The Hardware/Software Interface

Instructor:
Mara Kirdani-Ryan

Teaching Assistants:
Kashish Aggarwal Nick Durand
Colton Jobes. Tim Mandzyuk

Mort Garson's *Plantasia*
"Warm Earth Music for plants, and the people that love them"

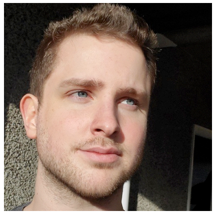
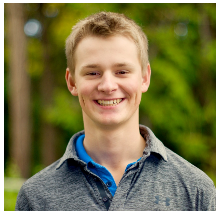
http://xkcd.com/66/

Introductions: Me

- Call me Mara! I use they/them
  - (no Dr., Prof., just Mara please!)
  - Trans/non-binary/queer/neurodiverse, I don't like boxes
- PhD student, Amy Ko
- Former/recovering computer architect
- Currently working in computing education
- Bikes, yoga, guitar, bird sounds, cat

…and a wonderful course staff!

- TAs:


- 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'd like change
- We're available in OH/Section/on Ed
  - We want to help!

Introductions: You!

- ~40 students registered

- CSE majors, ECE majors, and more
  - Lots of new pieces, regardless of background

- Get to know each other! Help each other!
  - Science says that learning happen best in groups!
  - Plus, it’s less lonely!
  - Most of your life will likely involve working with others
  - You’re limited by your own perspective, try to understand others!

ohyay!

- This is ohyay!
- It’s quirky, but much less industrial than Zoom
- Multiple forms of participation!
- More than 6 reactions!
- So far, 3 make sound!

Pick an Emoji!

How do you feel about ohyay?

Welcome to Summer 2021!

- Hopefully Ohyay Uni > Zoom Uni
- We're trying a few new things this quarter:
  - New Content!
  - New Tooling!
  - New Assessments!

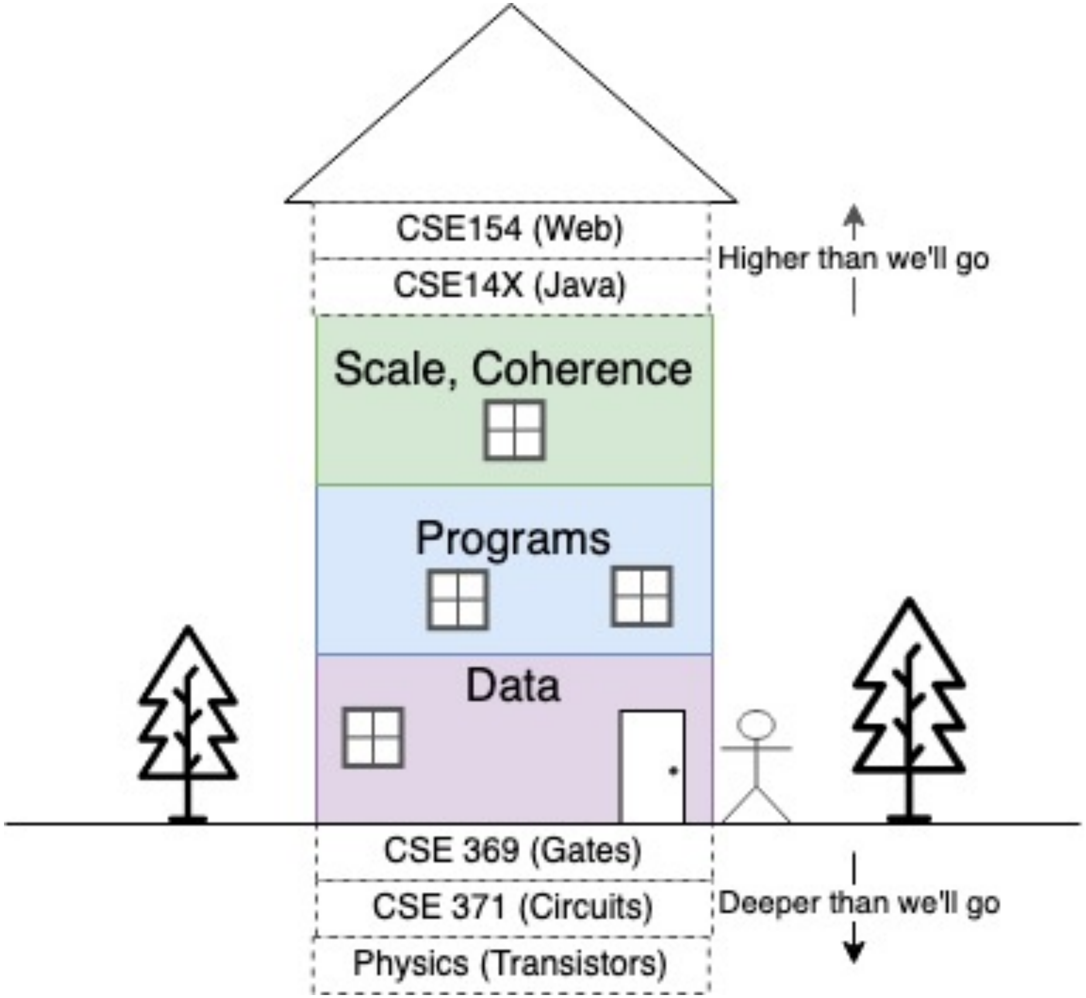
- We're open to feedback!
  - Come to office hours, send us an email!
  - We'd love to work with you to make this course better!

