Lecture 3: Bits, Bytes, Binary

Bits, bytes, binary numbers, and the representation of information

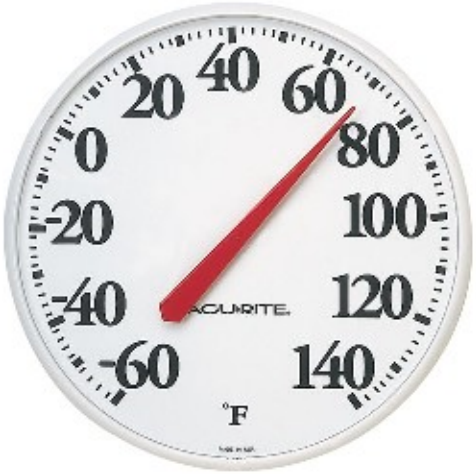
- **computers represent, process, store, copy, and transmit everything as numbers**
  - hence "digital computer"

- **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, ...

Analog versus Digital

- analog: "analogous" or "the analog of"
  - smoothly or continuously varying values
  - volume control, dimmer, faucet, steering wheel
  - value varies smoothly with something else
    - no discrete steps or changes in values
      - small change in one implies small change in another
      - infinite number of possible values
  - the world we perceive is largely analog
- digital: discrete values
  - only a finite number of different values
  - a change in something results in sudden change from one discrete value to another
    - digital speedometer, digital watch, push-button radio tuner, …
  - values are represented as numbers



Transducers

- devices that convert from one representation to another
  - microphone
  - loudspeaker / earphones
  - camera / scanner
  - printer / screen
  - keyboard
  - mouse
  - touch screen
  - etc.

- something is usually lost by conversion (in each direction)
  - the ultimate copy is not as good as the original

Digital pictures

- divide the picture up into a grid of little rectangles (“pixels”)
- assign a different numeric value to each different color value
- the finer the grid and the finer the color distinctions,
  the more accurate the representation will be

