CSci 3081W: Program Design and Development

Fall Semester 2018, 4 Credits
Course Website :https://canvas.umn.edu/courses/73375

Professor: Daniel Keefe
Email: dfk@umn.edu
Office: 6-211 Keller Hall
Phone: 612-626-7508

TA: Dan Orban
Email: dtorban@umn.edu

TA: Libby Ferland
Email: ferla006@umn.edu

TA: Nikki Kyllonen
Email: kyllo089@umn.edu

TA: Kiran Ravindra
Email: ravin047@umn.edu

TA: Shruti Verma
Email: verma125@umn.edu

Contact

The best way to contact us is the email alias, csci3081@umn.edu, which will forward to the professor and TAs.

Office Hours: See webpage for weekly professor and TA hours or email for an appointment.

Important Dates

Lectures: Tuesdays and Thursdays 9:45am - 11am, Tate Hall 105
Discussion Sections (register for one of the four sections):

Fridays 8:00am - 8:50am
Fridays 9:05am - 9:55am
Fridays 10:10am - 11:00am
Fridays 11:15am - 12:05pm

Homework and Programming Assignment Due Dates: See the webpage
Quizzes: In class, plan for one almost every week
Midterm / Final: None!

Introduction to the Course

In 3081W, you'll learn skills, tools, and theory related to becoming a good software developer. 3081W will prepare you to succeed in 4xxx- and 5xxx-level programming intensive classes. 3081W is a required course in the computer science undergraduate curriculum and is the capstone course for CLA. The topics covered in the course include:

· The software engineering process: overview of software engineering and different software process models; design basics; testing basics.
· C++: class basics, inheritance, polymorphism, assertions and exception handling, pointers, memory management, templates.
· UML: UML basics.
· Software development tools: Makefiles, debuggers, version control software.
· Coding practices: good coding practices including design patterns, documentation skills, writing correct loops, good program organization, naming conventions.
· Professional skills: group programming skills, project management skills, professional conduct.
· Writing in computer science: genres of writing utilized in computer science; using writing for design and technical communication to different audiences.

Learning Outcomes

In this course, you will learn to:

· Understand and analyze software projects.
· Develop formal descriptions of software design considerations.
· Implement solutions to software design problems.
· Identify and effectively utilize multiple genres of writing for program design and development.

Prerequisites

CSci 2021 and CSci 2041 are the formal prerequisites for this class. You must be accepted to CS upper division, be a CS graduate student, or receive department permission. You are expected to have some previous experience programming in C, C++, Java, or a similar language. You are expected to know basic data structures (such as lists), algorithms (such as search), recursion and data abstraction. You are expected to be proficient in English and using a computer to produce text documents and figures as required by the writing assignments. Please contact me if you have any questions about whether the course is a good fit for your background and academic goals.

Expected Effort and Participation

This course covers a large amount of material, requires significant programming and other development activities, and is writing intensive. Therefore the course will require a good amount of time to do all the reading, programming, project development, studying for exams, and writing. Most students should expect to spend an average 12 to 15 hours a week on this course. Some students will need more time.

Course Structure

A "Big" Program Design and Development Project

In previous courses you've learned about different programming languages, algorithms, and data structures, and you've written programs as part of your course assignments. But, there's a big difference between writing a function or two to plug into a homework assignment and the type of programming that happens "in the wild" – for example, in the type of work you may do in industry or graduate school in a few years. One of the major goals of this class is to provide a first academic experience with designing and writing a "big" program – one that requires working as a team with other programmers, takes months to complete, requires something close to 100 separate files of software code rather than one or two, involves linking with external libraries, and requires documentation and other good programming practices in order to be successful. In this course, you'll learn how to do all these things through hands-on practice.

Our Project Topic

This semester, we'll be working on a fun visual project using computer graphics. You are going to build an interactive Paint program for digital artists. You will create several different types of brushes and use these to paint onto a digital canvas. Later in the semester, you'll extend this tool to include some of the features that are common in photo editing software, such as Adobe Photoshop. Along the way, you'll learn a bit about image data formats and image processing algorithms, but the primary learning objective will be software design. In other words, this computer graphics program will be fun example, and you'll get to create some exciting visual results, but the key is that the underlying code will have enough complexity to provide some interesting challenges and opportunities for learning about good software design -- these program design and development lessons are the ones you should take with you after the course and apply to whatever you do in the future.

