CSC 252: Computer Organization
Fall 2021: Lecture 1

- Instructor: Alan Beadle

- Department of Computer Science
- University of Rochester

Action Items:

- **Get CSUG account**
- **Sign up for Blackboard**
- **Sign up for Piazza**

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

Note: This lecture is (hopefully) being recorded since there are students arriving late due to travel delays.

Another note: Please wear masks while indoors. This is a mandatory university-wide policy.
There is an exception for instructors who can remain 6 feet away

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.**

- Sign up for Blackboard (https://learn.rochester.edu/)
- Sign up for Piazza (link on blackboard)

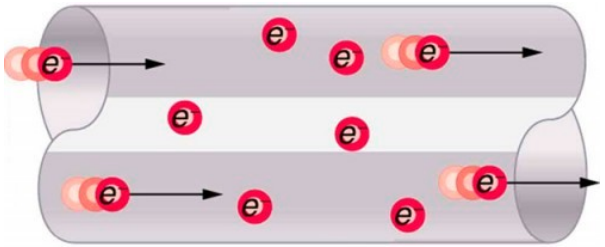
Where to Find Stuff

https://www.cs.rochester.edu/u/hbeadle/csc252/

- General info
- Programming assignments details
- Slides
- Link to Piazza forum
- Any other course materials or links

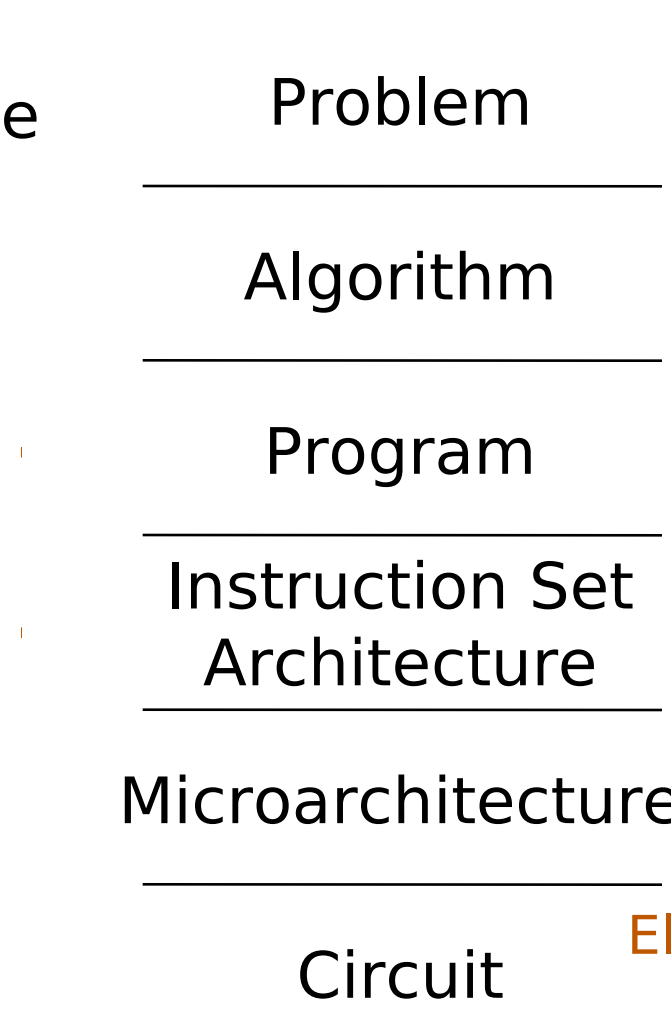
- CSUG machines for programming assignments submissions

What is Computer Organization?



Problem	Who scored 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.

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



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

- 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

Example: add.c

(Human-readable)

```
#include <stdio.h>
int main() {
  int a = 1;
  int b = 2;
  int c = a + b;
  printf(“%d\n”, c);
}
```

Executable Binary

(Machine-readable)

```
00011001   …
01101010   …
11010101   …
10100100   …
```

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
```

Assembler, Linker

Executable Binary