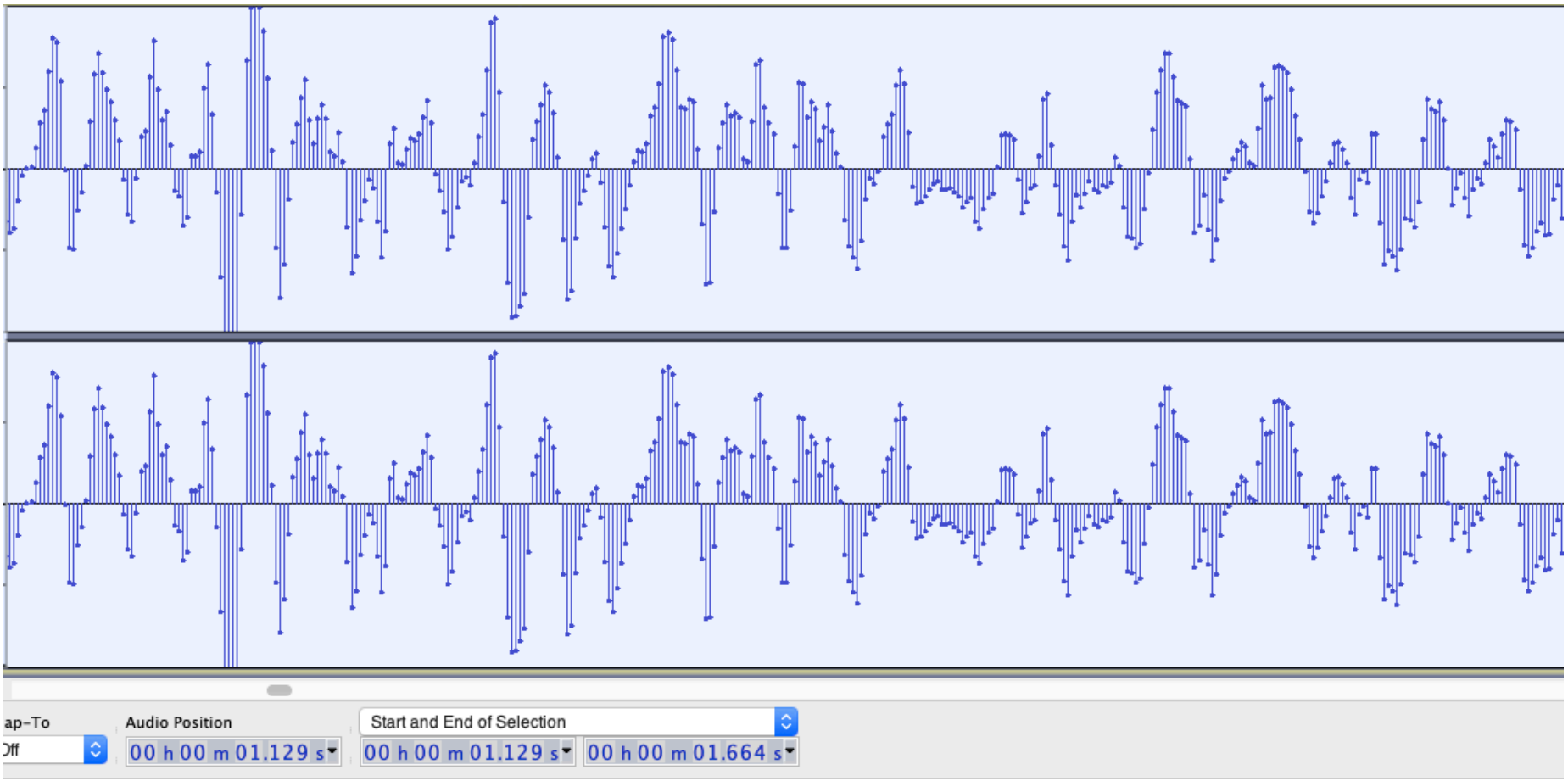
Digital sound

- need to measure intensity/loudness often enough and accurately enough that we can reconstruct it well enough
- higher frequency = higher pitch
- human ear can hear ~ 20 Hz to 20 KHz
  - taking samples at twice the highest frequency is good enough (Nyquist)

- CD audio usually uses
  - 44,100 samples / second
  - accuracy of 1 in 65,536 (= 2^16) distinct levels
  - two samples at each time for stereo
  - data rate is 44,100 x 2 x 16 bits/sample
    = 1,411,200 bits/sec = 176,400 bytes/sec ~ 10.6 MB/minute

- MP3 audio compresses by clever encoding and removal of sounds that won't really be heard
  - data rate is ~ 1 MB/minute

Digital sound sampling (using Audacity)



Why binary numbers? (from von Neumann's paper (§5.2)

In a discussion of the arithmetical organs of a computing machine one is naturally led to a consideration of the number system to be adopted. In spite of the longstanding tradition of building digital machines in the decimal system, we feel strongly in favor of the binary system for our device. Our fundamental unit of memory is naturally adapted to the binary system since we do not attempt to measure gradations of charge at a particular point in the Selectron but are content to distinguish two states.

The flip-flop again is truly a binary device. On magnetic wires or tapes and in acoustic delay line memories one is also content to recognize the presence or absence of a pulse or (if a carrier frequency is used) of a pulse train, or of the sign of a pulse.

A review of how decimal numbers work

- how many digits?

  we use 10 digits for counting: "decimal" numbers are natural for us

  other schemes show up in some areas

    clocks use 12, 24, 60; calendars use 7, 12

    other cultures use other schemes (quatre-vingts)

- what if we want to count to more than 10?

  0 1 2 3 4 5 6 7 8 9

    1 decimal digit represents 1 choice from 10; counts 10 things; 10 distinct values

  00 01 02 … 10 11 12 … 20 21 22 … 98 99

    2 decimal digits represents 1 choice from 100; 100 distinct values

    we usually elide zeros at the front

  000 001 … 099 100 101 … 998 999

    3 decimal digits …

- decimal numbers are shorthands for sums of powers of 10

  $1492 = 1 \times 1000 + 4 \times 100 + 9 \times 10 + 2 \times 1$

  $= 1 \times 10^3 + 4 \times 10^2 + 9 \times 10^1 + 2 \times 10^0$

- counting in "base 10", using powers of 10

Binary numbers: only use the digits 0 and 1 to represent numbers

- just like decimal except there are only two digits: 0 and 1
- everything is based on powers of 2 (1, 2, 4, 8, 16, 32, …)
  - instead of powers of 10 (1, 10, 100, 1000, …)
- counting in binary or base 2:

