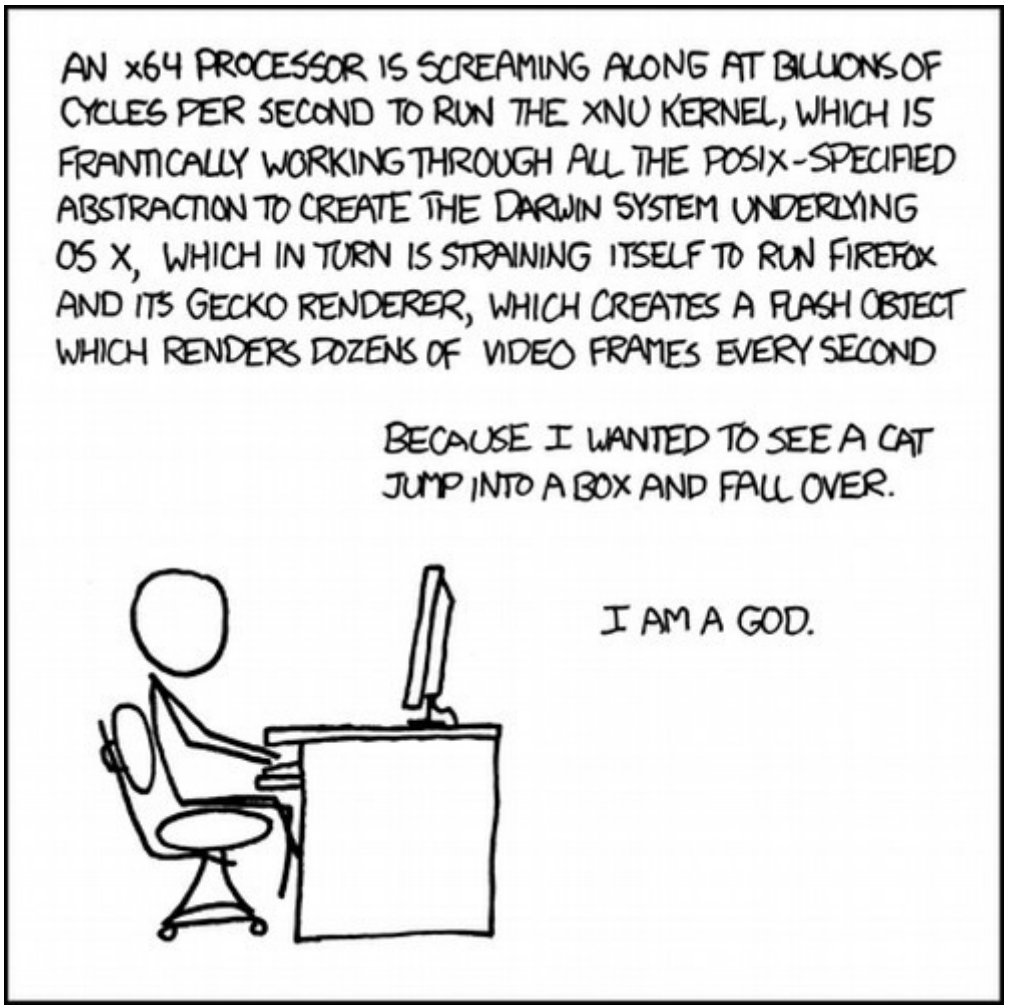
# CS 261
# Fall 2016

Mike Lam, Professor




## Computer Systems I: Course Overview

# Systems

- What is a “system?”

# Systems

- What is a “system?”
  - Set of interacting components
  - More than the sum of its parts

# Systems

- What is a “system?”
  - Set of interacting components
  - More than the sum of its parts

*"Multiple thingies working together" – my wife*

# Systems

- Why do we care about computer systems?

# Systems

- Why do we care about computer systems?
  - Because they’re everywhere

# Systems

- We use complex digital systems every day
  - Phones, tablets, PCs, embedded systems, etc.
- Also: it's your major!
- Our goal: peel back some of the complexity
  - See what's "under the hood"
  - It's also worth thinking about why the complexity was hidden; i.e., what is the purpose of abstraction?

# Systems

- What is a *process*? What is a *file*?

