CSC 252: Computer Organization
Spring 2025: Lecture 1

Instructor: Yuhao Zhu

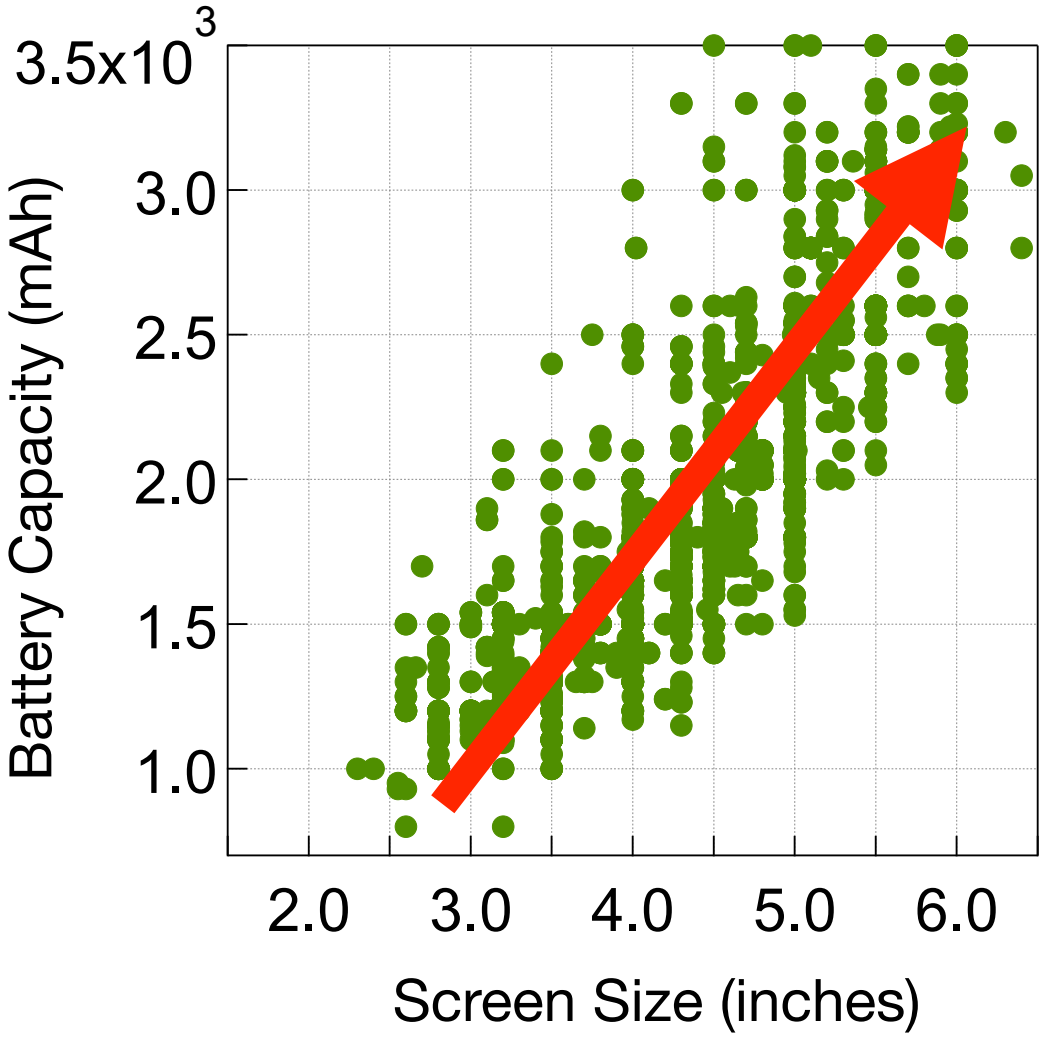
Department of Computer Science
University of Rochester

Action Items:

- **Get CSUG account**
- **Make sure you have VPN setup!!!!**

Computers are More Capable

“Improving” Energy Capacity



600 smartphone from 2006 to 2014 on http://www.gsmarena.com/makers.php3

Which Future Do You Want?

How to build ever-more capable computers but with lower energy consumption and smaller form factor?