  0 1

  1 binary digit represents 1 choice from 2; counts 2 things; 2 distinct values

  00 01 10 11

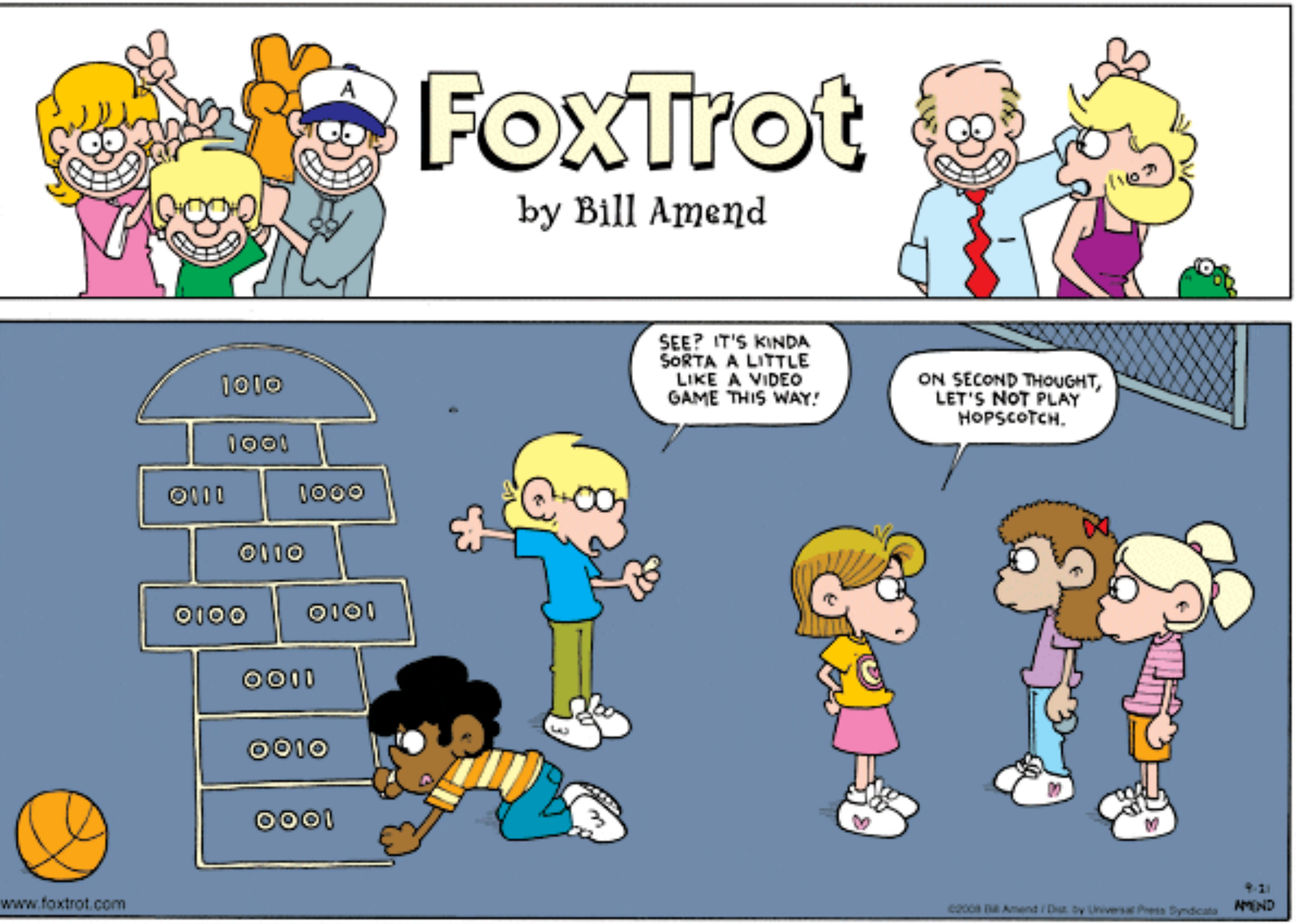
  2 binary digits represents 1 choice from 4; 4 distinct values

  000 001 010 011 100 101 110 111

  3 binary digits …
- binary numbers are shorthands for sums of powers of 2

  $11011 = 1 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$

  $= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
- counting in "base 2", using powers of 2


FoxTrot
by Bill Amend
1010
1001
0111
1000
0110
0100
0101
0011
0010
0001
SEE? IT'S KINDA SORTA A LITTLE LIKE A VIDEO GAME THIS WAY!
ON SECOND THOUGHT, LET'S NOT PLAY HOPSCOTCH.
www.foxtrot.com
©2008 Bill Amend / Dist. by Universal Press Syndicate
9-21
AMEND

Binary (base 2) arithmetic

- works like decimal (base 10) arithmetic, but simpler
- addition:

  0 + 0 = 0
  0 + 1 = 1
  1 + 0 = 1
  1 + 1 = 10

- subtraction, multiplication, division are analogous

Converting binary to decimal

from right to left:

if bit is 1 add corresponding power of 2

i.e. 2^0, 2^1, 2^2, 2^3

(rightmost power is zero)

$$1101 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3$$
$$= 1 \times 1 + 0 \times 2 + 1 \times 4 + 1 \times 8$$
$$= 13$$

Converting decimal to binary

repeat while the number is > 0:
    divide the number by 2
    write the remainder (0 or 1)
    use the quotient as the number and repeat
the answer is the resulting sequence
    in reverse (right to left) order

    divide 13 by 2, write "1", number is 6
    divide 6 by 2, write "0", number is 3
    divide 3 by 2, write "1", number is 1
    divide 1 by 2, write "1", number is 0
    answer is 1101

Python dec to binary conversion (adapted from Abby's version)

```
def dectobinary(num):
  if num == 0:
    return "0"
  binary = ""
  while num > 0:
    remainder = str(num % 2)
    binary = binary + remainder
    num //= 2
  return binary[::-1]

while True:
  num = input("Enter decimal number: ")
  bin = dectobinary(int(num))
  print("Binary representation of " + num + " is " + bin)
```

What's a bit?

