Course Information

Description of CS 3500

Large software systems are the most complex artifacts ever created. Students who are relatively new to programming can be unaware of the organizational and technical challenges posed in creating large systems, because approaches that succeed for small software projects often don’t scale and can be disastrous for larger projects. The field of software engineering encompasses the study and application of effective tools and techniques for organizing the efforts of teams of people towards designing, developing, deploying, and maintaining software systems. CS 3500 is an introduction to software engineering. Although you will learn about all the steps involved in creating a system, we will focus primarily on the software construction step by building on the programming background that you acquired from CS 1410 and CS 2420. CS 3505 will in turn build on this course by treating the rest of the software development process in more depth. In CS 3500, we will bridge the gap between introductory programming and full-blown software engineering.

Background

A portion of the course will be devoted to studying C#. All students should have had an introductory programming course (such as CS 1410) and a programming-intensive course in algorithms and data structures (such as CS 2420). Most of you will have learned some combination of Java, Python, and/or C++ in these two courses. Regardless of the particular language(s) you have used, you should be proficient in these topics:

- Using a development environment to create, compile, test, and debug programs
- Implementing classes (understanding member variables and their role in information hiding, constructors and their role in initialization, methods and their role in abstraction)
- Implementing constructors and methods (using method calls, assignments, conditionals, and loops)
- Using and implementing common algorithms (e.g., sorting and searching) and data structures (e.g., stacks, queues, lists, trees, hash tables, etc.) as well as the use of generics or templates
- Exploiting class libraries, especially those that provide common algorithms and data structures
- Understanding techniques for exploiting polymorphism via inheritance and interfaces

If you do not have this background, you should not take CS 3500.
Lectures

Lectures are Tuesdays and Thursdays from 2:00 - 3:20p in WEB L104.

Labs

Lab sections are Wednesdays at various times (see course catalog) in MEB 3225. Please attend the discussion section you are enrolled in.

Laptop Requirement

Per School of Computing policy, students enrolled in a CS class with a lab/discussion component are required to bring a laptop to the lab. For recommendations and further information, please see the official policy:

http://www.cs.utah.edu/laptop-policy/

Instructor

Daniel Kopta, Email: dkopta@cs.utah.edu Office: MEB 3124

TAs

- Haisley Brooking
- Nhan Nguyen
- Yerry Perez
- Zak Stroud
- Richard Timpson
- Avram Twitchell
- Jolie Uk
- Amber Zheng

Course Materials

Class Website

The class web site uses Canvas and is located here:

utah.instructure.com

or through the University of Utah web page. It will contain all pertinent course info and materials, such as lecture slides, code demos, and class announcements. It will also contain lab exercises and programming assignments. All of your grades will be posted on this site so you can keep up with them throughout the semester. It is absolutely necessary that you use this web page for this class, and you are accountable for checking it regularly.

Lecture Notes

Lecture slides will be posted online shortly after each class period, along with any other documents or code examples used in class. This does not mean that all of the material covered for that lecture is included online. Attend class and pay attention!
Lab Sections

When enrolling in this class, you also enrolled in a lab/discussion section. These are smaller, weekly meetings led by the TAs designed to give you practice with the concepts discussed in class and other necessary instruction. Lab activities and instructions will be posted online and you must attend and complete them for credit.

Grading

Your grade for this course will be determined by the following:

- assignments 50%
- mid-term exam 20%
- final exam 20%
- labs 10%

If $X$ is your overall course score, letter grades will be assigned using the below scale. Scores will not be rounded.

<table>
<thead>
<tr>
<th>Grade</th>
<th>Score Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>90 &gt; $X$ ≥ 87</td>
</tr>
<tr>
<td>A-</td>
<td>87 &gt; $X$ ≥ 83</td>
</tr>
<tr>
<td>B</td>
<td>80 &gt; $X$ ≥ 77</td>
</tr>
<tr>
<td>B-</td>
<td>77 &gt; $X$ ≥ 73</td>
</tr>
<tr>
<td>C</td>
<td>70 &gt; $X$ ≥ 67</td>
</tr>
<tr>
<td>C-</td>
<td>67 &gt; $X$ ≥ 63</td>
</tr>
<tr>
<td>D</td>
<td>60 &gt; $X$ ≥ 59</td>
</tr>
<tr>
<td>D-</td>
<td>59 &gt; $X$ ≥ 55</td>
</tr>
<tr>
<td>E</td>
<td>0 &gt; $X$ ≥ 0</td>
</tr>
</tbody>
</table>

Assignments

Roughly the first half of the semester will consist of weekly software development assignments using C# in Visual Studio. These assignments will all build towards one larger software system. The specifications and due dates of these assignments will be posted online each week.

The second half of the semester will consist of a team project implementing a multiplayer game. Development of this project will be mostly from scratch.

You will hand in your software through your Github repository (info on this forthcoming). Assignments will be graded on readability, comments, design of the code, as well as correctness in execution. Partial credit may be given for incorrect or incomplete assignments, but it must be clear that a strong attempt was made. If your software does not compile or run according to the specifications, no credit will be given unless you can fix it with minor changes.

