CS 261
Fall 2021

Mike Lam, Professor

https://xkcd.com/676/

Computer Systems I: Introduction

Question

- What will be the output of this C program?

```
#include <stdio.h>
int main() {
    int x = 40000;
    int y = 50000;
    if ((x * x) < (y * y)) {
        printf("Less than\n");
    } else {
        printf("Not less than\n");
    }
    return 0;
}
```

- A) “Less than”
- B) “Not less than”
- C) Neither of the above

Question

- What will be the output of this C program?

```
#include <stdio.h>
int main() {
    double a = 1e20;
    double b = -a;
    double c = 3.14;
    if (((a+b) + c) == (a + (b+c))) {
        printf("Equal!\n");
    } else {
        printf("Not equal!\n");
    }
    return 0;
}
```

- A) “Equal!”
- B) “Not equal!”
- C) Neither of the above

Question

- Which of the following versions of a “matrix copy” routine will run the fastest?
  - A)
```
for (int i = 0; i < 2048; i++) {
    for (int j = 0; j < 2048; j++) {
        dst[i][j] = src[i][j];
    }
}
```
  - B)
```
for (int j = 0; j < 2048; j++) {
    for (int i = 0; i < 2048; i++) {
        dst[i][j] = src[i][j];
    }
}
```
  - C) Neither; they will always run at approximately the same speed.

What's happening?

- Something about our **mental model** of these programs does not match the **system** on which we're running them.

Systems

- What is a “system?”

Systems

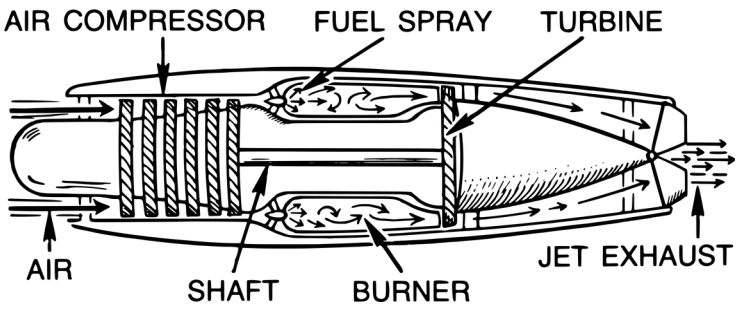
- What is a “system?”
  - Set of interacting components
  - More than the sum of its parts

