CMSC 461: Senior Capstone in Computer Science (3 credits)
Spring 2022
http://marmorstein.org/~robert/Spring2022/cs461.html

Instructor: Robert Marmorstein, 395-2185, marmorsteinrm@longwood.edu
Lecture: 10:00am-10:50am MWF, Stevens 118
Office Hours: 3pm-4pm MF, 12pm-3pm R (Rotunda 329) or by appointment

I am also available by appointment. My schedule is posted near my office door. To make an appointment, please check the schedule to see which times I am free, then contact me by e-mail and list some possible times we could meet. In general, I need at least 24 hours of notice to schedule an appointment.

Communications Policy:

The best way to reach me is to use **Slack**. Slack is a chat utility with clients for mobile devices and desktop computers. It will allow you to easily send me code snippets. Also, since I get notifications when a Slack message comes in, I am more likely to reply to your message quickly than if you send me e-mail. Slack also allows me to easily set up a Zoom meeting (or Google Hangout) if we need to video chat.

When you send me a Slack message, I instantly get a notification on my computer, tablet, and phone. Typically, I reply to Slack messages within 24 hours (often sooner) on weekdays and sometimes even on weekends. I am much slower at replying to e-mail (since I do not get a notification and have to explicitly check it). Typically, you can expect a reply to an e-mail within 48 hours (longer on weekends).

Slack is also a good way to communicate with other members of the class. You will be invited to a public #cmsc-461 channel in which you can discuss the projects and other course topics with other students in the class. Feel free to ask for and give help on this channel, but please stick to general answers rather than posting code.

You can also reach out to me by e-mail to marmorsteinrm@longwood.edu. However, please do not send me large files by e-mail. They take up space toward my limited quota on the mail server and cause me all sorts of headaches. **E-mail messages containing large files will be deleted unread**.

Asking for help

If you are asking for help with a project or homework problem, you can send me a direct message through Slack. You should attach your code or your work to a Slack message so that I can see where you are at. You should do this by using the “paperclip” icon to attach the file directly to your message or by copy/pasting the particular snippet of code you are working on to the body of the message.

Please do NOT attach screenshots or pictures taken on your phone. They are hard to read and do not allow me to compile your code without retyping it. Instead, attach the .cpp or .h file directly to the Slack message. You will probably need to do this from a browser running inside your Linux virtual machine (or other Linux system).

One last suggestion: don’t “ask to ask”. I am delighted to answer questions about the projects and homework assignments and you should feel free to ask questions at any time. Asking permission to ask a question wastes my time and yours.

Course Description:

A capstone course designed to consolidate experiences from a variety of other courses by working in groups on one or more large projects. Principles of software engineering will be covered, including traditional and object-oriented software design, software lifecycle models, software analysis, and management implications. The implementation of developing software using teams will be stressed along with various software tools. Reusability, portability, and interoperability will be discussed. A segment on assessment will be included.

This course is Speaking Intensive.

Prerequisites: CMSC 208, CMSC 262.

Course Student Learning Outcomes:
This course has three central themes: subject matter review, software engineering (the ability to collaborate to develop large software projects), and career preparation.

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

* Write a professional resume and cover letter

* Describe the elements of various Agile Programming paradigms

* Collaborate with peers to create and deploy a functioning, tested, and properly documented software product.

* Demonstrate mastery of computer science principles, such as common data structures and algorithms, security principles, network protocols, and

* Demonstrate proficiency of algorithms, data structures, programming language theory, computer organization, and other computer science topics on a standardized national test.

Course Structure and Student Expectations:
A significant part of this class comes from a single group project which you will complete outside of class. The assignment is designed to simulate, as closely as possible in an academic setting, a real world software project.

You will also be assigned reading and homework problems to cover the review portion of the course.

The in-class portion of the class will consist partly of lecture on software engineering topics and partly on student-led discussions of review topics. You will be responsible for leading a discussion on at least one, and probably two, review topics.

In addition to regularly attending classes, you should be prepared to spend a minimum of six hours a week working on the semester project and completing homework assignments.

Preparing to lead discussions will take an additional two to three hours, but not every week.

Textbook:
This course has two required textbooks:

"Beginning Software Engineering", 1st edition, Rod Stephens, Wrox Publishing,
ISBN: 978-1-118-96914-4 (print edition only)

"Cracking the Coding Interview: 189 Programming Questions and Solutions",
6th edition, Gayle Laakmann McDowell, CareerCup Publishing, ISBN: 978-0984782857

Course Requirements:

A significant part of your grade (50%) will be earned by completing the semester project. **Failure to complete this project successfully will result in a failing grade even if your numeric grade is high enough to pass the course.**

The remainder of your grade will come from short laboratory homework assignments and quizzes(25%), leading discussions (10%), participation (5%), and performance on the MFT, which will serve as our Final Exam(10%).

Grading Policy:
Late work will not be accepted unless you have a medical condition or family emergency which prevents you from completing the assignment on time. There will be no slip days in this course.

In the event of an emergency, you should contact me within twenty-four hours of the due date with details of why you were unable to complete the assignment. At my discretion, I may then extend the due date or find an equivalent alternative solution.

Grading Scale:
Letter grades will be assigned using the following scale. Note that there is no D- grade for this class. Any grade below a 64% is failing.

	B+: 89%	**C+:** 79%	**D+:** 69%
A: 91 - 100%	**B:** 81% - 88%	**C:** 71% - 78%	**D:** 64% - 68%
A-: 90%	**B-:** 80%	**C-:** 70%	**F:** 0% - 63%

Note: A failing grade on the semester project automatically results in an F for the class.

