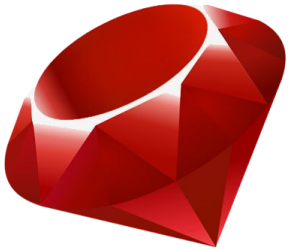
Ruby Monstas

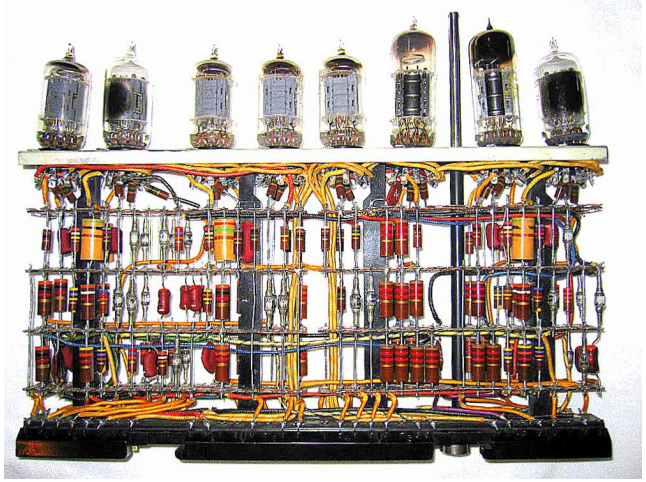
Session 12: Interlude

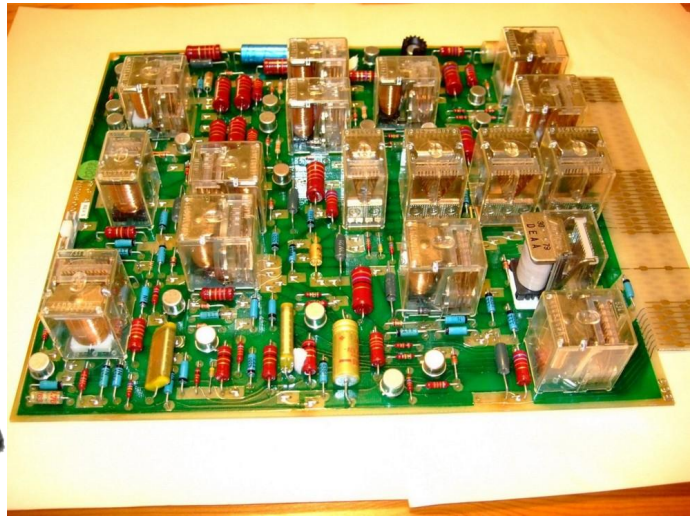
Agenda

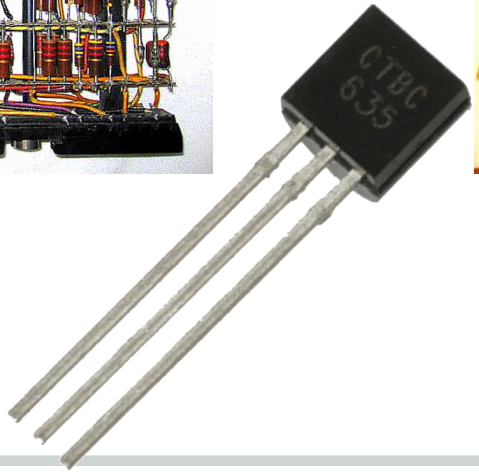
- Binary
- Data Representation
- Encoding



Computers work with 0 and 1







Numbers in binary

$142_{10} = 2 * 1 + 4 * 10 + 1 * 100$

$142_{10} = 2 * 10^0 + 4 * 10^1 + 1 * 10^2$

$10001110_2 = 0 * 2^0 + 1 * 2^1 + 1 * 2^2 + 1 * 2^3 +$
$0 * 2^4 + 0 * 2^5 + 0 * 2^6 + 1 * 2^7$

$10001110_2 = 0 + 2 + 4 + 8 + 0 + 0 + 0 + 128$

$10001110_2 = 142_{10}$

Doing math in binary

```
  10001110        142
+ 00101011      +  43
= 10111001        = 185
```

Floating point numbers

How do we represent floating point numbers?