Major Course Components

The project is such a defining aspect of the course that you'll find that almost everything we do in the course is timed or otherwise organized around supporting the project. Here are a few examples:

Homeworks – The first few weeks of the course will be like a C++ bootcamp. We'll focus on technical aspects of the language and related tools, such as version control systems and makefiles. During this time, you will have weekly homework assignments to practice and assess what you are learning. These will be accessed and turned in using git, which like the homework assignments themselves, will prepare you to succeed in the larger programming project that comes later in the semester.

Discussion Sections – When you registered for the course, you should have signed up for a Friday discussion session. Attendance is optional during the first 6 weeks where the topic will be the homework assignments. After this, the discussion sections will switch into a role of supporting the project work. Attendance will be required at the first discussion section after each new iteration of the project is handed out. After that, it will be optional.

Three Project Iterations – We'll divide the programming project into three iterations. Each iteration will add some new functionality to the program and/or otherwise provide a challenge for good software design and development. The result of Iteration #3 will be a software project release; you will prepare everything necessary to release it to the world, as one might do when kicking off an exciting new open source project. (Note: after the course is complete, please do not actually release your completed project online – we might like to continue to use this project in future versions of the course!)

Quizzes – There will be twelve (12) quizzes throughout the course, generally one each week. When calculating your final grades, we will automatically drop your two lowest quiz scores. We do this to provide some flexibility in case you have an unforeseen, unexcused absence, since we will not provide a make-up quiz in this situations. Make-up quizzes will only be considered for excused absences as defined by University policies, will require appropriate documentation, and — barring unforeseen emergencies — will require prior notice. If you think you meet the criteria for a make-up quiz and wish to take one (you can also just use one of your two drops), then notify me at least one week prior to the quiz.

Weekly Topics, Calendar, & Course Webpage

The course webpage is hosted on Canvas and already contains a planned schedule for weekly topics and assignments. All students registered for the course should already have access to it. We will make extensive use of this website, so it is a requirement of the course to check it regularly to stay on top of course announcements and assignments. As assignment dates and lecture topics may change slightly, I expect you to check the webpage regularly for updates.

We will often use an online forum feature. The built-in forums in Canvas do not support anonymous posts, which we have found to be a big help in the past, so we'll be using Piazza instead. One of the first homework assignments will be to initialize your Piazza account. The purpose of the online forums is to provide a place for the TAs and I to respond quickly to general questions that arise during the semester, particularly as you work on the programming assignments. Use the forum to ask general questions that apply to the entire class (e.g. a clarification about an assignment, a question about a topic discussed in class). It is important to not distribute part of your solution to the class when posting or responding to a post of the forum. In addition, when posting a question to the forum, it is essential that you keep in mind the default public nature of the communication. In particular, if you wish to discuss a grading decision or a personal concern, be sure you set the visibility to *private*, such that only you and the instructors can see the post.

Readings and Textbook

There is one official textbook for the course.

> Head First Design Patterns: A Brain-Friendly Guide, 1st Edition by Freeman and Robson
> **ISBN-10:** 0596007124

It is available at the UMN bookstore and also on Amazon.

We will also make use of several other texts and articles, which are available online. Required reading is assigned in preparation for many class sessions. For other sessions, we will note relevant supplemental readings that you can do if you wish.