- a bit represents one 2-way decision or a choice out of two possibilities
  - yes / no, true / false, on / off, up / down, north / south, ...
- the abstraction of all of these is represented as 0 or 1
  - enough to tell which of TWO possibilities has been chosen
  - a single digit with one of two values
  - hence "binary digit"
  - hence bit
- binary is used in computers because it's easy to make fast, reliable, small devices that have only two states
  - high voltage/low voltage, current flowing/not flowing (chips)
  - electrical charge present/not present (RAM, flash)
  - magnetized this way or that (disks)
  - light bounces off/doesn't bounce off (cd-rom, dvd)
- all information in a computer is stored and processed as bits

Using bits to represent information

- AB / BSE
  - 1 bit
- Fr / So / Jr / Sr
  - 2 bits
- grads, auditors, faculty as well
  - 3 bits
- a unique number for each person in 109
  - 6 bits
- a unique number for each freshman at PU
  - 11 bits
- a unique number for each PU undergrad
  - 13 bits

Powers of two, powers of ten

1 bit = 2 possibilities
2 bits = 4 possibilities
3 bits = 8 possibilities
...
n bits = 2^n possibilities

2^{10} = 1,024 is about 1,000 or 1K or 10^3
2^{20} = 1,048,576 is about 1,000,000 or 1M or 10^6
2^{30} = 1,073,741,824 is about 1,000,000,000 or 1G or 10^9

the approximation is becoming less good
but it's still good enough for estimation

- terminology is often imprecise:
  - " 1K " might mean 1000 or 1024 (10^3 or 2^{10})
  - " 1M " might mean 1000000 or 1048576 (10^6 or 2^{20})

Bytes

- "byte" = a group of 8 bits treated as a unit
  - on modern machines, the fundamental unit of processing and memory addressing
  - can encode any of 2^8 = 256 different values, e.g., numbers 0 .. 255 or a single letter like A or digit like 7 or punctuation like $
    ASCII character set defines values for letters, digits, punctuation, etc.
- group 2 bytes together to hold larger entities
  - two bytes (16 bits) holds 2^{16} = 65,536 values
  - a bigger integer, a character in a larger character set
    Unicode character set defines values for almost all characters anywhere
- group 4 bytes together to hold even larger entities
  - four bytes (32 bits) holds 2^{32} = 4,294,967,296 values
  - an even bigger integer, a number with a fractional part (floating point), a memory address
  - current machines use 64-bit integers and addresses (8 bytes)
    2^{64} = 18,446,744,073,709,551,616

- no fractional bytes: the number of bytes is always an integer

Interpretation of bits and bytes depends on context

- meaning of a group of bits depends on how they are interpreted
- 1 byte could be
  - 1 bit in use, 7 wasted bits (e.g., M/F in a database)
  - 8 bits representing a number between 0 and 255
  - an alphabetic character like W or + or 7
  - part of a character in another alphabet or writing system (2+ bytes)
  - part of a larger number (2 or 4 or 8 bytes, usually)
  - part of a picture or sound
  - part of an instruction for a computer to execute
    - instructions are just bits, stored in the same memory as data
    - different kinds of computers use different bit patterns for their instructions
      - laptop, cellphone, game machine, etc., all potentially different
  - part of the location or address of something in memory
  - ...
- one program's instructions are another program's data
  - when you download a new program from the net, it's data
  - when you run it, it's instructions

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	%	38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
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	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124	|	125	}	126	~	127	del

00010000 space 00010001 ! 00010010 " 00010101 # ...

