Lecture 1: Course overview

David Hovemeyer

August 31, 2026

601.229 Computer Systems Fundamentals

Welcome!

- Welcome to CSF!
- Today:
  - Administrative stuff
  - Course overview
  - Generative AI

Administrative stuff

About the course

- Instructor
  - David Hovemeyer, daveho@cs.jhu.edu, Malone 240A
- CAs
  - See course web page for details

Where to find stuff

- Course website: https://jhucsf.github.io/fall2026
  - Syllabus, schedule, lecture notes, assignments, etc.
  - All public course information will be here
- Courselore https://courselore.org/
  - Announcements
  - Discussion forum, Q/A: please post questions here!
- Canvas (accessible via MyJHU)
  - Non-public course information such as homework/exam solutions
  - Videos (e.g., lecture recordings)

Syllabus highlights

- Please read the syllabus carefully:
  https://jhucsf.github.io/fall2026/syllabus.html
- Highlights:
  - Grades: 30% homework, 65% exams, 5% participation
  - (Probably) 6 assignments, mostly programming based, expect them to be challenging!
  - Late policy: you have 120 late hours to use as needed
  - Three exams (two during semester, one during final exam period)
    - Exams will be in-class
    - Will focus on recently-covered material

Pair assignments

- For most/all assignments, you may (optionally) work with one partner
- If you feel your partner isn't making an adequate contribution, you can finish the assignment on your own and turn it in individually
- You may not use your partner's lack of contribution as an excuse for not finishing the assignment

Participation

- What counts as participation?
  - What officially counts:
    - Participation in clicker quizzes in class
  - Also valuable, but won't officially count:
    - Activity on Courselore (asking questions, answering questions)
    - Attending office hours
    - Reviewing lecture recordings
- I would like to see *reasonably consistent* participation
  - You can miss 6 clicker quizzes without penalty

Academic integrity

- Please read the academic integrity policy in the syllabus carefully
- Highlights:
  - Follow the CS Academic Integrity Code: `https://www.cs.jhu.edu/academic-integrity-code/`
  - Homework assignments: you can work with one partner, or you can choose to work individually
  - Exams are (obviously) individual effort
  - Violations of academic integrity will be reported to the Student Conduct office
- Be careful about using web as a resource
  - Do *not* copy code
  - *Always* cite sources used

Academic integrity (continued)

To be considered for credit, your work and your partner's:

- Must consist of original code, written by you
- Must be a "clean room" implementation (i.e., you didn't use someone else's code as a reference)

We need to know that you can create a working implementation from a specification, on your own

Code generated by AI tools (including but not limited to ChatGPT, Github Copilot, etc.) will not be considered original work, and submitting such code is a violation of academic integrity

Much more about this in a bit...

Public solutions

I am aware that there are solutions to CSF assignments posted publicly.

Please resist the temptation of copying code from these.

Because these projects are usually posted by students who have completed CSF, they are in my pool of previous submissions, meaning that they will come up in a code similarity comparison.

Class meetings

- Typical class meeting: lecture/discussion, peer instruction questions, occasional group activities, discussion of current assignment, time for free-form Q&A
- *Do the reading in advance!*
- Come prepared to actively engage with the material!
  - Learning is not passive
  - More productive class time $\rightarrow$ better outcomes
  - Ask questions!

Peer instruction

- How peer instruction works:
  - Slide with a multiple choice question
  - Answer individually, discuss with peers, then answer again
  - Shown to improve outcomes!
  - Questions may be challenging
  - Graded for participation only
- You may have done this in other courses

Getting an iClicker remote

- The CTEI and CS department have provided enough iClicker remotes for everyone in the course to use
  - Hopefully you have it already
- **Important**: if you borrow one for the semester, we expect that you will return it in good condition
- Note that the iClicker phone app will **not** be an option (you need an actual iClicker remote)

Peer instruction etiquette

- Be respectful:
  - Let everyone participate
  - Don't put down anyone else's ideas
- Work together and think carefully about the question!

First clicker quiz!

Clicker quiz omitted from public slides

Computing requirements

- All assignments will be done using x86-64 Linux
- Autograders will use Ubuntu 22.04
- **You will need an x86-64 Linux development environment!**
- Recommendations:
  - Ugrad machines (different version of Linux, but should work fine)
  - Run Linux on your laptop or PC
  - Run Ubuntu 24.04 using WSL2 under Windows (great option!)
- I'm not aware of any way to set up a usable development environment on an M1 Mac
  - VS Code and remote SSH to ugrad is a good option

Course overview

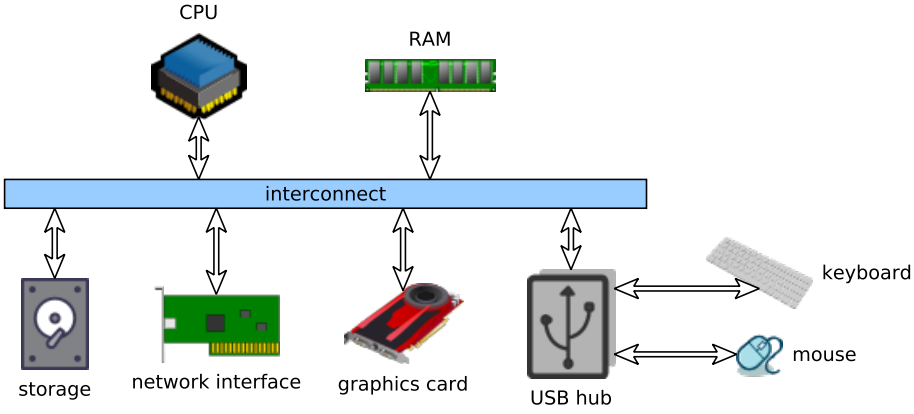
What the course is about

- Course is about *computer systems* from the *programmer's perspective*
- Computer system = hardware + software
  - Much of our concern is the interaction between hardware and software — how they work together