Jet engine

Computer

Systems

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

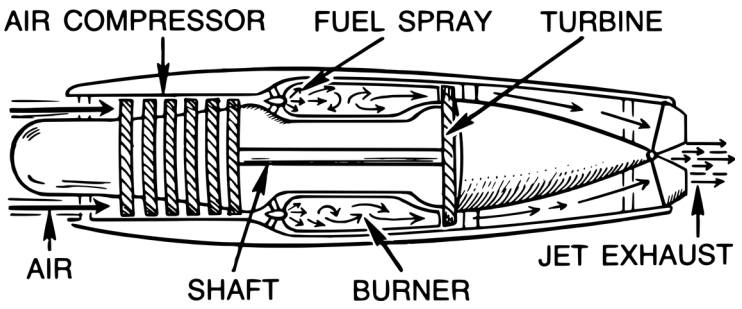


Jet engine

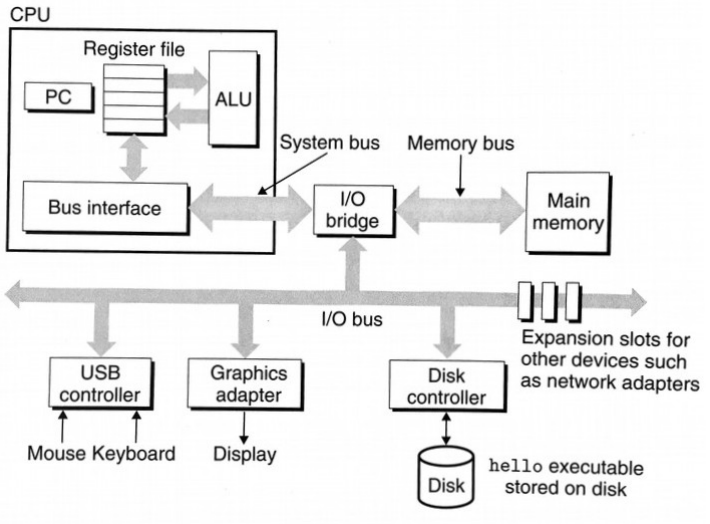
Computer

Systems

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



Jet engine



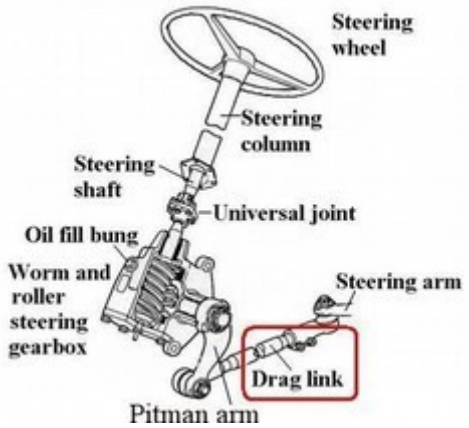
Computer

Systems

- A **computer system** consists of multiple hardware and software components that work together to run user applications.
  - We use complex computer systems every day
  - Our goal: peel back (some of) the complexity
    - See (some of) what's "under the hood"

Systems

- What is a *process*? What is a *file*?
  - These are examples of abstraction; "fake" views of reality that reduce complexity for users
  - Key ideas: **ignore details** and **focus on interfaces**
  - Especially important in large, complicated systems
  - Understanding abstractions can improve your ability to use them effectively



abstraction

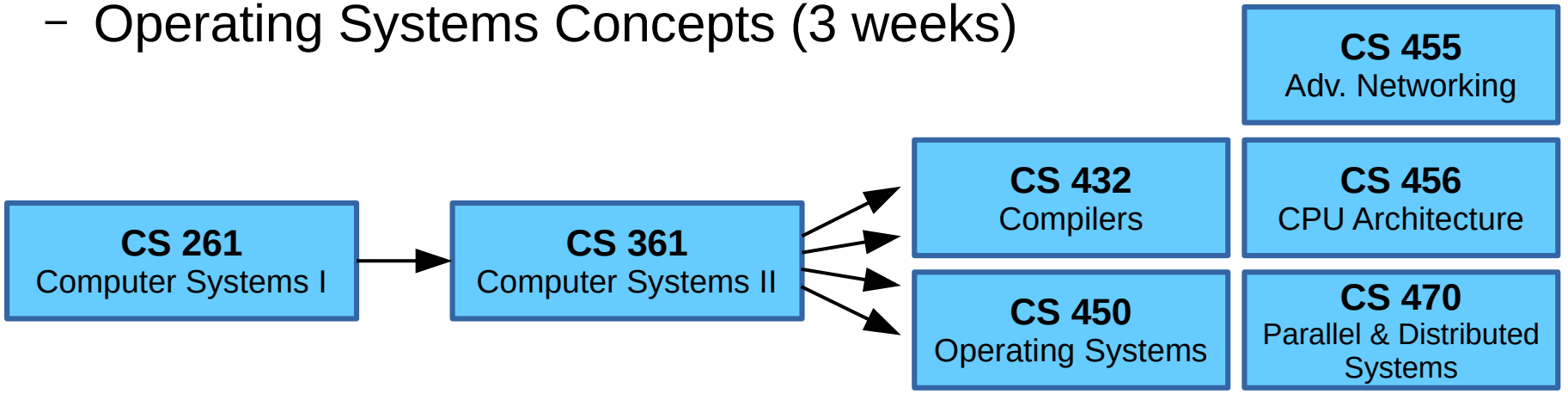
Course Objectives

- Explain machine-level representation of data and code
- Summarize the architecture of a computer
- Explain how complex systems are built from simple components
- Translate high-level code into assembly and machine language
- Write code to emulate the functionality of a computer

- Cultivate a sense of control over computer systems
- Gain an appreciation for software development tools
- Develop a sense of play when writing code
- Appreciate the complexity of systems-level software

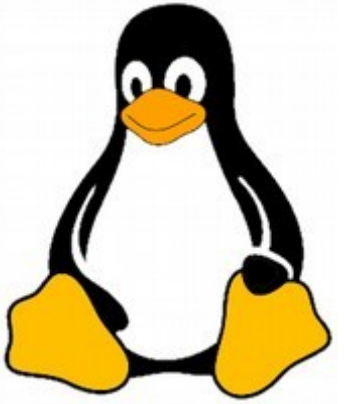
Systems courses

- CS 261 units:
  - C and Linux (3 weeks)
  - Binary Representations (2-3 weeks)
  - Assembly and Machine Code (2-3 weeks)
  - Computer Architecture (3 weeks)
  - Operating Systems Concepts (3 weeks)



