Syllabus

Essentials

Course Code:	CS 70
Course Title:	Data Structures & Program Development
Website:	http://www.cs.hmc.edu/cs70/
Piazza site:	https://piazza.com/hmc/fall2014/cs70
Professor:	Julie Medero ‹julie@cs.hmc.edu›, Olin 1269, x18072 Chris Stone ‹stone@cs.hmc.edu›, Olin 1271, x78975
Prerequisites:	CS 60 or CS 42
Credit Hours:	3
Class Times:	Tue/Thu 9:35 am–10:50 am Shanahan Center, 2450 (Section 1) Tue/Thu 1:15 pm–2:30 pm Shanahan Center, 2450 (Section 2) Tue/Thu 2:45 pm–4:00 pm Shanahan Center, 2450 (Section 3)

Overview

This course builds on the computer science foundation you received in CS 60/42 to develop your programming and problem-analysis skills. It also provides a grounding in fundamental data structures and a solid working understanding of C++. You will learn

- How to write elegant, readable, and maintainable programs
- How and when to use a range of common data structures, including lists, arrays, stacks, queues, trees, hash tables, and balanced trees.
- How use C++ effectively, including use of pointers and manual memory managements
- How to use standard software development tools, including subversion and makefiles.

You will also get lots of practice writing software, including some fairly large programs, which should allow you to improve your coding skills and speed.

Detailed objectives for the class are given in the *What You Learn in CS 70* handout.

Communication and Support

Questions and Discussions: Piazza

If you have questions about the course (e.g., about assignments, course policies, C++, etc.), you should post them on Piazza. By default, your posts should be public. That way,

we can get as many people as possible to see, respond to, and learn from your questions. You can always post anonymously, if you'd prefer that others not know who posted the question. In some cases, e.g., if you need to post a small snippet of your code, you can post privately, so that only the grutors and professor(s) will see your question. For long questions, you may be better off talking to someone in person. Come see us!

Notes and Materials: Wiki

Almost all class-related materials will be available on the course website:

> http://www.cs.hmc.edu/cs70/

This website is also reachable from the CS department's home page (via the Course Schedule link). As well as providing useful general information (such as how to find us when you have questions), homework assignments will be posted on the class website.

The website is a Wiki: it is editable, allowing you to post material about CS 70. If you would like to create a discussion area for any topic in CS 70, you can do so. Of particular importance are the *course notes* area where we post the content from our lecture slides and you add all the material that was not on those slides (including oral material and material written on the board). Included on the Wiki is a roster that associates every section of every class with a person who will act as "designated note taker." These note takers have primary responsibility for the notes for a given class, but you should check their work regularly. Since you can refer to these notes during exams, it is to your advantage to make them complete and free of errors.

Email Access

Almost all class-related discussions should be posted on Piazza. However, the course staff will sometimes send email to you, via the class mailing list: cs-70-l@hmc.edu. If you are registered in the course, you are be on this mailing list, which sends messages to your college email account. You are responsible for being familiar with announcements posted to the class mailing list(s) and with the contents of the class website.

Graders / Tutors (aka "Grutors")

There are several grutors for this course whose mission in life (at least some of the time) is to help convey the wonder of CS 70! They will be holding regular hours in the LAC lab and/or the Beckman CS labs in B102 and B105. They will also be grading your assignments.

Getting Help

If you're not getting the help you need on Piazza (or if you'd just like to see a friendly face), come talk to us in person. Talking in person is often the fastest way to resolve a

problem, especially if it's conceptual. If you speak to one of the course staff members and are not happy with (or convinced by) their answer, you should see it as their failure rather than yours, and seek us out for a better answer.

We believe that you learn best when you discover answers for yourself, so we (and the grutors) will often respond to questions with other questions, designed to help you find the answer for yourself. This approach is a little slower than just "telling you the answer," however. If you wait to ask your question until you are feeling very frustrated (or until the last minute), our approach might add to your frustrations. For that reason, ask your questions *early*.

Coursework & Grading

Assignments

All assignments come with a "grading rubric" explaining exactly what we will be looking for, and a table summarizing how your grade will be computed. We *highly recommend* you read the rubric before starting an assignment, as it explains what we are looking for and may hint at possible pitfalls.

There will be approximately ten homework assignments during the term. Assignments will usually be available on Thursdays and be due on the following Wednesday night. A followup self-assessment is due Friday afternoon.

The self-assessment gives you a chance to objectively critique your own work; the instructions are on the Wiki. Particularly good or bad (or missing) self-assessments can affect your homework score, most often by a third of a letter grade (e.g., B $\leftrightarrow$ B+ $\leftrightarrow$ A–).