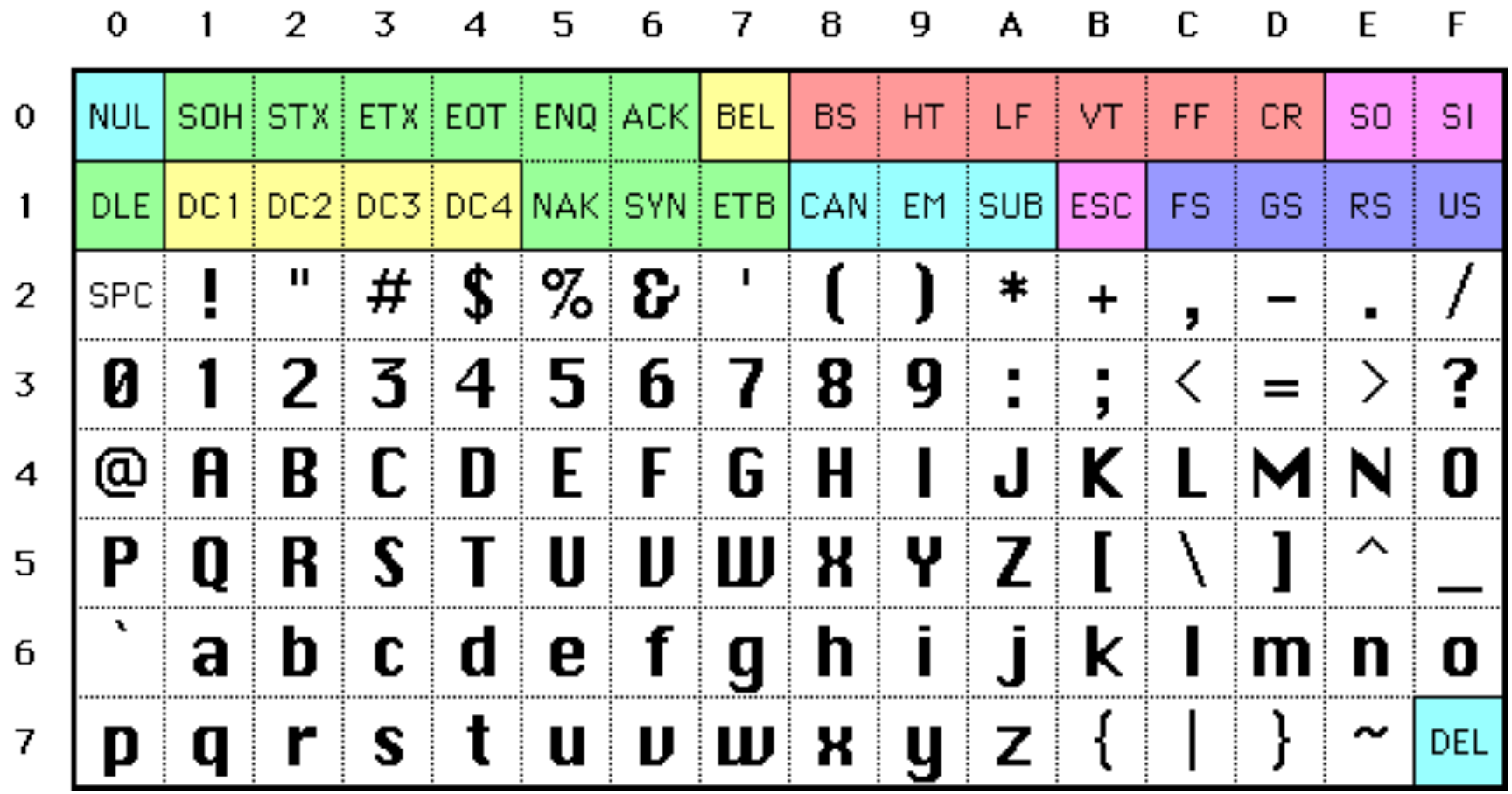
Hexadecimal notation

- 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

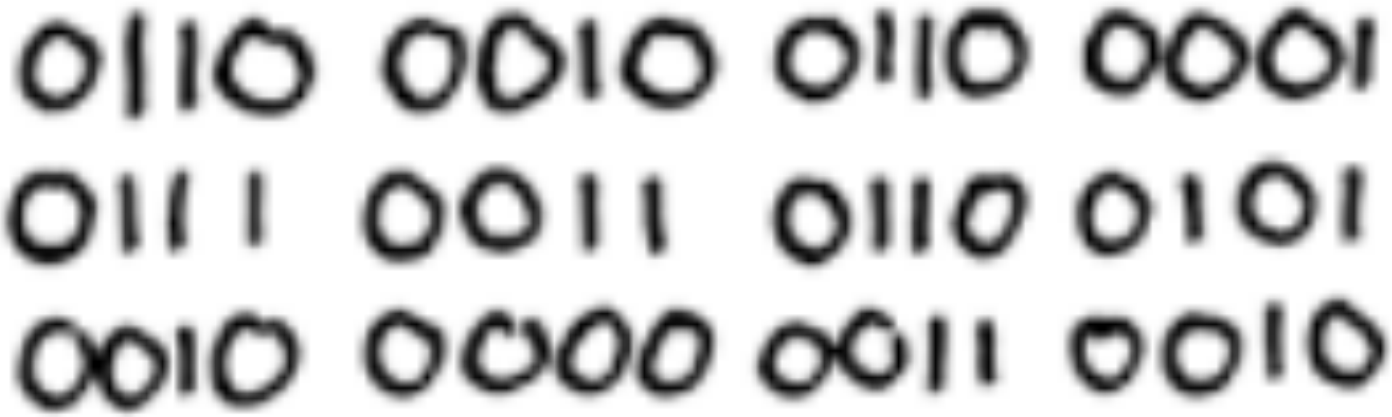
0	0000	1	0001	2	0010	3	0011
4	0100	5	0101	6	0110	7	0111
8	1000	9	1001	A	1010	B	1011
C	1100	D	1101	E	1110	F	1111

ASCII, using hexadecimal numbers



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SPC	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

What does this say?