Attendance:
I expect you to attend class unless you are sick or engaged in a school sponsored sports event or extra-curricular activity. I will rely on your honor to enforce the attendance policy. In accordance with Longwood policy, missing more than 10% of scheduled class time to unexcused absences may result in loss of one letter grade. Missing more than 25% of class (whether excused or unexcused absences) may, at my discretion, result in a failing grade.

Food and Drink:
I prefer that you do not eat in class (it distracts me and the other students). You may bring water or other non-alcoholic beverages to class. I occasionally make exceptions to this rule for students who would otherwise miss a meal or who have medical needs. If you feel that you need such an exception, you MUST make arrangements with me before you bring food to class. Violations of this policy will be considered an unexcused absence.

Cell Phones and Laptops:
Cell phones and laptops must be turned off and put away during lecture, unless I have specifically requested, usually by e-mail, that you bring them to class (e.g. for a lab day). Violations of this policy will be considered an unexcused absence.

Honor Code:
I take the honor code seriously in my classes. Students suspected of an honor code violation will be charged with honor offenses. Any student convicted of an honor offense will receive an F in the course in addition to any penalties imposed by the honor council.

All work in this class should be considered pledged. Tests and quizzes must be completed entirely on your own and will be taken closed-book and closed-notes. You *may* discuss homework problems and laboratory projects with other students subject to the following restrictions:

1. Your submitted work must consist of *your own answers in your own words* which you have typed or written yourself. You may discuss assignments verbally with other students, but do not share code or answers electronically.

2. You must acknowledge any help you receive from anyone outside your group, including any discussion of the homework problems, by leaving a short note in the margin of the assignment, or in the case of a project, placing appropriate comments in the code. Such acknowledgments should indicate which section or sections of your work you have discussed.

3. Do not copy large blocks of code or directly copy answers from other students, the Internet, or other resources. You can discuss the general approach to an assignment and you can help other students find syntax errors in their code, but any block of code longer than two or three lines should be entirely your own work.

Campus Policies

This course adheres to the campus policies listed at http://www.longwood.edu/academicaffairs/syllabus-statements/ including the Facemask policy, Intellectual Property policy, and policy on reporting crimes and sexual misconduct. If you have course accommodations through the disability services office, please contact me outside of class to discuss how we will implement those in this course.

Tentative Course Schedule:

Week 1: Jan. 12 – 14	Introduction, Resumes and Interviews The Software Lifecycle and The Mythical Man-Month (Read Chapter 2 of “The Mythical Man Month”, Chapter 1 and 2 of Software Engineering, and Chapters I through III of Programming Interviews)
January 17	**Martin Luther King Jr. Day, NO CLASS**
Week 2: Jan. 19 – 21	Big-O, Searching and Sorting Requirements Analysis, Use Cases, and User Stories Project Management (Read Chapters 3 and 4 of Software Engineering and Chapters IV through VII of Programming Interviews)
January 20	**Last day of Add/Drop (by 5pm)**
Week 3: Jan. 24 – 28	Arrays and Strings, Hash Tables, Permutations and Palindromes Agile Programming, Scrum (Read Chapter 14 of Software Engineering and Chapter 1 of Programming Interviews)
Week 4: Jan. 31 – Feb. 4	High-Level Design, Linked Lists, Stacks and Queues (Read Chapter 5 of Software Engineering and Chapter 2 and 3 of Programming Interviews) **Use Cases and User Stories Due: Feb. 4**
Week 5: Feb. 7 – 11	Trees and Graphs Low-Level Design (Read Chapter 6 of Software Engineering and Chapter 4 of Programming Interviews)
Week 6: Feb. 14 – 18	Development and Testing (Read Chapters 7 and 8 of Software Engineering and Chapter 11 of Programming Interviews)
Week 7: Feb. 21 – 25	Deployment Bit Manipulation and Logic Puzzles (Read Chapter 9 of Software Engineering and

	Chapters 5 and 6 of Programming Interviews) **Project Checkpoint 1 Due: Feb. 25**
Week 8: Feb. 28 – Mar. 4	Object-Oriented Programming, Recursion, Dynamic Programming Software Metrics (Read Chapter 10 of Software Engineering and Chapters 7 and 8 of Programming Interviews)
March 7 – 11	**SPRING BREAK : NO CLASS**
Week 9: Mar. 14 – 18	Scalability, Searching and Sorting (Read Chapters 9 and 10 of Programming Interviews)
Week 10: Mar. 21 – 25	C and C++ and Java Software Maintenance (Read Chapter 11 of Software Engineering and Chapters 12 and 13 of Programming Interviews) **Project Checkpoint 2 Due: Mar. 25th**
Week 11: Mar. 28 – Apr. 1	Databases, Threads and Locks Process Models (Waterfall, Iterative, Agile) (Read Chapters 12 and 13 of Software Engineering and Chapters 14 and 15 of Programming Interviews)
March 30	**Deadline to Withdraw (by 5pm)**
Week 12: Apr. 4 – 8	Graph and Geometry Problems (Read Chapter XI of Programming Interviews)
Week 13: Apr. 11 – 15	Finite Automata, Grammars, and Languages **User Manual Due: Apr. 15**
Week 14: Apr. 18 – 22	MFT Practice, **Project Presentations**
April 19	**NO CLASS: Symposium Day**
Week 15: Apr. 25 – 29	MFT Practice, **Project Presentations**
Final Exam: May 4	**MFT Exam (Wednesday, 8:00am – 10:30am) [Tentative]**