Exams

There are two exams, a midterm and a final. These exams are three-hour open-book take-home exams. The exams test more than recall of raw facts; they test your ability to *apply* what you know.

Course Grade

Your final grade in CS 70 will be calculated from the component grades as shown in Table 1.[1]

(Sometimes it is necessary to make small adjustments during the semester. If so, the changes will be announced in class and posted on the course website.)

55%	Assignments
15%	Midterm
20%	Final
5%	Class Participation
5%	Wiki Participation

Table 1: Grading Formula

[1] These weights apply in the normal case, but different rules may apply in extreme situations. You cannot, for example, pass the class, if you score 0% on the final, and 100% everywhere else.

Your Responsibilities

Pair Programming

CS 70 uses the increasingly prevalent *pair-programming* methodology for all homework assignments. All homework will be done as a pair, with a single, joint, assignment turned in by that pair. During the semester, you will be part of three pairs, the first will be for the first third of the assignments, the second for the next third, and final pairing for the final third. You choose your own partner, and record your choice on the ProgrammingPairs page of the wiki. When you pair anew, you may not work with someone you worked with in a previous pairing.

In the pair-programming model, you *must* spend the bulk of your time working together as a team, with one person at the keyboard and the other at their side watching and making suggestions. All work should be a joint effort.

> YOU WILL BE **VIOLATING THE HONOR CODE** IF YOU DIVIDE THE WORK SUCH THAT YOU WORK SEPARATELY, WITH ONE PERSON DOING ONE HALF OF THE WORK AND THE OTHER PERSON DOING THE OTHER HALF.

Proper pair programming style is described in detail on the course wiki (on the PairProgramming page). Make sure you read and understand the distinct roles and responsibilities involved.

Collaboration Outside Your Pair

We love for you to discuss the lecture and reading topics with any or all of your classmates; this can range anywhere from informal chats in the hallway to formal study groups organized on the course wiki.

You can even discuss high-level features of assignments and the ideas involved, including general approaches to the problems, bugs in the specification, how long you've spent working on a problem, and so forth. You may also help each other with basic issues related to completing the assignments—how to use UNIX, basic C++ issues, and the like.

Two good rules of thumb for appropriate collaboration on homework assignments are the "in your head rule" and "understanding, not rote learning" rules. When students help each other, they should leave with understanding in their heads, not physical or electronic artifacts. Thus you are not allowed to meet as a group and leave with notes on paper, nor can you help someone fix a bug and then leave without first reverting the bug to its unfixed state. Similarly, it is not okay to leave with an answer you don't understand (in the worst case, where such a situation seems unavoidable because you have found out the answer but still don't understand it and have to leave, it is *vital* for you to say so, since not doing so is dishonest). Also, if you are helping someone, remember that giving away answers can be detrimental to someone's understanding—often it is better to ask

leading questions instead. Working together as peers and coming to an understanding together is, of course, even better.

Reminder: The Honor Code

All students—even those from other colleges—are expected to understand and comply with Harvey Mudd College's Honor Code.[2] If you haven't already done so, you must read, sign, and abide by the computer-science department's interpretation of the Honor Code to participate in this course.[3] Specifically:

- You must not exchange literal copies of material, whether that material consists of code, program output, or English-language text (e.g., documentation). You also may not copy material from published or online sources, with or without cosmetic changes (such as altering variable names), without explicit permission. If you do have permission to use externally written material, you must attribute it properly and clearly indicate which material is yours and which material is not yours.
- You should not do anything that a reasonable student peer would describe as "subverting the clear intent of the assignment," unless you have asked for and received permission to do so. Finding open-sourced code that you can use to solve an assigned problem, for example, would typically be subverting the intent of the assignment because your shortcut means that you do not learn what the assignment aims to teach.
- If you use any sources to assist you, you *must* document them. For example, all assignments have a hints-and-tips page on the course website. If a "tip" from that page ends up incorporated into your graded work, you must credit the source. (A clarification about assignment requirements or a debugging tip, however, need not be credited, although it doesn't hurt to do so if you wish.)
- If any assigned material is substantially similar to material you have done before please contact us. In particular, in an situation where we are expecting you to do new work, you *may not* reuse or refer back to substantially similar work you have previously done. It is just as bad to copy from your past self as from someone else.
- If you aren't sure whether something you've done or plan to do is allowed, you should explicitly document what you did and—if at all possible—consult with the course staff, ideally *before* you take the questionable action. Similarly, document any extensive or particularly important help you obtain, even if that help seems legitimate. If you've been helped so much that we can't consider the work truly your own, you might not be able to get full credit for it but proper attribution will avoid an Honor Code violation.

