Lecture 4: Bits, bytes, binary numbers, and the representation of information continued

- computers represent, process, store, copy, and transmit everything as numbers
 - hence "digital computer"
- the numbers can represent anything
 - not just numbers that you might do arithmetic on
- the meaning depends on context
 - as well as what the numbers ultimately represent
 - e.g., numbers coming to your computer or phone from your wi-fi connection could be email, movies, music, documents, apps, Zoom meeting, ...

Some things are intrinsically discrete / digital

another kind of conversion

- letters are converted into numbers when you type on a keyboard
- the letters are stored (a Word document), retrieved (File/Open...), processed (paper is revised), transmitted (submitted by email), printed on paper
- letters and other symbols are inherently discrete
- encoding them as numbers is just assigning a numeric value to each one, without any intrinsic meaning
- what letters and other symbols are included?
- how many digits/letter?
 - determined by how many symbols there are
 - how do we disambiguate if symbols have different lengths?
- how do we decide whose encoding to use?
- the representation is arbitrary
- but everyone has to agree on it
 - if they want to work together

ASCII: American Standard Code for Information Interchange

- an arbitrary but agreed-upon representation for USA
- widely used everywhere

32	space	33	!	34	"	35	#	36	\$	37	00	38	&	39	,
40	(41)	42	*	43	+	44	,	45	_	46	•	47	1
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	9	65	A	66	В	67	С	68	D	69	E	70	F	71	G
72	Н	73	I	74	J	75	K	76	L	77	М	78	Ν	79	0
80	P	81	Q	82	R	83	S	84	Т	85	U	86	V	87	W
88	Х	89	Y	90	Z	91	[92	1	93]	94	^	95	_
96	`	97	a	98	b	99	С	100	d	101	е	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	0
112	р	113	q	114	r	115	S	116	t	117	u	118	v	119	W
120	х	121	У	122	Z	123	{	124		125	}	126	~	127	del

00100000 space 00100001 ! 00100010 " 00100011 # ...

Hexadecimal notation

- binary numbers are bulky
- hexadecimal notation is a shorthand
- it combines 4 bits into a single digit, written in base 16
 - a more compact representation of the same information
- hex uses the symbols A B C D E F for the digits 10 .. 15

0 1 2 3 4 5 6 7 8 9 A B C D E F

0	0000	1	0001	2	0010	3	0011
4	0100	5	0101	6	0110	7	0111
8	1000	9	1001	A	1010	В	1011
С	1100	D	1101	Ε	1110	F	1111

Decimal, binary, hex

dec	bin	hex
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	Α
11	1011	В
12	1100	С
13	1101	D
14	1110	E
15	1111	F

ASCII, using hexadecimal numbers

	0	1	2	3	4	5	6	7	8	9	Α	В	C	D	Ε	F
0	NUL	SOH	STX	EΤΧ	EOT	ENQ	АСК	BEL	BS	ΗT	LF	VT	FF	CR	SO	SI
1	DLE	DC 1	DC2	DC3	DC4	NAK	SYN	ЕТВ	CAN	ΕM	SUB	ESC	FS	GS	RS	US
2	SPC		11	#	\$	%	3	I	()	*	+	,	—	•	1
3	0	1	2	3	4	5	6	7	8	9	•	•	<	=	>	?
4	@	A	В	С	D	Ε	F	G	Η	I	J	Κ	L	Μ	Ν	0
5	Ρ	Q	R	S	Τ	U	V	Ш	X	Y	Ζ	Γ	١]	^	
6	``	а	b	C	d	е	f	g	h	Ī	j	k	I	m	n	0
7	р	q	r	S	t	u	υ	W	X	y	Ζ	{		}	~	DEL

2C80

Coptic

Co	ptic	(unicode.org)
		(

1	2C8	2C9	2CA	2CB	2CC	2CD	2CE	2CF
0	<u>ک</u>	O	П 2046	e Loss	₽	L .	Ψ	108
1	2	e an	П 2041	(1)	P 2001	L. 2001	Ψ	0 1001
2	B	l	P	1000		6	J	b
3	B	l za	P	<u>ب</u>	d	6	J	6
4	Г	K	C	<u>د</u>	3	i	K ICEA	
5	Г	K zes	C	¢ 1005	3	i	R 1005	
6	<u>Д</u> 208	λ **	T	1004	1	پ	tft acres	
7	A.	λ ≈₽	T 2047	1007	1	<u>کتب</u>	टटि	
8	6	M	Y	2	8	<u>11</u> 2006	+P	
9	6	M	Y	2	8	11 2009	*	100
A	5	N	ф	2004	9	4	OC JCEA	/. 2014
в	5	N 238	ф	2000	9	<u>A</u> 1000	200). 2018
с	Z	3	X	m	3	5	00 200	// XFG
D	2	X	X	m 2000	3	б	M	5
E	H	0	Y	Ht -	P		ы	4- 2015
F	H	O Xar	*	H+-	P	Г JCDF	۰ ۱۹۳۶	1017

