The Hardware/Software Interface

CSE 351, Winter 2023

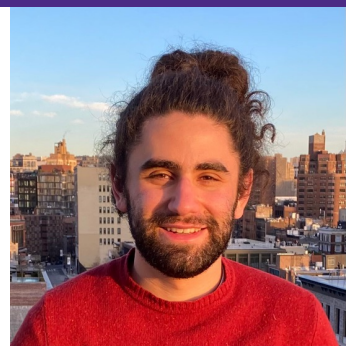
Instructor:
Sam Wolfson

Teaching Assistants:
Aman Mohammed
Angela Xu
Armin Magness
Clare Edmonds
David Dai
Jenny Peng
Maggie Jiang
Mara Kirdani-Ryan
Nayha Auradkar
Yoonseo Song

http://xkcd.com/676/

Lecture Outline

- **Course Introduction**
- Course Policies
  - https://courses.cs.washington.edu/courses/cse351/23wi/syllabus
  - Binary and Numerical Representation



Introductions: Course Staff

- Instructor: just call me Sam!
  - I graduated with a master's degree from UW CSE in 2020, and I now work at a company called ExtraHop
    - My "day job" is much further up the stack than what we'll talk about here, but I'm also happy to chat about Kubernetes, API design, Python hackery, and why I don't like Go 😁
  - I'm very excited to be teaching again (my 3rd time with 351)!
- Your wonderful 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

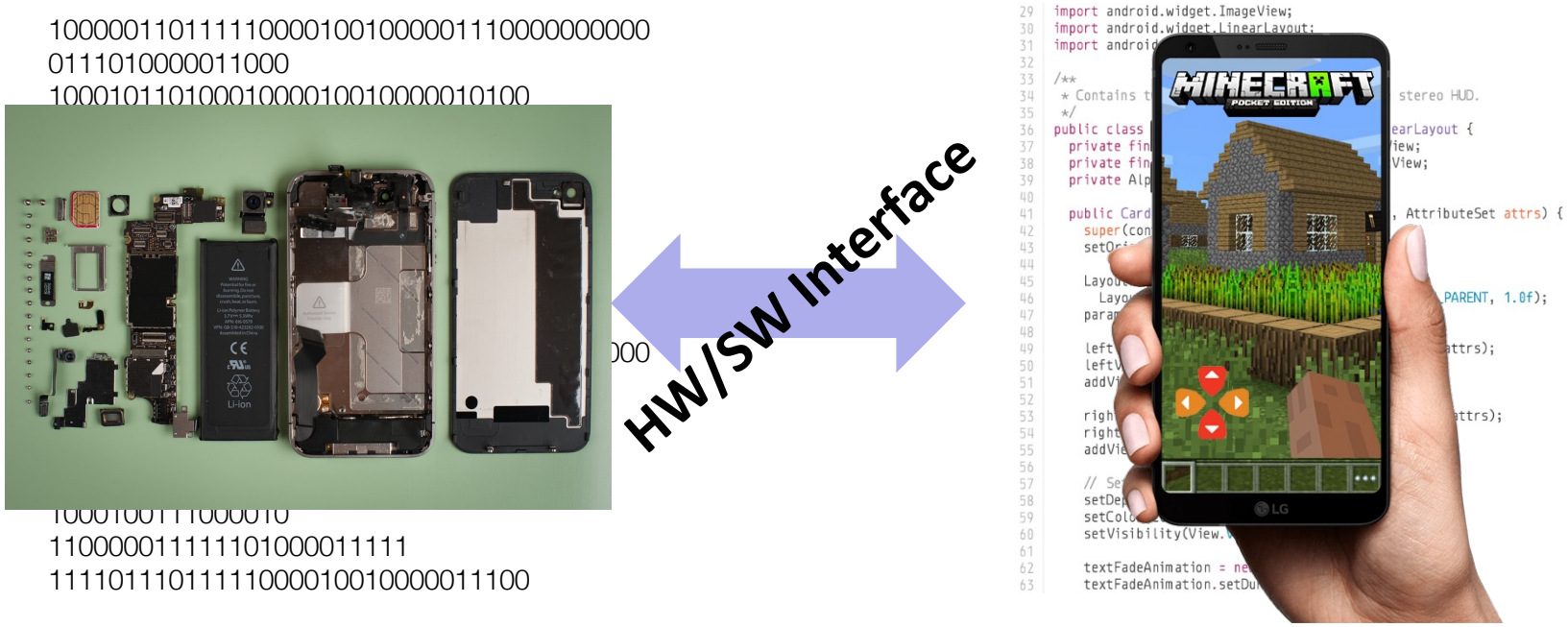
Introductions: You!