Some Definitions

- **Computing:**
  - Any activity involving computation (like CS, but including non-programming computation tasks)
- **Socio-technical:**
  - Interactions between society and technology
  - Computing exists beyond algorithms!
  - How is technology used? For what purposes?
- We shape our tools, our tools shape us

- *Emoji! How do you feel about socio-technical content?*

The House of Computing (HoC)

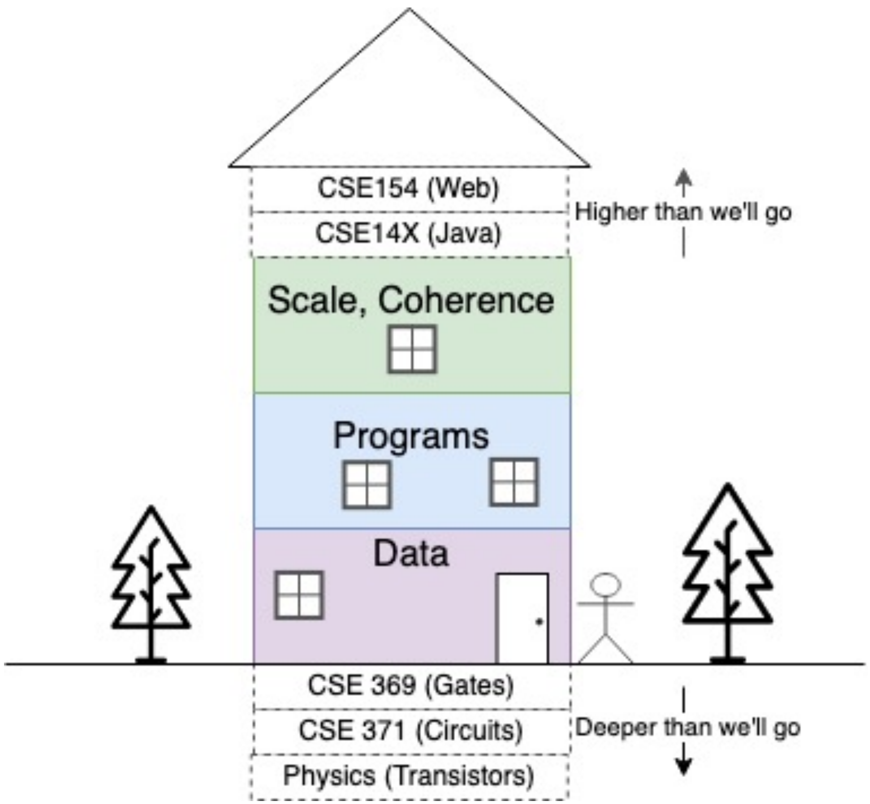


An introduction…

- We don’t have time to explore everything…
  - …but you can take more courses to explore more!
- You might want to linger in some space…
  - …notice and nurture those wants!
- You might miss Java…
  - We just ask you to keep your heart open
  - Something unexpected might pique your interest!

Remodeling

- This house doesn't work for everyone!
  - Accessibility issues
  - Obsolete priorities
  - Little regard for welfare
  - Much, much more

