

Encodings

Midterm Review

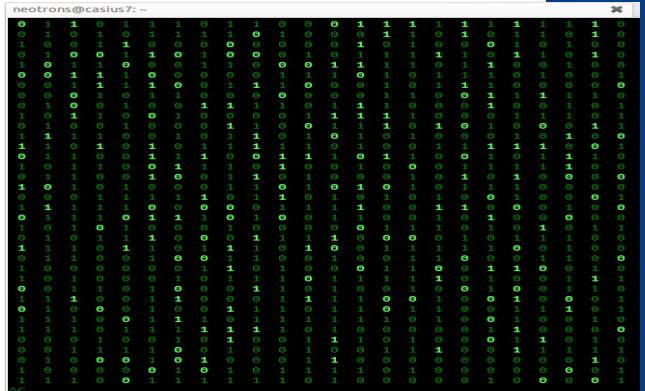
External vs. Internal Representation

- External representation:
 - Convenient for programmer
 - Decimal (base 10)
 - Internal representation:
 - Actual representation of data in the computer's memory:
Always binary (1's and 0's)



Binary representation (base 2)

- On a computer all data is stored in binary
- Only two symbols: 0 and 1
- Each position is called a *bit*
- *Bits take up space*
- 8 bits make a *byte*
- *Example of a 4-bit number*



A terminal window titled "neotrons@casius: ~" displaying a large amount of binary data (0s and 1s) on a black background with green text. The data is organized into several columns of 8-bit bytes.

- Actually the data is voltages
- We use the abstraction:
 - High voltage: 1 (true)
 - Low voltage: 0 (false)

Positional encoding for non-negative numbers

- Each position represents some power of the base
- Decimal (Base 10), Digits (0-9)
- Binary (Base 2), Digits (0,1)
- Hex (Base 16), Digits (0-9, A-F)

$101_5 = ?$ In decimal

A. 26

B. 51

C. 126

D. 130

Converting between binary and decimal

Binary to decimal: $10110_2 = ?_{10}$

Decimal to binary: $34_{10} = ?_2$



Hex to binary

- Each hex digit corresponds directly to four binary digits
- Programmers love hex, why?

$25B_{16} = ?$ In binary

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Hexadecimal to decimal

$25B_{16} = ? \text{ Decimal}$



Hexadecimal to decimal

- Use polynomial expansion
- $25B_{16} = 2*256 + 5*16 + 11*1 = 512 + 80 + 11$
 $= 603$
- Decimal to hex: $36_{10} = ?_{16}$

Binary to hex: 1000111100

- A. 8F0
- B. 23C
- C. None of the above

BIG IDEA: Bits can represent anything!!

Numbers Binary Code

0

1

2

3

How many (minimum) bits are required to represent the numbers 0 to 3?

What is the maximum positive value that can be stored in a byte?

- A. 127
- B. 128
- C. 255
- D. 256

BIG IDEA: Bits can represent anything!!

Colors

Binary code

Red

Green

Blue

How many (minimum) bits are required to represent the three colors?

BIG IDEA: Bits can represent anything!!

Characters

‘a’

‘b’

‘c’

‘d’

‘e’

N bits can represent at most 2^N things

BIG IDEA: Bits can represent anything!!

- Logical values?
 - $0 \Rightarrow$ False, $1 \Rightarrow$ True
- colors ?
- Characters?
 - 26 letters \Rightarrow 5 bits ($2^5 = 32$)
 - upper/lower case + punctuation
 \Rightarrow 7 bits (in 8) ("ASCII")
 - standard code to cover all the world's languages \Rightarrow 8,16,32 bits ("Unicode")
www.unicode.com
- locations / addresses? commands?
- MEMORIZE: N bits \Leftrightarrow at most 2^N things

Red

Green

Blue



Encodings in Python

`ord('a')` -> returns the ascii value of 'a' (97)