Outline: Class Introduction

- Introduction
  - What Are You Supposed to Learn in this Class?
    - What Is Computer Organization Anyways?
  - Instructor & TAs
  - What Do I Expect From You?
  - How am I Going to Teach?
  - Grading, Policies

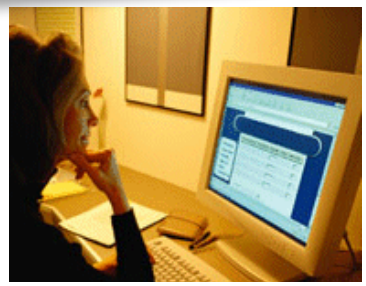
Action Items

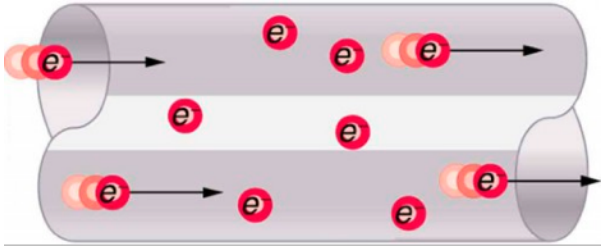
- Get a CSUG account.
  - cycle1.csug.rochester.edu (or cycle2, cycle3)
  - If you don't already have one, go to this link: https://accounts.csug.rochester.edu/
  - **YOU WILL NEED VPN to access these machines if you are not using campus WiFI!! Follow the instructions (https://tech.rochester.edu/remote-access-vpn-tutorials/) to set up the university VPN.**
  - **TAs will help with VPN setup too.**

Where to Find Stuff

- http://cs.rochester.edu/courses/252/spring2025/
  - General info
  - Programming assignments details
  - Slides
  - Practice problems, past exams
- Communication through Blackboard (https://learn.rochester.edu/)
- CSUG machines for programming assignments submissions

Computer Systems Match User Requirements to Hardware Technologies





Problem	Who scores the highest on the exam?
Algorithm	Quicksort
Program	Human-readable language (Java, C)
Instruction Set Architecture	Machine Language
Microarchitecture	Hardware Design
Circuit	Electrons, Resistors, Capacitors, etc.

Two Fundamental Aspects of Computer Systems

- How is a human-readable program translated to a representation that computers can understand?
- How does a modern computer execute that program?

Problem	Who scores the highest on the exam?
Algorithm	Quicksort
Program	Human-readable language (Java, C)
Instruction Set Architecture	Machine Language
Microarchitecture	Hardware Design
Circuit	Electrons, Resistors, Capacitors, etc.

The “Translation” Process, a.k.a., Compilation

C Program

```
void add() {
  int a = 1;
  int b = 2;
  int c = a + b;
}
```

Pre-processor
Compiler

Assembly program

```
movl    $1, -4(%rbp)
movl    $2, -8(%rbp)
movl    -4(%rbp), %eax
addl    -8(%rbp), %eax
```

The "Translation" Process, a.k.a., Compilation

Assembly program

```
movl    $1, -4(%rbp)
movl    $2, -8(%rbp)
movl    -4(%rbp), %eax
addl    -8(%rbp), %eax
```

Executable Binary

```
00011001   …
01101010   …
11010101   …
01110001   …
```

- It translates a text file to an executable binary file (a.k.a., executable) consisting of a sequence of instructions
- Why binary? Computers understand only 0s and 1s
  - The subject of next lecture

