EECS 280
Spring 2020 Syllabus

Instructors

James Juett	jjuett@umich.edu
John Kloosterman	jklooste@umich.edu

Big Picture

Our course website is at eecs280.org, where we will post everything you need to follow along with the course. We will use Canvas only for course announcements and to post grades.

Lecture videos and exercises are asynchronous. We generally have up to four lecture activities per week on Mondays-Thursdays. See the schedule of topics on the course website for details.

You will work in teams of four students to complete lab activities and exercises. You'll be paired with an IA as a coach, and the professors will sometimes call in to your team meetings, too. Lab team meetings occur twice each week, on either Tuesday/Thursday or Wednesday/Friday, depending on what works best for your team. At the start of the class, we'll send out a survey to help match you up with team members and work out scheduling logistics. The lab worksheets are turned in on Gradescope, and the coding exercises are turned in on autograder.io.

You earn your grade by completing lecture and lab exercises, five programming projects, and four assessments throughout the term.

The professors and IAs are here to help you have a great experience in EECS 280. You'll see us around in lecture, your team meetings, office hours, review sessions, and on Piazza.

For a detailed schedule, please refer to the weekly schedule, schedule of topics, or Google Calendar at eecs280.org. For all the details, please see below.

Contact

Please direct technical questions to our Piazza forums. For other questions, you can reach the course staff at eecs280staff@umich.edu.

Textbook

Course notes are available via our website at eecs280.org.

Optional: C++ Primer, by Lippman, Lajoie and Moo. 5th edition.

C++ Primer is available in print form, electronic Kindle edition, and for free through the University Library. If you decide to rely on the library version, please be aware that the number of simultaneous users is limited, and sometimes there may be a wait to access it.

Overview

This course provides a foundation for programming, the science of "how to" as a discipline. Students will learn a variety of techniques and principles to quickly write correct programs that are easy for others to understand and adapt to new purposes. This course focuses on programming in the small, writing short programs or program components that are easily understood by a single programmer. The concepts covered in this course are not just about C++ programming, but we will see how they can be concretely realized in the C++ language.

By the end of this course, a successful student will be able to:

- Take a problem and consider various possible approaches for solving it
- Select an approach or algorithm that provides for a simple, clean, well-structured solution
- Convert the algorithm into C++ code, using good design and style
- Test and debug the program using rigorous techniques
- Understand the concepts of top-down design, data encapsulation, information hiding, and procedural and data abstraction
- Design, implement and use complete classes, including constructors, destructors, and operator overloading
- Implement dynamic data structures for stacks, queues and lists
- Be able to quickly design, implement, test and debug a large scale project independently (1000+ lines of code).

Topics

- Functional abstraction
    1. Specification
    2. Recursion
    3. The recursion/iteration duality
    4. Recursive/iterative invariants
    5. Functional generalization

- Data abstraction
    1. Formal notions of type and type hierarchies
    2. Simple and aggregate types
    3. Abstract Data Types: ADTs
    4. Abstraction functions and representation invariants
    5. Polymorphism

- Dynamic resource management
    1. Creation and destruction
    2. Rules of safety
    3. Container types
    4. Container iteration

In our exploration of these topics, we will discuss many elements of the C++ language, such as:

- Arrays (1-dimensional and multidimensional)
- File I/O
- Structs, classes, and objects
- Overloading of functions and operators
- Friends
- Strings (C-style and C++ string class)
- Pointers and dynamic arrays
- Templates
- Linked lists, stacks, and queues
- Iterators
- Exceptions
- Recursion
- Functors

Prerequisites

The prerequisite for EECS 280 is ENGR 101 or ENGR 151 or EECS 180 or EECS 183 with a minimum grade of “C”. Students seeking to waive the prereq with transfer credit or by passing a proficiency exam should contact the EECS Undergrad Advising Office in Beyster 2808 or by email to ugadmin@eecs.umich.edu. (The EECS 280 instructors are not involved in the enforcement or waiving of prereqs and cannot grant permission.)

eecs280.org website

The first place to go for any course materials or resources is our course website at https://eecs280.org. All the course materials and assignments are made available there, and are considered required reading. A detailed schedule, including lecture topics, assignment due dates and assessment dates, is also available there. Other resources such as Canvas, Piazza, and the Autograder are linked from the site.