```
00011001   ...
01101010   ...
11010101   ...
01110001   ...
```

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

* *Actually there have been machines based on other number systems, but not many.*

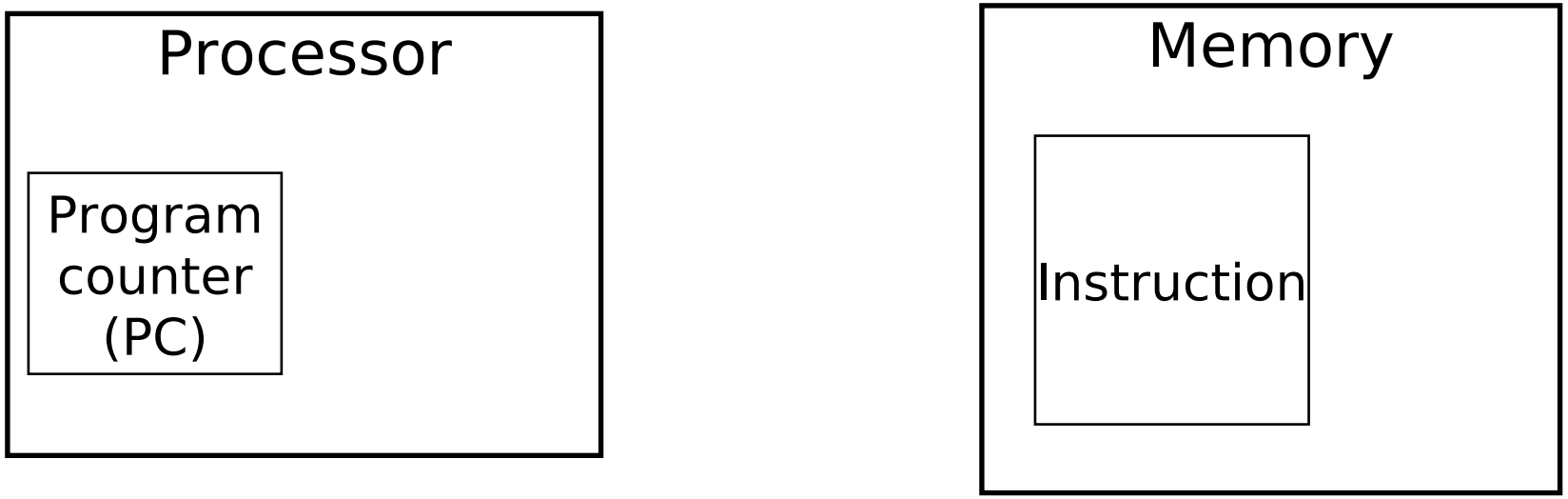
Back to Layers of Transformation…

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.

- Executables (i.e., instructions) are stored in “memory”



Assembly program: add.s

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

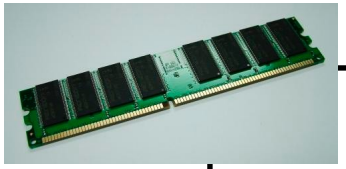
High-level Organization of Computer Hardware a.k.a., The Von Neumann Model

Processor

Program counter (PC)

Arithmetic logic unit (ALU)

Control Path



Memory

Instruction

Data

Bus

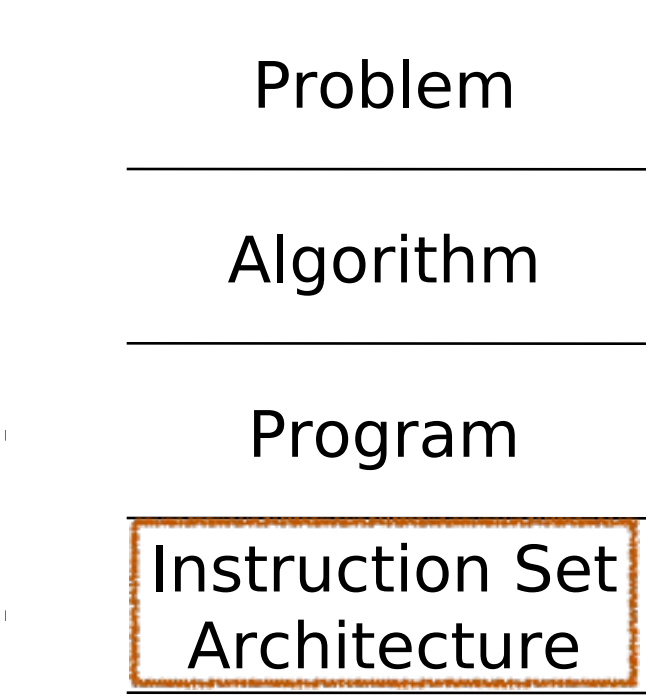
Input / Output Device

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

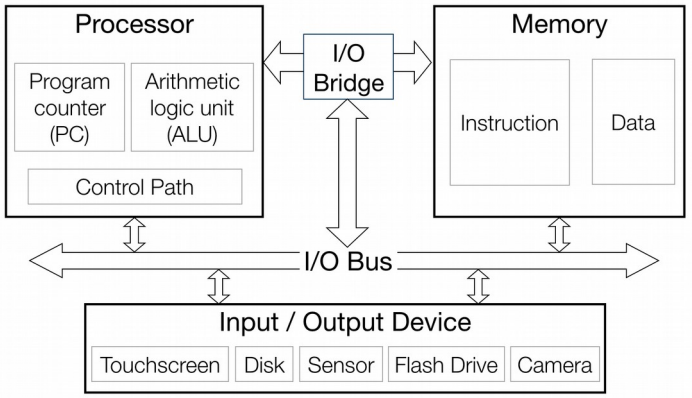
Electrons, Resistors, Capacitors, etc.

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*

Assembly program: add.s

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

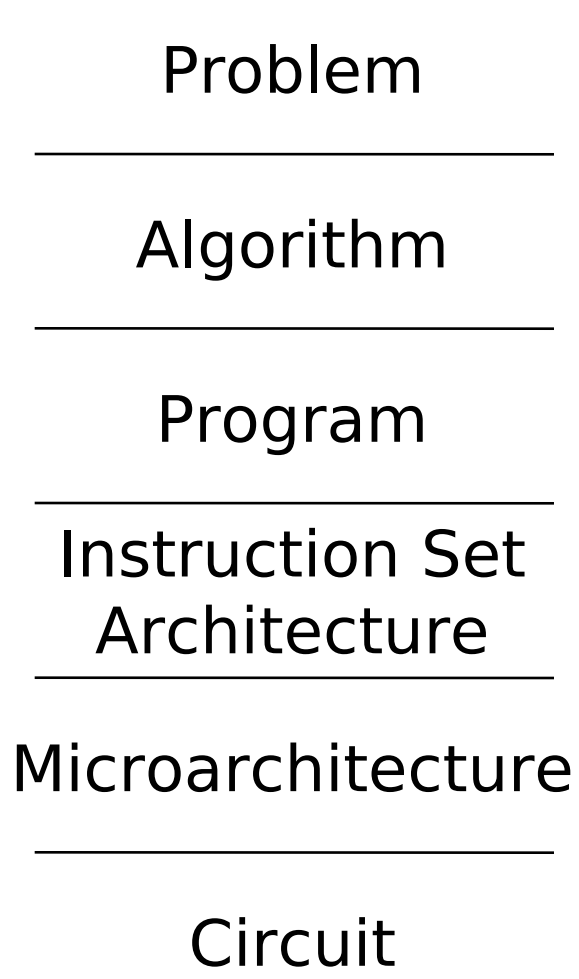


Abstraction

- 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 coffee-cup