Back to Layers of Transformation…

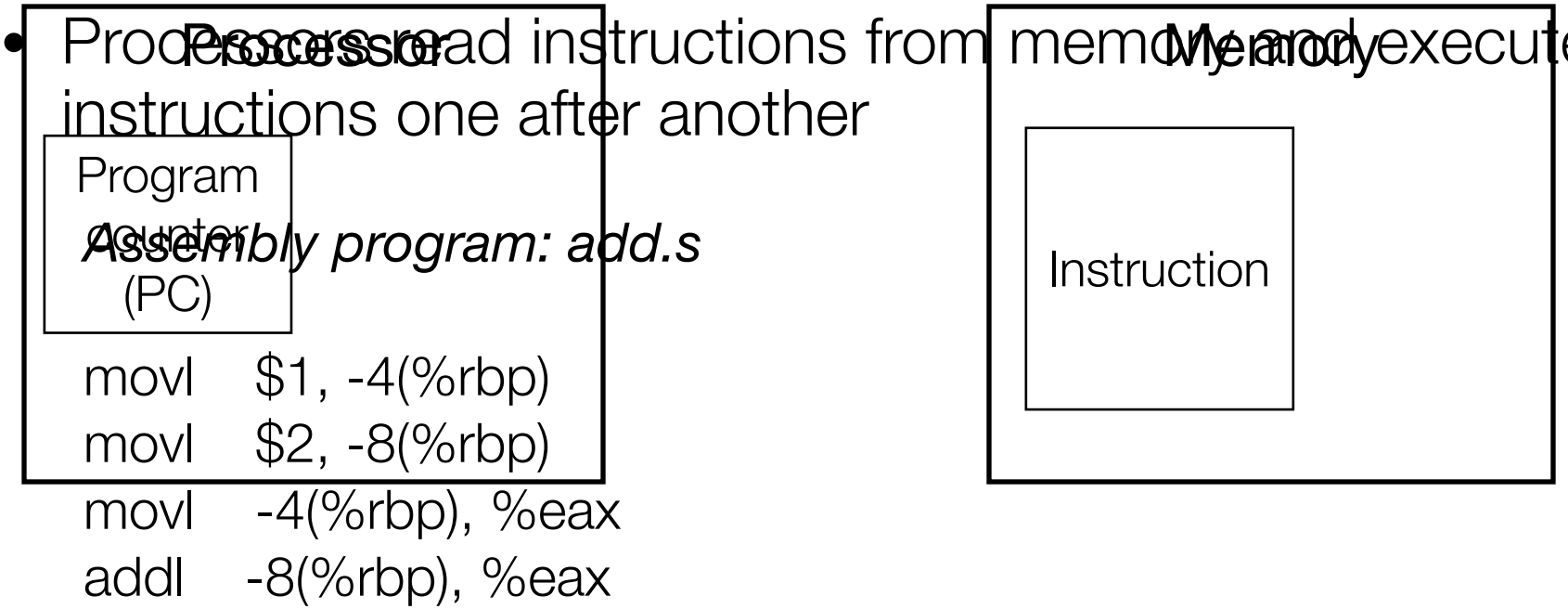
How is a human-readable program translated to a representation that computers can understand?

How does a modern computer execute that program?

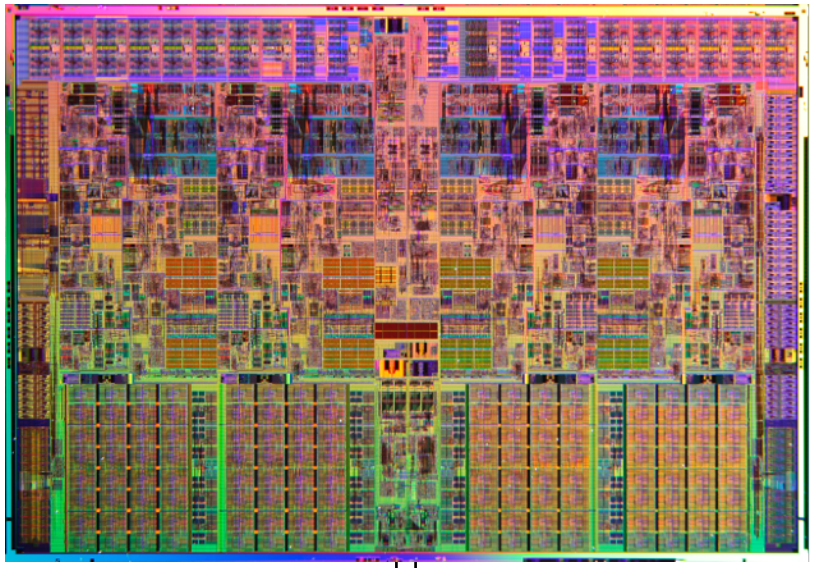
Problem	Who scores the highest on the exam?
Algorithm	Quicksort
Program	Human-readable language (Java, C)
Instruction Set Architecture	Machine Language
Microarchitecture	Hardware Design
Circuit	Electrons, Resistors, Capacitors, etc.