Canvas

We'll be using Canvas to make announcements and to help you track your grades. We use Canvas announcements to communicate critical information about the course. It is your responsibility to ensure you are able to receive Canvas announcements.

Piazza

We will be using Piazza to host a course forum. You are encouraged to read this regularly and post technical questions as it will be a significant source of help and hints on the projects.

We do not answer technical questions via email. In order to save everyone time, we want all students to have the benefit of seeing each question and its answer, so please use the forum.

We prohibit publicly posting your own solutions, project code, test cases, or output to the forum. Doing so is considered a violation of the Honor Code. Also, please search the forum before posting to avoid questions that have already been asked and answered.

Lab Exercises and Lab Teams

You will be assigned a 4-person lab team that will meet twice a week for 90 minutes. During this meeting, you will complete a worksheet and have time to ask questions of course staff.

Projects

Over the course of the semester, we will assign five programming projects. You'll apply core concepts from the course to write significant programs, including an image processing tool, a game of Euchre, a website backend, and an intelligent Piazza post classifier.

Programming Environment

For this course, you must have a CAEN account. Anyone enrolled in EECS 280 is eligible for one, but it takes a little while to get it set up. (http://caen.engin.umich.edu/accounts/obtain) Our grading system uses the same version of the g++ compiler as on the CAEN Linux systems. At the time of this writing, CAEN Linux provides version 4.8.5 of the g++ compiler.

You are free to develop your programs on any platform you like, but you may use only ANSI/ISO standard C++, and are responsible for any differences between your preferred platform and the grading platform. We recommend testing your code on CAEN Linux before submitting.

The Autograder

We will grade your programs in an autograder system that is similar to the CAEN Linux environment. Your code must be submitted, compile, and run correctly on the autograder before the deadline.

We use a web-based autograder for project submissions. You will receive 3 submissions per day. Each submission will receive feedback from the results of the public test cases released with the project. Public test cases contain incomplete feedback; just because your code passes the public test cases, that does not mean it is 100% correct!

After the deadline, the autograder also runs a second set of private test cases that will be used to determine the correctness score for your submission. Your final project grade will be the score of the submission that received the combined **best** score on the public and private tests. If multiple submissions share the best score on the tests, we will grade the last such submission.

The autograder also includes public tests for code style and good programming practices. For some projects, the autograder will also evaluate the thoroughness of test cases you submit.

Successfully Completing the Projects

We have found through many years of teaching experience that the most common reason for poor project performance is **not starting early enough**. Plan to do some work on the project every day and try to have it finished a few days ahead of the due date, because many unexpected problems arise during testing. In addition, the computing sites can become very crowded, making it difficult to get a computer to use, so plan for these things to happen.

The second most common reason for not doing well on the projects is not asking for help when you need it. We offer help in office hours and on the class forums. When you come to office hours, please be ready to provide access to your code, preferably electronic. Another good way to get help is to post a question to the course forum. Remember, if you find that you are stuck on a piece of your project for an undue amount of time, please see us!

One major goal of this course is for you to learn to test and debug your programs independently. We will not debug them for you. Instead, we will help you try to figure out how to test and debug your program yourself. We will also ask you to demonstrate what testing and debugging techniques you have already tried, and what the results were, before providing any advice.

Finally, always make multiple backup copies of your work! If you somehow lose your work, it is your responsibility. If you use source control, DO NOT make your code available in a public repository. This is considered a violation of the Honor Code.

Project Evaluation and Grading

Each project will be evaluated for behavioral correctness, adherence to course principles, and general style. Some projects may also be evaluated on the quality of test cases you supply.

A program is correct if it behaves as specified in the project handout, for example, by generating the correct output. A program adheres to course principles when it uses techniques taught in lecture and asked for in the specification. For example, respecting the ADT interfaces in project 2. Finally, a program should be readable by other programmers. We have posted general guidelines for good style on the course web site.

Due Dates and Extensions

Programs turned in after the exact time and date on the assignment will receive a zero. We do not generally offer extensions. For example, we do not offer extensions due to crowded computing sites, long queue times or slow response times on the autograder, internet access problems, accidental erasure or loss of files, or outside conflicting commitments.