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 and weekly meetings
    - https://www.jmunixusers.org



CS 261

- This is not an "**easy**" course
  - *But you **can** handle it!*
  - Be prepared to **read** and **work** a lot
  - Don't be afraid to experiment
  - Growth vs. fixed mindset
  - Take advantage of office hours and Piazza
  - Start assignments **early** and ask questions

Semester-specific info

- The remaining slides are specific to Fall 2021
  - All slides are posted on the website (calendar page)

- This is likely to be a challenging semester
  - I pledge to extend extra grace and patience
  - I pledge to keep the classroom as safe as possible
  - I ask that you do the same

Class Policies

- Masks must be worn in class
  - Must cover both your mouth and nose
  - *(I have spare disposable masks if you need one)*
- No food or drink is allowed
  - *(Quick sips are ok w/ me – stay hydrated!)*
- Recommended: sanitize your table before and after class
  - Spray sanitizer available at front podium
- If you are ill, please stay home
  - Contact me ASAP regarding missed class
- These policies may change
  - Changes will be announced via Canvas message

Course Design

- This is a **flipped class** (except for today)
  - Ahead of time: watch lecture, do reading, take quiz
  - During class: work on labs in small groups
  - Outside class: work on projects, take module tests

	Monday	Tuesday	Wednesday	Thursday	Friday
In-class		Lab		Lab	
Out-of-class	Lecture videos, reading, and quiz		Lecture videos, reading, and quiz		
	Project work	Project work	Project work	Project work	Project work (deadlines every 2-3 weeks)

Video playlists, quizzes, and labs all have a common tag (today's is "01")

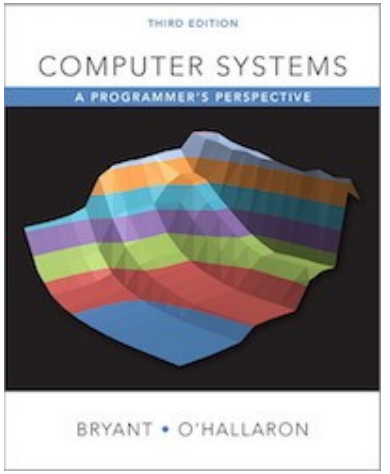
Course Components

- **Public website** (`w3.cs.jmu.edu/lam2mo/cs261`)
  - Syllabus, **calendar**, project descriptions, and resources
  - Links to lecture videos (YouTube, already posted)
    - Most recorded for Fall 2020; all still relevant this year
  - Links to slides (posted before class, may differ slightly from videos)
- **Canvas course**
  - Quizzes, lab submissions, and module tests
  - Grades and private files (e.g., lab solutions)
  - Access to Piazza Q&A
- **Student server** (`stu.cs.jmu.edu`)
  - Project development and submission
- **Piazza**
  - Q&A (especially re: projects)

Make sure you can access ALL of these!

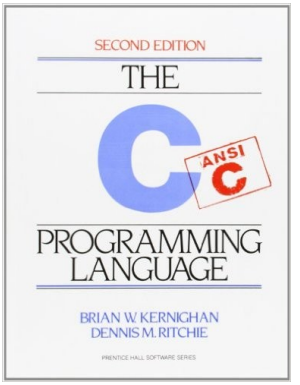
Textbooks

- Required textbook: “Computer Systems”
  - “CS:APP” textbook from Carnegie-Mellon
  - A practical, example-filled introduction
  - Electronic rental available via RedShelf
  - Reserve copy at the Rose library (*)



Important: Readings are listed on their associated quiz

- Recommended book: "The C Programming Language"
  - Brian Kernighan and Dennis Ritchie (creator of C)
  - This is “the book” about C (we’ll refer to it as “CPL”)
  - Scanned excerpts on Canvas (do not redistribute!)



