CMSC 162
Introduction to Algorithmic Design II
Fall 2019
http://marmorstein.org/~robert/Fall2019/cs162.html

Lecture: 10:00am - 10:50am MWF (Ruffner 352)
Lab: 2:00pm – 3:15pm T (Ruffner G56)

Instructor: Robert Marmorstein **Office:** Ruffner 329
Office Phone: 434-395-2185 **E-mail:** marmorsteinrm@longwood.edu
Office Hours: 2:00pm-4:00pm MWF, 3:00-4:00pm R *or by appointment*

To make an appointment to see me, please contact me by e-mail and send me your schedule. Include as much detail as you can about why you need to see me (this saves time). In general, I need at least 24 hours of notice to schedule an appointment.

Course Description:
A continuation of CMSC 160. Topics include algorithmic design, complexity analysis, abstract data types, encapsulation and basic data structures. Advanced topics include using a modern high-level programming language inheritance, overloading, and use of objects. 4 credits.

Prerequisite:
Grade of C- or better in CMSC 160.

Student Learning Outcomes:
At the end of this course, the successful student will be able to:

– identify appropriate implementations for abstract data types such as stacks, queues, lists, sets, trees, and maps

– explain, implement, and use data structures such as linked lists, trees, and hash tables

– compare and contrast standard algorithms using complexity analysis

– apply object-based principles to creating understandable and maintainable solutions to problems.

Course Structure and Student Expectations
This course is heavily project-driven, but also has a heavy theoretical component. In general, we will spend three hours a week in lecture and discussion sections and one hour a week in class working on laboratory projects. However, you expect to spend at least nine additional hours each week reading the textbook, preparing for exams and working on assignments.

Textbook and Other Resources:
The textbook for this class is "Data Structures and Algorithm Analysis in C++", 4th Edition, by Mark Allen Weiss, Pearson Books, ISBN: 978-0132847377. We will supplement readings from the textbook with readings from other sources such as the Unix Programmer's Manual (sometimes called the "man pages") and the TexInfo documentation (the "info" command). Other readings will be posted to the course web site.

Course Requirements:
Your grade will depend largely on completion of the weekly lab sessions. These projects will comprise 50% of your grade. The remainder of your grade will come from homework assignments and quizzes(25%), participation(5%), the midterm exam(10%), and the final exam(10%).

University Policies:
This course adheres to the university policies found at http://www.longwood.edu/academicaffairs/syllabus-statements/.

Grading Policy:

Your final grade in this course is computed using a weighted average of your scores on each assignment. The weights for each category are given in the course requirements section of this syllabus and can be used by applying the following formula:

Final Grade = .50*Projects + .25*(Homework and Quizzes) + .05*Participation + .10*Midterm + .10*Final

Each of the category grades (such as Projects) can be computed by summing the points you've earned on each assignment in that category and dividing by the total number of points possible. Numeric grades are translated to letter grades using the following grading scale:

		100-91:	A	90:	A-
89:	B+	88-81:	B	80:	B-
79:	C+	78-71:	C	70:	C-
69:	D+	68-64:	D		
63 or lower: F		(There is no grade of D- in this course. Anything below a 64 is failing)			

Late Work:
In general, I do not accept late work or grant extensions on assignments unless you have a serious medical or family emergency which prevents you from completing the assignment on time (however, see "Slip days" below). In such cases, you do not need a doctor's note, but you must notify me of the circumstances within a reasonable amount of time.

Since slip days do not apply to homework or quizzes, I may occasionally be persuaded to grant extensions on these assignments. However, in cases where I grant such extensions, I will impose a penalty of 25% per day overdue.

All requests for extensions (whether for an emergency or not), MUST be submitted by e-mail within a reasonable amount of time (typically twelve hours from the original due date). This e-mail should outline (in detail) the reasons your work is late. Granting of extensions is entirely at my discretion – if you have not turned an assignment in on time, you should expect to earn a 0%.

Slip Days:
You will be allocated a fixed number of slip days at the start of the semester. You may use your slip days to extend the due date of one or more *programming projects*. You can use all of your slip days on one assignment or you may use them over multiple assignments.

Slip days are calculated from the minute the assignment is due until you turn it in. The number of slip days used is rounded *up* to the nearest integer value. That means that if you turn an assignment in 24 hours and 1 minute after the due date, you will use up *two* slip days. The slip day clock runs over weekends and holidays. If a lab is due on Friday and you turn it in on Monday, you will have used three slip days, not one. Slip days cannot be shared, traded, bought, or sold, but can occasionally be earned by participation in relevant campus activities I select.