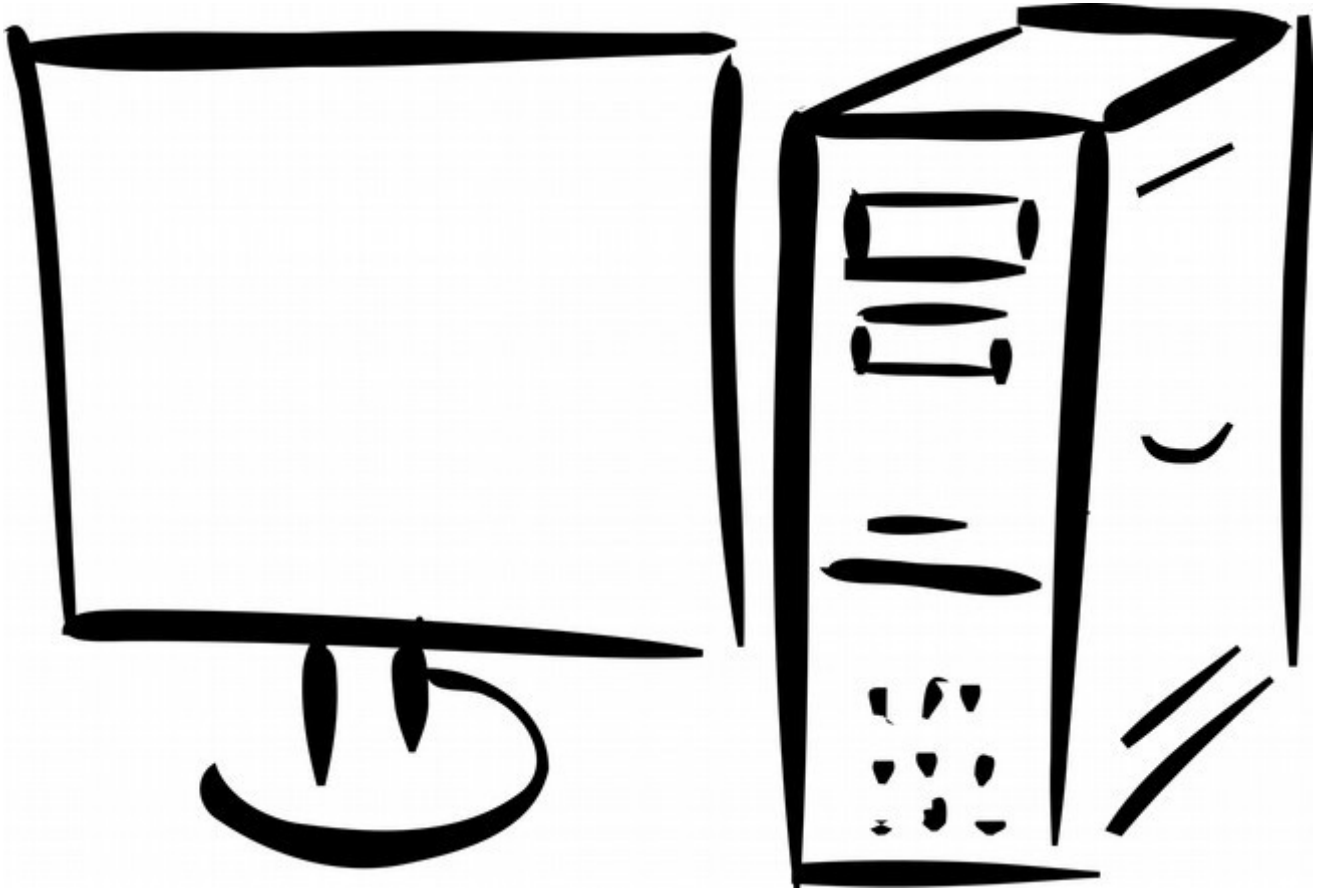
# Systems

- What is a *process*? What is a *file*?
  - These are examples of abstraction; "fake" views of reality that reduce complexity to users
  - Especially important in large, complicated systems
  - In this case, they are abstractions provided by the operating system to facilitate other software
  - Understanding abstractions can improve your ability to effectively use them

