The Hardware/Software Interface

CSE 351 Autumn 2021

Instructor:

Justin Hsia

Teaching Assistants:

Allie Pfleger
Anirudh Kumar
Assaf Vayner
Atharva Deodhar
Celeste Zeng
Dominick Ta
Francesca Wang
Hamsa Shankar
Isabella Nguyen
Joy Dang
Julia Wang
Maggie Jiang
Monty Nitschke
Morel Fotsing
Sanjana Chintalapati

http://xkcd.com/676/

Lecture Outline

- **Course Introduction**
- Course Policies
  - Return to in-person instruction
  - https://courses.cs.washington.edu/courses/cse351/21au/syllabus
- Binary and Numerical Representation

Introductions: Course Staff

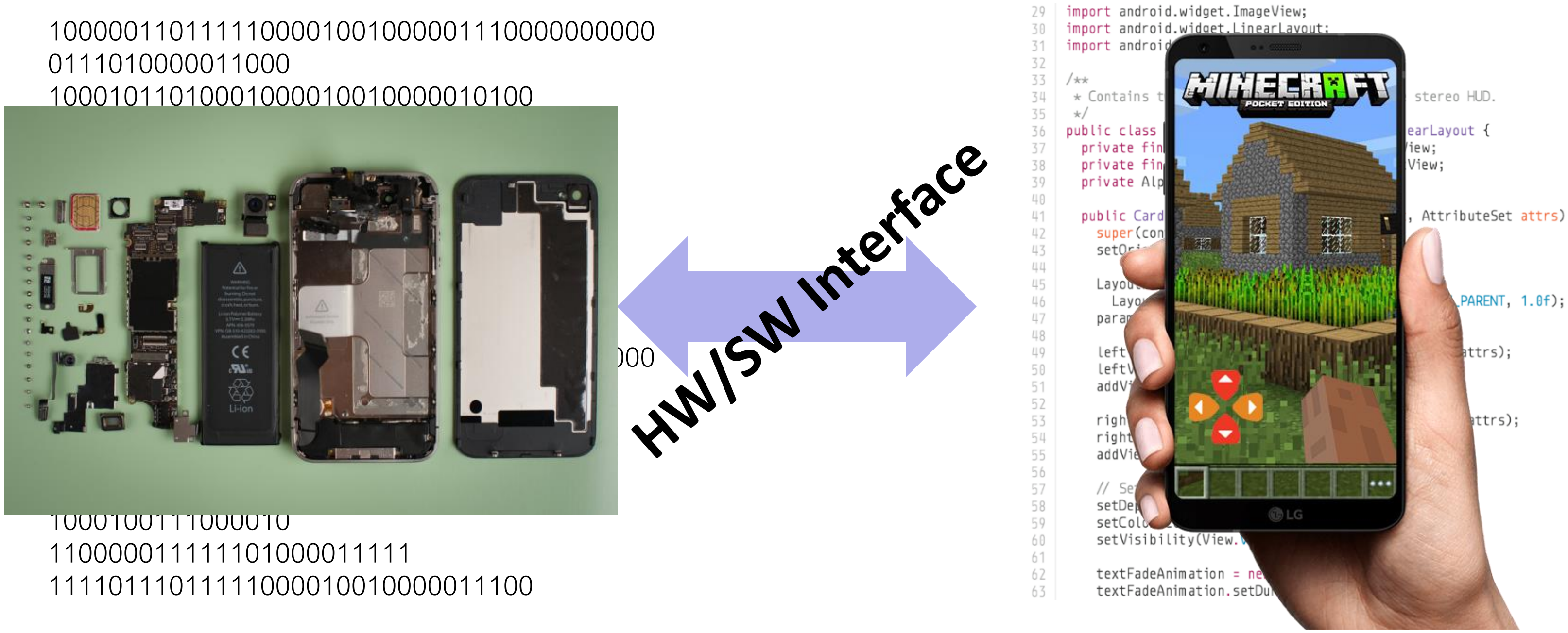
- Instructor: just call me Justin
  - CSE Associate Teaching Professor – 7th time teaching 351
  - Raising an infant this quarter (), will be tired
- TAs:

  - 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!