dec	bin	hex
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	Α
11	1011	В
12	1100	С
13	1101	D
14	1110	E
15	1111	F

2CFF

				1F60	1F61	1F62	1F63	1F64							
Emoji		0	1F600	1F610	1F620	1F630	15640	8	1F608	1F618	1F628	1F638	17648		
			1	1F601	1F611	1F621	1F631	17641	9	1F609	1F619	(C) 1F629	1F639	1F649	
dec	bin	hex	2	(j)	•		•••	•	A	() 1F60A	1F61A	1F62A	1F63A	990 1F64A	
0 1 2	0000 0001 0010	0001 1 0010 2 0011 3 0100 4 0101 5 0110 6 0111 7 1000 8 1001 9 1010 A 1011 B 1100 C 1101 D 1110 E	3	1F602	1F612	1F622	1F632	1642	в	3		8	3	ß	
3 4 5	0011 0100 0101		4	1F603	1F613	1F623	1F633	1F643	С	1F608	1F61B	1F628	1F638	17648	
6 7 8 9	0111 1000		7 8 9 A B	-	1F604	1F614	1F624	1F634	17644		1F60C	1F61C	1F62C	1F63C	1F64C
10 11 12	1010 1011 1100			5	1F605	1F615	1F625	1F635	1F645	D	1F600	1F61D	1F62D	1F63D	1F64D
13 14 15	1101 1110 1111		6	1F606	1F616	1F626	1F636	(🙂) 1F646	E	1F60E	1F61E	1F62E	1F63E	1F64E	
			7	3	••	3	9	Ä	F	•		•	3	A	
				1F607	1F617	1F627	1F637	1F647		1F60F	1F61F	1F62F	1F63F	1F64F	



• TV & computer screens use Red-Green-Blue (RGB) model



- each color is a combination of red, green, blue components
 - R+G = yellow, R+B = magenta, B+G = cyan, R+G+B = white
- for computers, color of a pixel is usually specified by three numbers giving amount of each color, on a scale of 0 to 255
- this is often expressed in hexadecimal so the three components can be specified separately (in effect, as bit patterns)
 - 000000 is black, FFFFFF is white
- printers, etc., use cyan-magenta-yellow[-black] (CMY[K])

= 16 * 1,000,000,000,000,000,000(18, 446, 744, 073, 709, 551, 616)

 $2^{64} = 2^{4} * 2^{60}$

 $2^{32} = 2^{2} * 2^{30}$

= 4 * 1,000,000,000(4,294,967,296)

= 16 * 1,000,000 (16,777,216)

 $2^{24} = 2^{4} * 2^{20}$

Approximations using 2ⁿ

A very important idea

- number of items and number of digits are tightly related:
 - one determines the other
 - maximum number of different items = base number of digits
 - e.g., 9-digit SSN: $10^9 = 1$ billion possible numbers
 - e.g., to represent up to 100 "characters": 2 digits is enough
 - but for 1000 characters, we need 3 digits
 - the same for bits: 9 bits can represent up to $2^9 = 512$ items
- interpretation depends on context
 - without knowing that, we can only guess what numbers mean

Things to remember

- digital devices represent everything as numbers
 - discrete values, not continuous or infinitely precise
- all modern digital devices use binary numbers (base 2) internally
 - instead of decimal (base 10)

it's all bits at the bottom

- a bit is a "binary digit", that is, a number that is either 0 or 1
- computers ultimately represent and process everything as bits

groups of bits represent larger things

- numbers, letters, words, names, pictures, sounds, instructions, ...
- the interpretation of a group of bits depends on their context
- the representation is arbitrary; standards (often) define what it is
- the number of digits used in the representation determines how many different things can be represented
 - number of values = base number of digits
 - e.g., 10², 2¹⁰