- 156 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!

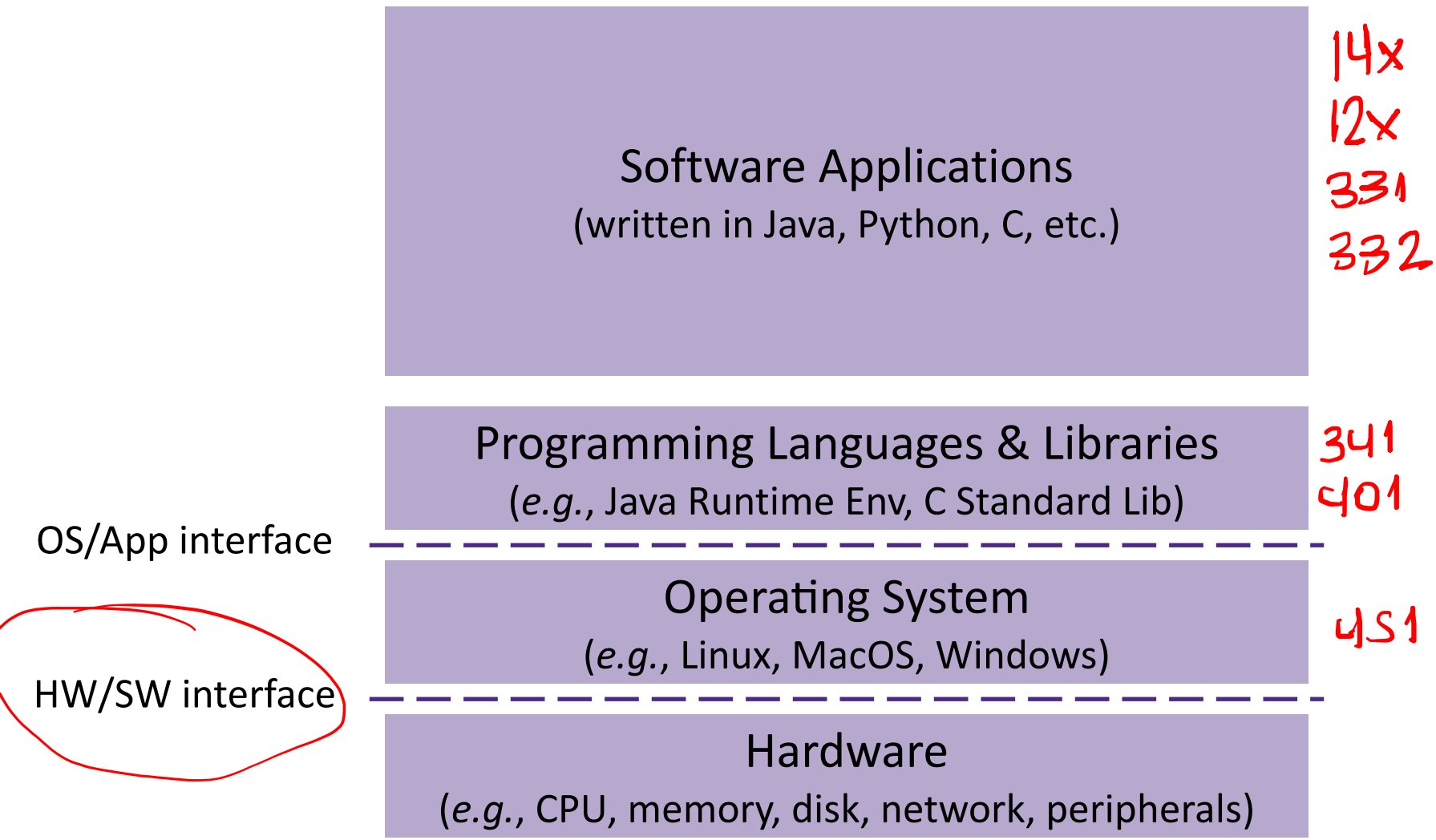
- See the key abstractions "under the hood" to describe "what really happens" when a program runs
  - How is it that "everything is 1s and 0s"?
  - Where does all the data get stored and how do you find it?
  - How can more than one program run at once?
  - How does your source code become something 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