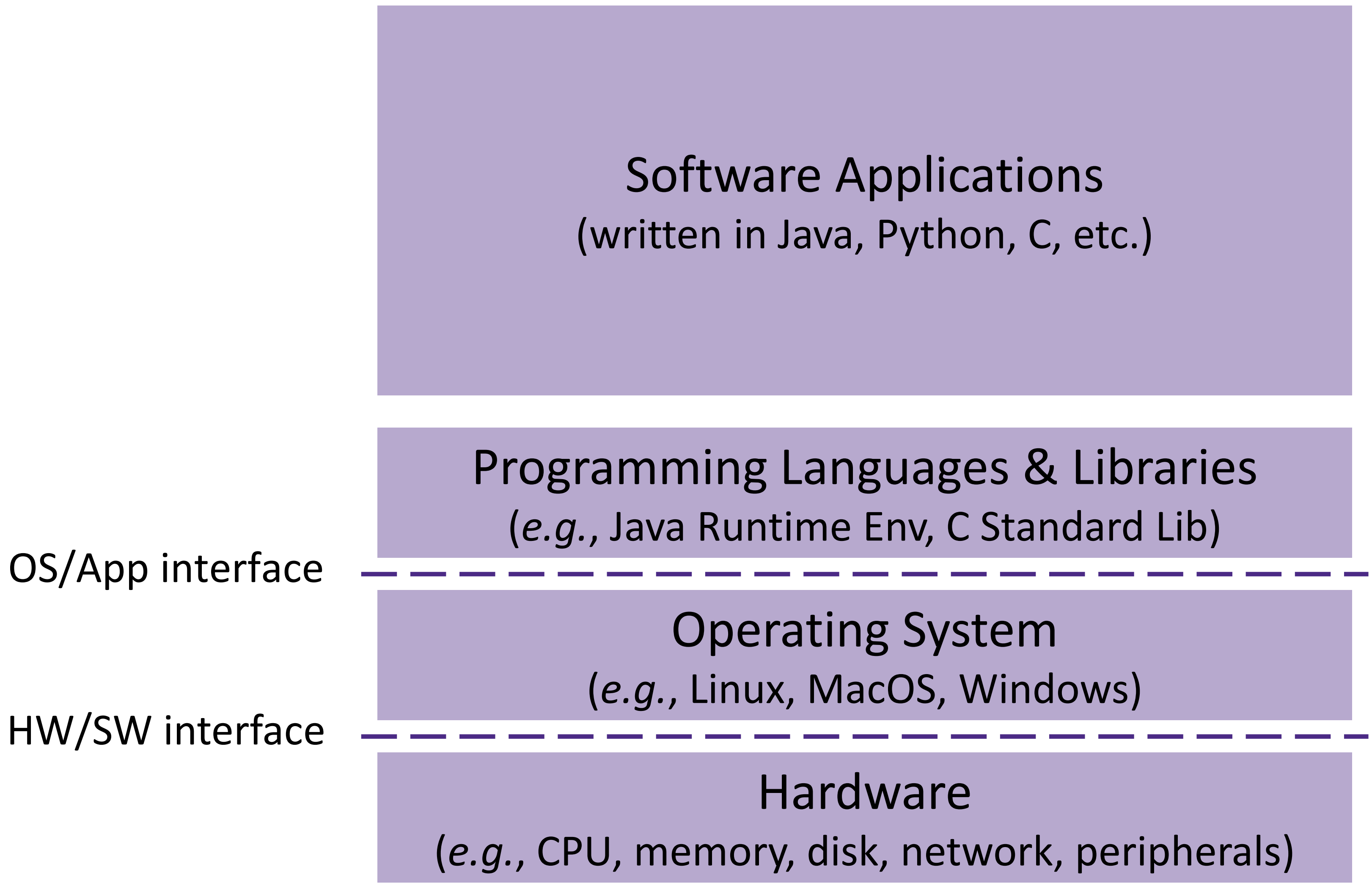
- ~320 students registered, split across two lectures
- CSE majors, ECE majors, and more
  - Most of you will find almost everything in the course new
  - Many of you are new to CSE and/or UW (and campus)!
- Get to know each other! Help each other out!
  - Science says that learning happens best in groups
  - 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 CSE351!