Coptic (unicode.org)

Other Coptic letters derived from Demotic are encoded in the Greek and Coptic block.

Coptic epact digits and numbers are encoded in the Coptic Epact Numbers block.

Bohairic Coptic letters

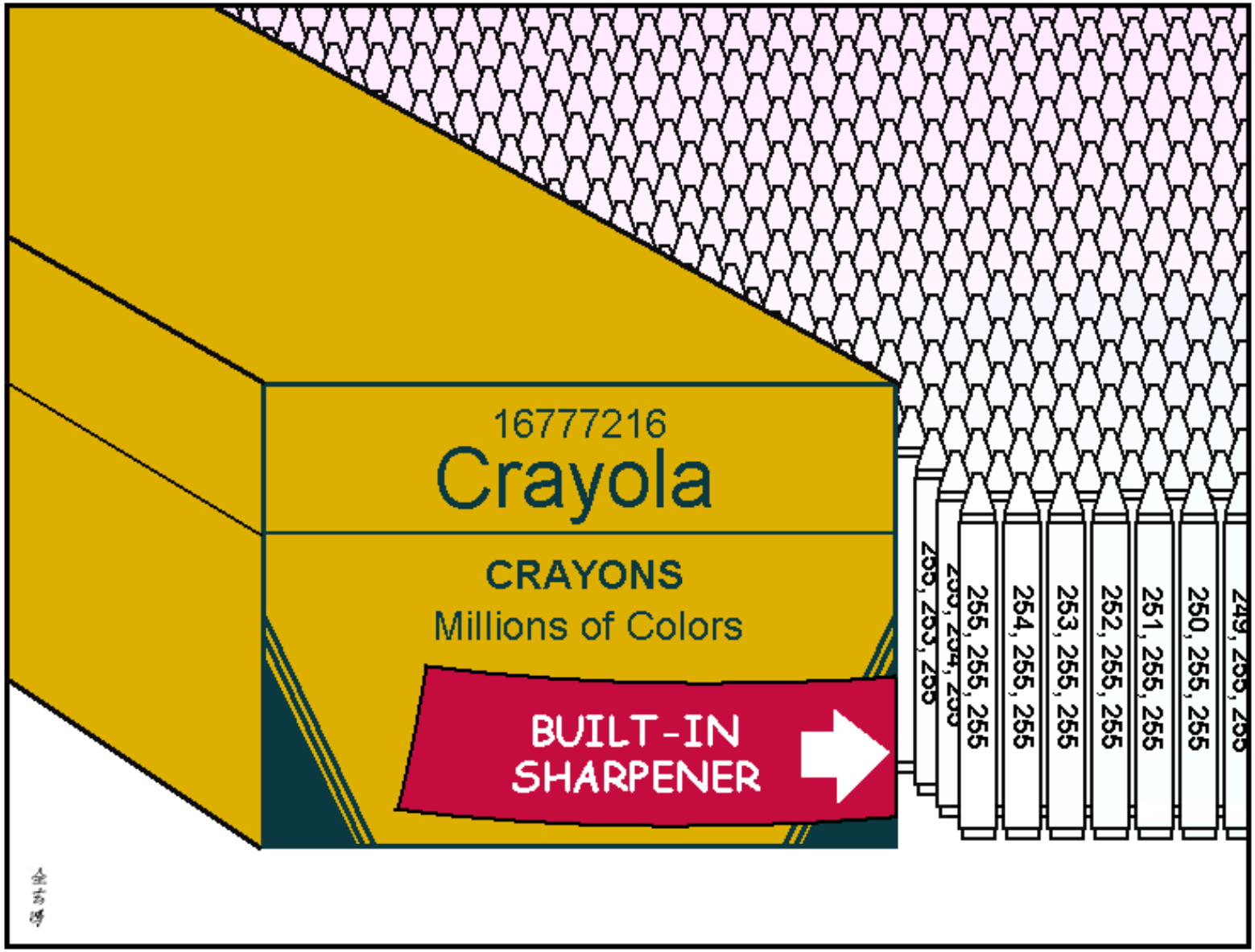
2C80 Ⲁ COPTIC CAPITAL LETTER ALFA
2C81 ⲁ COPTIC SMALL LETTER ALFA
2C82 Ⲃ COPTIC CAPITAL LETTER VIDA
2C83 ⲃ COPTIC SMALL LETTER VIDA
2C84 Ⲅ COPTIC CAPITAL LETTER GAMMA
2C85 ⲅ COPTIC SMALL LETTER GAMMA
2C86 Ⲇ COPTIC CAPITAL LETTER DALDA
2C87 ⲇ COPTIC SMALL LETTER DALDA
2C88 Ⲉ COPTIC CAPITAL LETTER EIE
2C89 ⲉ COPTIC SMALL LETTER EIE
2C8A Ⲋ COPTIC CAPITAL LETTER SOU
2C8B ⲋ COPTIC SMALL LETTER SOU
2C8C Ⲍ COPTIC CAPITAL LETTER ZATA
2C8D ⲍ COPTIC SMALL LETTER ZATA
2C8E Ⲏ COPTIC CAPITAL LETTER HATE
2C8F ⲏ COPTIC SMALL LETTER HATE
2C90 Ⲑ COPTIC CAPITAL LETTER THETHE
2C91 ⲑ COPTIC SMALL LETTER THETHE
2C92 Ⲓ COPTIC CAPITAL LETTER IAUDA
2C93 ⲓ COPTIC SMALL LETTER IAUDA
2C94 Ⲕ COPTIC CAPITAL LETTER KAPA
2C95 ⲕ COPTIC SMALL LETTER KAPA
2C96 Ⲗ COPTIC CAPITAL LETTER LAULA
2C97 ⲗ COPTIC SMALL LETTER LAULA
2C98 Ⲙ COPTIC CAPITAL LETTER MI
2C99 ⲙ COPTIC SMALL LETTER MI
2C9A Ⲛ COPTIC CAPITAL LETTER NI
2C9B ⲛ COPTIC SMALL LETTER NI
2C9C Ⲝ COPTIC CAPITAL LETTER KSI
2C9D ⲝ COPTIC SMALL LETTER KSI
2CB6 Ⲷ COPTIC CAPITAL LETTER CRYPTOGRAMMIC EIE
2CB7 ⲷ COPTIC SMALL LETTER CRYPTOGRAMMIC EIE
2CB8 Ⲹ COPTIC CAPITAL LETTER DIALECT-P KAPA
2CB9 ⲹ COPTIC SMALL LETTER DIALECT-P KAPA
