COMP 2710-1/2 Software Construction Syllabus

Term: Fall 2019
Course: COMP 2710 Software Construction

Schedule:	Sec1	TR	9:30-10:45am	BKENG 2117
	Sec2	TR	11:00-12:15pm	Shelby 2205

Instructor: Dr. Xuechao Li
Office: 3101U Shelby Center
Email: xcl@auburn.edu
Office Hours: TR 8:00-9:00am

GTA: (Effective from Aug 22nd Thursday)

Thao Phung	TR	11:00-12:30	Shelby 2319(desk15-17, 21-43)	tzp0047@tigermail.auburn.edu
	F	11:30 – 1pm		
Tian Xia	T	12:30-3:30	Shelby 2319(desk15-17, 21-43)	tzx0015@tigermail.auburn.edu
Tian Liu	T	2:00-3:30	Shelby 3134	tzl0031@tigermail.auburn.edu

Recommended Text: Savitch, Walter. *Absolute C++*, 6th Edition. Addison-Wesley.

A grading list:

Thao Phung:	Christopher Acosta ~ Nathan Henty
Tian Xia:	Ryan Hernandez ~ Byron Prather
Tian Liu:	Lindsey Rafalsky ~ Jingen Zhuo

Course Description

COMP 2710 is highly schizophrenic in that it is both a course on managing the complexity of large systems and an applied programming class. Managing software complexity requires some knowledge of software process. Applied programming means that you will be required to critically analyze real-world types of problems, design algorithms, and then implement those algorithms in high-level code to solve problems. COMP 2710 is as much about learning to solve problems as it is about C++ Programming. This course requires organization, effort, and discipline. You should prepare for every class and bring LOTS of questions – COMP 2710 is not a passive viewing experience. If at any time you feel that you are falling behind, you should contact the instructor immediately and come to office hours frequently. The keys to success in this course are attending every class, starting on homework assignments as soon as they are assigned, actively studying for exams, and always requesting help in a timely fashion.

This course typically requires 9 hours of time per week, on average for the average student. If you don't have it, drop.

Course Objectives

Upon successful completion of the course, the student should be able to:

1. Grasp both C++ Syntax and Semantics
2. Develop software on Linux
3. Use assertions and design by contract to develop high-reliability programs
4. Understand concepts of data abstraction, efficiency, and memory management
5. Understand how to perform unit and system-level testing
6. Apply recursion to solve problems

7. Compare and contrast mutability and immutability
8. Understand concurrency programming
9. Understand how to avoid deadlocks in programming

Two classes conducted by TA: Nov 19th/21st.

Assessment

Exams: Two Midterm Exams, Final Exam in room 2117 BKENG (2710-1) and 2205 Shelby Center(2710-2)
Exams will be closed book, closed notes. Questions will be derived from lectures, worksheets, quizzes, materials taught only in class.

Midterm 1: 9:30 – 10:45am Sep 26th Midterm 2: 9:30 – 10:45am Oct 31st Final exam: 8 – 10:30am Dec 10th (2710-001)

Midterm 1: 11:00 – 12:15pm Sep 26th Midterm 2: 11:00 – 12:15pm Oct 31st Final exam: 12 noon – 2:30pm Dec 12th (2710-002)

Short Homework Assignments and Activities: 6 projects
These activities will be take-home in nature and designed to reinforce concepts taught in class. They will be due in writing at the beginning of class. An electronic copy may also be necessary (specified in the assignment). Generally, these assignments are designed to be low-risk in the sense that they are designed to assess thinking and effort, rather than to strictly punish errors.

1. Write a “Flow of Control” program
2. Learn test drivers, assertions, random numbers
3. Use arrays, streams, and file I/O
4. Use structures, linked lists, multiple versions, and functions
5. Understand concurrency programming
6. Understand deadlock avoidances

Your work will be graded on Linux platform.

The instructor reserves the right to assess other penalties for any errors not strictly covered in the above rubric.
Grades:

- Exams 37% (Two midterm exams and one final exam)
  - Exam1 10%
  - Exam2 12%
  - Final Exam 15%
- Quizzes 9%
- Projects 54%

A [90, 100], **B** [80,90), **C** [70,80), **D** [60,70), **F** [0,60)

Course Policies

Scaling, Curves, etc: Grades will not be scaled, curved, or adjusted arbitrarily. Do not expect a low-performance assignment/project/exam can be replaced by a high-performance one.

Projects Due Dates: Projects will be submitted through Canvas. Projects will always be due at 11:59 pm on the due date. Late assignments will receive a grade of zero (0). Deadlines will be made as generous as possible to *a priori* take into account illness, other courses, Acts of God, and nearly all conceivable excuses. If you have a documented illness preventing you from completing your assignment, you may submit all of your partial work and request an extension. **This extension is not automatic**. **No Late Submission.**

Email Policy: Your questions posted through emails are less likely to be answered (see the **Piazza** Section below), because the questions by emails cannot benefit other students. Instructor will answer homework and project questions on Piazza and Canvas. If a student asks a particularly relevant question, Instructor will post the response on **Piazza** in Canvas for the benefit of the entire class. You are responsible for all announcements made in class or electronically. You should read your Canvas information at least once every two days.

Piazza: We will adopt Piazza for class discussion. Piazza is highly catered to getting you help fast and efficiently from students, the TA, and Instructor. Rather than sending questions to me through emails, you should post your questions on Piazza.
For bugs from your source code, please do not post your source code on Piazza.