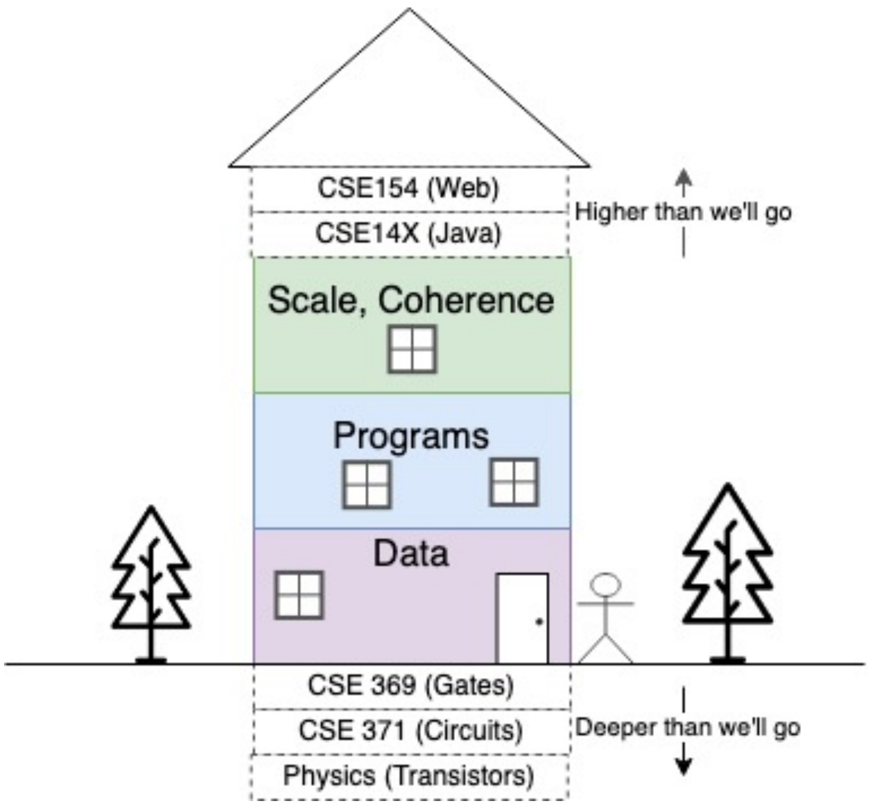


"More and more, the oppressors are using science and technology as unquestionably powerful instruments for their purpose, the maintenance of the oppressive order thorough manipulation and repression"

Paulo Freire, 1970

Remodeling, and you

- Things *need* to be fixed!
- Understand the foundation before you start hammering!
- Technical know-how and socio-technical understanding together!

