Syllabus: CSCI 2330 Computer Systems Fall 2026

Jeffrey Knockel

1 Course description

This course will provide a broad, programmer-oriented introduction to how modern computer systems execute programs, store information, and communicate. We examine the hardware and software abstractions and implementations required to go from a program expressed in a high-level programming language like C to the computer actually running the program. Topics include concepts of program compilation and assembly, machine code, data representation and computer arithmetic, caching and the memory hierarchy, processes, and system-level I/O.

2 Learning outcomes

Primary course goals include:

- To understand how programs are executed by computers (including key abstractions and their implementations underlying program execution).
- To grasp how these implementations impact the correctness, performance, and security of real-world programs.
- To gain proficiency in practical skills of systems and programming, including debugging, command-line environments, and managing memory.
- To build foundational knowledge for further study of many fields of computer science, such as operating systems, networks, distributed systems, programming languages, and computer architecture.

3 Attendance, location, and time

Attendance is not taken but is expected and will be essential for succeeding in the class. Bring a laptop to every class for completing some in-class exercises.

- Location: Searles Science Building 223
- Time: TR 2:50pm – 4:15pm, F 1:15pm – 2:40pm
- Website: https://tildesites.bowdoin.edu/~j.knockel/csci2330f26/

4 Textbook

There is no required textbook for this class. However, you may find the online book Dive Into Systems a helpful reference.

5 Grading

Your grade will be computed as 30% lab assignments, 70% exams. All assignments are weighted equally with respect to each other, and all exams are weighted equally with respect to each other.

Assignments may be resubmitted after they have been graded to fix mistakes. You are in fact encouraged to do this. Therefore, as long as you submit each assignment and meet all resubmission deadlines, you can ultimately earn 100% on the assignments. Assignments must be resubmitted within a week of them being returned (or rereturned, in the case of reresubmissions, etc.).

Assignments receiving a grade of "0" or receiving any other penalty due to academic misconduct cannot be resubmitted.

To provide reasonable no-questions-asked flexibility with final due dates, you are allotted five flex days for the semester, each of which may be used to submit an assignment up to 24 hours late without penalty. A maximum of three flex days may be applied to a single assignment. For group projects, applying a flex day uses a flex day from each group member's allotment, but can be applied as long as at least one group member has a flex day remaining. Beyond the use of your flex days, work will not be accepted after the due date unless alternate arrangements have been approved in advance of the deadline.

6 Guidelines on group work

Collaboration is important and valuable in computer science and some assignments in this course will permit (but not require) working in a group. This section describes expectations and requirements for completing group work in this course.

For all group work, a single copy of the assignment will be submitted. Group members are expected to fully collaborate on the work, and all group members are responsible for understanding all parts of the assignment. In other words, you should not plan to split the work between the group members and complete the parts independently; instead, the expectation should be that you will work collectively as a group. For programming assignments completed in groups, most or all programming should be done with all group members working in front of a single (physical or virtual) computer. Group members should take turns "driving" (writing code) and "directing" (looking at the code and offering suggestions and feedback).

Note that this model of group work means that you should choose your group members carefully and deliberately. In particular, you should consider both your individual working styles and schedules; you cannot be an effective

team if you cannot find suitable chunks of time to collaborate synchronously. If you want to work in a group but don't have a partner in mind, let me know, and I can try to match you. Working in a group on one assignment does not commit you to working in the same group (or in any group) on future assignments.

Each student working in a group is required to email me an individual group report at the conclusion of each group assignment. Your group report must (1) identify your partner(s), (2) summarize your own contributions to the assignment, and (3) summarize your partners' contributions to the assignment. The purpose of these reports is to provide mutual accountability within your group and to ensure that groups are functioning well. These reports do not need to be overly detailed and are often of the form "I worked with my partner on the entirety of the project together at the same time." Note that we reserve the right to adjust individual grades up or down from the group grade in cases of clear inequity. Group reports should be emailed to me and are due at the same time of the assignment itself. Your assignment will not be considered submitted until both the assignment and your group report are received!

7 Collaboration Policy and Honor Code

Please review the Computer Science Collaboration Policy. You are responsible for reading, understanding, and adhering to this policy.

Group work follows the standard guidelines described above, with the exception that collaboration between members within the group is unrestricted and does not need to be cited. Any collaborations outside of the group must follow the standard guidelines.

8 AI policy

In this class, you may not *explicitly* make use of generative AI technology (e.g., you may not use ChatGPT, Claude, etc.) to complete assignments. While some exposure to generative AI technologies is unavoidable (e.g., when Googling something, you may inevitably see an AI summary), you are also encouraged to avoid these *implicit* uses as much as possible.