Every Layer in CS is an Abstraction

- Depending on which layer you want to live at, you have different views of the computer
- This course aims to un-abstract many of these layers so that you can be a more effective programmer



Instruction Set Architecture

- There used to be many ISAs
  - x86, ARM, Power/PowerPC, Sparc, MIPS, IA64, z
  - Very consolidated today: ARM for mobile, x86 for others
- There are even more microarchitectures
  - Apple/Samsung/Qualcomm have their own microarchitecture (implementation) of the ARM ISA
  - Intel and AMD have different microarchitectures for x86
- ISA is lucrative business: ARM’s Business Model
  - Patent the ISA, and then license the ISA
  - Every implementer pays a royalty to ARM

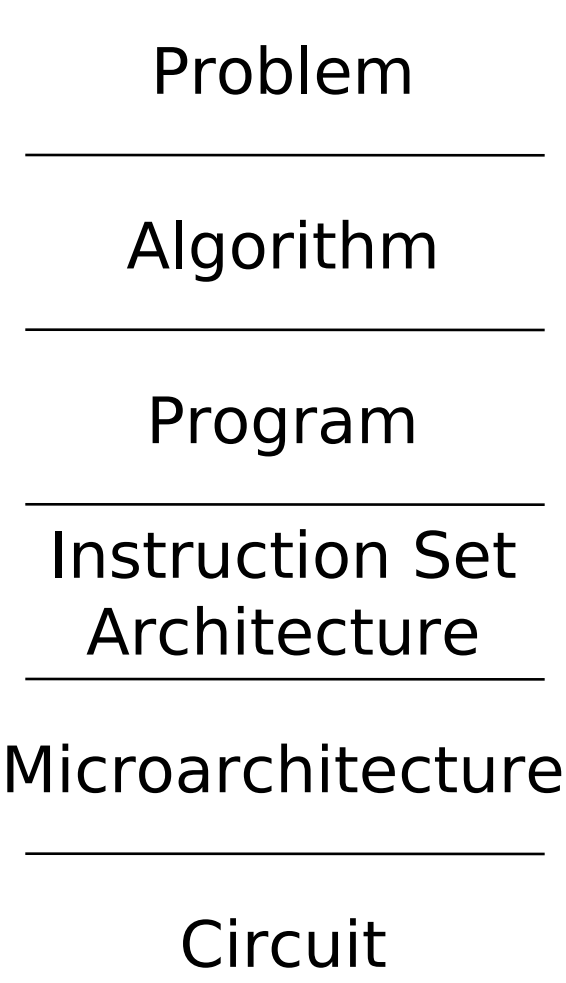
Instruction Set Architecture

- Little research on ISA, much more microarch. research
  - ISA is a mostly stable area now.
  - Free, open ISA: RISC V (UC Berkeley, https://riscv.org/)

  - Instead, current focus is mostly on optimizing the implementation.

The Role of a Computer System Designer

- Look Upwards (Nature of the problems)
- Look Downwards (Nature of the circuit technology and physics)