Welcome to CSE351!



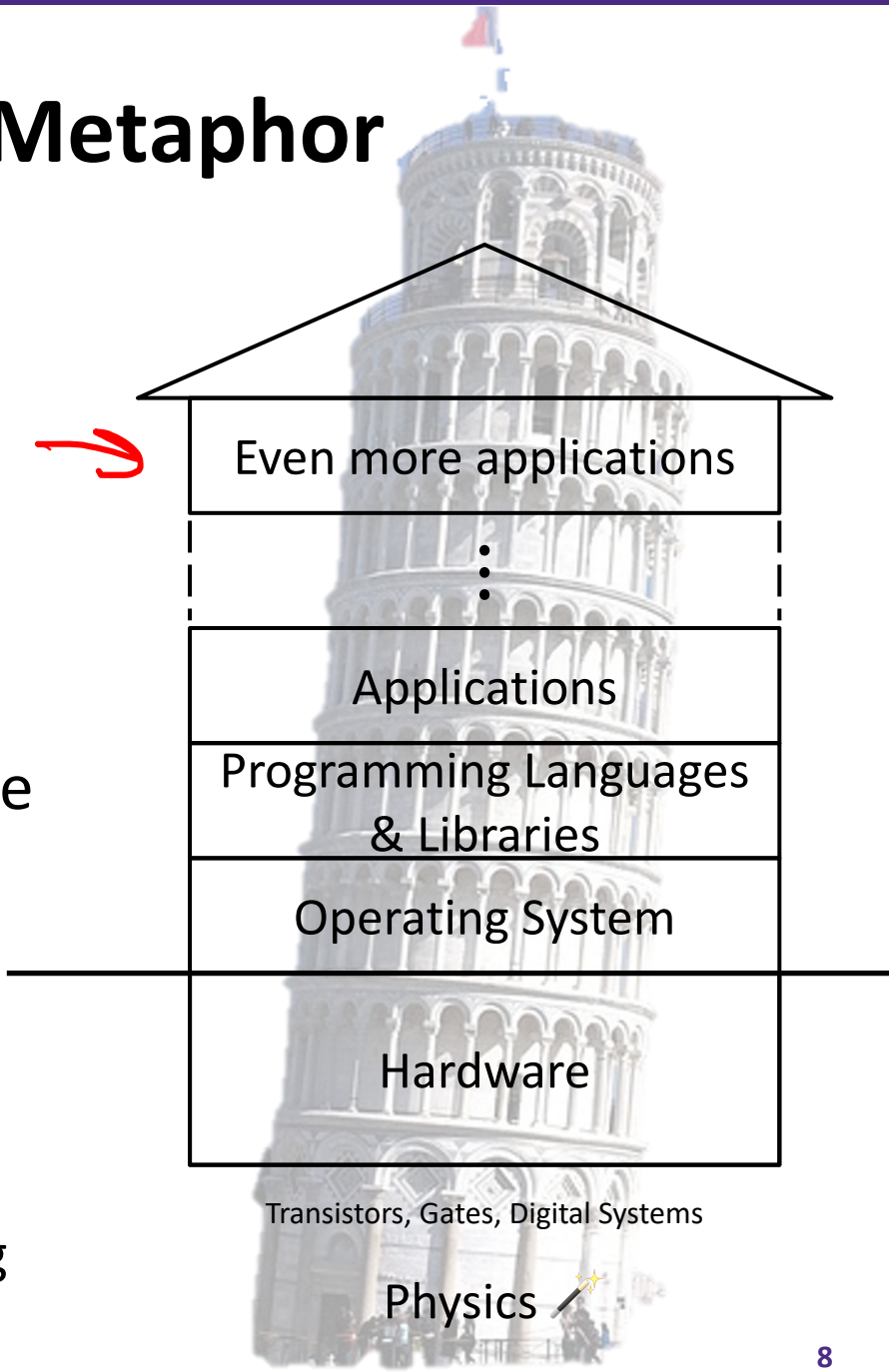
- Our goal: to teach you the key abstractions "under the hood" of your computer
  - How does your source code become something that your computer understands?
  - What happens as your computer executes one or more processes?

