information or misunderstanding policies that are essential for succeeding in the course. If you experience a negative outcome in CS 11 (e.g., losing points on an assessment, missing a regrade request window, not begin granted an extension) due to a failure to meet one of these expectations, that outcome will not be changed.

By taking CS 11, the instructor expects that a student will:

- Read this entire syllabus during the first week of class.
- Follow our TA Office Hour policies.
- Follow the three CS 11 axioms:
    1. **Start early**. This is the key to tackling programming assignments in CS 11! Unlike, say, a set of individual math problems, a program has to function as a whole. The more you build, the more carefully you must think about how the various components fit together. If you're rushing, it's far more likely that you'll break something than build something functional; give yourself as much time as possible!
    2. **Think before your code**. A program is just a faster and more reliable version of a procedure that can be done with a pencil and paper. If you do not have a clear idea of how you would solve the problem with a pencil and paper, then you are not ready to write code for it.
    3. **Write a little, test a little**. Finding a bug in 5 lines of code is far easier than finding a bug in 500 lines of code. A good program is written in small, testable increments. You should not write the next section of your program unless you have a clear plan for how to test it.
- Read all Piazza posts carefully throughout the semester (Piazza is explained below). This is the main mechanism for distributing information to the whole class, including changes to policies, deadlines, etc.
- Adhere to all the policies outlined in this syllabus.

Inclusivity, Accessibility, and Religious Holy Days

This course strives for inclusion of all participants, regardless of their background or identity. Everyone is expected to treat everyone else with dignity and respect. If you feel unwelcome or mistreated for any reason by either another student, a TA, or the course itself, please let an appropriate member of the teaching staff know. If you wish to speak with someone outside of the course, you're encouraged to contact Simone Nicholson (Simone.Nicholson@tufts.edu), Diversity and Inclusion Program Administrator for the CS department.

CS 11 can be especially difficult for first-generation college students and for members of historically underrepresented groups in computing, who may not have the family or social support that helps them develop their skills in "how to be a college student." If you are a student in either of these categories, you are strongly encouraged to meet with Richard or contact him early in the term to talk about your support system.

Accommodations for Students with Disabilities

Tufts University values the diversity of our body of students, staff, and faculty and recognizes the important contribution each student makes to our unique community. Tufts is committed to providing equal access and support to all qualified students through the provision of reasonable accommodations so that each student may fully participate in the Tufts experience. If you have a disability that requires reasonable accommodations, please head to the StAAR Center website to make an appointment with an accessibility representative to determine appropriate accommodations. **Please be aware that accommodations cannot be enacted retroactively; all accommodation notes must be provided to the instructor within one week of the note being written to guarantee consideration.**

Academic Support at the StAAR Center

The StAAR Center offers a variety of free resources available to all enrolled students; they are an excellent resource! Students may make an appointment to work on any writing-related project or assignment, attend subject tutoring in a variety of disciplines, or meet with an academic coach to hone fundamental academic skills like time management or overcoming procrastination. Students may access the StAAR Center through the StAAR Center website.

Religious Holy Days

We will reasonably accommodate any student who, for reasons of observing religious holy days, will be absent from a lab or experience any hardship in the completion of their work during the holy days. Please contact the instructor in advance of the holy day if you need any additional accommodation (such as for assignments); reasonable accommodations will be allowed at no penalty **only if the instructor is notified well in advance**.

Technical Resources

Textbook

This course has one optional textbook, which serves as a helpful supplementary reference: Problem Solving with C++ by Walter Savitch. ISBN: 0133591743 (7th Edition or higher).

Piazza

The preferred means of contacting the course staff is via Piazza, an online forum where students can ask and answer questions. General questions about the homework, course policies, C++, or Linux should be posted publicly so that your classmates can benefit from the answers and any resulting discussion (before posting a question, please check to see whether your question has already been asked!). Questions that are personal in nature, or that pertain to specific pieces of code that you have written should be posted privately to the course staff.

Make sure you sign up for the CS 11 Piazza page. **Piazza will host all of our major course announcements; it is your responsibility to check in regularly to avoid missing any information.**

Course Techware

This course relies on a small number of critical programs and scripts that support the basic routine of completing and submitting labs and homeworks. It is crucial that students arrive at a basic understanding of these technologies in order to complete their work without issues. You will learn about these technologies during the first few weeks of class, but further descriptions and instructions for them can be found on the course website in the CS 11 Tech Guide.

TAs and Office Hours

This course is challenging, but we want to help you succeed! If you need help understanding a concept, tackling an assignment, or dealing with a pernicious bug, you can participate in our teaching assistant (TA) office hours (OH) sessions, which take place in the large collab room overlooking the baseball fields on the 4th floor of the Joyce Cummings Center (Room 401). When you arrive, you will "join the queue" by writing your name on a list maintained on a whiteboard. If you plan to wait somewhere other than the collab room, you must write your location next to your name. When it's your turn, a TA will come find you! The full OH schedule is posted and kept up to date as a pinned Piazza post.