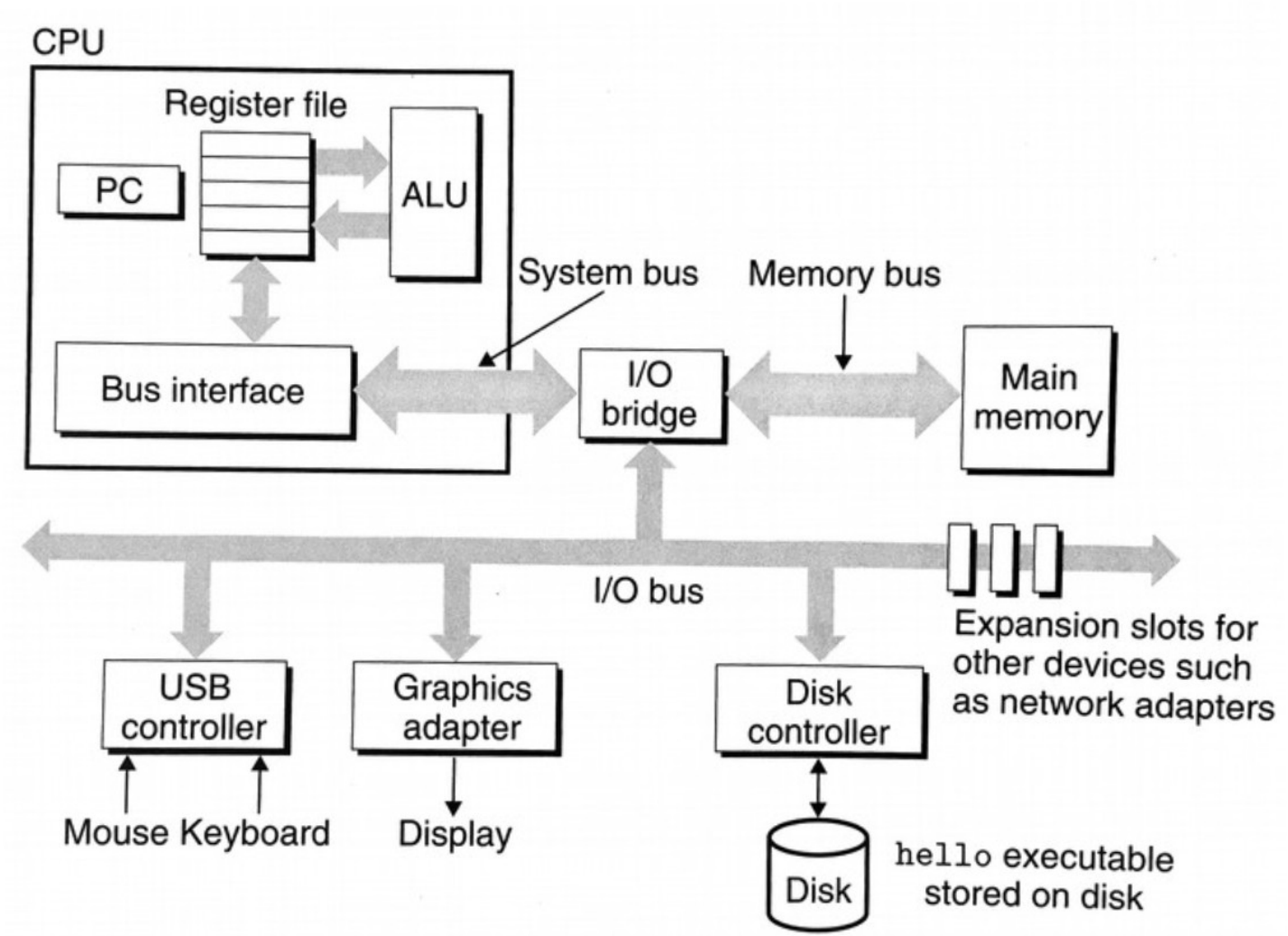
# Caveat

- Software system vs system software
  - Former: interconnected software components
  - Latter: software providing services to other software
  - We are concerned with both!
    - Example: operating systems, compilers, distributed systems

