The Hardware/Software Interface

CSE 351, Winter 2022

Instructor:
Sam Wolfson

Teaching Assistants:
Angela Xu
Anirudh Kumar
Catherine Guevara
Dara Stotland
Nick Durand
Harrison Bay
Ian Hsiao
Mara Kirdani-Ryan
Sanjana Sridhar

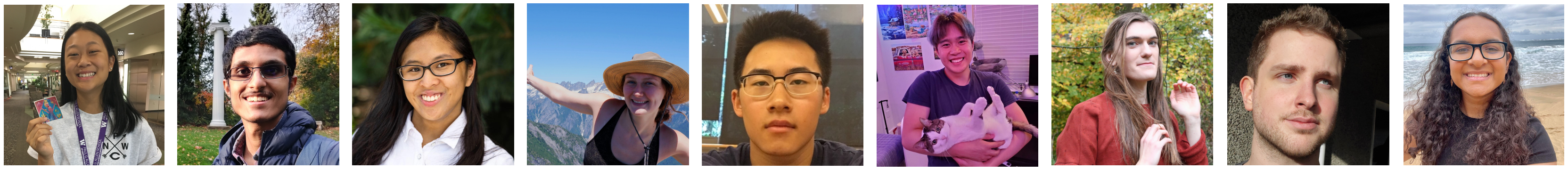
http://xkcd.com/676/

Lecture Outline

- **Course Introduction**
- Course Policies
  - https://courses.cs.washington.edu/courses/cse351/22wi/syllabus
  - Binary and Numerical Representation

Introductions: Course Staff

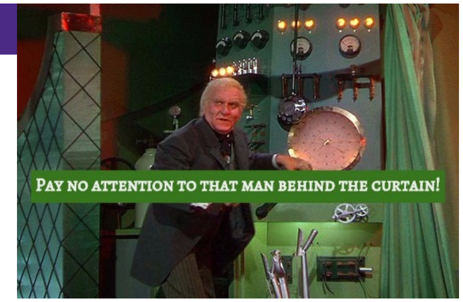
- Instructor: Sam Wolfson
  - This is my 2nd time teaching CSE 351
  - I graduated with a master's degree from UW CSE in 2020
  - Very excited to be teaching again!
- TAs:



  - Available in section, office hours, and on Ed Discussion
- More than anything, we want you to feel…
  - ✓ Comfortable and welcome in this space
  - ✓ Able to learn and succeed in this course
  - ✓ Comfortable reaching out if you need help or want change

Introductions: You!

- ~140 students registered!
- CSE majors, ECE majors, and more
  - Most of you will find almost everything in the course new
  - Many of you are new to CSE and/or UW!
- Get to know each other! Help each other out!
  - Science says that learning happens best in groups
  - Working well with others is a valuable life skill
  - Diversity of perspectives expands your horizons
  - Take advantage of group work, where permissible, to *learn*, not just get a grade



Welcome to CSE 351!

- See the key abstractions "under the hood" to describe "what really happens" when a program runs
  - How is it that "everything is 1s and 0s"?
  - Where does all the data get stored and how do you find it?
  - How can more than one program run at once?
  - How does your source code become something that your computer understands?
- An *introduction* that will:
  - Profoundly change/augment your view of computers and programs
  - Connect your source code down to the hardware
  - Leave you impressed that computers ever work
  - Help you understand the values that have informed the history of computing, and how you can think critically about them

Layers of Computing Below Programming

Software Applications
(written in Java, Python, C, etc.)

Programming Languages & Libraries
(*e.g.*, Java Runtime Env, C Standard Lib)