2CBA Ⲻ COPTIC CAPITAL LETTER DIALECT-P NI
2CBB ⲻ COPTIC SMALL LETTER DIALECT-P NI
2CBC Ⲽ COPTIC CAPITAL LETTER CRYPTOGRAMMIC NI
2CBD ⲽ COPTIC SMALL LETTER CRYPTOGRAMMIC NI
2CBE Ⲿ COPTIC CAPITAL LETTER OLD COPTIC OOU
2CBF ⲿ COPTIC SMALL LETTER OLD COPTIC OOU
2CC0 Ⳁ COPTIC CAPITAL LETTER SAMPI
2CC1 ⳁ COPTIC SMALL LETTER SAMPI
2CC2 Ⳃ COPTIC CAPITAL LETTER CROSSED SHEI
2CC3 ⳃ COPTIC SMALL LETTER CROSSED SHEI
2CC4 Ⳅ COPTIC CAPITAL LETTER OLD COPTIC SHEI
2CC5 ⳅ COPTIC SMALL LETTER OLD COPTIC SHEI
2CC6 Ⳇ COPTIC CAPITAL LETTER OLD COPTIC ESH
2CC7 ⳇ COPTIC SMALL LETTER OLD COPTIC ESH
2CC8 Ⳉ COPTIC CAPITAL LETTER AKHMIMIC KHEI
2CC9 ⳉ COPTIC SMALL LETTER AKHMIMIC KHEI
2CCA Ⳋ COPTIC CAPITAL LETTER DIALECT-P HORI
2CCB ⳋ COPTIC SMALL LETTER DIALECT-P HORI
2CCC Ⳍ COPTIC CAPITAL LETTER OLD COPTIC HORI
2CCD ⳍ COPTIC SMALL LETTER OLD COPTIC HORI
2CCE Ⳏ COPTIC CAPITAL LETTER OLD COPTIC HA
2CCF ⳏ COPTIC SMALL LETTER OLD COPTIC HA
2CD0 Ⳑ COPTIC CAPITAL LETTER L-SHAPED HA
2CD1 ⳑ COPTIC SMALL LETTER L-SHAPED HA
2CD2 Ⳓ COPTIC CAPITAL LETTER OLD COPTIC HEI
2CD3 ⳓ COPTIC SMALL LETTER OLD COPTIC HEI
2CD4 Ⳕ COPTIC CAPITAL LETTER OLD COPTIC HAT
2CD5 ⳕ COPTIC SMALL LETTER OLD COPTIC HAT
2CD6 Ⳗ COPTIC CAPITAL LETTER OLD COPTIC GANGIA
2CD7 ⳗ COPTIC SMALL LETTER OLD COPTIC GANGIA
2CD8 Ⳙ COPTIC CAPITAL LETTER OLD COPTIC DJA
2CD9 ⳙ COPTIC SMALL LETTER OLD COPTIC DJA

Color

- 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])

"More than 16 million colors!"