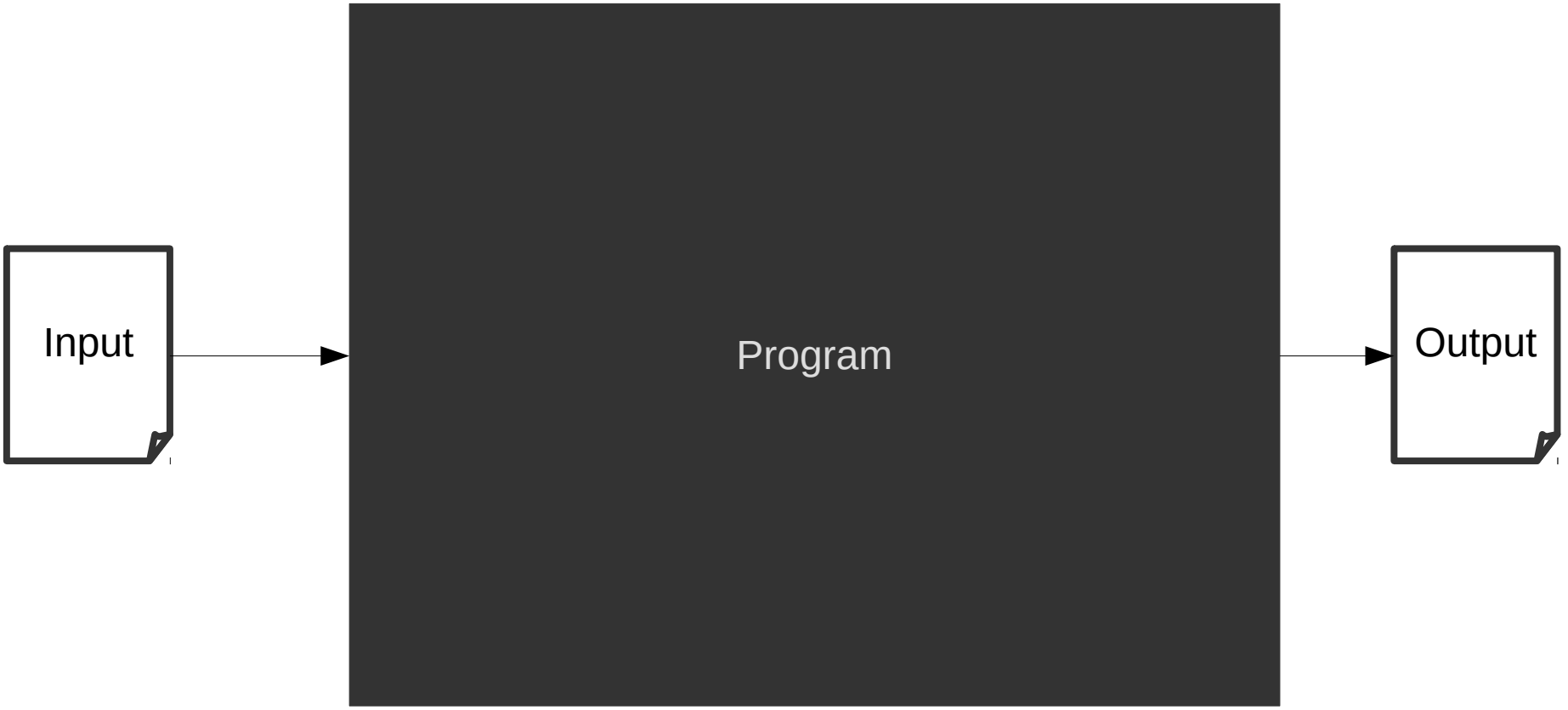
# Example: Computer



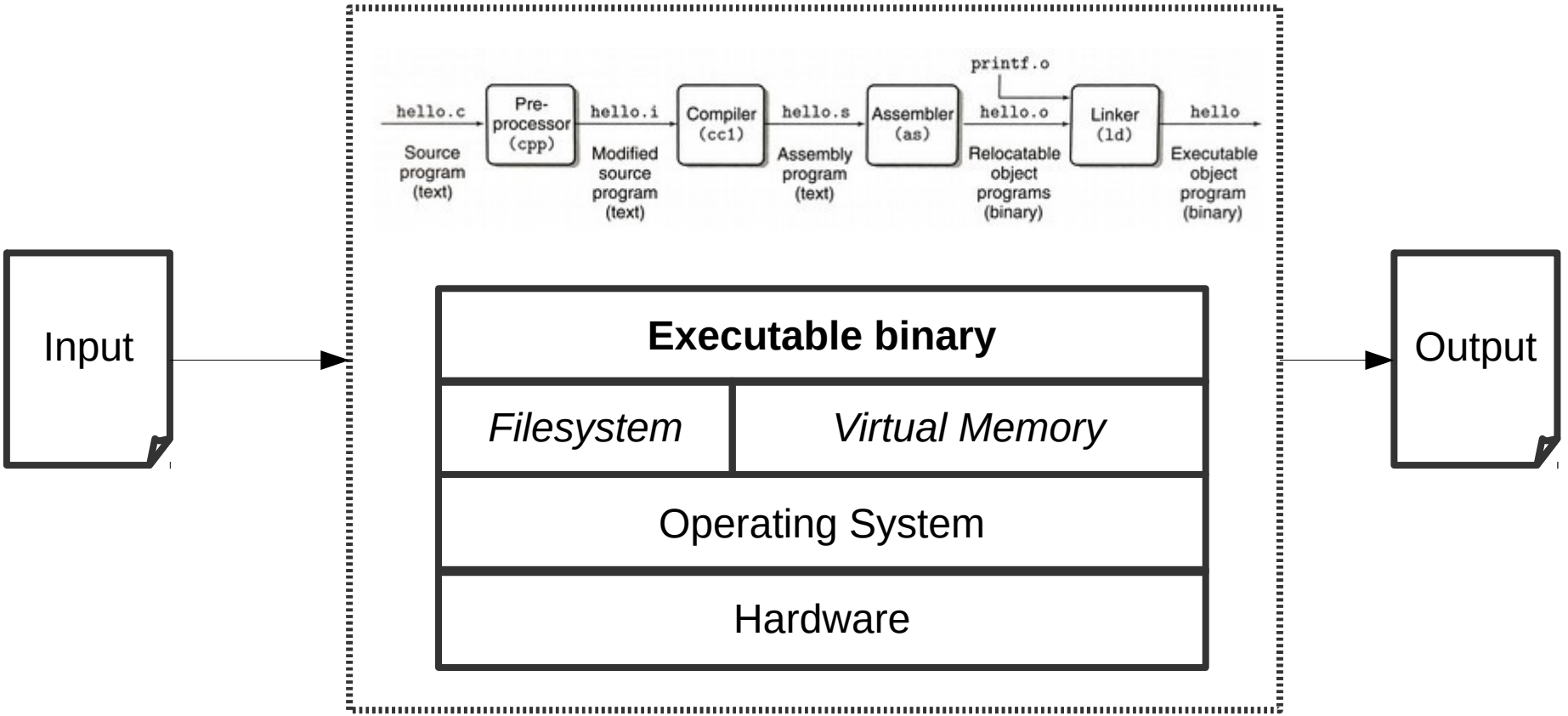
# Example: Computer



# Example: Program



# Example: Program



# Example: Memory

```
int age;

double temps[500];
```

# Example: Memory

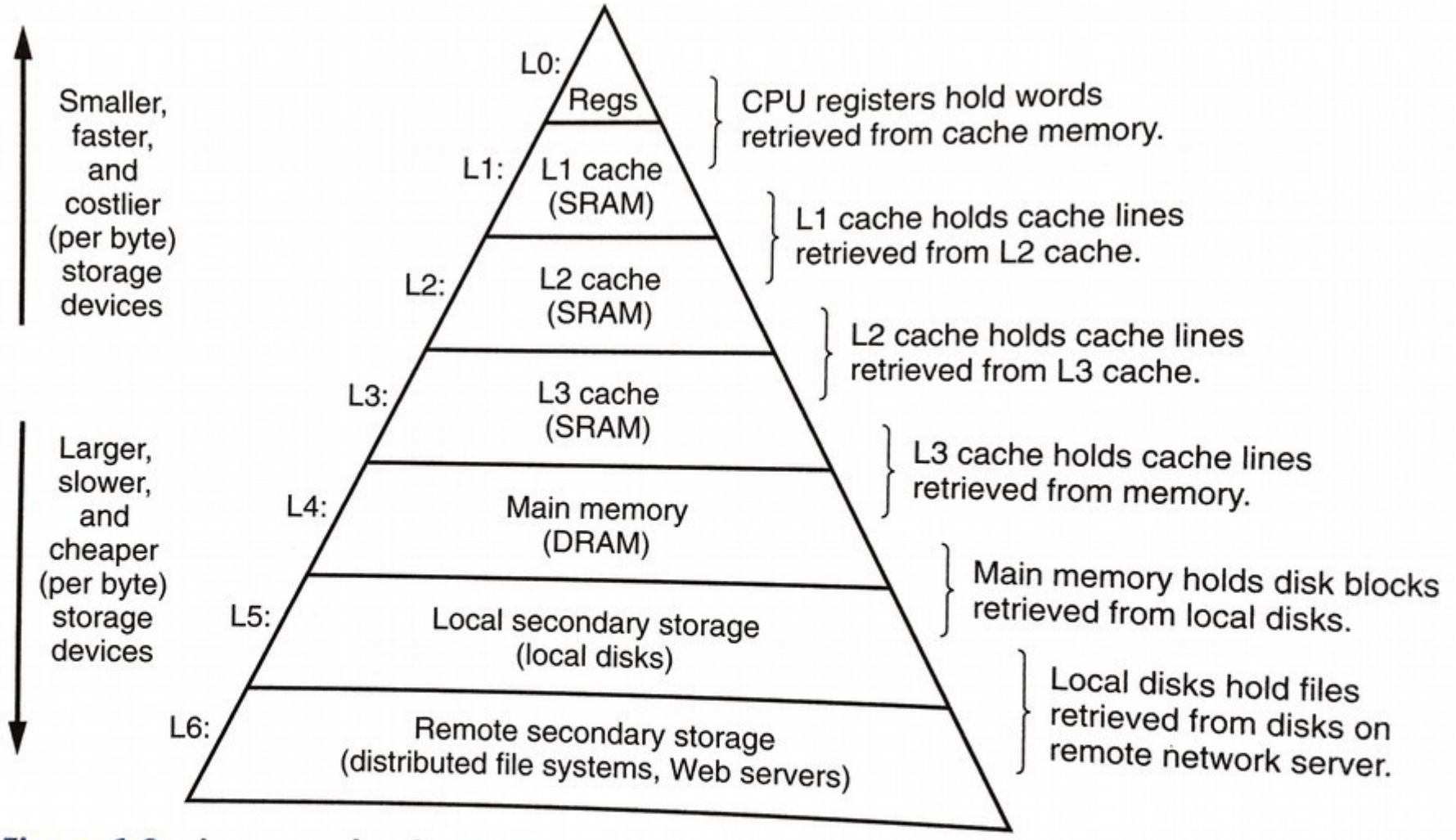



**Figure 1.9** **An example of a memory hierarchy.**

# Systems Themes

- **Information = bits + context**
- System design involves tradeoffs
- Systems are foundational platforms
- Digital models of an analog world
- Computer as other
- From abstract specifications to complete implementations