The Single Most Important Idea of Computers

- Executables (i.e., instructions) are stored in “memory”
- Processors read instructions from memory and execute instructions one after another



High-level Organization of Computer Hardware a.k.a., The Von Neumann Model



Memory

Bus

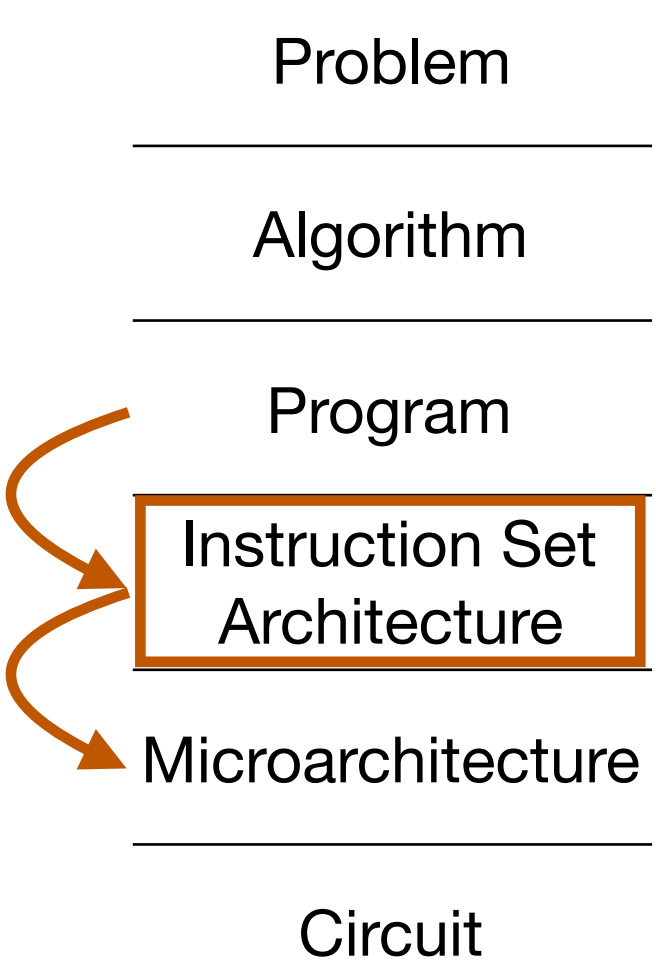
Input / Output Device

Touchscreen | Disk | Sensor | Ethernet | Camera

Back to Layers of Transformation…

How is a human-readable program translated to a representation that computers can understand?

How does a modern computer execute that program?



Who scores the highest on the exam?

Quicksort

Human-readable language (Java, C)

Machine Language

Hardware Design

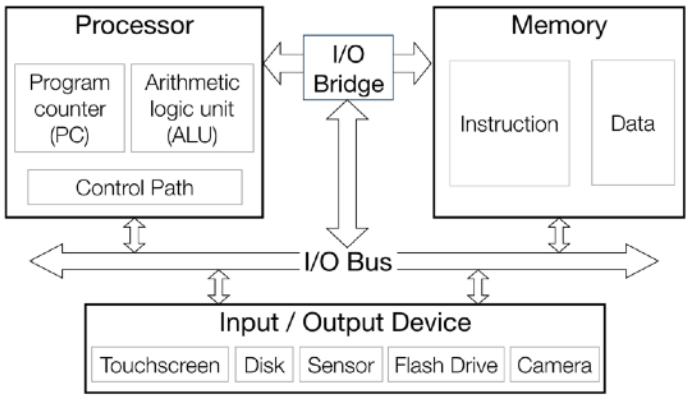
Electrons, Resistors, Capacitors, etc.