Examples of forbidden uses of generative AI technology include but are not limited to:

- Uploading any assignment description, text, or code to a generative AI system
- Using text or code generated by AI in your submitted assignment

Getting stuck on a difficult problem is normal, and when we finally solve it and have that breakthrough moment — that is when we have developed a new way of thinking that we can resource for the rest of our lives. Unfortunately, there is no shortcut to this process. By using generative AI or otherwise appropriating solutions from other sources we deprive ourselves of becoming better

thinkers. Of course, sometimes we get *too* stuck on a problem, and even if we aren't stuck, it is natural to still have questions. We, your class instructor and this class's learning assistants, are available to help you think about a problem without depriving yourself of those crucial opportunities to cultivate new ways of thinking critically.

Violating this AI policy constitutes academic misconduct and is subject to review by the Conduct Review Board. If you have questions as to what is or is not acceptable, please let me know. If you have accidentally violated this policy or you think that you may have accidentally violated it, please let me know as soon as possible. My intention is for no one to get into trouble accidentally.

9 Office hours and contact information

During my office hours, I would like you to talk to me about anything directly related to the class, such as assignments, but also other topics adjacent to the class like fellowships, graduate school opportunities, etc.

1. Office location: Searles Science Building 219
2. Office hours: TR 4:20–5:20pm, F 2:45–3:45pm, or by appointment
3. Email: j.knockel – bowdoin.edu

10 Accommodations

Please don't hesitate to let me know if there is anything I can do to make our shared classroom experience more conducive for you. Also, please feel free to discuss any accommodations you may require at any point during the semester. We can talk after class, in office hours, or over email.

The college welcomes students to self-identify as an individual needing accommodations. If you have a disability or medical condition that impacts your learning or academic ability and have not yet been authorized to receive academic accommodations, I encourage you to contact the Student Accessibility Office. They will help determine the options that are available for you.

11 Tentative schedule

The following is a tentative schedule for the material covered in this course and is subject to change:

Introduction

1. Sept. 1: Introduction
2. Sept. 3: Terminals, Shells, Files, Text Editors

Data Representation

3. Sept. 4: Numbering Systems
4. Sept. 8: Bitwise Operators
5. Sept. 10: Integer Representations
6. Sept. 11: Two's Complement
7. Sept. 15: Integer Logic & Floating-Point
8. Sept. 17: IEEE 754 Representation

Memory

9. Sept. 18: Memory, Endianness, & Pointers
10. Sept. 22: Arrays and Strings
11. Sept. 24: Memory Layout & Allocation
12. Sept. 25: Debugging
13. Sept. 29: GDB

Machine Code

14. Oct. 1: x86-64 ISA, Data Movement
15. Oct. 2: Addressing & Arithmetic
16. Oct. 6: Procedures & Condition Codes
17. Oct. 8: Conditionals & Loops
18. Oct. 9: **Exam 1**
19. Oct. 13: **Fall break**
20. Oct. 15: Reverse Engineering
21. Oct. 16: Switches & Jump Tables
22. Oct. 20: Procedures & Stacks
23. Oct. 22: Stack Vars & Saved Registers
24. Oct. 23: Arrays & Structs
25. Oct. 27: Buffer Overflows
26. Oct. 29: Code Injection Attacks
27. Oct. 30: Return-Oriented Programming

Memory Hierarchy

28. Nov. 3: Caching & Cache Designs
29. Nov. 5: The Memory Hierarchy
30. Nov. 6: **Exam 2**

Processes

31. Nov. 10: Processes & Exceptional Control Flow
32. Nov. 12: Process Management
33. Nov. 13: Process Control & Shells
34. Nov. 17: Signals & Zombies
35. Nov. 19: Reaping & Signal Handlers
36. Nov. 20: Files & File Descriptors
37. Nov. 24: **Thanksgiving break**
38. Nov. 26: **Thanksgiving break**
39. Nov. 27: **Thanksgiving break**
40. Dec. 1: Unix pipelines
41. Dec. 3: Concurrency & Threads
42. Dec. 4: Threads & Assignment 6 Discussion

Miscellaneous

43. Dec. 8: Virtual memory
44. Dec. 10: Other Computer Architectures
45. Dec. 11: Final Review
46. Dec. 20 1:30–4:30pm: **Final exam**

12 About me

In my research I fight to defend our safety and freedom online. For more on my research see my personal site. If you are interested in working with me please come chat with me!