- Look Backward (Evaluating old ideas in light of new technologies)
- Look Forward



Questions?

Who Are We?

- Myself: Alan Beadle
  - WH 2209, hbeadle@cs.rochester.edu
  - Office hours Friday 1pm – 2pm;
  - Graduate student, not a professor
  - Call me Alan

- TAs: 1 Grad + 5 UG
  - Did very well themselves in this course before
  - Really care about you learning the material and succeeding
  - Ask questions on Piazza forum so they can all see it (and see when it has been answered)
  - Each TA has one hour of office hours per week, will be posted on website and announced later

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 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

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)
  - The book will likely just slow you down
  - **The exam will make you think**

Programming Assignments and Exams

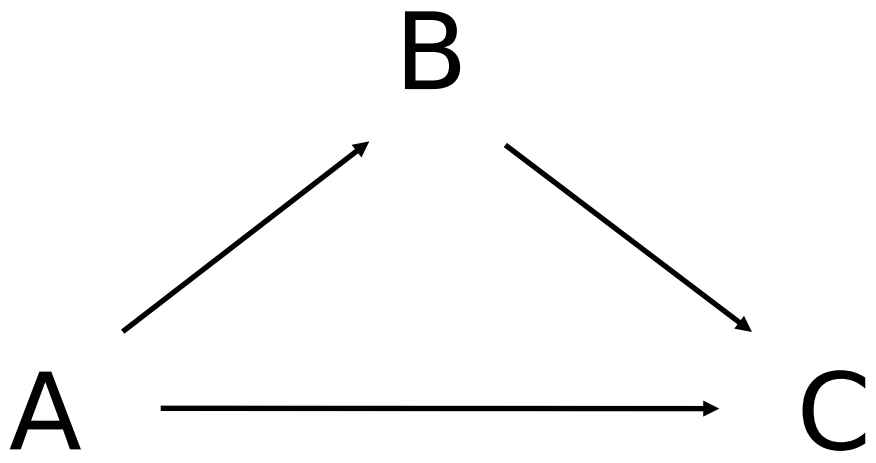
The assignments were often very different from what we were learning in the course, causing a lot of frustration in the beginning. People had to rely on outside sources to get a grounding of how to do an assignment.

