CMSC H251: Principles of Computing Systems

Instructors: John Dougherty [*a.k.a.,* jd] and Dakotah Lambert
Email: jdougher@haverford.edu || dlambert@haverford.edu
TAs: Kai, Matt, Leo, Jaxon, and Raphael
Consultation times: See the course Moodle page and Google calendar

Course Meetings Fall 2024 (TBD)

- Lectures T & Th 8:30 am - 10:00 am ET KINSC H 11
- Labs M 11:30 am -12:30 pm ET **XOR** 2:30 - 3:30 pm ET KINSC H110

Course Description:

What happens when you hit "run", after writing your program? This course introduces the elements of hardware and language/OS software that executes a program, providing insights into computing efficiency that may be important to a wide range of programmers, and serving as a foundation for later work in computing systems. It includes weekly lab exercises, focused on principles covered in the lecture and details from both lecture and self-teaching (the latter according to principles presented in the course).

Our central theme focuses on *reasoning across, rather than within, abstraction boundaries*. High-level programming in languages like Python, Java, or C++ allows programmers to reason about the correctness of their code and, for the most part, about the asymptotic resource-use complexity. However, the "constant-factor" influences on complexity do not respect abstraction boundaries; adding computing steps to an existing program can be considerably more or less resource-intensive than performing those steps in isolation. Low-level programming in languages like assembly language, C, or C++ can help to expose and manage these effects, but can also create situations in which we must reason about *correctness* across abstraction boundaries.

Understanding and controlling these cross-abstraction effects requires knowledge of computer hardware, programming language implementation, and operating system software. CMSC 251 is designed to introduce students to these issues. Discussion of computer hardware will focus on the central processing unit (CPU) and memory system (cache, RAM, virtual memory), with labs exploring implementing these components. Operating-system discussions will focus on the memory system and discuss processes, threads, and file-system issues. Programming labs will make use of the aforementioned hardware and system features, using both high- and low-level techniques, illustrated with C++ and assembly-language code. These labs will also highlight the translation of C++ into assembly language, and, in particular, the role of type information in producing efficient code.

Due to the rule against getting credit for different courses with the same content, students cannot receive credit for this course and either Byrn Mawr CMSC 223 or Swarthmore CS 31.

Learning Goals:

This course touches on each of the CS Department's core learning goals:

- The ability to **think deeply about computation**, by
    - understanding the relationship between linear/linguistic representations of algorithms (in high-level or hardware-design languages), and graphical representations (*e.g.,* circuit diagrams)
    - moving from one-element-at-a-time reasoning to reasoning that spans many different levels of computation abstraction, e.g., about memory use patterns.
- **Clear and effective communication**, by expressing thoughts about computation both as executable computer software and hardware, and as comments, lab write-ups, or oral presentations about the correctness and resource requirements of those components.
- Interpretation of the **practical implications** of computing, in that we focus on minimizing resource usage, which itself has ethical implications in terms of carbon footprint.

Reference Materials

- Text: ***Dive into Systems,*** by Mathews, Newhall, and Webb (free online; paper copies from the Haverford Bookstore).
- HERA Lab Manual: *HERA, the Haverford Educational RISC Architecture*, by Wonnacott (free online)
- Recommended: a good source of information about the C++ language and STL, such as the websites *cplusplus.com* (which includes tutorials) and *cppreference.com* (for details of the very latest features), or reference books such as *Programming: Principles and Practice Using C++ (2nd Edition)*, by Stroustrup, or the series *Effective C++, Effective STL, Effective Modern C++*, by Meyers

Course Resources

- Piazza [signup]
- Moodle
- Google Calendar
- Logisim 3.7.2

Requirements, Evaluation, and (tentative) Grading:

- 70% Lab projects, including the final project
    - ~55% labs (5-6)
    - ~15% project
- 15% Midterm Exam
- 5% Participation, including attendance activities in class/lab/individual lab review, participation in online discussions, in-class/online assessments

Course Policies

These policies are inspired by the **Honor Code of Haverford College**.

- **Collaboration and Academic Integrity**
  - Work will be governed by the CS Department collaboration policy; the core principles focus on
    - collaborative thinking with your peers
    - appropriate use of outside resources
    - a complete citation of peers and outside resources
    - not directly copying work to be handed in unless this is explicitly allowed in an assignment
  - The full current policy is as follows:
    - Collaboration is critical to learning and the practice of programming and thus is a fundamental part of students' current and future work in computer science. Our overall goal is to allow collaborative engagement that leads to individual understanding; this is realized, within the context of the Haverford Honor Code, through the details below. Note that, in all Haverford CS courses, each student who is part of a collaboration, or each student who is giving or receiving help, is responsible for making sure that this collaboration happens in a way that contributes to everyone's growth and learning and is following the Haverford Honor Code.
    - When students work together in any way, all collaborators should be cited on the resulting assignment. Any outside sources (meaning anything other than your course textbook and your notes) should be similarly cited, though before using outside sources you should check with your professor to make sure outside sources are allowed. No outside sources are allowed on exams unless otherwise stated.
    - While collaboration is encouraged, there are also times when it can cross a line and be considered inappropriate help in violation of the Honor Code. The responsibility for ensuring this collaboration remains appropriate falls on all parties. Sometimes, it is easier for a student *providing* help to tell when they are crossing the line from collaboration to doing the work for the other student. Remember, it is never helpful to your friend to deny them the chance to learn the material.
    - Work done in collaboration should never be copied from another student (e.g., from their computer or joint work on the board). Work from previous semesters should never be shared with current students, or looked at by students in the current semester, though it is fine to share notes you make about lectures or the textbook. Code and other material should never be copied from another student or outside sources unless permission is explicitly given *in advance* by your professor and the code is cited.
    - If you are uncertain about whether you have, or are about to, cross the line of acceptable collaboration, ask your instructor or the T.A. for the course, or consult our list of examples of appropriate collaboration on