OS/App interface

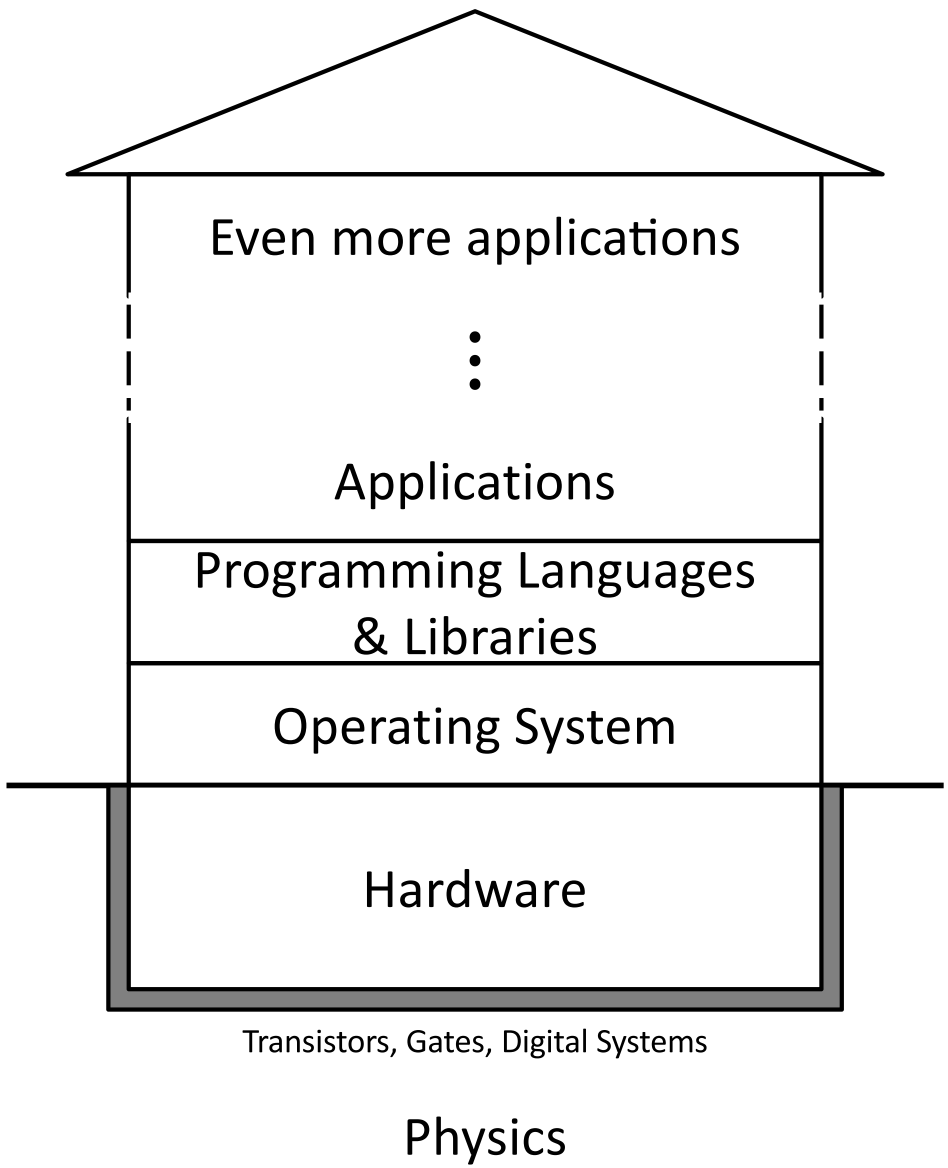
Operating System
(*e.g.*, Linux, MacOS, Windows)

HW/SW interface

Hardware
(*e.g.*, CPU, memory, disk, network, peripherals)

"House" of Computing Metaphor

- We continue to build upward but everything relies on the base & foundation
  - We'll explore parts of Hardware, OS, and PL
- Built a long time ago
  - Some parts have been updated over the years, some have not
  - More remodeling necessary, but should understand *how* and *why* things are this way before demolishing anything



The Hardware/Software Interface

- Topic Group 1: **Data**
  - Memory, Data, Integers, Floating Point, Arrays, Structs
- Topic Group 2: **Programs**
  - x86-64 Assembly, Procedures, Stacks, Executables
- Topic Group 3: **Scale & Coherence**
  - Caches, Processes, Virtual Memory, Memory Allocation

- Learning in this class
  - You might miss Java, but we just ask you to keep your heart open; something unexpected might pique your interest!
  - Notice and nurture any wants to linger in some space
    - Many future classes to explore this space more

Some fun topics that we will touch on

Lecture Outline

- Course Introduction
- **Course Policies**
  - https://courses.cs.washington.edu/courses/cse351/22wi/syllabus
- Binary and Numerical Representation