Instruction Set Architecture

- The programmer's view of the computer is called the "instruction set architecture" (*ISA*)
- For programmer: no need to care how the instructions are implemented as long as they are somehow implemented
- Implementation of an ISA is called *microarchitecture*
- ISAs *abstract* away details of microarchitecture

Assembly program: add.s

```
movl    $1, -4(%rbp)
movl    $2, -8(%rbp)
movl    -4(%rbp), %eax
addl    -8(%rbp), %eax
…
callq   _printf
```

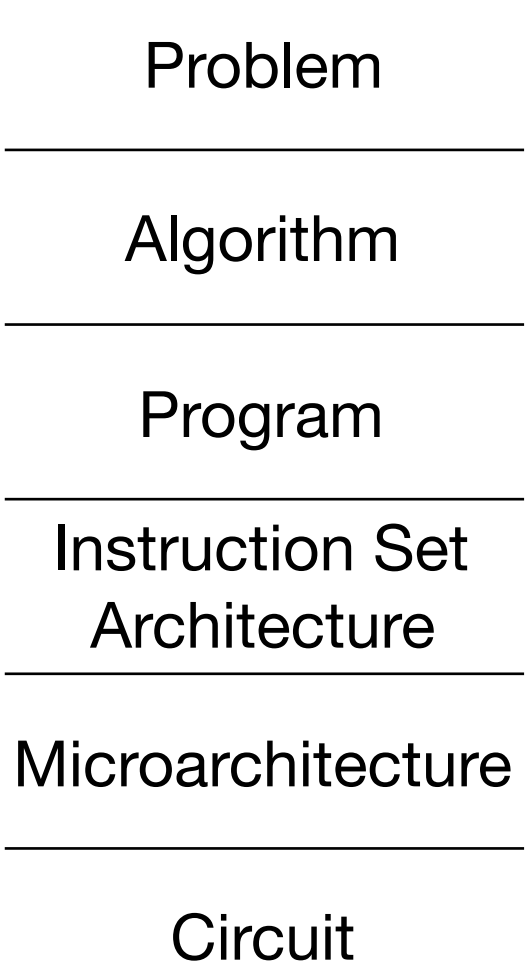


Abstraction

- Think of car versus engine, transmission, brakes, …
- Leaving out one or more properties of a complex object so as to focus on others
  - ISA leaves out *how* “ADD” is implemented
  - ISA also leaves our *how long* an “ADD” instruction takes
- Bad abstractions throw away essential features of problem
  - Topologist is someone who does not know the difference between a doughnut and a coffee cup
  - Bad ISAs don’t tell you whether the hardware can do multiplication

Every Layer in CS is an Abstraction

- Depend on which layer you want to live at, you have different views of the computer
- This course expands your layers of abstractions



Who scores the highest on the exam?

Quicksort

Human-readable language (Java, C)

Machine Language

Hardware Design

Electrons, Resistors, Capacitors, etc.

Instruction Set Architecture

- There used to be many ISAs
  - x86, ARM, PC, Spar, z
  - Very cons ARM for others
- There are microarc
  - Apple/Sa mm have architecture (impleme RM ISA
  - Intel and rent micro r x86
- ISA is lucr ss: ARM Model
  - Patent th license t
  - Every imp a royalty
  - Apple/Sa RM whene smartphone