Students adding EECS 280 before the Drop/Add deadline may petition to complete assignments due before they added the course. Email the course staff immediately after adding the course if this is the case.

We will consider extension requests made at least two weeks in advance, for example, for religious holidays or planned medical procedures.

Additionally, we will consider requests for extensions due to documented, unanticipated medical or personal emergencies. If you can't see the instructor in advance due to the emergency, then contact them as soon as you possibly can. In all cases, we require written documentation of the emergency.

Guidelines for Project Partnerships

Form your partnership early during the project. Don't partner with someone who has already written most of the code, or vice versa. You won't get as much out of the project.

Plan your strategy for completing the project. Talk about your expected workflow. When will you meet? Do you plan to attend office hours? Do you prefer to work during the day, at night, or on the weekends? Will you start early or ~~save the project for the last few days~~?

Work on all parts of the project together, so that each partner gains experience with each of the concepts involved. This will be valuable practice for assessments. It's also helpful to have someone to bounce ideas off of and two pairs of eyes on the code to avoid bugs.

Do NOT split the work in half and work separately. This may harm your or your partner's understanding of parts of the project. You also have no control over your partner's contribution.

Academic Integrity

Encouraged Collaboration

You may give or receive help on any of the concepts covered in lecture, lab, or the textbook, and on the specifics of C/C++ syntax. You are allowed to consult with other students in the class to help you understand the assignment specification (the definition of the problem).

Partnerships

You must complete project 1 alone. You may complete the other projects and labs either alone or with a partner.

- You may collaborate with your partner in any way to complete the assignment.
- Your partner must be another student registered for (any section of) EECS 280 this term.
- You are required to register your partnership on the autograder.
- You and your partner turn in the same solution (This is enforced by the autograder).
- For those retaking the course: if you submitted an assignment in a previous term, you may **not** partner on that same assignment this term.
- You may choose different partners for different assignments. You may **not** change partners during a single assignment. If a conflict arises between you and your partner, you may contact course instructors for guidance.
- If you are aware that your partner has engaged in an honor code violation while contributing to your assignment solution (e.g. copying their portion of the code), you must report it. If you suspect this and are concerned, please reach out to course instructors immediately.

Honor Code Violations

The following are considered **honor code violations**:

- Submitting others' work as your own.
- Copying or deriving portions of your code from others' solutions.
- Collaborating with others in constructing writing your code, to the extent that your solutions are identifiably similar.
- Sharing your code with others to use as a resource when writing their code.
- Receiving help from others to write your code.

- Sharing test cases with others. These are considered part of your solution.
- Sharing your code code in any way, including making it publicly available in any form (e.g. a public GitHub repository or personal website).

You are still responsible for following these rules even after finishing the course.

Students may be nervous about being reported for coincidental similarities between their code and others, but we only report clear cases of academic misconduct. You will not be reported for:

- Using starter code provided by course instructors.
- Having the same idea as someone else.
- Receiving similar help/guidance from the same course staff member in office hours.

If you are retaking the course, you may reuse your own code, presuming it was wholly written by you and/or your partner and not derived from another source, following all the rules outlined here. It is possible for instructors to miss an honor code violation in a previous term, but catch and report it when the code is reused on a course retake.

If you have any questions as to what is allowed, please talk to an instructor right away.

The Honor Council Process

We report suspected violations to the Engineering Honor Council. To identify violations, we use both manual inspection and automated software to compare present solutions with each other, with solutions, and with code found online. The Honor Council determines whether a violation of academic standards has occurred, as well as any sanctions. Read the Honor Code for detailed definitions of cheating, plagiarism, and other forms of academic misconduct.

Here's what you can expect if you are reported for an honor council violation:

- The EECS 280 instructors submit an official report to the honor council.
- The honor council notifies you of the report, and explains the next steps of the process. You receive a copy of the report, including the evidence of the suspected violation.
- The course instructors play no role in adjudicating reported cases.
- The honor council notifies course instructors when your case is resolved. Any penalties they prescribe are applied to your grade. If you are found not responsible, your grade is unaffected.
- If you have a pending honor council case at the end of the term:
    - You receive an "I" (incomplete) grade until the case is resolved.
    - We will send you a grade projection via email to help with planning.
    - Your grade is updated once the case is resolved. The "I" should not remain on your transcript.