Assessment

Final course grades will be calculated based upon the following percentages:

5%	In-class Writing and Small Exercises
10%	Homework Assignments
15%	Quizzes (two lowest scores automatically dropped)
23%	Project Iteration #1
23%	Project Iteration #2
24%	Project Iteration #3

Final grades will be assigned based the following scale:

Weighted Score (x)	Letter	S/N
93.0% ≤ x ≤ 100.0%	A	S
90.0% ≤ x < 93.0%	A-	S
87.0% ≤ x < 90.0%	B+	S
83.0% ≤ x < 87.0%	B	S
80.0% ≤ x < 83.0%	B-	S
77.0% ≤ x < 80.0%	C+	S
73.0% ≤ x < 77.0%	C	S
70.0% ≤ x < 73.0%	C-	S
65.0% ≤ x < 70.00	D+	N
60.0% ≤ x < 65.0%	D	N
0% ≤ x < 60.0%	F	N

This course will follow the University's Uniform Grading and Transcription Policies, which are described on the web at http://policy.umn.edu/Policies/Education/Education/GRADINGTRANSCRIPTS.html.

Late Work Policy

Late work is not accepted in this class. The primary reason for this policy is that we plan to discuss the solutions to assignments immediately after they are due. For some project iterations we will even release to the class the source code for our own solution to the assignment immediately after it is due. If for some reason your solution was not complete, you can use ours as the starting point for the next iteration of the project. We enjoy working this way; being open with these solutions provides a great learning opportunity, but it requires everyone to turn their work in on time. Requests for extensions on project iterations are, therefore, nearly impossible for us to grant. You will have a least a week to complete other course assignments (e.g., homework assignments), so extensions would also only be given in the most exceptional circumstances.

Regrades/Grade Changes

Items requested to be regraded will re-examined in their entirety, not just the problem(s)/part(s) requested to be regraded. Marks may go up or down when regraded. Requests for items to be regraded must be made **within 5 business days** of the marks being posted. Be aware, this means that students may not ask for an assignment from early in the term be regraded after they receive their final grade.

It is the student's responsibility to verify their marks are recorded properly. If a mistake has been made in recording a student's marks, the student must bring this to the attention of the course staff prior to the end of the term.

Incomplete

The I grade indicates that the instructor has (1) reasonable expectations that the student can complete an unfinished course on her/his own no later than the end of the next semester and (2) believes that legitimate reasons exist to justify extending the deadline for course completion. The only acceptable reasons will be documented illness or personal emergency. A written explanation (including supporting documentation) must be submitted to your instructor; if the explanation is acceptable, an Agreement for the Completion of Incomplete Work will be filled out as a contract between the student and the instructor.

Tips for Doing Well

1. *Embrace this opportunity.* Many of us are in computer science because we find software development rewarding. This class is an invitation to become a better programmer and a better software developer. We are lucky to have this hands-on, very practical course as a part of our curriculum. This course teaches all the things “I wish” I had learned in school, including the most important thing, how to write about computer science!
2. *Show up.* Our class meetings will mix lectures with in-class activities. Although we will reference a few textbooks, this is the kind of course that relies upon practice and participation rather than book learning. To really understand the process of program design, you need to participate in the hands-on design exercises in class. The quizes will draw directly upon the material discussed in class.
3. *Understand the nature of the class*. This class is not an introductory programming class or a theoretical class. It is a "learn by doing" class; a class that assumes you already know programming and computer science basics; and a class that involves professional skills such as group work, communication skills, and a larger view of software development issues.
4. *Give yourself a buffer on assignment deadlines*. One rule of software development is there are often (and on a truly large project, *always*) unexpected delays. This class has rigid deadlines on many assignments, so stay on pace in course activities. And plan to get crucial work done ahead of time so if/when unexpected issues arise you have time to address them and still meet the deadline.

Academic Integrity

Do Your Own Program Design and Development

All work submitted for this course is required to be your original work, or that of your group in the case of group work. You are expected to do your own thinking about how to solve an assignment, your own design, and your own coding. You are encouraged to discuss the content of the lectures and the texts with your peers. With respect to programming assignments, you are also permitted to discuss (and post to the forum regarding) programming in C++ in general (e.g., a syntax error you are stuck on). However, your discussions with others must stop before discussing a solution to the homework or assignment. If you have any question about whether discussing something with peers might go beyond what is permitted, then stop and ask us first for clarification on the policy.