(*) work in progress

Course Grades

Quizzes and Labs	25%
Programming Projects	30%
Unit Tests	25%
Exams	20%

- Quizzes and labs are **formative**
  - Designed to help you learn
- Tests/exams are **summative**
  - Designed to assess what you have learned
- Projects are **both**
  - Designed to help you learn C and reinforce other course concepts
  - Also designed to assess whether you are ready for CS 361

Class Policies

- Class attendance is necessary and expected
  - We will be completing labs most class periods
  - Find a group (2 or 4 people) to work with consistently
  - Make a name card and bring it every day
- Every person should fill out a separate copy of the lab
  - Work together and check each other
  - Ask for help if you are stuck or want to confirm something

Class Policies

- Submit as PDF on Canvas when done
  - **Scan as a black-and-white PDF**
  - Recommended apps: Scanner Pro, Scannable, Genius Scan
  - Other instructions: https://wiki.cs.jmu.edu/student/canvas/start
  - You may NOT submit raw photos

- Labs are “lightly graded” (w/o individual mistakes marked)
  - Solutions will be posted on Canvas (under Files → Lab Solutions)
  - Come to office hours if you wish to discuss individual questions

Course Policies

- The projects in this course are VERY important!
  - One purpose of this course is to ensure you are ready to tackle harder projects in CS 361 and the system electives

- Projects are **individual** and **mandatory**
  - A “good faith” submission shows evidence of significant work and investment in writing a solution
  - A “good faith” submission gets you an “F” (50 or 60 points) instead of a zero!

Course Policies

- The JMU Honor Code applies on ALL assignments
  - Violations may be sent to the honor council
  - See relevant section in the syllabus
  - All quizzes, module tests, and exams must be done by yourself with no assistance aside from what is allowed in the assignment description in Canvas
- All submitted labs must represent YOUR work
  - You will work in groups to discuss the answers
  - By submitting a PDF on Canvas, you are asserting that these answers are YOUR answers and that you understand WHY you have answered the way you have

Course Policies

- All submitted project code must be YOUR work entirely
  - You may work in groups to discuss general approaches (in fact, I encourage this; use *pseudocode* if necessary)
  - However, one goal of the projects in this course is to develop individual competency, so **you may NOT share code** with anyone who is not a TA or CS 261 instructor
  - This includes letting someone examine or take a screenshot of your code, or “talking it through” with them line-by-line
  - Do not store your solution in a public repository
  - If you have questions about this, please ask!

Course Policies

- There are a total of three sections of CS 261
  - Two Lam sections (T-Th) and one Weikle section (M-W)
  - Many course materials are shared
  - You are welcome to study with students from other sections, but you must attend and submit assignments to the section you are registered for

Office hours

- My office hours: **Monday through Friday, 10:00-11:00am**
  - In person: ISAT/CS 227
    - Please avoid congregating in large groups in the hallways!
    - If I'm unavailable when you arrive in person, use the bit.ly link below
    - You'll join the same queue as virtual attendees and I will call you when I'm available (you'll leave a cell number as part of sign-up)
  - Virtual via Zoom: bit.ly/lam-office-hours-f21
    - This is **strongly preferred** for coding questions
    - Be prepared to share your screen!
  - Outside office hours via appointment: calendly.com/lam2mo
- CS TAs: in-person and virtual office hours: bit.ly/CS-TAs
  - 261-specific TAs: Ryan Showalter and Kyle LaCanna

TODOs in the next few days

- If you haven't already:
  - **Take welcome survey on Canvas**
  - **Take syllabus quiz on Canvas**
  - **Read CS:APP Ch. 1 and take Quiz 01** (due tomorrow)

- Before class next Tuesday:
  - Review these slides and the syllabus and come with questions
  - **Watch "Command line and C compilation" lecture videos**
  - **Read 02-CPL excerpts** (on Canvas under Files → Readings)
  - **Take Quiz 02** (due Monday)
  - **Make sure you can log into `stu`**
    - Instructions at the top of Tuesday's lab: w3.cs.jmu.edu/lam2mo/cs261/02-cmd_line.html
  - Make sure you can access Piazza
  - Skim the project guide and Project 0 description (on website)

Intro lab

- Material from Chapter 1
  - Front page: **Computer Organization**
  - Back page: **C Compilation**

- Submit as PDF on Canvas when done
  - Scan as a black-and-white PDF
  - Recommended apps: Scanner Pro, Scannable, Genius Scan
  - Other instructions: https://wiki.cs.jmu.edu/student/canvas/start
  - You may NOT submit raw photos
  - Let me know after you submit and I will check it on my end
  - Once you have verified a satisfactory submission, please feel free to leave – have a great weekend and I'll see you next Tuesday!