Grading

Your final grade is based on scores from programming projects, lecture worksheets, lab team worksheets and participation, and four assessments. The tentative point distribution is included in the following table. It is not likely that this will change, but circumstances might occur which would make changes necessary, at the discretion of the instructor.

Lecture exercises	3%
Team/Lab exercises	12%
Programming projects (p1 4%, p2-p5 8%)	36%
Assessments (12% each)	48%
Computing CARES entry and exit surveys	1%

There are no letter grades for individual projects or assessments. The final course letter grade is based on the total weighted score earned. To pass EECS 280, your weighted average project score must be a passing score, and your weighted average assessment score must be a passing score. If you score 60% overall AND your project average is above 60% AND your assessment average is above 50% AND your lecture/lab average is above 80% you can expect to pass the course with a C or better. Final grades will be assigned based on the distribution of earned scores, consistent with past incarnations of EECS 280. In the past, the median student has received a grade in the B range. We may adjust our distribution and the threshold for passing if this semester's assessments are more difficult than in the past. Please note, we do not offer the opportunity for "make-up" or extra credit work to improve your grade.

We will drop up to two missed lab participation points and up to three missed lecture exercise worksheets.

Assessments

There are four written assessments which will be administered online. You are expected to take the assessment the day it is administered. If you miss an assessment, and a medical or personal emergency is not involved, you will receive a zero for that assessment . If you anticipate an exam in another course or a religious holiday that conflicts with our assessment day, you must notify the course staff during the **first** two weeks of the term.

Regrade Requests

While we work hard to grade accurately, we sometimes make mistakes. If you believe we graded an assessment of yours incorrectly, you can submit a regrade request by the announced deadline for that assessment . We will then regrade your entire assessment , which can cause your grade to go up, but it can also go down. Regrade requests should only be made if you feel a grading error occurred - not if you are unsatisfied with the grading criteria.

Projects and labs are automatically graded and are thus not open to regrade requests.

Please refer to course announcements for the regrade procedure and deadline for individual assessments.

Tips for Doing Well in the Class

You will maximize your grade, and learn a lot at the same time, if you:

- Attend all lectures and lab sections (note that many tips and hints about projects are given during class!)
- Read the provided readings (course notes, handouts, Piazza forum, web pages)
- Hand in your work on time (even if a program does not work, turn in whatever you have done)
- Start the programming projects early and come for help as soon as you need it
- Follow the program specifications carefully

Accommodations for Students with Disabilities

If you think you need an accommodation for a disability, please let your instructor know during the first three weeks of the semester. Some aspects of this course may be modified to facilitate your participation and progress. As soon as you make us aware of your needs, we can work with the Services for Students with Disabilities (SSD) office to help us determine appropriate academic accommodations. SSD (734-763-3000; http://ssd.umich.edu) typically recommends accommodations through a Verified Individualized Services and Accommodations (VISA) form. Any information you provide is private and confidential and will be treated as such.

Commitment to Equal Opportunity

As indicated in the General Standards of Conduct for Engineering Students, we are committed to a policy of equal opportunity for all persons and do not discriminate on the basis of race, color, national origin, age, marital status, sex, sexual orientation, gender identity, gender expression, disability, religion, height, weight, or veteran status. Please feel free to contact us with any problem, concern, or suggestion. We ask that all students treat each other with respect.

Students' Mental Health and Well-being

University of Michigan is committed to advancing the mental health and well-being of its students. If you or someone you know is feeling overwhelmed, depressed, and/or in need of support, services are available. For help, contact Counseling and Psychological Services (CAPS) at (734) 764-8312 and https://caps.umich.edu during and after hours, on weekends and holidays, or through its counselors physically located in schools on both North and Central Campus. You may also consult University Health Service (UHS) at (734) 764-8320 and https://www.uhs.umich.edu/mentalhealthsvcs, or for alcohol or drug concerns, see www.uhs.umich.edu/aodresources. For a listing of other mental health resources available on and off campus, visit: http://umich.edu/~mhealth.