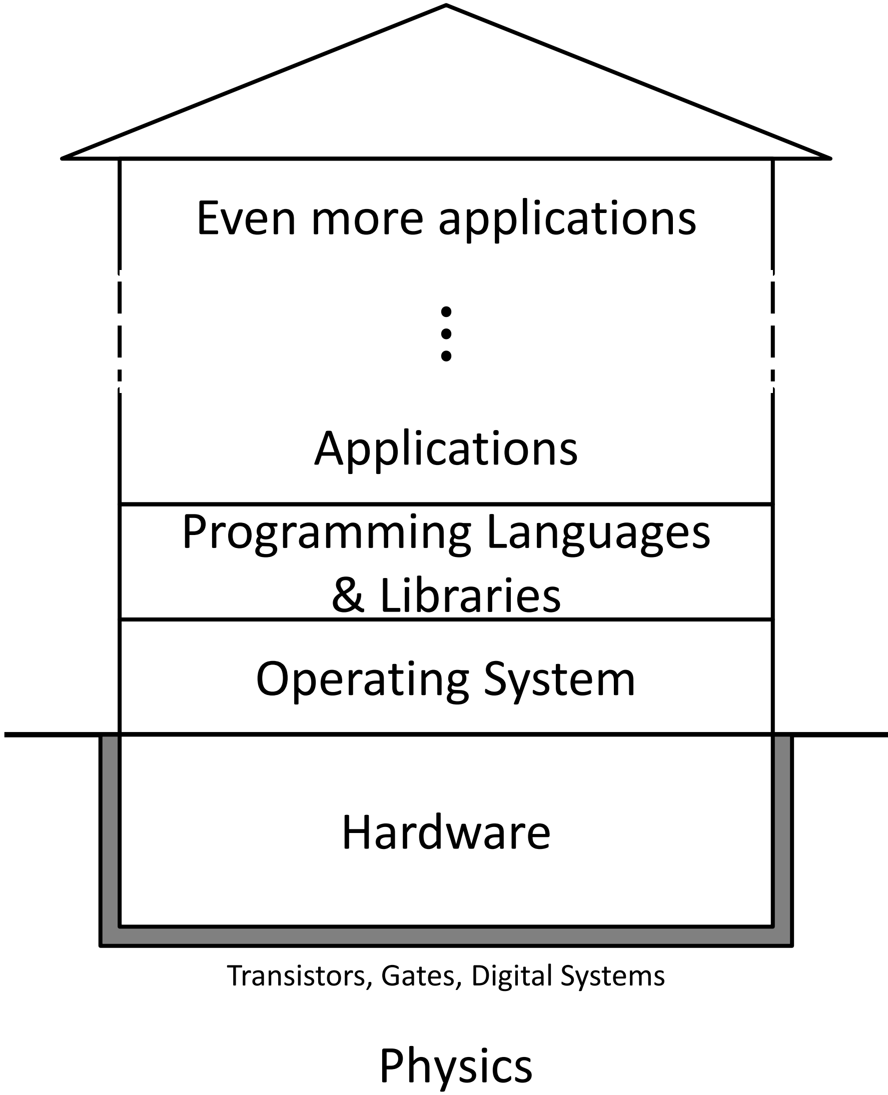
- Our goal is to teach you the key abstractions “under the hood”
  - How does your source code become something that your computer understands?
  - What happens as your computer is executing one or more processes?

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 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

- 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.*, # of cores, size of cache)

Lecture Outline

- Course Introduction
- **Course Policies**
  - Return to in-person instruction
  - https://courses.cs.washington.edu/courses/cse351/21au/syllabus
- Binary and Numerical Representation