Rebuttal Period: You will be given a period of three business days to read and respond to the comments and grades of your projects. The TA may use this opportunity to address any concern and question you have. The TA also may ask for additional information from you regarding your projects.
Classroom activities: The following activities are prohibited:

- alcohol drinks (i.e. martini, beer)
- headset
- video/video games/online video game with classmates(i.e. keyboard noise, quick movement of eyes, colors reflected from eyes glasses/pupil, quiet smile/leaning body to friends with digital devices, selfies)
- video dating
- snore and then whistles

Announcements: You are responsible for all announcements made in class or electronically. You should read your e-mail at least once a day. If a student asks a particularly relevant question, I may forward the response to the entire class for their benefit.
The instructor rejects to reply your emails if no course **name/number** and **section number** in the Subject line.

GTAs will reply your emails within 120 minutes in regular working days. 8am – 6pm Mon – Fri (except weekends/national/University holidays). It is at GTAs' discretions for the rest of time. For "Ping-pong" emails, if students do not reply GTAs' emails within 120 minutes, GTA will decide if he/she continues to communicate with students.
Special Accommodations: Students who need accommodations are asked to arrange a meeting with your instructor as soon as possible. If you have a conflict with my office hours, an alternate time can be arranged. To set up this meeting, please contact me by email. If you do not have an Accommodation Memo but need accommodations, make an appointment with The Program for Students with Disabilities, Office of Accessibility, 1228 Haley Center, 844-2096 (V/TDD). https://accessibility.auburn.edu/
Please see the following link for more information on accommodations (modified 09/24/2018):
https://sites.auburn.edu/admin/universitypolicies/Policies/ADAAccommodationsPolicy.pdf

Students should proactively contact Instructor for a face to face meeting within 48 hours since students send a special request to Office of Accessibility. Otherwise, students should following regular rules without any exception.

Academic Integrity: Students will be expected to understand and follow Academic Honesty policies in place by the university. All work is to be done **individually**. Students should NOT share any project code or even detailed algorithm information with each other. Your programming code is exclusive to you.

Approved references:
The following constitute acceptable references to help you complete assignments.

- The course textbook is always approved and content may be used without citation.
- My course notes, lectures, and advice I give in my office may be used without citation
- Online general web references are fine, provided you give a citation for the website at the top of your code AND clearly label any lines of code that you use (it should never be ambiguous which lines of code you used from a website)
- Other books/textbooks on the language are fine, but require citations
- You are allowed to discuss broad conceptual ideas (for example, the idea of polymorphism) with other students, but never to share code. If you discuss something with another student (even casually), you should always cite that reference in clear terms.

Unapproved references (these constitute Academic Dishonesty):
This is not a complete listing and cases of ambiguity should always be referred to the instructor for approval prior to use.

- Solution manuals for the text (or the like)
- Websites that sell custom code to individuals
- Code written by others (students or otherwise) for this class or similar classes
- Anything not listed under "Approved References" or approved by the instructor

You MUST document references clearly. If you discuss a project with another student or professor, you should indicate what you discussed and who you discussed it with clearly in the header of your project documentation (and/or code).
For example:

//Xuechao Li
//Project1.cpp
//Dr. Li helped me debug a syntax error in my "for" loop.
//I used Wikipedia.org in order to learn how a genetic algorithm works.
//I spoke with Bob Smith in the class about identifying objects in C++.

If you don't need any sources for an assignment, clearly state "I did not use any external sources for this assignment" in your source code. Failing to document sources is plagiarism and will be penalized.

If you are unsure whether or not to document a source, it is better to document. Breaches of Academic Honesty will be referred to the Academic Honesty Committee and the strictest sanctions possible (including expulsion and failure) will be my recommendation.

If you are ever unclear about whether or not a course of action is unacceptable, you are always encouraged to consult the instructor.

Attendance: You are responsible for all materials and announcements presented in class (even if absent). Being late more than 15 minutes and/or leaving earlier than 15 minutes may loss a chance to get worksheets.

1) For exams:
   a. if you have a planned university-approved absence you must make me aware before the test in writing (with appropriate documentation).
   b. if you have an unplanned absence, you must provide written, documented, and verifiable justification in the Exam date except Emergency such as coma.
2) If you are late for a test, you do not receive any extension.

Getting Help: Projects may prove challenging and time-consuming. You are always welcome to bring questions concerning labs to the class, as well as to office hours. A good strategy is to always start early on projects, so that if you run into difficulties, you can get help as soon as possible. I will do my best to answer questions concerning labs within 24 hours of receiving them; however, I do not guarantee that I will always have time to debug code via e-mail (I prefer not to do so). For time-consuming problems dealing with code, office hours are always preferable. I will not help debug code via e-mail on the day an assignment is due.

Office Hours: You are always welcome to drop by during office hours to discuss projects or general concepts. To get urgent help or advice out of office hours, it is recommended to send an email in advance to make an appointment.

Course Difficulty: Typically, the course starts off relatively easy and gets harder as time goes on. Often, students are deceived by the (slower) initial pace and develop lazy habits at the beginning of the course. By mid-semester, they have thrown away many grade opportunities and find themselves in a bad situation with respect to grades. No amount of effort at the end of the class will compensate for consistent, dedicated effort throughout the class. Whether or not you have past experience with programming (or even with C++), my strongest recommendation is that you respect the class and come to class ready to engage every single class period. If you do this, you will dramatically increase your chances of success.