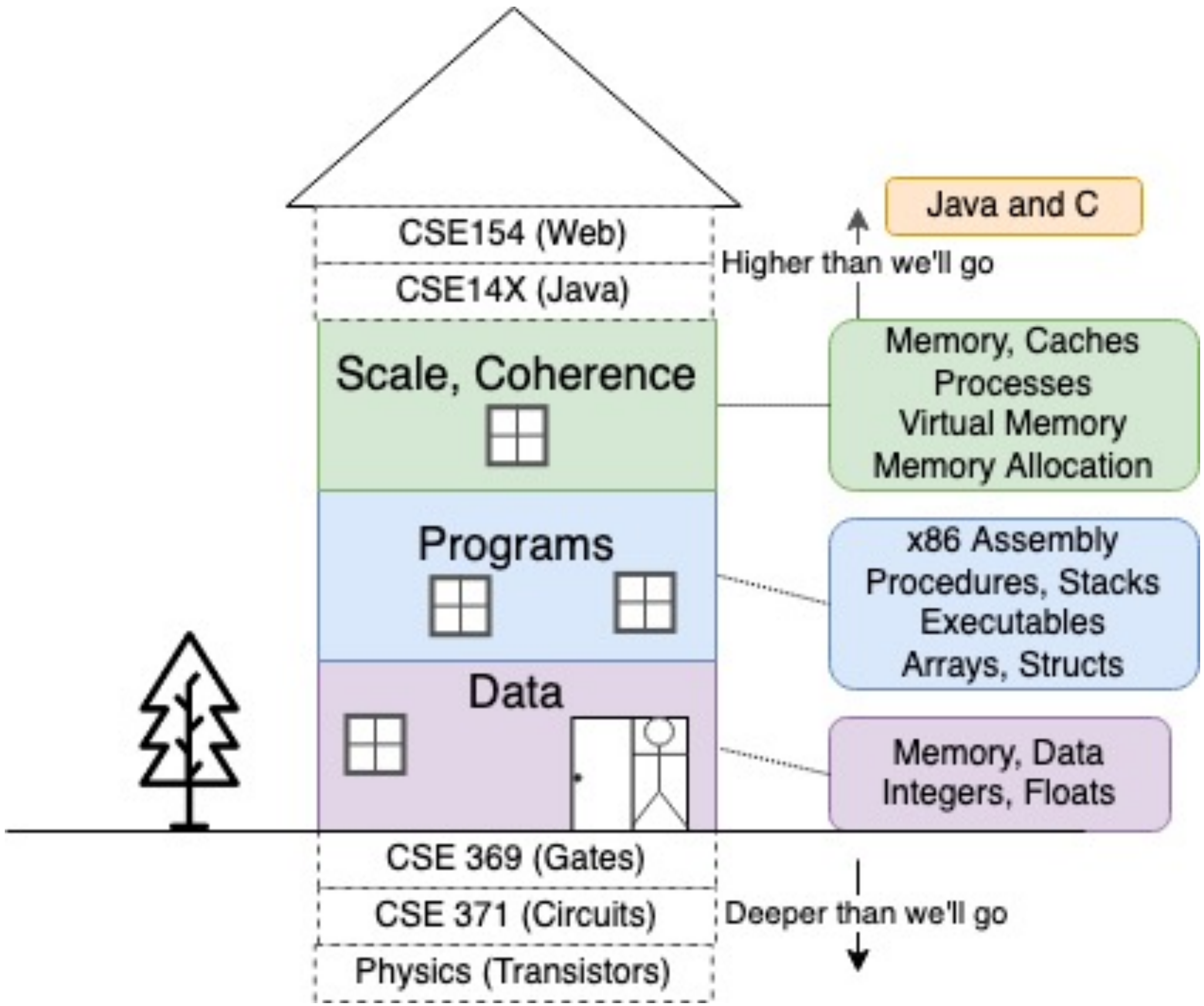


A loving disclaimer

- It's summer!
- I'm a grad student!
- I wanted to try something new!

- **This is going to be different!**
  - Come to my OH/send email if this isn't working.
- 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'd like change

Course Overview



Goals for this course

- Learn how computers work!
  - (With some necessary cuts, we only have 8 weeks)
- Learn why they were designed to work that way
  - Computers didn't come out of nothing, they were built!
- Learn about CS from an ideological perspective
  - The values encoded in systems stuck around!
  - We'll connect to some modern incarnations

Lecture Outline

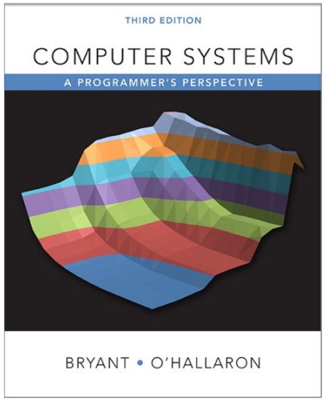
- Course Introduction
- **Course Policies**
  - https://cs.uw.edu/351/syllabus
- Binary

Bookmarks

- Website: https://cs.uw.edu/351
  - Schedule, policies, materials, videos, assignments, etc.
- Discussion on EdStem (link on course page)
  - Announcements here!
  - Ask and answer questions!
- Lessons on EdStem (link on course page)
  - Pre-lecture Readings, lecture questions, HW
- Gradescope: (link on course page)
  - Lab/Unit Summary submissions and grades
- Canvas: (link on course page)
  - Calendar, grade book

Recommended Textbooks

- *Computer Systems: A Programmer's Perspective*
  - Website: http://csapp.cs.cmu.edu
  - Latest edition: (North American) 3rd edition
    - http://csapp.cs.cmu.edu/3e/changes3e.html
    - http://csapp.cs.cmu.edu/3e/errata.html
  - It's a good, *popular*, systems book
    - Recommended lecture readings on website
    - Starkly lacking critical content



- A good C book – any will do
  - *The C Programming Language* (Kernighan, Ritchie)

Course Components:

- Lectures (26)
  - Via Ohyay, review concepts, discuss content
  - Ideally slides posted before, recordings after
- Sections (9)
  - Via Ohyay, short review then mainly group work
  - Not recorded, but materials/helpful videos posted after
- Office Hours
  - Via Ohyay, schedule on the course site. Not recorded.
  - Come ask questions! (course material or others)
  - We may add a queue, stay tuned

Course Components:

- Pre-quarter and Mid-quarter surveys (on Canvas)
  - Meant to check in and get to know you better
- Online Readings
  - Before lecture
- Labs (6)
  - In depth applications/investigations of course material
  - Specs on website, submitted via Gradescope
- Unit Summaries (3)
  - We'll talk more on Wednesday
- Can use up to 7 late days on labs and unit summaries (see syllabus for more details)

Grading

- **Readings/Section Worksheets:** ~10%
  - One attempt per question (completion)
- **Homework:** ~20%
  - Unlimited submission attempts (autograded correctness)
- **Labs:** ~40%
  - Last submission graded (correctness)
- **Unit Summaries:** ~30%
  - Meant to replace the review, summarizing, and reflecting that studying for exams provides. More info on these later.