Bookmarks

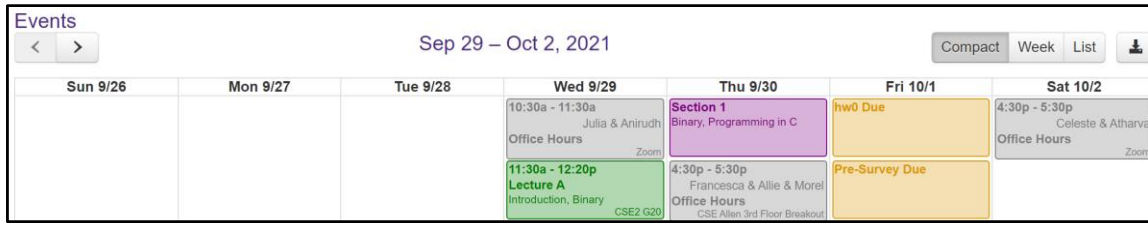
- Website: https://courses.cs.washington.edu/courses/cse351/21au/
  - Schedule, policies, materials, videos, assignment specs, etc.
- Ed Course: https://edstem.org/us/courses/7371
  - Discussion: announcements, ask and answer questions
  - Lessons: readings, lecture questions, homework
  - Resources: links to other tools and information
- Linked from website and Ed
  - Canvas: grade book, Zoom links
  - Gradescope: lab submissions
  - Panopto: lecture recordings

Return to In-person Instruction

- You should be prepared for the possibility of suddenly switching to remote instruction (temporarily or indefinitely)
  - This class is designed to allow for asynchronous learning
- Face coverings required during all indoor, in-person interactions (lecture, section, in-person office hours)
  - Short breaks to sip water are okay
- Maintain physical distancing as much as possible
- You are allowed to attend either lecture and any section, provided there is enough seating/room
  - Please give priority to those officially enrolled

Return to In-person Instruction

- Some office hours will be in-person and others virtual
  - Find scheduled office hours on the course website Events weekly view:



  - Zoom meeting links found in Zoom tab within Canvas
    - We encourage you to chat with other students in the lobby if that TAs are in breakout rooms
  - *All* office hours will use a Google Sheets queuing system
  - Allen 3rd floor breakout limited to 19 people, please wait for "Enter" status:

Concept/Clarifications Question Queue (<5 mins)					Debugging Queue (>10 mins)				
Name	TA	Status	Question Description	Time Queued	Name	TA	Status	Question Description	Time Queued
Example 1		Done	Question about floating point encoding range.		Example 2		Done	Lab 5: running into a segfault in mm_malloc after reaching end of the heap.	
Leslie		Done	two's complement negation		Yutong		In Progress	Lab 1a segfault in selection sort	
Gabriela		Enter	bit shifting: logical vs arithmetic		Keysha		Enter	lab 1a withinSameBlock incorrect values	
Ishaan		Enter	endianness		Amadeus		Waiting	Lab 1a selectionSort edge case	

Return to In-person Instruction

- 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

Grading

groupwork allowed

- **Pre-lecture Readings:** 5%
  - Can reveal solution after one attempt (completion)
- **Homework:** 20% total
  - Unlimited submission attempts (autograded correctness)
- **Labs:** 40% total ← partners allowed
  - Last submission graded (correctness)
- **Exams:** Midterm (16%) and Final (16%) ← individual work
  - Take-home; individual, but some discussion permitted
- **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!
    - 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
  - Pre-Course Survey and hw0 due Friday (10/1)
  - hw1 and Lab 0 due Monday (10/4)
  - Pre-lecture readings due before each lecture – 2 pm

Lecture Outline

- Course Introduction
- Course Policies
  - Return to in-person instruction
  - https://courses.cs.washington.edu/courses/cse351/21au/syllabus
- **Binary and Numerical Representation**

Reading Review

- Terminology:
  - numeral, digit, base, symbol, digit position, leading zeros
  - binary, bit, nibble, 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

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

$64 + 0 + 7 = 71$

position: 2 1 0

- What is the decimal number 108 in hex?
  - A. 0x6C
  - B. 0xA8
  - C. 0x108
  - D. 0x612

(base 16)

$16^0=1$

$16^1=16$

$16^2=256$

$108 = 96+12 = 6\times16^1 + 12\times16^0 = 0x6C$

- Represent 0b100110110101101 in hex.

$16 = 2^4$

1 hex digit ⟷ 4 bits

0x4DAD

- Represent 0x3C9 in binary.