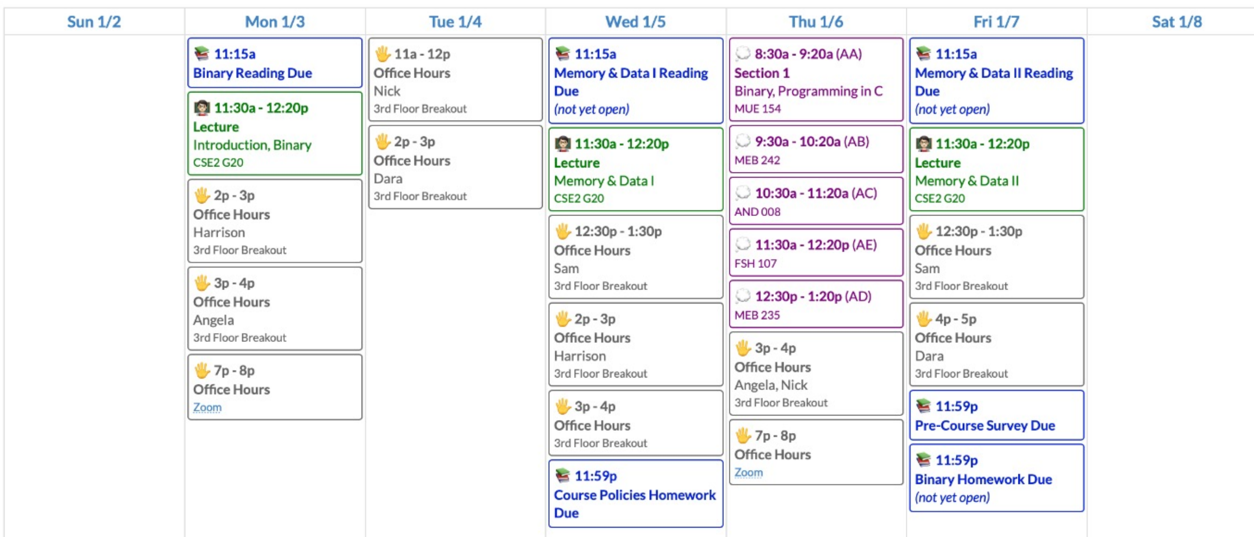
Bookmarks

- Website: https://courses.cs.washington.edu/courses/cse351/22wi/
  - Schedule, policies, materials, videos, assignment specs, etc.
- Ed Course: https://edstem.org/us/courses/16207
  - Discussion: announcements, ask and answer questions
  - Lessons: readings, lecture questions, homework
  - Resources: links to other tools and information

- Linked from website and Ed
  - Canvas: gradebook, Zoom links
  - Gradescope: lab submissions
  - Panopto: lecture recordings

Virtual Instruction (This Week)

- All lectures, sections, and office hours will take place via Zoom. Links are provided on the course website, under "Tools" or the Zoom tab within Canvas.
- Find scheduled office hours on the course calendar



- **The locations of office hours are for next week and beyond, please ignore them for this week.**

Extenuating Circumstances

- Students (and staff) still face an extremely varied set of environments and circumstances
- For formal accommodations, go through Disability Resources for Students (DRS)
- We will try to be accommodating otherwise, but the earlier you reach out, the better

- Don’t suffer in silence – talk to a staff member!
  - We have a 1-on-1 meeting request form

Inclusiveness

- It is very important to us that you have a positive experience in CSE 351 this quarter.
- If at any point you are made to feel uncomfortable, disrespected, or excluded by a staff member or student, please let us know.
  - You may talk with a staff member, email me directly, or send anonymous feedback (via the "Tools" menu on the website).

Grading

- **Pre-Lecture Readings:** 5%
  - Can reveal solution after one attempt (completion)
- **Homework:** 20% total
  - Unlimited submission attempts (autograded correctness)
- **Labs:** 40% total
  - Last submission graded (correctness)
- **Exams:** Midterm (16%) and Final (16%)
  - Exact format TBD; individual
  - If in person, accommodations/makeups will be given
- **EPA:** Effort, Participation, and Altruism (3%)

Group Work in 351

- Group work will be *emphasized* in this class
  - Lecture and section will have built-in group work time – you will get the most out of it if you actively participate!
    - In Zoom: TAs and I will monitor chat
    - In-person: TAs will circle around the room and interact with groups
      - Raise your hand to get the attention of a staff member
  - Most assignments allow collaboration – talking to classmates will help you synthesize concepts and terminology
    - *The major takeaways for this course will be the ability to explain the major concepts verbally and/or in writing to others*
  - However, the responsibility for learning falls on *you*

Lab Collaboration and Academic Integrity

- All submissions are expected to be yours and yours alone
- You are encouraged to discuss your assignments with other students (*ideas*), but we expect that what you turn in is yours
- It is NOT acceptable to copy solutions from other students or to copy (or start your) solutions from the Web (including GitHub, Chegg, and similar sites)
- Our goal is that **you** learn the material so you will be prepared for exams, interviews, and the future

To-Do List

- Admin
  - Explore/read the course website *thoroughly*
  - Check that you can access Ed Discussion & Lessons
  - **Get your machine set up to access the CSE Linux environment (CSE VM or attu) *as soon as possible***
  - Optionally, sign up for CSE 391: System and Software Tools
- Assignments
  - Course Policies due Wednesday (1/5)
  - Pre-Course Survey and Binary Homework due Friday (1/7)
  - Pre-lecture readings due before each lecture – 11:15 am
    - Optional Computer Systems reading given on course calendar
  - Lab 0 due next Monday (1/10)

Lecture Outline

- Course Introduction
- Course Policies
  - https://courses.cs.washington.edu/courses/cse351/22wi/syllabus/
- **Binary and Numerical Representation**

Reading Review

- Terminology:
  - numeral, digit, base, symbol, digit position, leading zeros
  - binary, bit, nibble (nybble?), byte, hexadecimal
  - numerical representation, encoding scheme

- Questions from the reading?