$1.2345 = 12345 * 10^{-4}$

$\text{Significand} * \text{Base}^{\text{Exponent}}$

Floating point numbers

IEEE 754

Significand * Base$^{\text{Exponent}}$

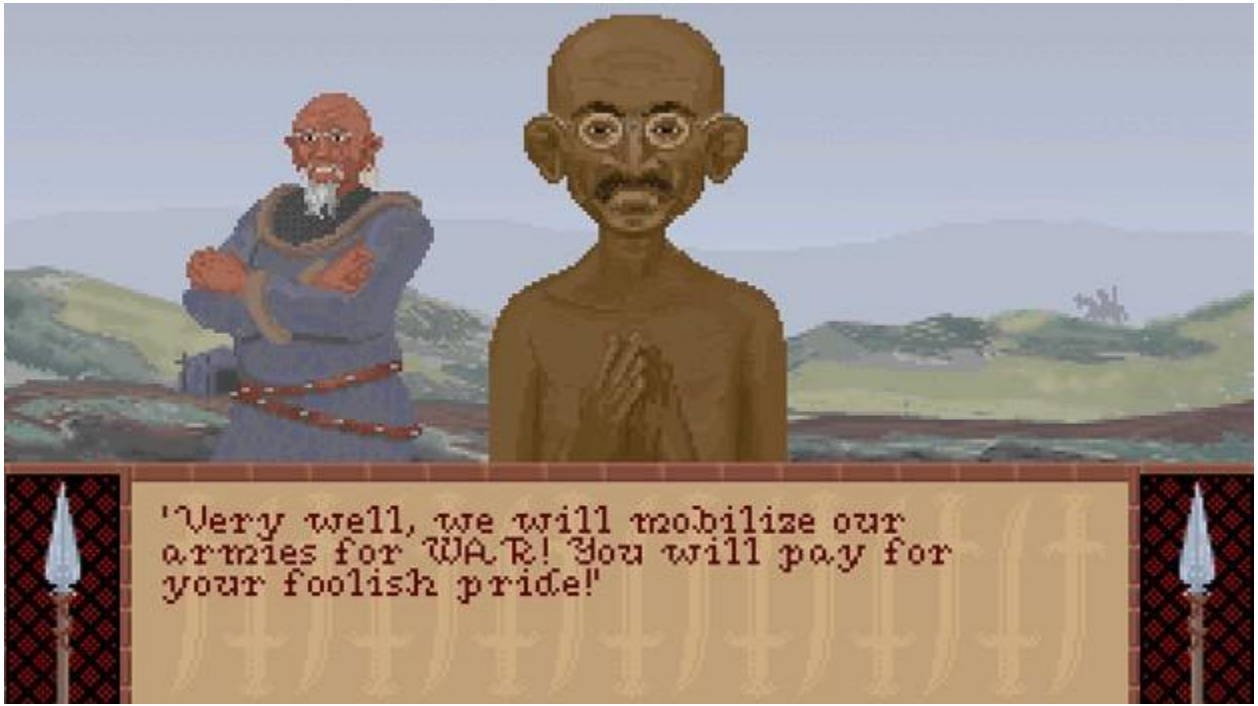
Type	Sign	Exponent	Significand field	Total bits		Exponent bias	Bits precision	Number of decimal digits
Half (IEEE 754-2008)	1	5	10	16		15	11	~3.3
Single	1	8	23	32		127	24	~7.2
Double	1	11	52	64		1023	53	~15.9
x86 extended precision	1	15	64	80		16383	64	~19.2
Quad	1	15	112	128		16383	113	~34.0

Overflow

Overflow

```
   11111111          255
+  00000001        +   1
   00000000        =   0 ?!
```

Overflow


'Very well, we will mobilize our armies for WAR! You will pay for your foolish pride!'