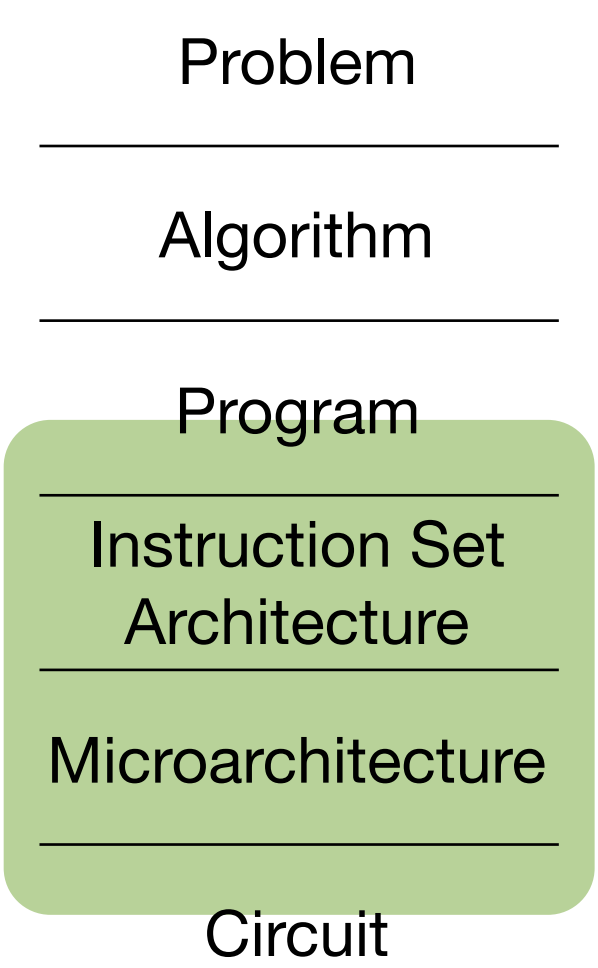
Instruction Set Architecture

- Little research on ISA, much more microarch. research
  - ISA is stable now. “One ISA rules them all.”
  - Free, open ISA: RISC V (https://riscv.org/)

  - Instead, focus on optimizing the implementation.
- Interesting question: can we have one microarchitecture (implementation) for different ISAs?
  - Can a microarchitecture designed for ISA ***X*** execute ISA ***Y***?
  - Yes but you either need something that translates programs written in ISA ***Y*** to ISA ***X*** while you are executing it: *dynamic binary translator*
    - E.g., Transmeta executes x86 ISA programs on their own ISA
  - Or you need to implement the superset

The Role of a Computer System Designer

- Look Up (Nature of the problems)
- Look Down (Nature of the circuit technology and physics)
- Look Backward (Evaluating old ideas in light of new technologies)
- Look Forward (Listen to dreamers and predict the future)



Questions?

Who Are We?

- Myself: Yuhao Zhu
  - WH 3501, yzhu@rochester.edu
  - Office hours Tuesday 4pm — 5pm.
  - Got a good education
  - Got some industry experience
  - Interested in computer systems for Augmented/Virtual Reality, Computational Phototography, etc.
- TAs: 2 Grads + 4 UGs
  - Their office hours times are available on the course website.
  - Did very well themselves in this course before
  - Really care about you learning the material and succeeding
- **Coming to office hours does NOT mean you are weak!**

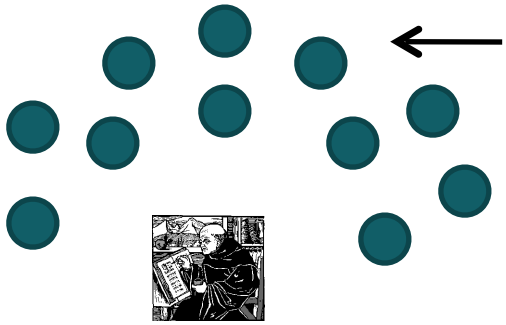
A Word about Lectures and Medieval Times

- Lecture: It's a large part of what you pay for

- But why do we have the "lecture" format?
  - Why does someone stand at the front and tell you things?
  - Why do you take "notes" on what they say?

All The Way Back to Medieval Times..

Lecture Halls



people

Modern Times

- You don’t have to trust the monk!
  - The printing press: a revolutionizing development
  - The web: order your knowledge up for yourself on Amazon!
- Read books and analyze for YOURSELF!
  - If I rephrase it for you, what purpose does that serve?

amazon.com

FAQ: "But, wouldn't it be more efficient if you just taught us with the right answer to begin with?"

- Have you ever heard of a workout class where the instructor did all the exercises while everyone else just watched attentively?
- To learn, you must do the work with your own muscle (brain)!

Textbook

- Required textbook
  - Bryant and O'Hallaron's *Computer Systems: A Programmer's Perspective* (3rd edition)
- Some recommended (but not required) textbooks
  - *Introduction to Computing Systems: From Bits and Gates to C and Beyond*, 2/e. This is where I learnt Computer Systems.
  - *Structured Computer Organization*, 6/e. More emphasis on SW.
  - *Computer Organization and Design: The Hardware Software Interface*, ARM Edition. More emphasis on hardware.

How Will You Be Evaluated?