Lab Collaboration and Academic Integrity

- Ideally, I would've remade the course assignments so that *cheating wasn't possible*
  - I didn't have time!
  - This collaboration policy doesn't match the "real world", but we want assignments to gauge *your* understanding, so we can step in and help if needed
- Read the syllabus! There's a quiz!
- Collaboration allowed on some assignments, some must be independent.
  - Your own words, your own ideas, your own work

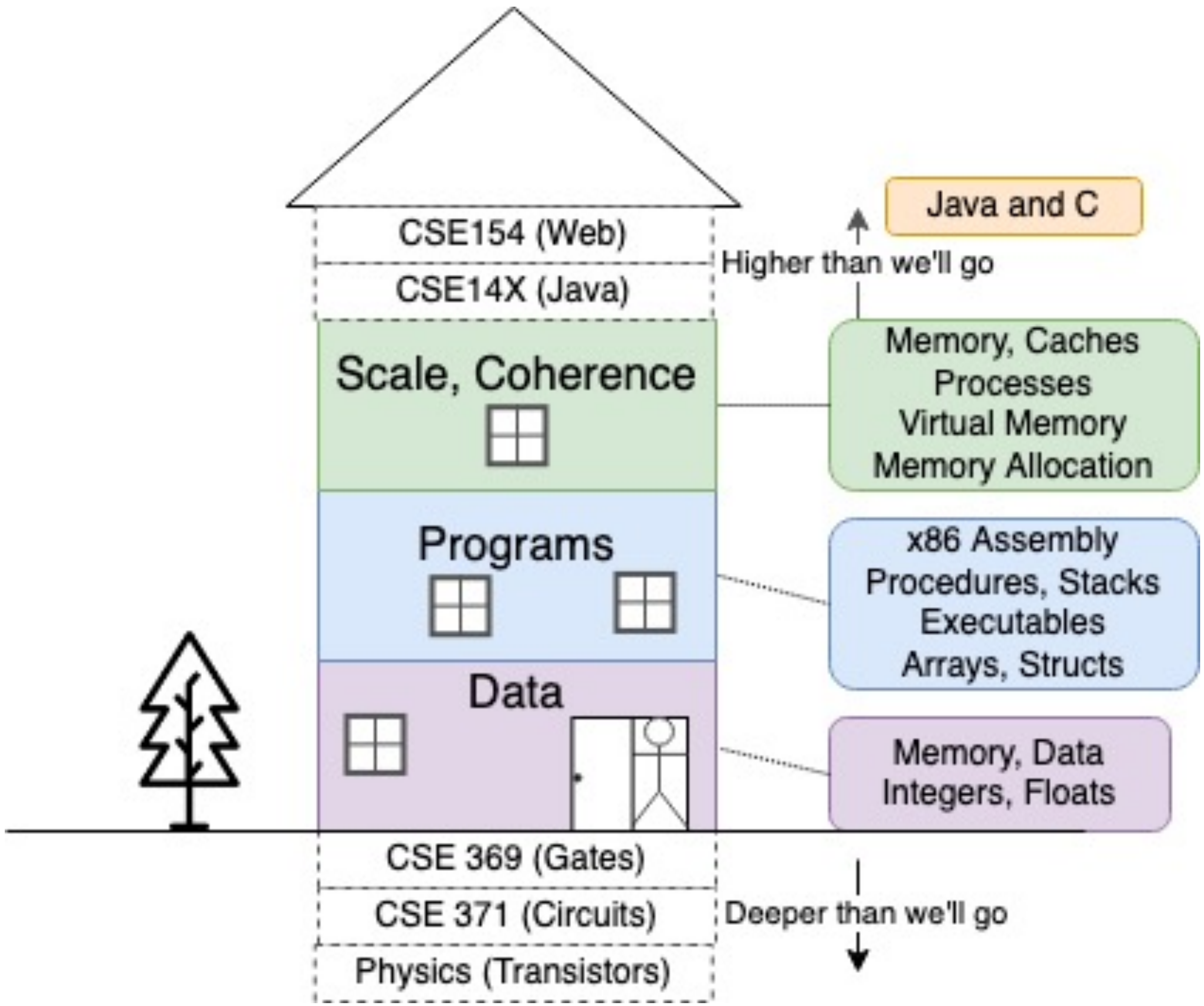
Course Environment and Culture

- Your physical and emotional well-being are much more important than course material!
- Let us know what we can do to help!