0b 0011 1100 1001

↑ could drop leading zeros

Base Comparison

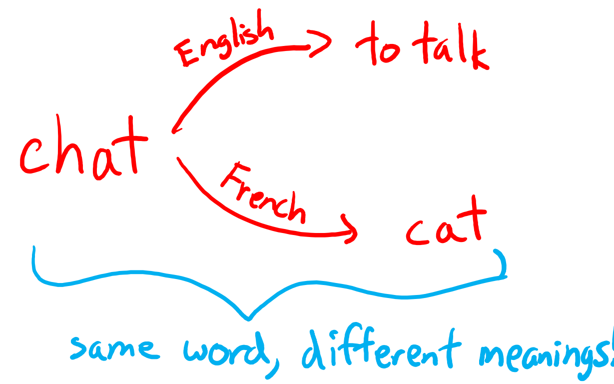
MEMORIZE ME!!!

- 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!
- 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

Binary Encoding

1 bit:
0b0 → thing 1
0b1 → thing 2

2 bits:
0b00 → thing 1
0b01 → thing 2
0b10 → thing 3
0b11 → thing 4

- With n binary digits, how many "things" can you represent? 2^n things
  - 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
- 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 real number 7.203034×10^{-39}
    - The characters "No!"
    - The background color of this slide

- It is up to the program/programmer 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	G	G	103	[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	[illegible]	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	(data link escap[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] acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	[illegible]	[illegible]	[illegible]nchronous idle)	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

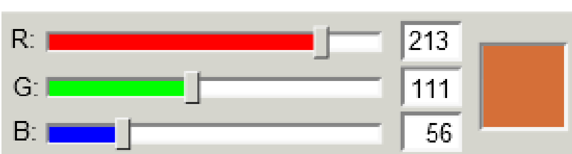
- ASCII Encoding (www.asciitable.com)
  - *American* Standard Code for Information Interchange
- Created in 1963
  - Memory was expensive, 32KB in brand new machines
  - *Economic incentive* to use fewer bits for encoding
- **Design Goals:**
  - Represent everything on an *American* typewriter as *efficiently*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

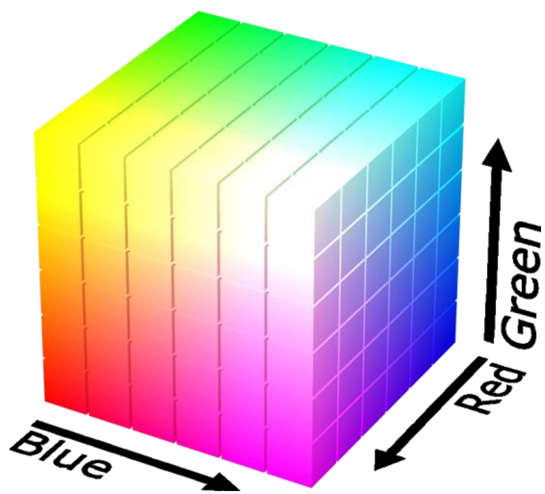
Binary Encoding – Unicode & Emoji

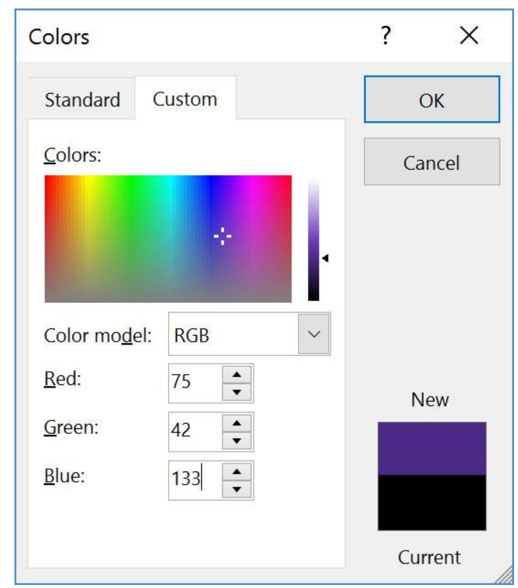
- Unicode Standard is managed by the Unicode Consortium
  - "Universal language" that uses 1-4 bytes to represent a much larger range of characters/languages, including emoji
  - Adds new emojis every year, though adoption often lags: ⍰
    - https://emojipedia.org/new/
- Emojipedia demo: http://www.emojipedia.org
  - 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 actually 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