- Programming Assignments: 40%
  - 5 assignments, 8% each

- 1 midterm exam, 25%
- 1 comprehensive final exam, 35%

Programming Assignments

- Check syllabus and labs to figure out when they are due (there is a date and a time specified)
- They take time, so start early!
  - Thinking/design time
  - Programming time
  - Test design + debug (and repeat)
- 3 slip days. Use it wisely!
  - Other than slip days, late submission counts 0 point
- You could work in pairs
  - Only 1 submission per pair
- Share ideas but not artifacts (e.g., code, sketch)

Programming Environment

- Develop code (or at least test it) on the CSUG Linux boxes (csug.rochester.edu)
  - Microsoft Visual Studio could be nice, but it's not what we use
  - The lack of Unix knowledge is a major problem according to our industry contacts
- Projects will be mostly in C and x86 assembly.
- We only accept ANSI-C that can be compiled by the default GCC on the CSUG Linux boxes

Exams

- Two exams: one in-class midterm and one final
  - Midterm covers everything up until the second last lecture
  - Final will cover everything, including materials before midterm
- No collaboration on exams
- "I don't know" is given 15% partial credit
  - You need to decide if guessing is worthwhile
  - Saves grading time
  - You have to write "I don't know" and cross out /erase anything else to get credit: A blank answer doesn't count
- All exams are open book (means your book won't help)
  - They will in fact probably **hurt**
  - **Memorization won't help. Thinking will.**

Loved how the final exam was set up. Actually made me THINK.

Programming Assignments and Exams

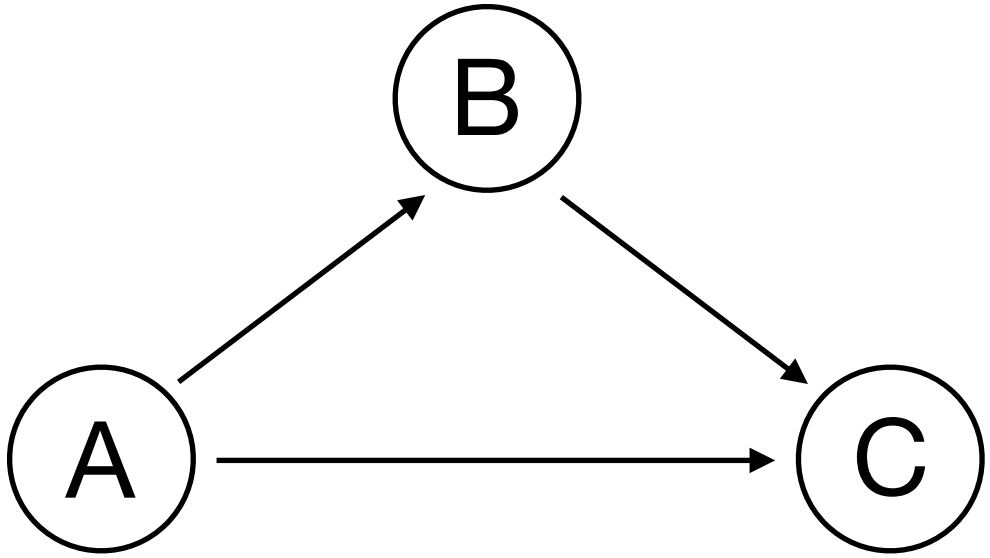
The assignments were often very different from what we were learning in the course, causing a lot of frustration in the beginning. People had to rely on outside sources to get a grounding of how to do an assignment.

The exams are structured to be significantly harder than the examples covered in class which means there is strong need for critical thinking on the spot during exams.

This is a feature, not a bug.

Here is how to think of the projects, exams, and lectures (loosely)…

- Lectures teach you how to go from **A** to **B**
- Projects ask you to go from **B** to **C**
- Exams test whether you can go from **A** to **C**



What Should You Expect From Us?