[2] See page 17 of the HMC student handbook: http://www.hmc.edu/files/dos/student-handbook.pdf.
[3] See http://www.cs.hmc.edu/honesty.html for details.

- Academic integrity also involves being careful enough to avoid unintentionally breaking the rules. Thus, you *must* read instructions in assignments and exams carefully so that you are aware of any limitations they place on you, such as time restrictions or restrictions on information sources you may consult. Similarly, if you see something that plausibly seems like it ought to be off-limits to you, such as a Subversion directory belonging to another student or files from a previous semester, you should immediately contact us to let us know that something doesn't seem right, rather than looking further at something that perhaps should have been off-limits.

These principles apply to all methods and media of discussion or exchange (voice, writing, email, etc.).

Attendance & Participation

You are expected to attend *every* class. We will not be formally taking attendance, but many classes will have group exercises that will affect your final grade (both directly, because we grade for class participation, and indirectly, because questions on the exams are often similar to the group-exercise questions). If you wish to miss a class for any reason, you should ask beforehand about the make-up work you will need to do. If you are sick, you can send word to us through another student or by email.

You are expected to participate actively in each class. *The only way to receive a high grade for class participation is to be an active participant in the class.*

Due Dates & Late Policy

Late Work Is Strongly Discouraged

The penalties for submitting late almost always outweigh the benefits. With proper planning, there is almost always a way to avoid suffering the penalties that arise from turning work in late. For example, if you know of an upcoming commitment (such as an exam in another course, or a family event you must attend) that could affect your ability to get an assignment done by its deadline, you may ask for an assignment *early*. Similarly, if, after starting an assignment, you think it is impossible to accomplish the necessary work in the available time, consult us before the due date. It may be that there is some help we can offer you that would allow you to proceed more quickly, or it may be that everyone in the class is having the same problem.

Extenuating Circumstances — Unforeseeable Situations Only

Extenuating circumstances (such as illness) are dealt with on a case-by-case basis. In general, you are only excused for situations you could not have foreseen, and only if you explain the situation at your soonest opportunity (either directly or via someone else, such as the Dean of Students), *before* the due date.

Late Penalty Formula

If, against the preceding advice, you do submit work late, your score will be scaled using the multiplier returned by the function shown in Figure 1. This function (graphed in Figure 2) is swift to penalize lateness, but slows down with time. At, and beyond, the 9-hour point, the multiplier is zero.

Times are based on the time that the submission process finished, not when you began submitting. For assignments due at 11:59 PM, we count late minutes from midnight and round down to whole numbers of minutes. Thus, you have about two minutes of grace after the clock ticks over to reading 11:59 PM. (It is, of course, risky to cut things close, especially since the submit system takes time to perform its work.)

```
double lateMult (double mins)
{
    const double MAX_LATE_PERIOD = 9 * 60;
    /* i.e. nine hours (in minutes) */

    /* This formula is strange, but gives a good penalty curve. */
    double  lateness = mins / MAX_LATE_PERIOD;
    double  root     = sqrt(lateness);
    double  squared  = lateness * lateness;
    double  penalty  = 1.0 - (root * (1.0 - root) + squared * root);
    return (penalty < 0.0) ? 0.0 : penalty;
}
```

Figure 1: The Late Penalty Function

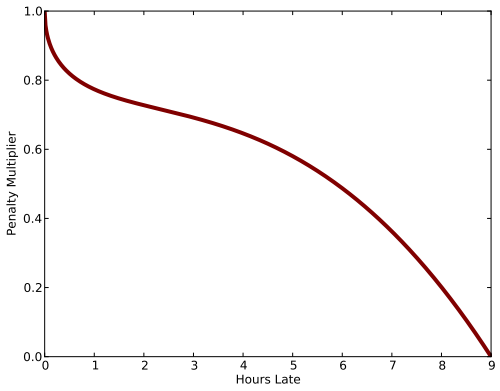

Figure 2: Graph of the Late Penalty Function

Illness

If you get sick during the term, notify us immediately, even if you think that being sick will not affect your ability to complete your assignments. You should also notify us any time that you're sick enough to miss *any* classes (not just CS 70) or find that your

performance is below par for any reason.

Textbooks

Required Texts

There are two required textbooks for this course:

- Steve McConnell, *Code Complete*, second edition. Microsoft Press, 2004. ISBN 0735619670.
- Stanley B. Lippman et al., *C++ Primer*, fifth edition. Addison-Wesley, 2012. ISBN 0321714113.

Optional Texts

The following book is recommended, but not required:

- Robert Sedgewick, *Algorithms in C++, Parts 1-4: Fundamentals, Data Structures, Sorting, Searching*, 3rd Edition. Addison-Wesley, 1998. ISBN 0201350882.