Layers of Computing Below Programming



"House" of Computing Metaphor

- We continue to build upward but everything relies on the base & foundation
  - We'll explore parts of the hardware, the operating system, and programming languages
- Built a long time ago, by people with biases and priorities
  - 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 getting out the sledgehammer



Your computer before you take CSE 351

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

- Which of the following seems the most interesting to you? (vote in Ed Lessons)

a) What is a GFLOP and why is it used in computer benchmarks?

b) How and why does running many programs for a long time eat into your memory (RAM)?

c) What is stack overflow and how does it happen?

d) Why does your computer slow down when you run out of disk space?

e) What was the flaw behind the original Internet worm, the Heartbleed bug, and the Cloudbleed bug?

f) What is the meaning behind the different CPU specifications? (e.g., number of cores, size of cache)

Lecture Outline

- Course Introduction
- **Course Policies**
  - https://courses.cs.washington.edu/courses/cse351/23wi/syllabus
- Binary and Numerical Representation

Bookmarks

- Website: https://courses.cs.washington.edu/courses/cse351/23wi/
  - Schedule, policies, materials, videos, assignment specs, etc.
- Ed Course: https://edstem.org/us/courses/32033/
  - 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

Grading

- **Pre-Lecture Readings:** 5% 
  - Can reveal solution after one attempt (completion)
- **Homework:** 20% total 
  - Unlimited submission attempts (auto-graded correctness)
- **Labs:** 40% total 
  - Last submission graded (correctness)
- **Assessments:** Midterm (16%) and Final (16%)
  - Exact format TBD
- **Polling Questions**: 3% 

A note on participation

- I love it when you participate in the course!
- But I recognize that ways in which students participate may not be accurately captured by a simple "participation" score.

- We won't allocate any of your course grade to participation points. We ask that you to show up as you and participate in however works best for you.

Group Work in 351

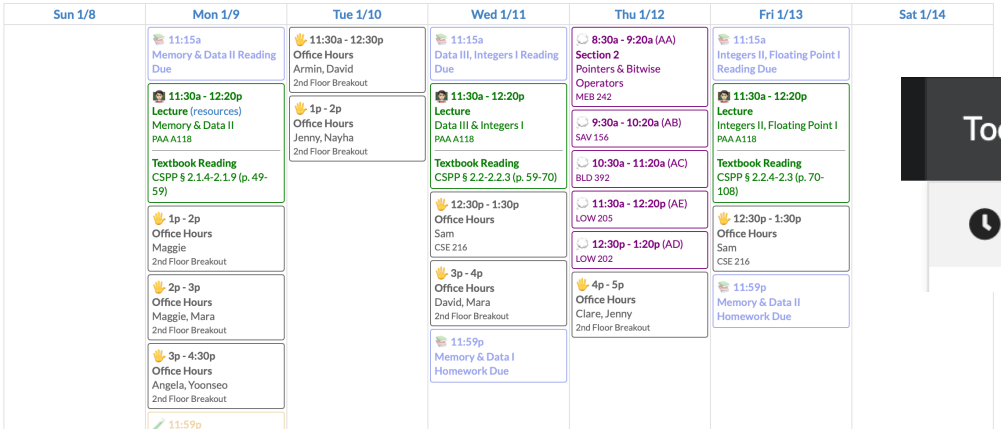
- Group work will be *emphasized* in this class
  - Lecture and section will have built-in group work time – you will get the most out of it if you actively participate!
    - TAs will circle around the room and interact with groups
      - Raise your hand to get the attention of a staff member
  - Most 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*