Exams

The midterm exam will be given during the regular class time in the regular class room on Thursday, October 17. The final exam will be held on Tuesday, December 10 from 1:00 - 3:00pm in the regular class room. All exams are written exams.
Getting Help

Office Hours
My office hours in MEB 3124 will be posted on Canvas shortly after the start of the semester. You are welcome to stop by any time if I am there. In general, it is best to come during my office hours or email me to set an appointment to guarantee that I will be there.

TA Consulting Hours
The teaching assistants will have regularly scheduled consulting hours held in the CADE lab. They will help with any course content, and are primarily there to help you with the weekly homework assignments. The TA consulting schedule will be posted on Canvas shortly after the start of the semester.

Other Issues
Send email to the teach staff via Piazza if you have issues or questions that can’t be resolved on the student help forums, such as a grading issue, or if your question requires posting code. See the “Getting Help” page on Canvas to set up your Piazza account. If you have a question about a personal matter, or one that does not relate to the course, feel free to email me directly at dkopta@cs.utah.edu.

We will send email and announcements to everyone in the class, such as corrections to assignments, changes to due dates, clarifications, etc. through the class web page. Students are required to check their email and the class web page regularly.

Course Guidelines

Working Together
Some of the work in this class will be completed in pairs with another student. When pair work is required, students must adhere to the techniques of pair programming. Partners are required to contribute equally half of the work and problem solving required. Students are encouraged to discuss high-level solution strategies with fellow classmates, but each student is responsible for writing their own answer. Cheating is: sharing (outside of a partnership) written or electronic work either by copying, retyping, looking at, or supplying a copy. Cheating is not: discussing concepts, answering questions about concepts or clarifying ambiguities, or helping someone understand how to use the class tools and software. There must be no collaboration during tests or the final exam. See the class web page for the cheating policy for this course. See the University of Utah Student Code for a detailed description of the University policy on cheating. Any student found cheating will fail the course. Supplying cheated materials is considered cheating just as using them is.

College of Engineering Guidelines

For information on withdrawing from courses, appealing grades, and more, see:

coe.utah.edu/students/academic-affairs/academics/semester-guidelines/
Attendance

By enrolling in this course you are implying your availability to complete and attend all lectures, labs, assignments, and tests. There will be no extensions on assignments or makeup tests due to absence, unless there is a legitimate emergency or civic duty. Specifically, vacation time and job responsibilities are not a legitimate emergency.

Late Work

Late assignments will incur a penalty of 15% of the assignment’s value if submitted within the 24-hour period following the due date. This penalty increases by 15% per 24-hour period, up to three days. Work submitted more than three days late will not receive credit. A submission is considered late if submitted any amount of time past the deadline, as measured by the submission system. **Do not risk submitting at the last minute.** Late days apply to programming assignments only; other work, such as labs, will not be accepted late.

Github

Your code for this class will be stored on a private Github repository owned by the course staff. Your repository may be deleted after the end of the semester. Make sure you backup your work if you want to keep it after the end of the semester.

Students with Disabilities

The University of Utah seeks to provide equal access to its programs, services, and activities for people with disabilities. If you need accommodations in this class, reasonable prior notice needs to be given to the Center for Disability Services, 162 Olpin Union Building, 581-5020 (V/TDD). CDS will work with you and the instructor to make arrangements for accommodations.

Safety

The University of Utah values the safety of all campus community members. To report suspicious activity or to request a courtesy escort, call campus police at 801-585-COPS (801-585-2677). You will receive important emergency alerts and safety messages regarding campus safety via text message. For more information regarding safety and to view available training resources, including helpful videos, visit safeu.utah.edu.

Violence and harassment based on race, national origin, color, religion, age, disability, sex or gender (which includes sexual orientation and gender identity/expression) is a civil rights offense and will not be tolerated. If you or someone you know has been harassed or assaulted, you are encouraged to report it to the Title IX Coordinator in the Office of Equal Opportunity and Affirmative Action, 135 Park Building, 801-581-8365, or the Office of the Dean of Students, 270 Union Building, 801-581-7066. For support and confidential consultation, contact the Center for Student Wellness, 426 SSB, 801-581-7776.
Course Objectives

Students will improve on the programming skills learned in CS 1410 and CS 2420. Students will build large and complex software systems that have a longer lifetime and development cycle. Upon completion of CS 3500, students will be able to:

- Design and implement large and complex software systems (including concurrent software) through the use of libraries (both standard and custom), and modern software development tools (such as debuggers, profilers, and revision control systems)
- Perform input validation and error handling, as well as employ advanced testing principles and tools to systematically evaluate software
- Apply the model-view-controller pattern and event handling fundamentals to create a graphical user interface
- Exercise the client-server model and high-level networking APIs to build a networked software system
- Operate a modern relational database to define relations, as well as store and retrieve data
- Appreciate the collaborative nature of software development