Attendance:
I expect you to attend class unless you are sick or engaged in a school-sponsored sport or extracurricular activity. Please do NOT come to class if you are sick. Instead, contact me within 12 hours of the absence to

check whether you've missed any work and make arrangements to make up any missed quizzes. You should also make arrangements to get notes from another student in the class.
You should also check the course web site for announcements, new assignments, and other important updates.

I will rely primarily on your honor for enforcement of the attendance policy. However, I will keep a record of your attendance. In accordance with Longwood policy, missing more than 10% of scheduled class time (5 class sessions) to unexcused absences may, at my discretion, result in loss of one letter grade and missing 25% of class or more (14 sessions), whether excused or not may result in an automatic failing grade.

Cell Phones and Laptops:
Cell phones, music players, and laptops are to be turned off and put away during class, except as needed for the lab sessions. Violations of this policy will be considered an **unexcused** absence. I will not interrupt class to notify you if you have been counted absent for use of a prohibited device. Feel free to contact me by e-mail at any point in the semester to check on the number of absences you have in my class.

Food and Drink:
You may bring non-alcoholic beverages, including soft drinks, to class. However, please do not eat in class (it distracts me and the other students). Violations of this policy will be considered an **unexcused** absence. I will not interrupt class to notify you if you have been counted absent for violation of this policy. Feel free to contact me by e-mail at any point in the semester to check on the number of absences you have in my class.

I occasionally grant exceptions to this rule for students who must otherwise forgo lunch or have medical needs that require them to eat in class. If you feel that you need such an exception, you must make arrangements with me in advance (i.e. before bringing food to class).

Honor Code and Collaboration:
I firmly believe in the honor code. As such, I encourage you to actively collaborate with other students and to discuss homework problems. However, there is a point at which collaboration becomes cheating. To help you understand the line between acceptable discussion of a project and dishonorable behavior, I ask you to observe the following rules:

1. Exams and quizzes are to be completed entirely on your own. You may not discuss them with anyone or use any resources except those specifically outlined on the exam handout.

2. You must give proper attribution.

Whenever you receive help or use an online resource, you should comment your code to give proper credit. A simple comment like:

/ based on http://codewarrior.com */*

or

/* Jessica helped me with the curly braces here */

is fine. This comment should go directly above or on the same line as the code on which you received help, so that it is clear exactly which parts of your program are original and which are not. You do NOT need to cite material you obtain directly from me (in lecture, the assignment handout, or office hours). In general, you also do NOT need to cite material taken from the textbook.

3. The work you submit should, in general, be either your own original work or material which I have provided and you have suitably modified yourself.

At no point should another student touch your keyboard while helping you with a project. ***For homework and projects, everything you turn in should be something YOU have personally typed or hand-written. You may NOT copy code electronically from other students or the Internet.***

You MAY NOT share code with other students using flash drives, cell phones, e-mail, web sites, floppies, CDs, or other means unless I specifically direct you to do so. You MAY NOT print out copies of your code to share with other students (personal copies or copies to show me during office hours are fine).

You MAY use web sites, books, and the man pages as reference materials. However, you must cite them appropriately and you MUST re-type any code you find and not just download it or copy/paste it.

4. Do not copy large blocks of code from other students or the Internet.

You MAY assist other students or get assistance with simple problems like syntax errors, but you MAY NOT copy large blocks of code, such as entire classes or functions, from each other. A good guideline of what "large" means is that copying more than three complete programming statements is usually too much.

5. You are responsible for securing your code.

Helping other students to cheat is also cheating. Furthermore, it is your responsibility to make sure that other students do not use your work to cheat. Be careful with who you let access your account and report any missing files, flash drives, or other devices to me promptly.

Infractions of these policies will be dealt with harshly under the Longwood Honor Code. Any student convicted of an honor offense involving this class will automatically receive a final course grade of **F** in addition to any penalties imposed by the Honor Board. You should consider all work in this class to be pledged work, whether or not the pledge appears on the assignment.

If you have questions about the honor code policy, PLEASE ask me. It is much better to receive a late penalty on a single assignment than to fail the course and face honor board charges.

You may find the scenarios at https://integrity.mit.edu/handbook/writing-code helpful in understanding this policy. While their honor code policy is not identical to mine it is very, very similar.

Computing Environment:
In order to complete the programming assignments, you will need to use a Unix-based open-source operating system such as Linux or BSD. ***You are responsible for getting a development environment set up and working correctly on your system.*** There are two options for doing this:

A. You can install Linux directly onto your hard drive in a "dual-boot" configuration. This means that you can have both Linux and another operating system on your computer and can reboot to switch between them.