To-Do List

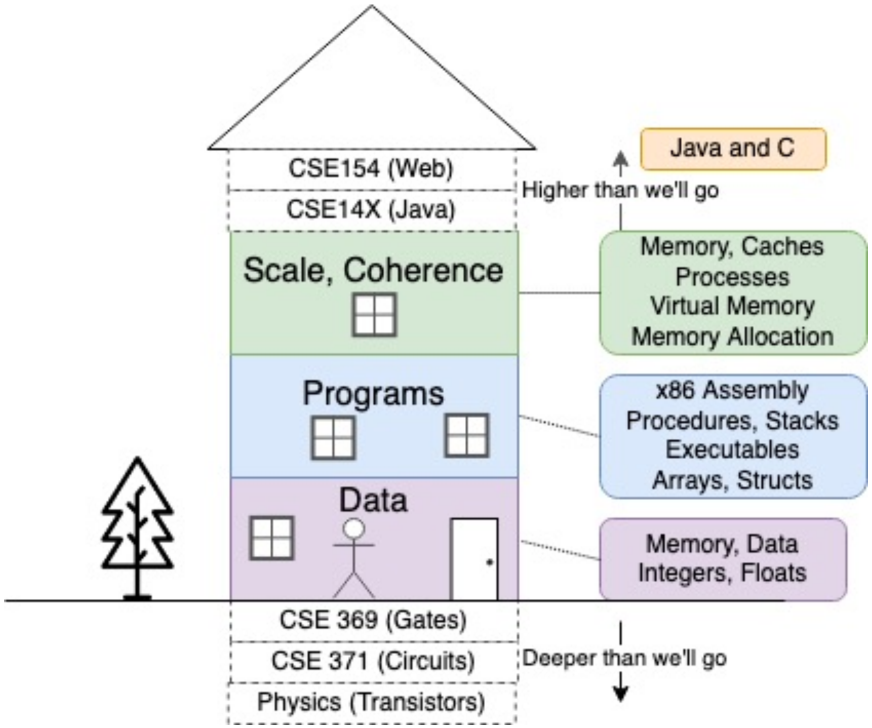
- Admin
  - Explore/read website *thoroughly*
  - Check that you can access Ed Discussion & Lessons
  - **Get your machine set up to access the CSE Linux environment (CSE VM or attu) *as soon as possible***
  - Optional: CSE 391: System & Software Tools
- Assignments
  - Pre-Course Survey and hw0 due Wednesday (6/23) – 8:00pm
  - Hw1 due Friday (6/25) – 8:00pm
  - Lab 0 due Monday (6/28) – 8:00pm
  - Readings due before each lecture – 10am
  - Lecture activities from today are due before NEXT lecture -- 10am

Course Overview



First Floor: Data

- How do we represent data (strings, numbers) computationally? What limits exist? Why?
- What values were encoded into data representations?



Lecture Outline

- Course Introduction
- Course Policies
- **Binary**
  - **Decimal, Binary, and Hexadecimal**
  - **Base Conversion**
  - **Binary Encoding**

Decimal Numbering System

- Ten symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Represent larger numbers as sequence of digits
  - Each digit is one of the available symbols
- Example: 7061 in decimal (base 10)
  - $7061_{10} = (7 \times 10^3) + (0 \times 10^2) + (6 \times 10^1) + (1 \times 10^0)$

Octal Numbering System

- Eight symbols: 0, 1, 2, 3, 4, 5, 6, 7
  - Notice that we no longer use 8 or 9
- Base comparison:
  - Base 10: `0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12…`
  - Base 8: `0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14…`
- Example: What is 7061_8 in base 10?
  - $7061_8 = (7 \times 8^3) + (0 \times 8^2) + (6 \times 8^1) + (1 \times 8^0) = 3633_{10}$

Warmup Question

- What is 34_8 in base 10?

✨ 32_{10}

🦗 34_{10}

🥶 7_{10}

🐈 28_{10}

🧄 35_{10}

Binary and Hexadecimal

- Binary is base 2
  - Symbols: 0, 1
  - Convention: $2_{10} = 10_2 =$ 0b10
- Example: What is 0b110 in base 10?
  - 0b110 = $110_2 = (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) = 6_{10}$
- Hexadecimal (hex, for short) is base 16
  - Symbols? 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, …? 9, **A, B, C, D, E, F** 🤯
  - Convention: $16_{10} = 10_{16} =$ 0x10
- Example: What is 0xA5 in base 10?
  - 0xA5 = $A5_{16} = (10 \times 16^1) + (5 \times 16^0) = 165_{10}$

Polling Question

- Which of the following orderings is correct?