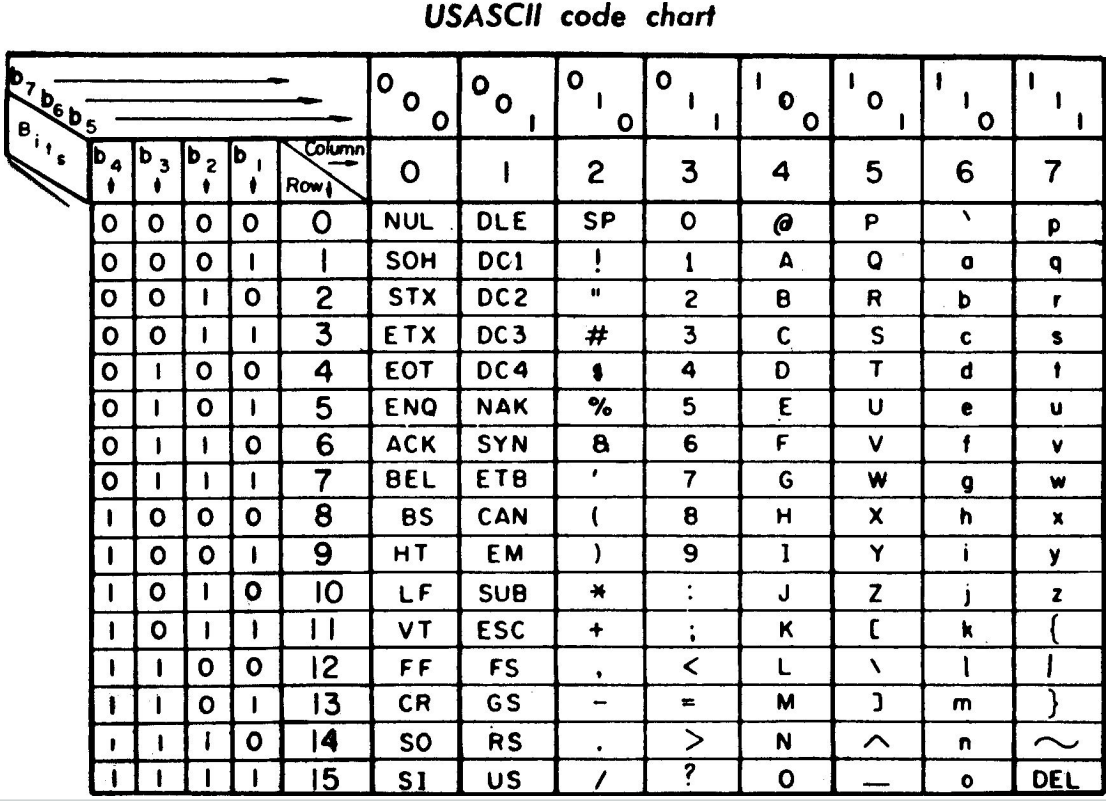
Encoding

How do we represent characters?

The answer is encoding!

ASCII

American Standard Code for Information Interchange



USASCII code chart

b7 b6 b5 → / Bits	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4 b3 b2 b1 / Row ↓ Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

UTF-8

Character		Octal code point	Binary code point	Binary UTF-8	Octal UTF-8	Hexadecimal UTF-8
$	U+0024	044	010 0100	00100100	044	24
¢	U+00A2	0242	000 1010 0010	11000010 10100010	302 242	C2 A2
€	U+20AC	020254	0010 0000 1010 1100	11100010 10000010 10101100	342 202 254	E2 82 AC
𐍈	U+10348	0201510	0 0001 0000 0011 0100 1000	11110000 10010000 10001101 10001000	360 220 215 210	F0 90 8D 88

Playing with binary data in Ruby

Array#pack

String#unpack

See also

CS50 2017 - Lecture 0 - Scratch