A very important idea

- number of items and number of digits are tightly related:
  - one determines the other
  - maximum number of different items = base $^{\text{number of digits}}$
  - 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

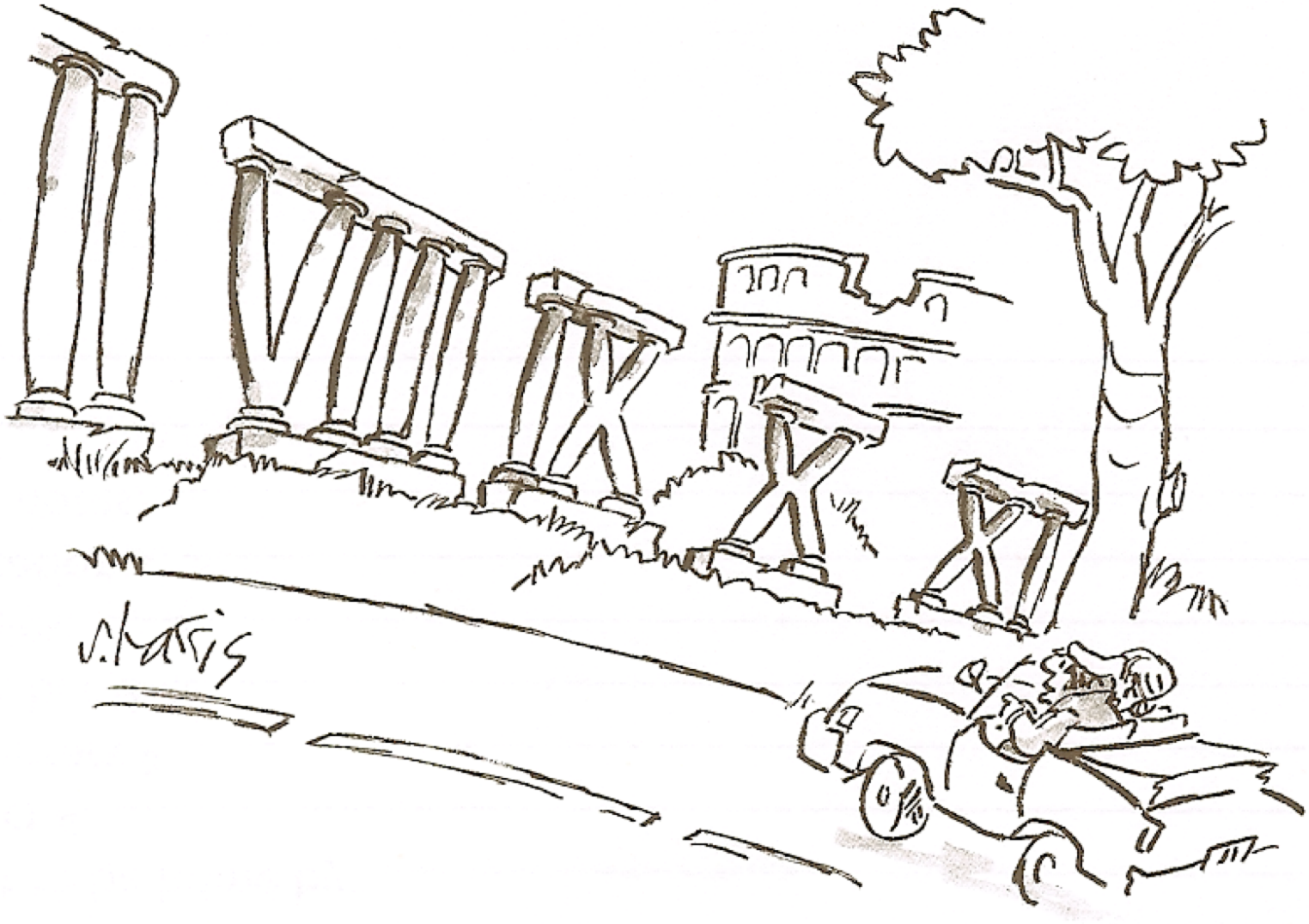
  - the same for bits: 9 bits can represent up to $2^9 = 512$ items

- 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)
  - 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 $^{\text{number of digits}}$
  - e.g., 10^2 , 2^{10}

THE NEW YORKER



"If you think this is annoying, just wait. In a few miles, they switch to binary."