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**



Academic Honesty

- Please ask us for help when you need it. We are eager to assist.
- Answering your questions is way less work than filling out academic honesty violation forms.
- All suspected cases of academic dishonesty will be reported to the university by the instructor (This is a university-wide policy)

Academic Honesty

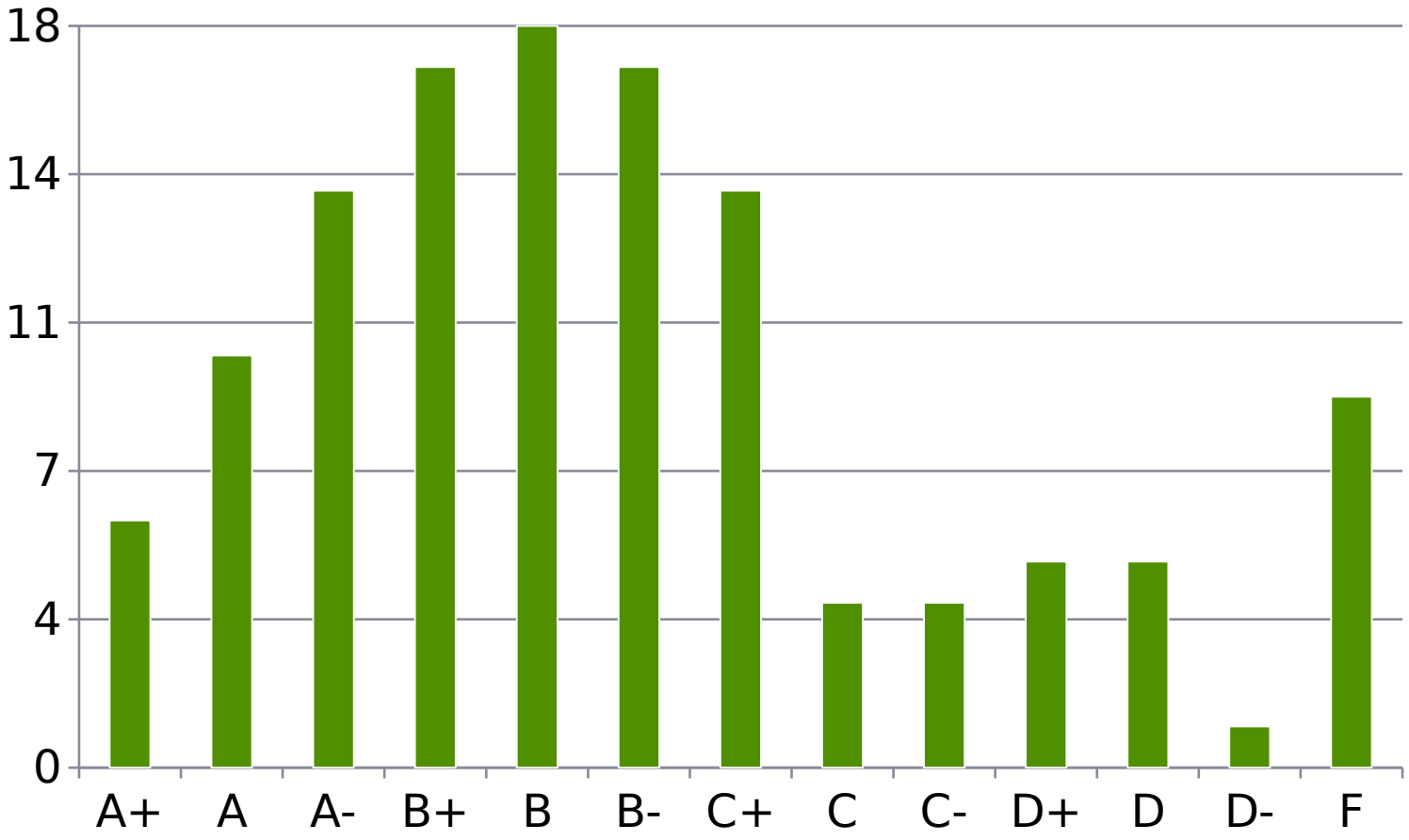
- Please be mindful of where you are looking during exams
- On assignments you can work in pairs, but only pairs and don’t change pairs after you’ve started an assignment
- If your learning needs are not being met, give us a chance to address that by telling us!
- Protect your work from others (sharing work is an honesty policy violation)

What Should You Expect From Us?

- *Think of us as your tutors*
  - *Be your guide in inducing you to explore concepts*
  - *Create situations and pose 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*

- *We understand that you will be busy, but if you make time to ask for help, we can provide it*
  - *Piazza is a great tool for this!*

Final Grades Spring 2018



Final Grades Spring 2019