- Use the right tool for the job

# Example: Data

- What does “$2^{10}$” mean?

Systems Theme:
**Information = Bits + Context**

# Example: Data

- What does “$2^{10}$” mean?

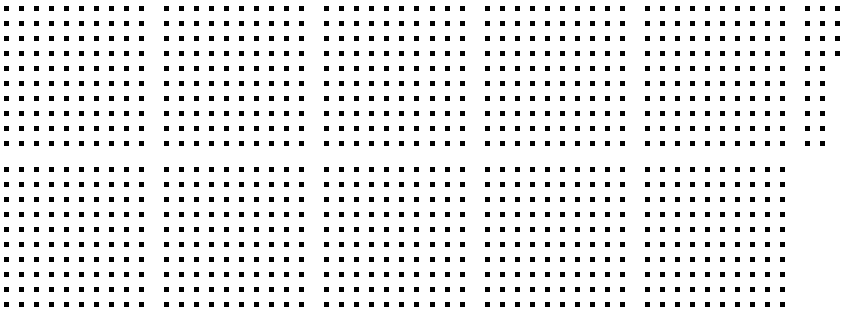
1<<10

1024

0x400

0100 0000 0000

Systems Theme:
**Information = Bits + Context**

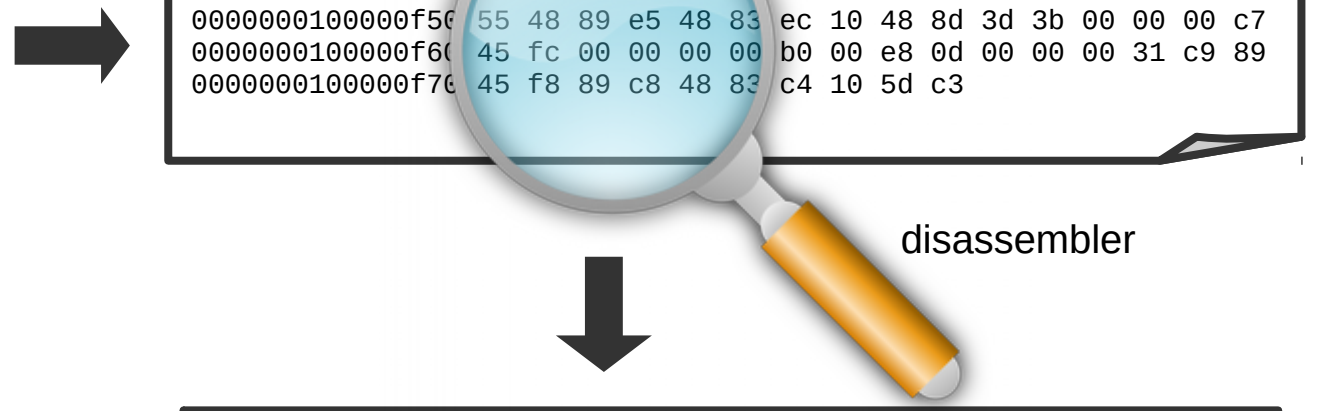


# Example: Program

```
#include <stdio.h>

int main()
{
    printf("Hello, world!\n");
    return 0;
}
```

compiler




disassembler

```
_main:
0000000100000f50            pushq       %rbp
0000000100000f51            movq        %rsp, %rbp
0000000100000f54            subq        $0x10, %rsp
0000000100000f58            leaq        0x3b(%rip), %rdi
0000000100000f5f            movl        $0x0, -0x4(%rbp)
0000000100000f66            movb        $0x0, %al
0000000100000f68            callq       0x100000f7a
0000000100000f6d            xorl        %ecx, %ecx
0000000100000f6f            movl        %eax, -0x8(%rbp)
0000000100000f72            movl        %ecx, %eax
0000000100000f74            addq        $0x10, %rsp
0000000100000f78            popq        %rbp
0000000100000f79            retq
```