OH assistance is meant to help you along or provide a nudge in the right direction; we expect that you will try to grapple with your issue yourself before asking for help. To help you follow this practice, you are expected to provide a specific topic or question to receive help. Here are some *poor* examples of specific topics/questions: "why doesn't my program work?", "debugging", "I don't know what's going on". Here are *much better* examples: "why does my program print a number when it should print a letter?", "debugging a compiler error", "I don't understand the assignment's directions".

To make your OH experience as pleasant and helpful as possible, we have a TA Office Hours Policy that you are required to read and understand before asking for help.

Grading, Labs, and Homeworks

Your course grade will be produced as a weighted sum of your scores in four categories:

1. Labs (10%)
2. Homework (60%)
3. Midterm (10%)
4. Final Exam (20%)

We do not expect to apply a curve to any portion of this grade.

All grades will be posted on Gradescope for students to review. Labs are graded as Complete or Incomplete, while homeworks, the midterm, and the final exam are each graded out of 100 points. Your final lab grade is the number of labs you completed divided by 12 (the total number of labs). Your final homework grade is the average of all your assignment submissions, with the final assignment weighted by 150% and the lowest-scoring assignment weighted by 50% (your final assignment is not considered for this lower weight).

CS 11 uses the following breakdown of letter grades and percentages, with any grade below 60% receiving an F:

98–100% A+	87–89% B+	77–79% C+	67–69% D+
93–97% A	83–86% B	73–76% C	63–66% D
90–92% A–	80–82% B–	70–72% C–	60–62% D–

Labs

Lab sessions are collaborative spaces where you will work with a partner to complete a programming problem or activity, guided by one of our stellar TAs. Attendance in weekly labs is mandatory, as they are designed expressly to prepare students for the current homework.

Ideally, you will finish the lab in the allotted time, but it's totally ok if you don't; simply submit what you have at the end of your lab session and, if you feel comfortable with the concepts involved, move on to your weekly homework assignment. You're also welcome to keep working on the lab after your lab session; labs are due by the Friday of the week the lab was released. As long as you've put in a good faith effort, you will receive a Complete lab grade. Lab grades will be posted to Gradescope as a component of the weekly homework grade and adhere to the same regrade deadlines described below.

Homework

In general, homeworks will go out every Wednesday and be due at 11:59:00PM EST the following Tuesday (there will be a larger project towards the end of the term). Each homework will have two components that together are worth a total of 100 points:

1. A **written component** (10 points), where you will answer questions that either prepare you for the rest of the assignment or gauge your understanding of recently covered concepts.
2. A **programming component**, where you will use computational thinking and C++ programming to solve one or more problems. The functional correctness of your programs (80 points) is determined by automated testing software; adherence to our coding style guide (10 points) is assessed by our TA staff.

To receive a grade, all homework files must be submitted via the specific instructions described below and on the assignment itself. You may submit as many times as you wish before the deadline, but **we will always grade the latest submission.**

Written Component Details This component must be submitted as a **.pdf** file to Gradescope. Failing to submit your written component by the assignment deadline or submitting a non-pdf document will result in 0 out of 10 points for this component. **It is your responsibility to double-check the pdf component of your submission after submitting it and make sure it's viewable on Gradescope; if the course staff cannot view it on Gradescope, you will lose credit for this component of the homework.**

Programming Component Details You will submit the programming component of each assignment as a collection of files via our `submit11` system. First and foremost, these programs **must compile** on the Halligan servers in order to receive any credit. Once your submission has compiled, it is evaluated by comparing its output to the output that our solution code produces.

With the exception of HW0, programming submissions will be graded by an automated system that uses the `diff` program to compare a submission's output to our solution's output. Students will learn to use `diff` during their second lab session and are expected to use it for testing before submitting all of their subsequent homeworks. **It is not reasonable to assume that a program is working properly if it is not passing the `diff` tests.**

For the majority of the homework assignments, we will be providing one or more sample input files that students may use for preliminary testing. Just passing these tests doesn't guarantee full credit on an assignment though; we will be running additional tests too! To account for this, students are expected to create their own input scenarios and test them using `diff`.

The triplicate of **compile-diff-submit** should become the cornerstone of every student's submission process.

Late/Make-Up Policies

The Late Token System

Homeworks are expected to be submitted on time. However, we recognize that the exigencies of college life occasionally thwart a homework deadline. In such circumstances, you can utilize the **late token system**. Each student is automatically issued **five "late tokens"** to be used on assignments; a maximum of **two** tokens can be used on a single assignment. A late token grants you a 24-hour extension on an assignment, and requires no action on your part; our grading software will automatically check the date of your submission and deduct the appropriate number of tokens. Expenditure of late tokens is governed by these rules:

- Once you are out of late tokens, late homework will no longer be accepted and will receive a score of 0.
- If you don't want to use a token, don't submit anything after the due date. We grade your most recently submitted work, so if that work came in after the due date it will be counted as late.

The total number of tokens used on an assignment is reported along with your homework and lab grades on Gradescope. Unfortunately, Gradescope does not provide a summary of how many late tokens you've used over the semester; it is your responsibility to keep track of your late token usage.