✨ 0xC < 0b1010 < 11

🦗 0xC < 11 < 0b1010

🥶 11 < 0b1010 < 0xC

🐈 0b1010 < 11 < 0xC

🧄 0b1010 < 0xC < 11

Converting to Base 10

- Can convert from any base *to* base 10
  - 0b110 = $110_2 = (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) = 6_{10}$
  - 0xA5 = $A5_{16} = (10 \times 16^1) + (5 \times 16^0) = 165_{10}$

- We learned to think in base 10, so this is fairly natural for us

- **Challenge:** Convert into other bases (*e.g.* 2, 16)

Challenge Question

- Convert 13_{10} into binary

- Hints:
  - $2^3 = 8$
  - $2^2 = 4$
  - $2^1 = 2$
  - $2^0 = 1$

Converting from Decimal to Binary

- Given a decimal number N:
  1. List increasing powers of 2 from *right to left* until ≥ N
  2. Then from *left to right*, ask is that (power of 2) ≤ N?
     - If **YES**, put a 1 below and subtract that power from N
     - If **NO**, put a 0 below and keep going

- Example: 13 to binary

$2^4=16$	$2^3=8$	$2^2=4$	$2^1=2$	$2^0=1$

Converting from Decimal to Base B

- Given a decimal number N:
  1. List increasing powers of B from *right to left* until ≥ N
  2. Then from *left to right*, ask is that (power of B) ≤ N?
     - If **YES**, put *how many* of that power go into N and subtract from N
     - If **NO**, put a 0 below and keep going

- Example: 165 to hex

16^2=256	16^1=16	16^0=1

Converting Binary ↔ Hexadecimal

- Hex → Binary
  - Sub hex, then drop leading zeros
  - Example: 0x2D to binary
    - 0x2 is 0b0010, 0xD is 0b1101
    - Drop two zeros; 0b101101
- Binary → Hex
  - Pad leading zeros until multiple of 4, then sub each group of 4
  - Example: 0b101101
    - Pad to 0b 0010 1101
    - Substitute to get 0x2D

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

Binary → Hex Practice

- Convert 0b100 1101 1010 1101
  - How many digits?
  - Pad?
  - Substitute?

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

Base Comparison

- Why does all of 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 of the amazing things that they do!

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

- **You can represent *anything* countable using numbers!**
  - *Though, not everything is countable*
  - 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?
  - N binary digits to represent n things, where $2^N \geq n$
  - Ex: 5 binary digits for alphabet because $2^5 = 32 > 26$

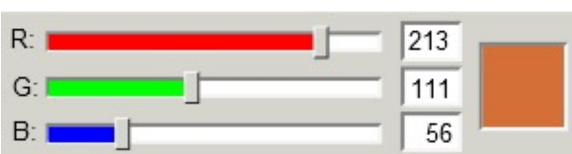
- A binary digit is known as a bit
- A group of 4 bits (1 hex digit) is called a nibble
- A group of 8 bits (2 hex digits) is called a byte
  - 1 bit → 2 things, 1 nibble → 16 things, 1 byte → 256 things

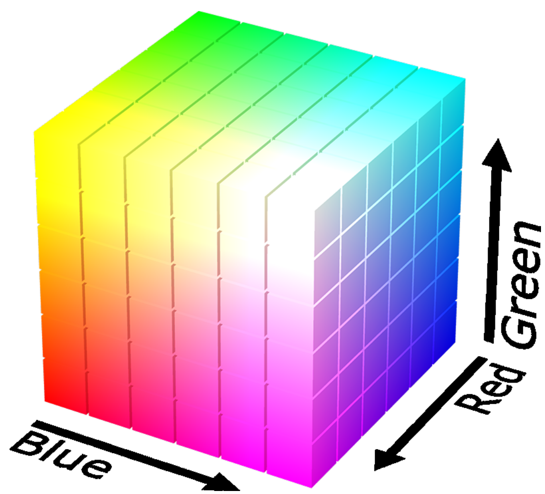
So What's It Mean?

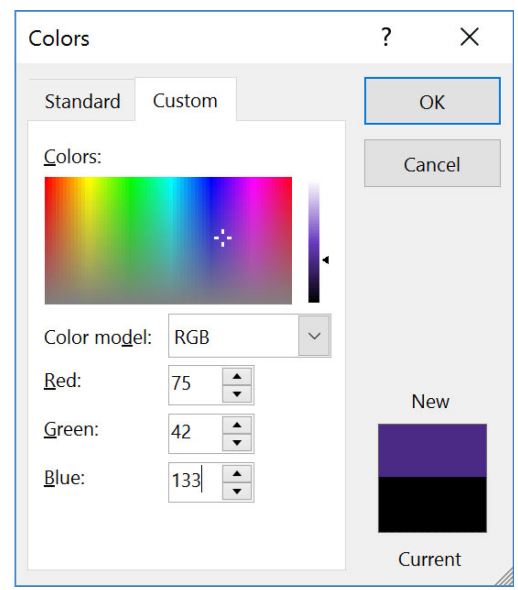
- *A sequence of bits can have many meanings!*
- Consider the hex sequence 0x4E6F21
  - Common interpretations include:
    - The decimal number 5140257
    - The characters "No!"
    - The background color of this slide
    - The real number 7.203034×10^{-39}
- It is up to the program/programmer to decide how to interpret the sequence of bits

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 – Digital Text

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

What's Missing?