Systems Theme:

**Information = Bits + Context**

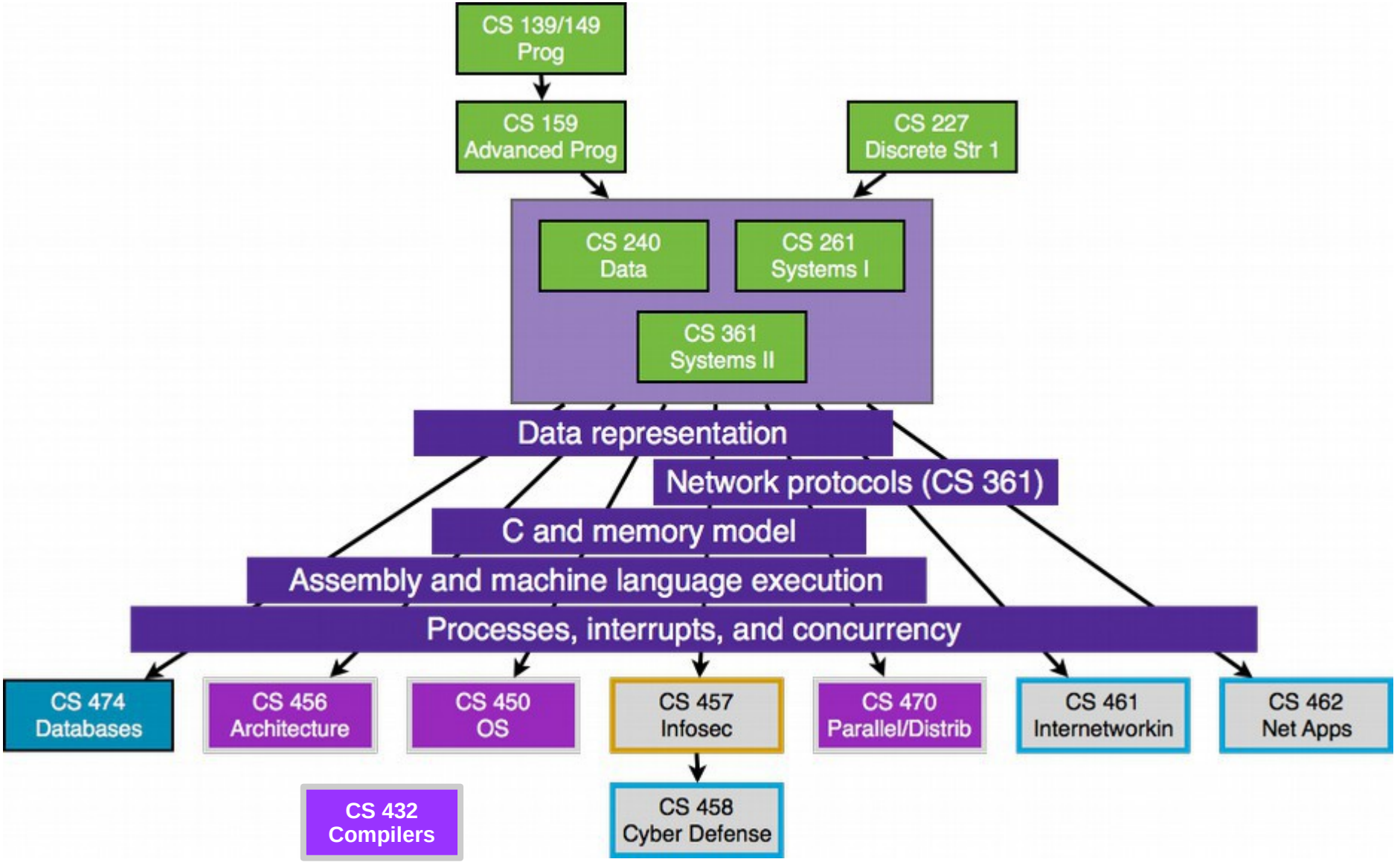
# Systems

- Why should you care?

# Systems

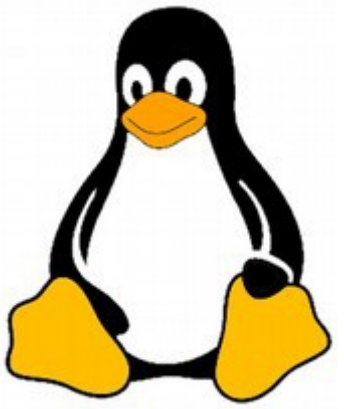
- Why should you care?
  - Sometimes $x^2 < 0$
  - Sometimes $(x + y) + z \neq x + (y + z)$
  - Machine code is important for optimization, debugging, and security
  - Memory is not uniform in performance
  - Black boxes should concern/intrigue you!