Late tokens are designed to accommodate short-term setbacks like catching a cold or an ill-timed deadline in another class. There is no need for any e-mails or explanations. Just turn in the assignment when you get it done, and the late token accounting will happen automatically.

Further Extensions

We understand that students can experience extraordinary difficulties or sudden situations that prevent them from completing work on time. Examples include serious illness, family emergencies, or other extraordinary unpleasant events. In these cases, you must *proactively contact your advising dean as soon as you can*: explain the situation to them and ask them to contact Richard. Your dean will work with Richard to make appropriate arrangements. **IMPORTANT: The earlier you notify your dean, the more flexibility the course staff will have to make appropriate arrangements.** We rarely grant extensions that are requested on or past the due date.

Regrades and Grade Explanations

The grade reports and feedback provided on Gradescope can sometimes confuse students instead of helping them. You are *always* welcome and encouraged to ask for an explanation of the grade you received; simply post on piazza privately.

If you want to request a regrade for an objective grading error on our part, you must submit a request via Gradescope on the corresponding assignment's page within **one week** of the assignment grades being released. In cases where it is not immediately obvious how to submit a regrade request for a particular component of an assignment, it is always fine to submit the request on the first problem of the assignment or pose your request as a private Piazza post.

A regrade request may or may not result in a new grade being assigned. In some cases, a regrade request will be granted but incur a point penalty. If we have to make a small manual change to your code to regrade your assignment successfully, your grade for that component will take a 10 point penalty (so your max grade would be a 90%). Starting with HW1, if this change could have been avoided by running the `diff` program appropriately before submitting the assignment, a 30 point penalty will be applied (so your max grade would be an 70%). Finally, since written components are worth 10 points themselves, we do not accept regrade requests for written components that would require uploading a new pdf to Gradescope.

Collaboration and Academic Honesty

All students are expected to know these policies in full, as ignorance is not an excuse for violating them. While some of these policies may seem overly restrictive, they are motivated with our students' learning in mind: the most effective way to learn and retain the fundamentals of computer science and programming is through predominantly independent work. If you have **any** questions about these policies or a specific academic integrity situation, do not hesitate to ask Richard. All students are expected to read and adhere to the Tufts Academic Integrity Policy as well.

Collaboration

In general, you are encouraged to discuss general CS 11 concepts with *anyone*, including other current students. This includes lecture slides and coding demos, lab handouts, reading material, and C++ concepts (functions, pointers, classes, etc.) and syntax ("how do I write a for loop?", "how do I create an integer variable?"). You are also welcome and encouraged to fully collaborate on labs with a partner.

The collaboration policy for homework assignments is much more strict and nuanced than what you may be used to in other, non-CS courses. For each component of a homework, we have slightly different policies about which forms of outside help and collaboration are acceptable:

Written The goal of most of our written problems is to give students practice at finding pieces of information that they have not expressly been given in class and lab. This skill allows you to eventually function independently as a programmer, and is critical for all computer scientists to master. As such, students are expected to find the answers to written problems themselves. They may not confer with anyone other than the course staff. From there, however, any means of finding an answer is fair game. Students may search the internet, consult their textbook, write test programs - anything.

Programming Here are our policies regarding the programming portion of assignments:

1. You may search the internet and talk to other students about general programming concepts and C++ syntax, but **not** about assignment-specific code or strategies. For example, it is fine to ask "How do you structure a 'while' loop in C++?", but it is not fine to ask "How do you write a Caesar Cipher decryptor in C++?" Assignment-specific questions should only be posed to the course staff.
2. **You should never be looking at or discussing another student's code and no one outside of our course staff should be looking at or discussing your code.** This also applies to testing and debugging: you may not help test or debug another student's code, and you may not receive help testing or debugging your code from anyone other than the current course staff. If a TA sees you participating in any of these activities, it will be reported.
3. Only submit code that you can explain. We reserve the right at any time to ask you to explain a piece of code that you submitted. If you cannot explain the code, then we will have no choice but to assume that it is not your work.
4. Do not plagiarize code! Lifting partial or complete solutions from anyone (classmates, online sources, strangers) is completely prohibited. You must not submit code that others wrote or

that you wrote for a previous class (or a previous iteration of CS 11).

5. No questions, student solutions, or instructor-provided solutions should be posted online in any capacity (except as a submission to Gradescope); **this means that you are prohibited from posting any of your work for 11 in a public Github repository**.

Any violation of the above policies will be considered cheating, and all students involved in any capacity will be forwarded directly to the Office of Student Affairs, who will investigate the case independently. Their sanctions range from horrible to inconceivably horrible. It's not worth it.

Policies for Students Retaking CS 11

If you are repeating the course or any part of it (e.g., you withdrew or dropped after doing one or more assignments), you must let Richard know at the beginning of the semester. In this case, you are expected to abide by the following policies:

- If you have written your own solutions for past semesters, it is acceptable to consult them for ideas, and it is acceptable to submit parts verbatim. Such use must be explicitly acknowledged in a README file (you will learn about these files in the assignment directions).
- In every homework, your README file must note what work is new, what work is based on work from a prior semester, and what work is submitted verbatim from a prior semester. Even if nothing is from a prior semester, your README file must disclose this information.