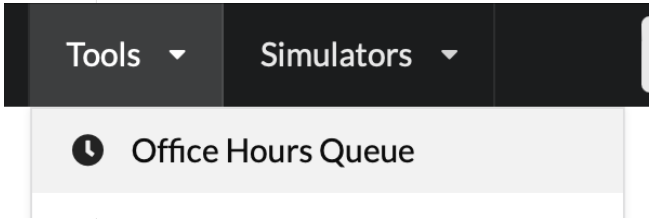
Lab Collaboration and Academic Integrity

- 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, ChatGPT, and similar sites)
- Our goal is that **you** learn the material so you will be prepared for exams, interviews, and the future

Office Hours

- Will either be in-person or on Zoom, but not hybrid.
  - We'll add Zoom office hours to the calendar soon.

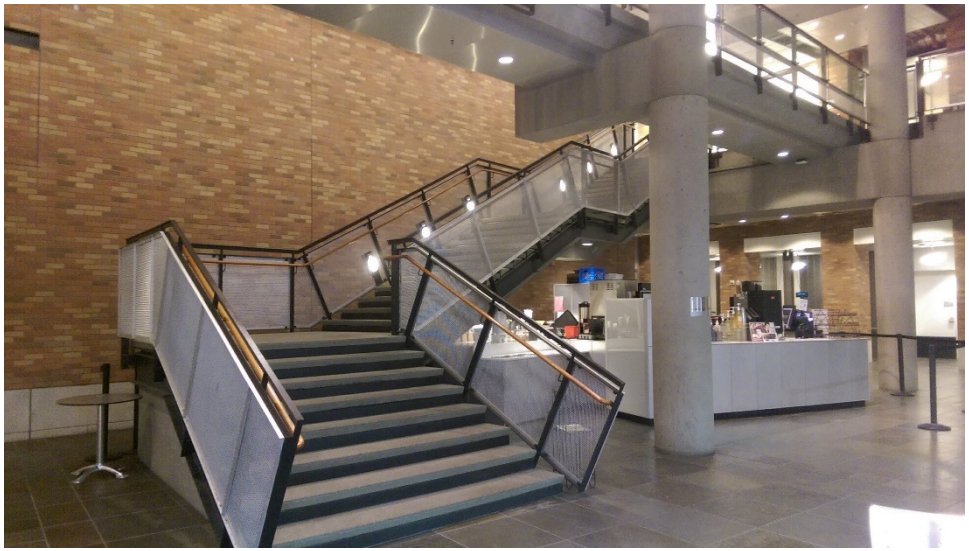




- We'll use a queueing system (e.g., a Google Doc) linked on the course website for Zoom office hours
  - For in-person, just use the whiteboard as needed
- Feel free to chat with other students while waiting for the TA! (In Zoom, use the main room)

In-Person Office Hours

- Allen 2nd floor breakout
  - Up the stairs in the CSE Atrium (Allen Center, not Gates)
  - At the top of the first flight, the open area with the whiteboard wall is the 2nd floor breakout!
- Sam's office (CSE 216) is on the same floor, just across from the atrium



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).