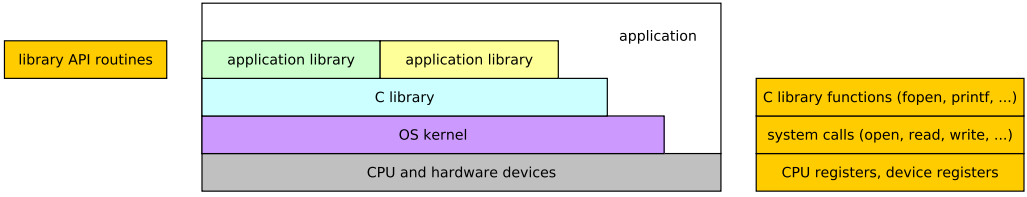
Goals of course

- "Deep" understanding of how computers work (down to hardware)
  - OS and runtime library interfaces
  - Machine-level ISA / assembly language
  - Processor features
  - Operating system features
- Apply this understanding to...
  - Optimize application performance
  - Avoid pitfalls such as security vulnerabilities
  - Take full advantage of the computer's and operating system's capabilities

A computer system (hardware)

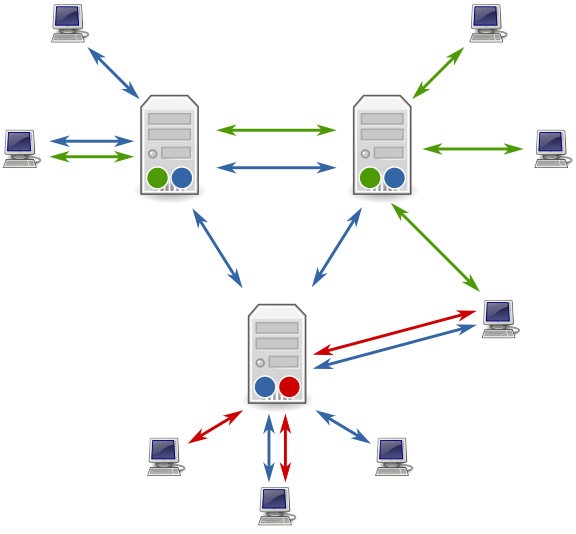


A computer system (software)



- Your application program is supported by lower layers of software and hardware
- Each layer provides an interface to the layer above

A computer network



Computer networks allow your program to communicate with peer systems.

Thanks to the global Internet, the peer systems could be anywhere on earth!

Generative AI

Generative AI

We're at the point where

- It's possible to give a programming assignment to an AI model and have it output a working solution with no effort on your part
- LLMs can answer questions (often correctly!) that you might otherwise have asked the instructor, a CA, or a peer

What are the implications?

AI vs. Learning

Having AI generate an assignment solution denies you a learning opportunity

The purpose of undergraduate education is to help you develop expertise in subject areas

Over-reliance on AI means you don't learn things you could have learned, and don't develop skills you could have developed

I believe strongly that to have true expertise in software development, you need lots of hands-on experience at all levels of abstraction

- If you consistently defer the actual programming work to AI, you will never truly have expertise at that level of abstraction

AI vs. Communicating with People

Asking questions of chatbots is convenient (they're available any time)

Asking questions of a machine potentially avoids anxiety or embarrassment ("Maybe my question is a dumb question? I'll just ask ChatGPT...")

However:

- A chatbot tends to give you an answer rather than challenging you to think
  - This can create a false perception that you are learning
- Interactions with the course staff and your peers are valuable!
  - One of the most valuable things you will do here at JHU is to develop a network of connections
  - Avoiding contact with people means you will miss opportunities for collaboration, references, research opportunities, etc.

AI vs. Academic Integrity

Because generative AI can (in many cases) produce reasonably high-quality solutions to assignments, there is an obvious temptation to use it to do exactly this

This creates a perverse situation where students who do the work without AI assistance feel penalized

- Which then leads to a "race to the bottom" effect where everyone feels they have to use AI

What We Will Do In CSF

Several policies and practices in CSF will attempt to avoid allowing generative AI to impede learning:

1. Git repository requirement
2. Code walkthroughs
3. Exams

Git Repository Requirement

You will need to fill out two Google forms to let us know your Github account information, and to register yourself or your team (you are allowed to work with one partner.) See Courselore post #5 for details.

We will create a Github repository for you (and your partner if you are working in a pair) to use for your assignment work.

We will review your commit history as part of assignment grading. Documenting your development progress in the Git history is a requirement for receiving credit for each assignment.

Code Walkthroughs

Within one week of submitting an assignment, you will need to meet with me or a CA for a code walkthrough. We will ask you to explain how your code works and to justify your implementation decisions.

A satisfactory code walkthrough is a requirement for receiving credit for the assignment submission.

If you are working with a partner, both you and your partner must actively participate in the code walkthrough.

Exams

The exams will have questions that are closely connected to the work you did on the assignment(s) preceding the exam.

If you haven't fully benefited from the learning experience the assignments are designed to provide, you will be at a disadvantage in answering such questions.

Next time...

Data representation (binary numbers, machine data types, addresses and memory)