Review Questions

- What is the *decimal value* of the numeral 107_8?
  - A. 71
  - B. 87
  - C. 107
  - D. 568
- What is the decimal number 108 in hex?
  - A. 0x6C
  - B. 0xA8
  - C. 0x108
  - D. 0x612
- Represent 0b100110110101101 in hex.
- Represent 0x3C9 in binary.

Base Comparison

- Why does all this matter?
  - *Humans* think about numbers in **base 10**, but *computers* "think" about numbers in **base 2**
  - Binary encoding is what allows computers to do all the amazing things that they do!

- You should have this table memorized by the end of the class
  - Might as well start now ☺

Base 10	Base 2	Base 16
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Numerical Encoding

- **AMAZING FACT: You can represent *anything* countable using numbers!**
  - Need to agree on an encoding
  - Kind of like learning a new language

- <u>Examples</u>:
  - Decimal Integers: 0→0b0, 1→0b1, 2→0b10, etc.
  - English Letters: CSE→0x435345, yay→0x796179
  - Emoticons: 😃 0x0, 😞 0x1, 😎 0x2, 😇 0x3, 😈 0x4, 🙋 0x5

Binary Encoding

- With n binary digits, how many "things" can you represent?
  - Need n binary digits to represent N things, where $2^n \geq N$
  - Example: 5 binary digits for alphabet because $2^5 = 32 > 26$

- A binary digit is known as a bit
- A group of 4 bits (1 hex digit) is called a nibble (nybble?)
- A group of 8 bits (2 hex digits) is called a byte
  - 1 bit → 2 things, 1 nibble → 16 things, 1 byte → 256 things

So, What Does It Mean?

- *A sequence of bits can have many meanings!*

- Consider the hex sequence 0x4E6F21
  - Common interpretations include:
    - The decimal number 5,140,257
    - The real number 7.203034×10^{-39}
    - The characters "No!"
    - The background color of this slide

- It is up to the program/programmer (you!) to decide how to **interpret** the sequence of bits

Binary Encoding – Characters/Text

- ASCII Encoding (www.asciitable.com)
  - American Standard Code for Information Interchange

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH	(start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX	(start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX	(end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT	(end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	[illegible]	[illegible]	d
5	5	005	ENQ	(enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	[illegible]	[illegible]	e
6	6	006	ACK	(acknowledge)	38	26	046	&	&	70	46	106	F	F	102	[illegible]	[illegible]	[illegible]	f
7	7	007	BEL	(bell)	39	27	047	'	'	71	47	107	[illegible]	G	[illegible]	[illegible]	[illegible]	[illegible]	g
8	8	010	BS	(backspace)	40	28	050	(	(	72	48	110	[illegible]	[illegible]	[illegible]	[illegible]	150	[illegible]	h
9	9	011	TAB	(horizontal tab)	41	29	051	)	)	73	49	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	i	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	*	*	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	j	j
11	B	013	VT	(vertical tab)	43	2B	053	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	107	6B	153	k	k
12	C	014	FF	(NP form feed, new page)	44	2C	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	L	L	108	6C	154	l	l
13	D	015	CR	(carriage return)	45	2D	[illegible]	[illegible]	-	[illegible]	[illegible]	115	M	M	109	6D	155	m	m
14	E	016	SO	(shift out)	[illegible]	2E	[illegible]	[illegible]	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI	(shift in)	[illegible]	2F	[illegible]	[illegible]	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE	[illegible]	[illegible]	[illegible]	060	[illegible]	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1	[illegible]	[illegible]	[illegible]	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2	[illegible]	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	[illegible]	[illegible]	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	[illegible]	[illegible]	[illegible]	[illegible]	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	[illegible]	[illegible]	[illegible]	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	[illegible]	[illegible]	[illegible]	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	[illegible]	[illegible]	(end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN	(cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM	(end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB	(substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC	(escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS	(file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS	(group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS	(record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US	(unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

What's Missing?