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 or calgary) *as soon as possible***
  - Optionally, sign up for CSE 391: System and Software Tools
- Assignments
  - Course Policies due Friday (1/6)
  - Pre-Course Survey and Binary Homework due Monday (1/9)
  - Pre-lecture readings due before each lecture – 11:15 am
    - Optional Computer Systems reading given on course calendar
  - Lab 0 due next Monday (1/9)

Lecture Outline

- Course Introduction
- Course Policies
  - https://courses.cs.washington.edu/courses/cse351/23wi/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?

  $1\times8^2+0\times8^1+7\times8^0$

  2^{10}

  A. 71
  B. 87
  C. 107
  D. 568

- What is the decimal number 108 in hex?

  A. 0x6C
  B. 0xA8
  C. 0x108
  D. 0x612

  $16^0=1$
  $16^1=16$
  $16^2=256$
  $96+12=108$

- Represent 0b100110110101101 in hex.

  0x4DAD

- Represent 0x3C9 in binary.

  0b0011 1100 1001

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

- Examples:
  - Decimal Integers: 0→0b0, 1→0b1, 2→0b10, etc.
  - English Letters: CSE→0x435345, yay→0x796179
  - Emoticons: 😀 0x0, 😞 0x1, 😎 0x2, 😇 0x3, 😈 0x4, 🙋 0x5
    - Well, not literally, but you get the idea…

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	1	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	1	101;	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	46	102;	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	10		47	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	72				150	104;	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	11	I					i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	11	J		1			j	j
11	B	013	VT (vertical tab)	43	2B	053	+		75	4B	113			107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	0	,		76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	0	4	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	05	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escap	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (d ce cor	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (e nc	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	e	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	ev e 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15		ga e acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	0	N ynchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	B (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

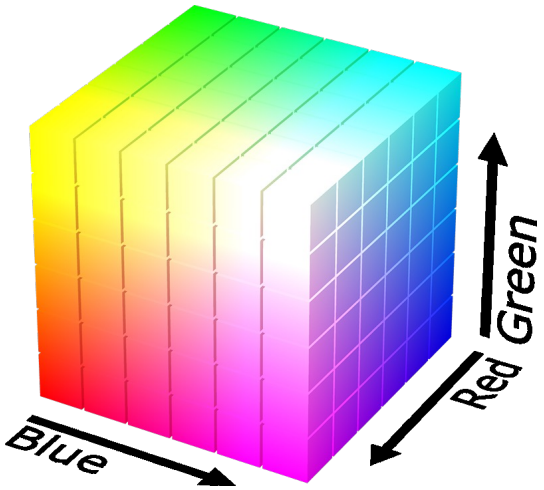
- ASCII Encoding (www.asciitable.com)
  - *American* Standard Code for Information Interchange
- Created in 1963
  - Memory was expensive, 32KB in brand new machines
  - *Economic incentive* to use fewer bits for encoding
- **Design Goals:**
  - Represent everything on an *American* typewriter as *efficiently*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

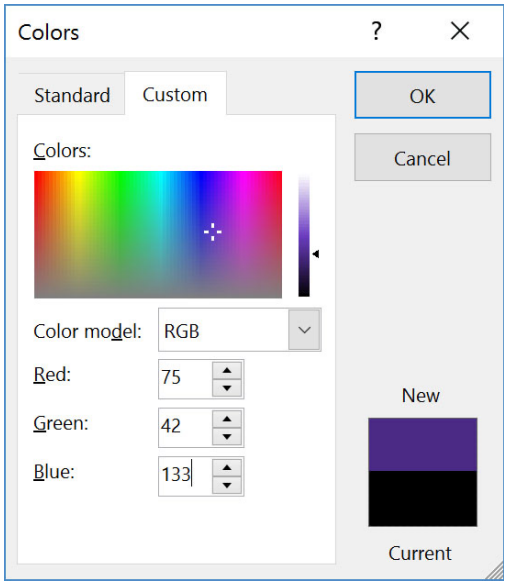
Binary Encoding – Unicode & Emoji

- 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