The web is a fantastic resource for learning about programming, but it is also a potential source for solutions to assignments, so it is important that we are all on the same page with regard to what is a permitted use of an online resource for this class and what is not. Use of the web in a way that supplements the type of information you will find in your textbook is great. This is encouraged. For example, if we talk about the Factory Method Design Pattern in class, you are encouraged to search for Factory Method Design Pattern online and read more about this as a generic technique that can be used within many programs. However, when it comes to programming a solution to your project, you are not allowed to search online for something like "use a factory method to implement a paint program". This crosses a line into searching out the solution to an assignment rather than learning about design patterns or C++ programming in general. Here's another example. In the first iteration of our project, you will need to implement a virtual paintbrush. If you are having trouble getting your program to compile and the compiler is giving you an error that you don't understand, by all means, use google to search for the error and see if you can understand what the C++ compiler is stuck on. This is fine. You cannot, however, scour various computer graphics programming online forums to see if other programmers have created a paintbrush before and how they have implemented it. You need to do your own program design and development – this is the topic of the course.

Scholastic Conduct: Course, Department, and University Policies

Scholastic dishonesty includes any deceptive means whereby a student attempts to gain an unfair advantage. Examples of scholastic dishonesty include violating the course policies outlined here, especially "Do Your Own Program Design and Development"; plagiarizing; cheating on assignments or examinations; or engaging in unauthorized collaboration on academic work, either with other students or via the internet. In order to be as clear as possible about your scholastic conduct responsibilities and how these relate specifically to the types of courses that we teach in the Department of Computer Science & Engineering, the faculty have prepared a CS&E Department Academic Conduct Polic*y* (https://www.cs.umn.edu/academics/graduate/ms-mcs/academic-conduct). Our course will follow this policy, which stands alongside the broader Board of Regents Student Conduct Code (http://regents.umn.edu/sites/regents.umn.edu/files/policies/Student_Conduct_Code.pdf).

Within the course, a student responsible for scholastic dishonesty can be given a penalty, including an "F" or "N" for the course. I am also required to report the incident to the Office for Student Conduct and Academic Integrity (http://www.oscai.umn.edu/), and further disciplinary action may occur.

You are responsible for knowing and following the policies on scholastic conduct that are described here in the syllabus and in the related documents discussed above (see especially the CS&E Department Academic Conduct Policy).

Additional Information

Disability Information

University policy is to provide, on a flexible and individualized basis, reasonable accommodations to students who have documented disability conditions (e.g., physical, learning, psychiatric, vision, hearing, or systemic) that may affect their ability to participate in course activities or to meet course requirements. Students with disabilities are encouraged to contact Disability Services and their instructors to discuss individual needs for accommodations. Disability Services, McNamara Alumni Center, Suite 180, 200 Oak Street, East Bank. Staff can be reached at http://ds.umn.edu or by calling (612) 626-1333 (voice or TTY).

Mental Health Information

As a student you may experience a range of issues that can cause barriers to learning, such as strained relationships, increased anxiety, alcohol/drug problems, feeling down, difficulty concentrating and/or lack of motivation. These mental health concerns or stressful events may lead to diminished academic performance or reduce your ability to participate in daily activities. University of Minnesota services are available to assist you with addressing these and other concerns you may be experiencing. You can learn more about the broad range of confidential mental health services available on campus via http://www.mentalhealth.umn.edu.

Equal Access and Opportunity

The University of Minnesota shall provide equal access to and opportunity in its programs, facilities, and employment without regard to race, color, creed, religion, national origin, gender, age, marital status, disability, public assistance status, veteran status, sexual orientation, gender identity, or gender expression.

Sexual Harassment

University policy prohibits sexual harassment as defined in the University Policy Statement adopted on December 11, 1998. Complaints about sexual harassment should be reported to the University Office of Equal Opportunity, 419 Morrill Hall, East Bank.
http://www1.umn.edu/regents/policies/humanresources/SexHarassment.html

Visibility on the Web

In this class, our use of technology will make students' names, U of M Internet IDs, and/or coursework (e.g., images, videos, and descriptions of the computer graphics projects you produce) visible within course-related websites, including the main course website. Some of these websites are not secure and can be accessed by anyone. If you have concerns about the visibility of your name, Internet ID, or images and descriptions of your coursework appearing in any of these public forms, please contact me for further information.