### syllabus for
data structures
comp 116

**Instructor:**  Mark LeBlanc  
SC-103  508.286.3970  
mleblanc@wheatoncollege.edu  
Hours:  MW 2-3, T 2-3 or appt.  
http://cs.wheatoncollege.edu/mleblanc

Meeting Times:  MWF 11:30-12:20, T Lab 3:30-5:20  
A118(lecture)  A102(lab), SciCenter


*I also strongly recommend that you buy a 3-ring binder. I’ll pass out lots of handouts.*

**Content:**  This course is the second half of a year-long introduction to computer science for majors and minors. The content includes an introduction to the theoretical and practical aspects of data structures. Emphasis is on abstract data types and the use of the class mechanism to support object-oriented implementations, including exposure to the C++ Standard Template Library (STL). Examples include stacks, queues, linked lists, and general trees and their applications. Pointers, memory management, and recursion are discussed in length and used in some implementations.

**Pedagogy:**  This course exposes students to the theory of and techniques for “structuring data.” Moving students to a point where they can appreciate object-oriented programming (OOP) is one underlying goal in the course, including of course the reasons for and methods toward encapsulation, information hiding, and reuse. Following practice with multiple data structures from a client perspective, students spend the second half of the course “under the hood” as they learn to implement their own classes. A second goal of the course is to emphasize good software engineering, including steps that lead to verify “program correctness.” Significant portions of time are spent on learning how to write specifications and documentation, including assertions, pre- and post-conditions, and loop invariants. A number of assignments will use a software emulation of an embedded medical chip under the skin of a person. Called “medPing”, the chip performs real-time monitoring of one’s “health” (e.g., blood pressure, pulse rate, glucose level) and automatically beams the medical status to an (emulated) iPhone display window.

**Your Grade:**

<table>
<thead>
<tr>
<th>Things to do</th>
<th>Grading</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labs</td>
<td>10%</td>
<td>weekly</td>
</tr>
<tr>
<td>5 Programs</td>
<td>45%</td>
<td>TBA</td>
</tr>
<tr>
<td>Quizzes (approximately 8)</td>
<td>10%</td>
<td>TBA</td>
</tr>
<tr>
<td>Exams (2)</td>
<td>20%</td>
<td>TBA</td>
</tr>
<tr>
<td>Final Exam</td>
<td>15%</td>
<td>Thur., May 13, 9am</td>
</tr>
</tbody>
</table>
Late Submissions:
Due is due. Always turn in whatever you have on time. Something turned in on time is much
better than not having it accepted because it is late. Late is not an option. (Good, glad we can all
agree with this). Note that the moodle (onCourse) page associated with this course will have
submission areas that are time triggered; if you are late, submissions will be blocked.

Honor Code Revisited:
It goes without saying that all submitted work will be the student's own, in keeping with the
Wheaton Honor Code, unless the assignment has assigned groups. For labs, you may get “help”
from fellow classmates, but remember that all completed work must be your own. Use
discretion; don't ask your colleague for “the” answer or for lines of code. However, I do
courage you to discuss the problem in general, such as the type of statements or functions one
might use. For programming assignments, your answers and software must be your own from
beginning to end. Here is an analogy. Almost no one would every “use/steal” a line or two from
another person’s poem. Consider it the same with your programs. Don't “borrow/use” lines or
sections of code from another classmate. Your program is (like) your poem; everyone’s program
should be unique. Be wise. If a colleague is asking you for too much help, be honest and remind
them your program is just that, your program.

Homework: It is expected that you spend at least 2++ hours on reading and practice problems for
every 50 minutes of lecture. This computes to at least 6 hours of work in the Data Structures text
per week. This should be done throughout the semester and not just when studying for quizzes. The material is cumulative in a big way; for example, week 5 depends heavily on weeks 1 through 4. It is expected that you spend
at least 6 hours per week on your current programming assignment.
WARNING: Programmers typically underestimate the time it takes to
complete a software project; 6 hours per week on your programming
assignment may be one of those “underestimations.”

Labs: The labs are a critical part of the course. In almost all cases, the current lab will be
preparing you for the current programming assignment. That is, if you complete and understand
the lab, you should be well on your way to a solution for the programming assignment. In order
to best grasp the material in some labs, I strongly suggest that you completely redo any labs that
you find difficult. (Read that last sentence again, unless of course you've already reread it once).

Quizzes: In addition to two exams spaced evenly over the semester, you will be quizzed on a
(near) weekly basis. Each quiz may address material that has been assigned in the text, but not
yet presented by the instructor. NOTE: you'll have to read the material and practice the sample
problems BEFORE coming to class in order to be successful with the quizzes. Quizzes start at
the beginning of a lecture and last approximately 10-15 minutes. After some quizzes, immediate
(peer) grading will be followed by a discussion which will focus on the concepts in the quiz.
Quizzes and Exams: There will be no makeups, nor will the lowest grade be dropped. If you are an athlete and/or you have a conflict with a quiz date, please see me. When studying for exams and quizzes, have someone pick problems from the notes and text, mix them up, and present them to you as if you were taking a quiz (not telling you from what section a problem comes). While practicing, do not peek at the textbook or an earlier solution. NOTE: it’s easy to “read” code; it’s much more difficulty to “write” code from scratch. Quizzes and exams will sometimes ask you to write code from scratch; practice ownership of the concepts!

Working with MacOS X and Xcode: C++ programming assignments will often require the use of the medPing API and thus at times you must work on a computer with MacOS 10.5 and Xcode v3.1. The following labs are available for after-hours work:

- A102 (our lab room) day hours (when class is not in session)
- ICUC (2nd floor science center) Sun-Thur, 7-10:00pm, most day hours also
- Graphics Design Lab (2nd floor Meneely) Sun 12noon – midnight, Mon-Thur 9am-Midnight, Fri 9-3

HELP
I have listed my office hours on the syllabus, but be bold: schedule a time to meet. Study, study, study and talk about the material with me as often as you can.

Please don’t wait too long before you see me;
a quick chat in my office can often clear things up.
I’m here a lot...

Wondering what you’ll be doing with a degree in computer science?
Check out this website for the exciting potential of a career in computing ....