Binary Encoding – Characters/Text

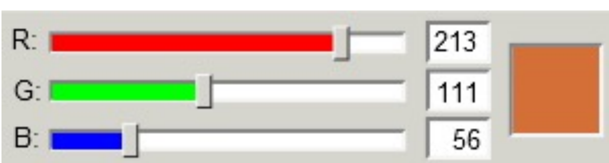
- ASCII Encoding (www.asciitable.com)
  - *American* Standard Code for Information Interchange
- Created in 1963
  - Memory was expensive, 32KB in brand new machines
  - *Economic incentive* to use fewer bits for encoding
- **Design Goals:**
  - Represent everything on an *American* typewriter as *efficiently* as possible
  - Organize similar characters together
    - Numbers, uppercase, lowercase, then other stuff

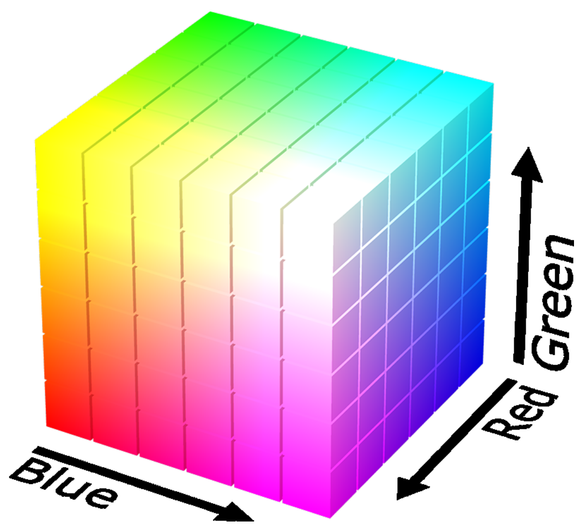
Binary Encoding – Unicode & Emoji

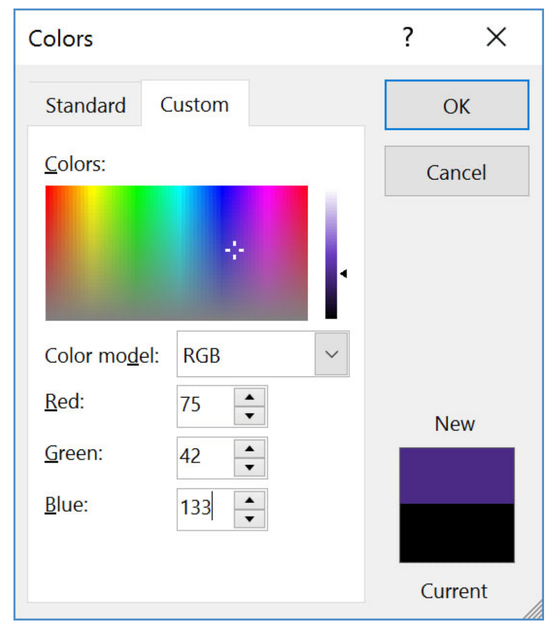
- Unicode Standard is managed by the Unicode Consortium
  - "Universal language" that uses 1-4 bytes to represent a much larger range of characters/languages, including emoji
  - Adds new emojis every year
    - Offer opportunities to be more inclusive of race and gender diversity
    - However, adoption often lags: 👸 and 🤴 added in 2015 and 2016, but non-gendered "person with crown" only added in 2021: ⍰
    - https://emojipedia.org/new/
- Emojipedia demo: http://www.emojipedia.org
  - Desktop Computer: 🖥️
  - Code points: U+1F5A5, U+FE0F
  - Display: 

Binary Encoding – Colors

- RGB – Red, Green, Blue
  - Additive color model (light): byte (8 bits) for each color
  - Commonly seen in hex (in HTML, photo editing, etc.)
  - Examples: **Blue**→0x0000FF, **Gold**→0xFFD700, **White**→0xFFFFFF, **Deep Pink**→0xFF1493







Binary Encoding – Files and Programs

- At the lowest level, all digital data is stored as bits!

- Layers of abstraction keep everything comprehensible
  - Data/files are groups of bits interpreted by program
  - Program is groups of bits being interpreted by your CPU

- Computer Memory Demo (if time)
  - From vim: `%!xxd`
  - From emacs: `M-x hexl-mode`

Summary

- Humans think about numbers in decimal; computers think about numbers in binary
  - Base conversion to go between them
  - Hexadecimal is more human-readable than binary
- All information on a computer is binary
- Binary encoding can represent *anything*!
  - Computer/program needs to know how to interpret the bits
  - Encodings aren't "neutral"; priorities are baked in