COMP 2710 Software Construction

Fall 2014

Instructor: Dr. Alvin Lim
TR 12:30pm - 1:45pm, Shelby Center, Room 1103
Office: 3110 Shelby Center
Office hours: TR 3:00pm - 4:00pm (or by appointment)

GTA: Steffi Mariya Gnanaprakasa P Arasu (smg0033@tigermail.auburn.edu)
Office: 2319 Shelby Center
Office Hours: MWF 11:00am – 12:00noon (or by appointment)

Announcements | Syllabus | Lectures | Calendar

Announcements

- The webpage of COMP 2710 is launched in Canvas.

Course Information

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 12 hours of time per week, on average for the average student. If you don't have it, drop.

Prerequisite: COMP 2210 Fundamentals of Computing.

Course Objectives:

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

- Analyze problems to determine system requirements
- Develop object-oriented software designs that map to requirements identified in analysis
- Develop software using sound programming principles
- Grasp both C++ Syntax and Semantics
- Have experience in developing non-trivial software applications
- Understand concepts of data abstraction, efficiency, and memory management
- Understand how to perform software testing.

Textbooks: Required Text: Savitch, Walter. Absolute C++, 5th Edition. 2013 . Addison-Wesley.

Topic List (not necessarily strictly in chronological order):

- Administrative Stuff (Lecture 1)
- C++ Intro: History, Basics, through Flow of Control (Lectures 1-2)
- Basic I/Oand Expressions (Lecture 2)
- Basic Functions and Strings (Lecture 3)
- Structures & Classes: Basics, Constructors (Lecture 4)
- Software Process (Basics of Analysis, Design, and Testing) (Lectures 5-6)
- Function Overloading and Templates (Lecture 7)
- Arrays (Lecture 8)

- Midterm Exam
- File IO (Lecture 9)
- Pointers & Dynamic Arrays (Lectures 10)
- Function Templates and Vectors (Lecture 11)
- Operator Overloading (Lecture 12)
- Inheritance and Polymorphism (Lectures 13-14)
- Linked Data Structures (Lecture 15)
- Additional Topics covered as they occur: Separate Compilation, Makefiles, Templates
- Final Exam will be Tuesday on December 9th, 2014, 12:00 noon - 2:30pm

---

Assessment:

Exams: Midterm Exam, Final Exam
Exams will be closed book, closed notes. Questions will be derived from lectures, material taught only in class, and from assignments. Question format will be mixed.

Short Homeworks and Activities: 3 homeworks
These activities will be take-home in nature and designed to reinforce concepts taught in class. They will be due in Canvas. An electronic copy is 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.

Individual Construction Projects: 4 Lab Assignments
These projects will consist of the creation of design artifacts (turned in prior to the implementation) and correct C++ implementations of project specifications. All projects should be made to compile under the g++ compiler on Linux. You may use any development platform or compiler, but your projects will be graded ONLY on a g++ compiler running on Linux. If your project does not work in that environment, you will NOT get credit. Always test it yourself in the Lunix Lab (Davis 123)!

Individual Projects will be graded as follows:
Analysis, Design, and Testing Documents: 20%
A useful and descriptive README file: 10%
Adhering to coding style: 10%
Program meets specifications and implements key features correctly: 60%
(Note that efficiency is not a grading criteria in this class)

Getting Help: Assignments 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 e-mails concerning labs within 48 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. The Canvas Discussion Board is a great way to ask questions so that everyone benefits from the answer to your question!

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.

Attendance: Class attendance is mandatory. Students will have to actively participate in class. It is believed that if you miss many classes (more than 6), there is a strong likelihood that you will not pass the class.

Grades:

Mid-term	20%
Final Exam	25%
Quizzes	10%
Homework Assignments	10%
Lab Assignments	35%

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

Note: In order to pass the class, you must receive at least 60% credit on the Individual Construction Projects and Homework, regardless of performance on exams.

Project Due Dates: Projects will be submitted through Canvas. Projects will always be due at 11:55 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.

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.

Special Accommodations: A student in need of special accommodations must bring that need to my attention within the first two weeks of class. The need must be properly documented.

Study Hints

- Ask questions in class.
- At the first sign of difficulty, talk to your instructor and teaching assistant.
- Form a study group and meet regularly.
- Construct chapter summaries noting concepts, definitions, & procedures.

Misc Related Material

Instructor Wiley's C++ Tips
gdb Tutorial
Reference to the C++ Language Library