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 [illegible]g)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start [illegible]	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end [illegible]	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end [illegible]	[illegible]	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (en[illegible]	[illegible]	[illegible]	[illegible]	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (ack[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	[illegible]	[illegible]	[illegible]	[illegible]	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new li[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	[illegible]	[illegible]	[illegible]	[illegible]	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	[illegible]	[illegible]	[illegible]	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	[illegible]	v	v
23	17	027	ETB (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

Source: www.LookupTables.com

How do encodings get created?

- Usually, a committee gets together
  - Collectively decides on design priorities
  - Collectively creates standard
- This committee is usually made up of senior, well-established members of large, powerful companies*
- Members bring existing ideas, company priorities

*Also, similarly powerful academics, industry researchers (e.g. Bell Labs)

Computing Standards: Qualification

- ASCII: should encode all american digital text
- Created in 1963, who was well-established?

Robert W. Bemer

Hugh McGregor Ross

Computing Standards: Qualification

- ASCII: should encode all american digital text
- Created in 1963, who was well-established?
- White/Cis/straight men, English-primary

- Only their interests were represented when deciding on priorities!
- ASCII: Represent everything on an american typewriter, as efficiently as possible
- Unicode: “Universal” language, still with problems

ASCII Design Goals

- Represent everything on an american typewriter, as efficiently as possible
  - Fewer bits for encoding is better!
  - Memory was expensive, 32KB in brand new machines
  - *Economic incentive to be efficient*
- Organize similar characters together
  - Numbers, uppercase, lowercase, then other stuff

Standards always “encode” the priorities of their creators into data!

Breakout rooms!

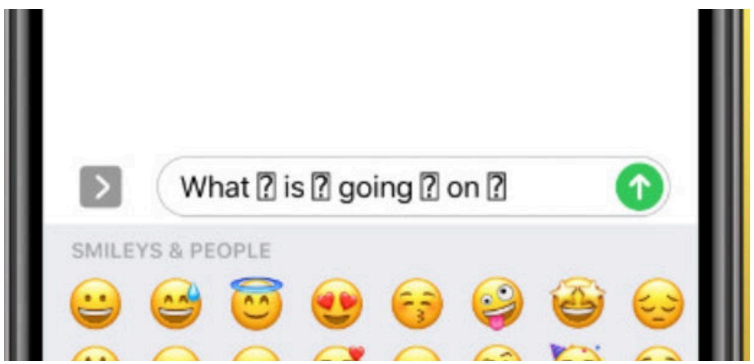
- Join any breakout corresponding to your section
  - Section AA – Any room from A1 to A6
- I'll bring y'all back in 5 minutes

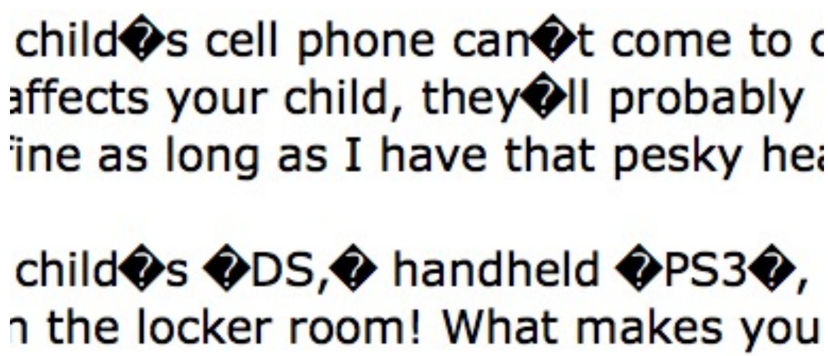
If you designed ASCII, what would you have done?

We'll share out with Ohyay reactions!

But…they fixed it, right?

- Unicode: “Universal language” uses 8-32 bits
  - ASCII uses 7, for reference
- Unicode still has issues!
  - OS/Applications need to support new characters
  - ASCII’s still around, sometimes apps “guess” wrong





Standards stick around, consider priorities carefully!

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 bits (binary)
- Binary encoding can represent countable things!
  - Program needs to know how to interpret the bits
  - Encodings aren't "neutral", priorities are baked in

Learning Objectives (added late)

By the end of this lecture, you should be able to:

- Explain the house of computing, what exists, and what you might need to know before remodeling
- Understand the course policies, and where to look if you forget something (the syllabus!)
- Convert between decimal, binary, and hexadecimal numbers
- Explain how encodings and standards get created, and how that process can be problematic