# JMU CS Curriculum



# CS 261

- What this course is NOT:
  - Programming 101 – I will assume you can program
    - However, we will spend a few weeks learning C
  - Electronics 101 – we won't be going THAT deep
    - If you're interested, check out PHYS 240/250
  - Linux 101 – but you have the Unix Users Group
    - InstallFest on Wed, Sep 7 at 7pm in ISAT/CS 259



# CS 261

- This is not an **easy** course
  - Especially if you haven't "tinkered" with a Unix-based computer system before
  - Be prepared to **read** a lot and **work** a lot
  - Learn the **why** and not just the **what**
  - Some stuff is worth memorizing (e.g., powers of two and hex characters)
  - For other stuff, **Google** is your friend
  - **Piazza** is also your friend (literally)
  - Start assignments early and ask questions

# Syllabus & Course Website

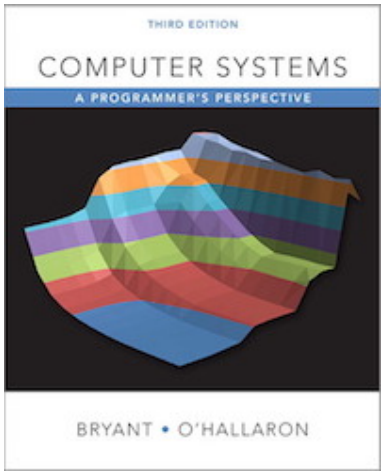
- This is the main course website:
  - `http://w3.cs.jmu.edu/lam2mo/cs261/`
  - Lots of useful stuff:
  - Syllabus (Read it! Especially the parts marked in **red**)
  - Calendar (Will be updated regularly)
  - Assignments
  - Resources
- Bookmark it and check back often

# Online Systems

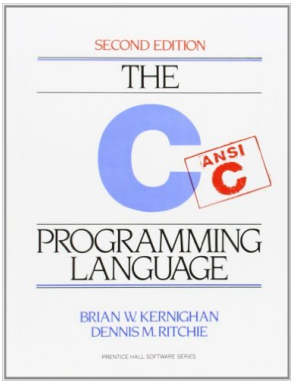
- `student.cs.jmu.edu` (“stu”)
  - Log in with your eID and password
  - Project development and debugging
- Canvas
  - Grades and online quizzes
  - Project submissions
  - Private files (i.e., solutions)
- Piazza (accessed via Canvas)
  - Q&A and discussions
- Make sure you can access all of these!

# Textbook(s)

- Required textbook: “Computer Systems”
  - “CS:APP” textbook from Carnegie-Mellon




- Recommended book: "The C Programming Language"
  - Brian Kernighan and **Dennis Ritchie** (creator of C)
  - Available on Safari Books through the library



# Course Policies

- Check Canvas daily for reading quizzes
- Class attendance is **strongly** recommended
  - If the class periods are not worth attending, tell me so that I can make them better!
- Slides will be posted on the website
- Please silence your cell phones during class
  - Be respectful with laptop and tablet usage

# Course Policies

- Submit programming projects as specified in the project description
  - No thumb drives, CDs, or emails (unless requested)
- Project grading will be based on automated test results
  - You will not have access to all test code
  - Some portion of the grade will be granted for style and documentation based on a manual inspection of the code
- Late submissions up to 72 hours will receive a 10% penalty per 24 hr period

# Course Policies

- The JMU Honor Code applies on ALL assignments
  - I **will** use software to detect plagiarism
  - Violations may be sent to the honor council
  - See relevant section in the syllabus
- All submitted code must be YOUR work entirely
  - You may work in groups to discuss assignments (in fact, I encourage this), but do NOT share code!

# Course Grades

| | |
|---|---|
| Quizzes and Activities | 25% |
| Programming Projects | 40% |
| Midterm Exams | 20% |
| Final Exam | 15% |

# Course Policies

- Exams will be held in ISAT/CS 246
  - Final exam times are on the website
- If you ask for a re-grade, I may re-grade the entire assignment
  - This applies to homework and projects, too
- If you have to miss a due date or exam because of an excused absence, let me know ASAP

# Have a great semester!

- Please take the intro survey on Canvas
- Make sure you can log into stu
- Start reading Chapter 1
- See you on Wednesday!