B. You can install Linux in a "Live" configuration, where it runs off a DVD or external drive (such as a USB flash drive). You will then need a way to save your work to another device (such as a second flash drive or in the cloud).

Either of these can work in this course, though students who successfully configure their systems to dual-boot usually experience fewer problems. **I do not recommend the use of virtual machines to run Linux for this class.**

In addition to the Linux operating system, you should have (at a minimum) the following software tools installed: the vim editor, the “gcc” and “g++” compilers, a web browser (such as chromium or brave).

If you have a Macintosh, you have an additional option. Your operating system already provides many Unix tools through the terminal utility. Most of the projects in this class can be completed directly from the Mac Terminal. To do that, you will need to install the XCode developer tools, which are available free from Apple. Be aware that you may also need to adapt the instructions of some of the programming labs to account for differences in the programming environment. If you elect to choose this option, it is your responsibility to get everything set up properly and make things work.

Tentative Course Schedule:

Aug. 26–30	Introduction, UNIX review, Design: Code Reuse and Code Modularity Review of CMSC 160 topics **Read Syllabus** **Read Sections 1.1 – 1.3**
Aug. 27	**Lab 0: Writing C++ Programs in Linux using Vim**
Sept. 3	**Lab 1: C++ Review** **Last day of Add/Drop (by 5 pm)**
Sept. 4–6	Structs and Classes, Top-Down Design, Implementation and Interface Encapsulation and Abstraction, Methods and Operators, Accessors and Mutators, Constants and Mutability **Read Section 1.4**
Sept. 9–13	Pointers, Lvalues, Rvalues, and References Dynamic Memory and Dynamic Arrays, Smart Pointers
Sept. 10	**Lab 2: Dynamic Memory**
Sept. 16–20	Constructors and Destructors, Initializer Lists Return-by-value vs Return-by-constant-reference The swap and move functions, The “big five” operators **Read Section 1.5**
Sept. 17	Lab: Catchup and Review
Sept. 23–27	Inheritance, Composition, and Polymorphism, Casting Objects, Static Members, Using the Debugger, Virtual functions and classes Linking Libraries, Makefiles, Testing, In-line Functions Templates, Vectors, Lists, and Maps, **Read Sections 1.6 – 1.7**
Sept. 24	**Lab 3: Object-Oriented Programming and Inheritance**
Sept. 30–Oct. 4	Mathematics Review, Algorithmic Analysis, Inequalities, Big-O, Subset Sum, Searching, Binary Search Version Control **Read Chapter 2**

Sept. 31	Lab: Catchup and Review
Oct. 7–11	Linked Lists, Array and Pointer based Lists, Abstract Data Types, Iterators, Using Valgrind **Read Sections 3.1 – 3.5**
Oct. 8	**Lab 4: Linked Lists, Stacks and Queues**
Oct. 14–15	**Fall Break: NO CLASS**
Oct. 16–18	Catchup and Review, **Midterm Exam**
Oct. 21–25	Stacks and Queues, Time Complexity, Using profilers **Read Sections 3.6 – 3.7**
Oct. 22	Lab: Catchup and Review
Oct. 28–Nov. 1	Trees, Tree Traversals, Balanced Binary Search Trees, B-Trees **Read Chapter 4**
Oct. 29	**Lab 5: Stacks, Graphics, and Exceptions**
Nov. 4–8	Hash Functions, and Hash Tables, Open and Closed Hashing Using static analysis tools **Read Chapter 5**
Nov. 5	Lab: Catchup and Review
Nov. 11–15	Priority Queues (Heaps), **Read Chapter 6**
Nov. 12	**Lab 6: Hashing and using an IDE (integrated development environment)**
Nov. 18 – 20	Sorting (Selection Sort, Insertion Sort, Merge Sort, Heap Sort) **Read Chapter 7**
Nov. 19	**Lab 7: Review**
Nov. 22	**Symposium Day: NO CLASS**
Nov. 25 Nov. 26	Catchup and Review Lab: Catchup and Review
Nov. 27 – 29	**Thanksgiving Holiday: NO CLASS**
Dec. 2 – 6	Disjoint Sets and Graphs, Dijkstra's Shortest Path Algorithm, Exam Review **Read Chapters 8 and 9**
Dec. 12	**Final Exam** (3:00 – 5:30pm, Thursday)

Major Assignments:

In addition to the two exams, there will be several laboratory projects in this course.

Exams:

The midterm exam will be taken in class on October 18th and is worth 10% of your final grade. The final exam will be held on Thursday, December 12th at 3:00pm and is also worth 10% of your grade.

Projects:

Projects are worth 50% of your grade. There will be six to eight laboratory projects. For tentative due dates, see the course schedule above.