- *Think of us as your tutors*
  - Be your guide in inducing you to explore concepts
  - Create situations and post problems that set the scene for your exploration
  - Answer your questions
  - Not spend lecture reading the textbook to you with slightly different words
- *Think of us as your friends, not enemies*
  - I'd love to give you an A, but give me a reason to do that

If I am Traveling…

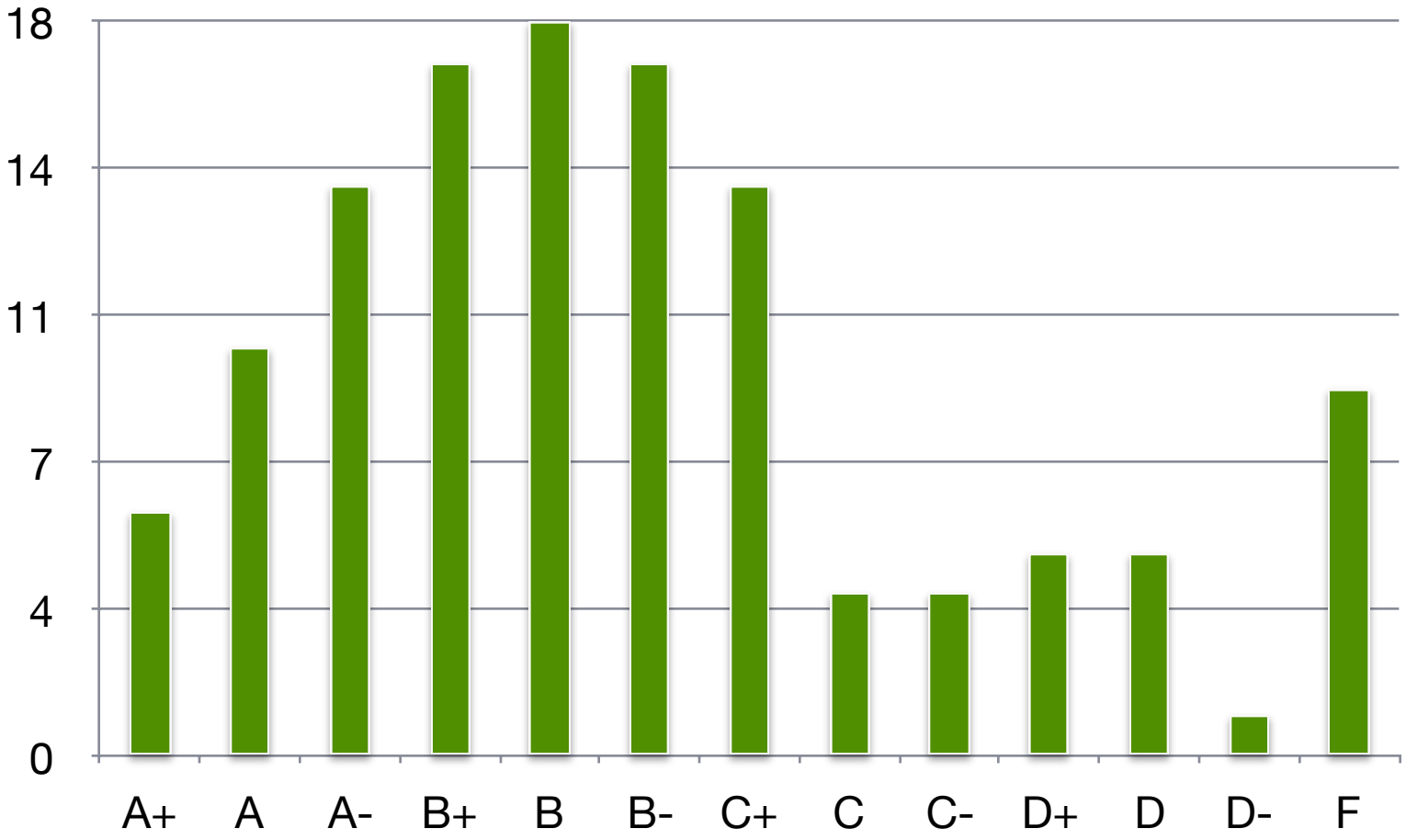
- I will try teaching over zoom; if impossible:
- I will try to get someone to teach for me; if impossible
- I will ask you to watch pre-recorded videos from 2021

- Canceling lectures is very unlikely
- 2021 recordings have been made available so that you can consume them offline anyways
- No zoom option otherwise
- I don’t take attendance

Grading

If your final score is at least	You will at least get a grade of
92	A
89	A-
87	B+
82	B
79	B-
77	C+
72	C
69	C-
60	D
0	E

Final Grades Spring 2018



Final Grades Spring 2019