details of this policy. As part of your education in CS, we expect you to develop an understanding of appropriate collaboration and attribution, so it is fair/appropriate for an instructor to include exam questions about collaboration and this policy.
    - This policy resulted from a departmental discussion and was significantly influenced by the policy at Grinnell and the 2016-07-22 "Counting From Zero" blog post about a related policy at Whitman.
- **Available Academic Resources**
    - Haverford College is committed to supporting the learning of all students. Please contact us as soon as possible if you are having difficulties with the course. There are also campus resources available to you, including the Office of Academic Resources (OAR); the Office of Access and Disabilities Services (ADS); Counseling and Psychological Services (CAPS); and the Writing Center. The staff members in each of these offices are highly trained and will preserve the confidentiality of your visit.
    - To request accommodations due to a disability, contact the Office of Access and Disability Services at hc-ads@haverford.edu. If you have already been approved to receive academic accommodations and would like to arrange accommodations in this course, please meet with me privately during the first few weeks of the semester with your verification letter from ADS.
- **Learning Accommodations**
    - We are committed to partnering with you on your academic and intellectual journey. We also recognize that your ability to thrive academically can be impacted by your well-being and that stressors may impact you over the semester. If the stressors are academic, we welcome the opportunity to discuss and address those stressors with you to find solutions together. If you are experiencing challenges or questions related to emotional health, finances, physical health, relationships, learning strategies or differences, or other potential stressors, We hope you will consider reaching out to the many resources available on campus. These resources include CAPS (free and unlimited counseling is available), the Office of Academic Resources, Health Services, Professional Health Advocate, Religious and Spiritual Life, the Office of Multicultural Affairs, the GRASE Center, and the Dean's Office. Additional information can be found at https://www.haverford.edu/deans-office-student-life/offices-resources.
    - Additionally, Haverford College is committed to creating a learning environment that meets the needs of its diverse student body and provides equal access to students with a disability. If you have (or think you have) a learning difference or disability – including mental health, medical, or physical impairment – please contact the Office of Access and Disability Services (ADS) at hc-ads@haverford.edu. The Director will confidentially discuss the process to establish reasonable accommodations. It is never too late to request accommodations – our bodies and circumstances are continuously changing.
    - Students who have already been approved to receive academic accommodations and want to use their accommodations in this course should share their accommodation letter and make arrangements to meet with the instructor **as soon as possible** to discuss how their accommodations will be implemented in

this course. Please note that accommodations are not retroactive and require advance notice to be successfully implemented.

- If, at any point in the semester, a disability or personal circumstances affect your learning in this course or if there are ways in which the overall structure of the course and general classroom interactions could be adapted to facilitate full participation, please do not hesitate to reach out to us.
- *It is a state law in Pennsylvania that individuals must be given advance notice that they may be recorded. Therefore, any student who has a disability-related need to audio record this class must first be approved for this accommodation by the Director of Access and Disability Services and then must speak to the instructor. Other class members need to be aware that this class may be recorded.*

Tentative Weekly Schedule

Wk	Topic	DiS	Due
1	Goals; C & C++ overview	0 - 1	
2	Memory Allocation & Program Scope; Performance	2; 3.3	**Lab 1C**
3	Data Representations, Binary; HERA assembly	4, 6	
4	Microarchitecture	5	**Lab 1H**
5	From HLL to assembly to machine execution	0-6	
	Fall Break		
6	C/C++ pointer management	2	Exam
7	C/C++ memory management (heap)	2, 11	**Lab 2C**
8	Registers and RAM	5.4.3; 5.5.2	
9	Functions and Parameters (stack)	7.5	**Lab 2H**
10	Inheritance in C++		
11	Optimization; more C (esp. memory management)	12	Lab 3C
12	Smart Pointers in C++	13	
13	Cache, Multicore, and other system topics		
14	**Finals Period**		Project

Grade from Raw Score

Raw Score (x)	Grade
x >= 95	4.0
90 <= x < 95	3.7
85 <= x < 90	3.3
80 <= x < 85	3.0
75 <= x < 80	2.7
70 <= x < 75	2.3
65 <= x < 70	2.0
60 <= x < 65	1.7
55 <= x < 60	1.3
50 <= x < 55	1.0
x < 50	0.0