`chr(97)` -> returns the character corresponding with ascii value 97

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	40	 	space	64	40 100	100	@	0	96	60 140	140	`	'
1	1 001	SOH	(start of heading)	33	21 041	41	!	!	65	41 101	101	A	A	97	61 141	141	a	a
2	2 002	STX	(start of text)	34	22 042	42	"	"	66	42 102	102	B	B	98	62 142	142	b	b
3	3 003	ETX	(end of text)	35	23 043	43	#	#	67	43 103	103	C	C	99	63 143	143	c	c
4	4 004	EOT	(end of transmission)	36	24 044	44	$	\$	68	44 104	104	D	D	100	64 144	144	d	d
5	5 005	ENQ	(enquiry)	37	25 045	45	%	%	69	45 105	105	E	E	101	65 145	145	e	e
6	6 006	ACK	(acknowledge)	38	26 046	46	&	&	70	46 106	106	F	F	102	66 146	146	f	f
7	7 007	BEL	(bell)	39	27 047	47	'	'	71	47 107	107	G	G	103	67 147	147	g	g
8	8 010	BS	(backspace)	40	28 050	40	((72	48 110	110	H	H	104	68 150	150	h	h
9	9 011	TAB	(horizontal tab)	41	29 051	41))	73	49 111	111	I	I	105	69 151	151	i	i
10	A 012	LF	(NL line feed, new line)	42	2A 052	42	*	*	74	4A 112	112	J	J	106	6A 152	152	j	j
11	B 013	VT	(vertical tab)	43	2B 053	43	+	+	75	4B 113	113	K	K	107	6B 153	153	k	k
12	C 014	FF	(NP form feed, new page)	44	2C 054	44	,	,	76	4C 114	114	L	L	108	6C 154	154	l	l
13	D 015	CR	(carriage return)	45	2D 055	45	-	-	77	4D 115	115	M	M	109	6D 155	155	m	m
14	E 016	SO	(shift out)	46	2E 056	46	.	.	78	4E 116	116	N	N	110	6E 156	156	n	n
15	F 017	SI	(shift in)	47	2F 057	47	/	/	79	4F 117	117	O	O	111	6F 157	157	o	o
16	10 020	DLE	(data link escape)	48	30 060	48	0	0	80	50 120	120	P	P	112	70 160	160	p	p
17	11 021	DC1	(device control 1)	49	31 061	49	1	1	81	51 121	121	Q	Q	113	71 161	161	q	q
18	12 022	DC2	(device control 2)	50	32 062	49	2	2	82	52 122	122	R	R	114	72 162	162	r	r
19	13 023	DC3	(device control 3)	51	33 063	51	3	3	83	53 123	123	S	S	115	73 163	163	s	s
20	14 024	DC4	(device control 4)	52	34 064	52	4	4	84	54 124	124	T	T	116	74 164	164	t	t
21	15 025	NAK	(negative acknowledge)	53	35 065	53	5	5	85	55 125	125	U	U	117	75 165	165	u	u
22	16 026	SYN	(synchronous idle)	54	36 066	54	6	6	86	56 126	126	V	V	118	76 166	166	v	v
23	17 027	ETB	(end of trans. block)	55	37 067	55	7	7	87	57 127	127	W	W	119	77 167	167	w	w
24	18 030	CAN	(cancel)	56	38 070	56	8	8	88	58 130	130	X	X	120	78 170	170	x	x
25	19 031	EM	(end of medium)	57	39 071	57	9	9	89	59 131	131	Y	Y	121	79 171	171	y	y
26	1A 032	SUB	(substitute)	58	3A 072	58	:	:	90	5A 132	132	Z	Z	122	7A 172	172	z	z
27	1B 033	ESC	(escape)	59	3B 073	59	;	:	91	5B 133	133	[[123	7B 173	173	{	{
28	1C 034	FS	(file separator)	60	3C 074	60	<	<	92	5C 134	134	\	\	124	7C 174	174	|	
29	1D 035	GS	(group separator)	61	3D 075	61	=	=	93	5D 135	135]]	125	7D 175	175	}	}
30	1E 036	RS	(record separator)	62	3E 076	62	>	>	94	5E 136	136	^	^	126	7E 176	176	~	~
31	1F 037	US	(unit separator)	63	3F 077	63	?	?	95	5F 137	137	_	_	127	7F 177	177		DEL

Source: www.LookupTables.com

ASCII Table

Unicode

- Universal character encoding (extends ASCII to handle other languages)

```
>>> chinese = '\u4e16\u754c\u60a8\u597d!'
>>> print(chinese)
世界您好!
```

- Includes all ASCII characters using the same ascii encoding

```
>>> print('\u0048\u0049')
HI
```

```
>>>
```

Midterm Review

- We'll go over a practice exam from Fall 2017:

<https://drive.google.com/file/d/1OcgXYZhtfrNQgtMdyZ1a8koHKbO9Ll6c/view?usp=sharing>

Hand out:

https://drive.google.com/file/d/1wOLtOH3vLANqUEG_tBmA6cl9ld2eQ0V3/view?usp=sharing