MAKE
SCHOOL

NUMBER BASES

All your base are belong to us.

MAKE SCHOOL

BASE-10

Decimal

Humans use base-10, a.k.a. “decimal.”

We have 10 digits in base-10. (`0` through `9`)

`9` rolls over to `10`. `99` rolls over to `100`.

BASE-2

Binary

Computers use base-2, a.k.a. “binary.”

There are 2 digits in base-2. (`0` and `1`)

`1` rolls over to `10`. `11` rolls over to `100`.

“There are 10 kinds of people in the world. Those who understand binary, and those who don’t.”

–Old Programming Proverb

COUNTING IN BINARY

0,	1,	10,	11,	100,	101,	110,	111,
1000,	1001,	1010,	1011,	1100,	1101,	1110,	1111,
10000,	10001,	10010,	10011,	10100,	10101,	10110,	10111,
11000,	11001,	11010,	11011,	11100,	11101,	11110,	11111,
100000,	100001,	100010,	100011,	100100,	100101,	100110,	100111,
101000,	101001,	101010,	101011,	101100,	101101,	101110,	101111,
110000,	110001,	110010,	110011,	110100,	110101,	110110,	110111,
111000,	111001,	111010,	111011,	111100,	111101,	111110,	111111,

...

BINARY TO DECIMAL

1 0 1 0 0 1 0 1 0 0 1 1

*

2^{11} 2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0}

=

2048 0 512 0 0 64 0 16 0 0 2 1

=

2643_{10}

BASE-16

Hexadecimal

Computers sometimes also use base-16, a.k.a. “hexadecimal” or simply “hex.”

There are 16 digits in base-16. (**`0–9`** and **`A–F`**)

`9` continues to **`A`**. **`F`** rolls over to **`10`**.

`99` continues to **`9A`**. **`FF`** rolls over to **`100`**.

BASE-16

Hexadecimal

There are 16 digits in base-16. (**0–9** and **A–F**)

$A_{16} = 10_{10}$ $D_{16} = 13_{10}$

$B_{16} = 11_{10}$ $E_{16} = 14_{10}$

$C_{16} = 12_{10}$ $F_{16} = 15_{10}$

BASE-16

Hexadecimal

Hex is often prefixed **`0x`**

$$\texttt{0xAF320100} == \texttt{AF320100}_{16}$$

COUNTING IN HEX

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

10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1A, 1B, 1C, 1D, 1E, 1F,

20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2A, 2B, 2C, 2D, 2E, 2F,

…

90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 9A, 9B, 9C, 9D, 9E, 9F,

A0, A1, A2, A3, A4, A5, A6, A7, A8, A9, AA, AB, AC, AD, AE, AF,

B0, B1, B2, B3, B4, B5, B6, B7, B8, B9, BA, BB, BC, BD, BE, …

HEX TO BINARY

Every hex digit is 4 binary digits (bits)

$5_{16} \leftrightarrow 0101_2$	$10_{16} \leftrightarrow 0001\ 0000_2$
$8_{16} \leftrightarrow 1000_2$	$72_{16} \leftrightarrow 0111\ 0010_2$
$B_{16} \leftrightarrow 1011_2$	$A6_{16} \leftrightarrow 1010\ 0110_2$
$F_{16} \leftrightarrow 1111_2$	$FF_{16} \leftrightarrow 1111\ 1111_2$

NEGATIVE INTEGERS

SIGNED MAGNITUDE

Most significant (leftmost) bit indicates sign

`0` is positive, `1` is negative

Range is $-(2^{n-1} - 1)$ to $2^{n-1} - 1$

Two representations of zero: `00000000` (zero) and `10000000` (negative zero?)

More info: [signed number representations]()

SIGNED MAGNITUDE

$72_{10} \leftrightarrow 0100\ 1000_2$

$-72_{10} \leftrightarrow 1100\ 1000_2$

$-1_{10} \leftrightarrow 1000\ 0001_2$

$-0_{10} \leftrightarrow 1000\ 0000_2$

More info: [signed number representations]()

ONES' COMPLEMENT

Negative number is positive magnitude with bitwise NOT applied (invert/flip all bits)

Range is $-(2^{n-1}-1)$ to $2^{n-1}-1$

Two representations of zero: **`00000000`** (zero) and **`11111111`** (negative zero?)

More info: ones' complement

MAKE SCHOOL

ONES' COMPLEMENT

$72_{10} \leftrightarrow 0100\ 1000_2$

$-72_{10} \leftrightarrow 1011\ 0111_2$

$-1_{10} \leftrightarrow 1111\ 1110_2$

$-0_{10} \leftrightarrow 1111\ 1111_2$

More info: ones' complement

ONES’ COMPLEMENT ADDITION

Add the two numbers

If there’s a carry, do an *end-around carry*
(add it back in to the sum)

More info: ones' complement

ONES' COMPLEMENT ADDITION

$$
\begin{array}{rrr}
 & 1111\ 1010_2 & -5_{10} \\
+ & 0000\ 1000_2 & 8_{10} \\
\hline
 & 1\ 0000\ 0010_2 & 2_{10} \\
+ & 1_2 & 1_{10} \\
\hline
 & 0000\ 0011_2 & 3_{10}
\end{array}
$$

More info: ones' complement

TWO'S COMPLEMENT

Standard way most processors represent negative numbers

Addition, subtraction, multiplication algorithms are the same as the ones for unsigned integers

Range is $-(2^{n-1})$ to $2^{n-1}-1$

One representation of **`0: 00000000`**

More info: [two's complement]()

TWO'S COMPLEMENT NEGATION

To negate a two's complement number, invert (flip) all the bits and then add **1**

Alternative technique (easier by hand):

1. Starting from the right, find the first **1**
2. Invert all the bits to the left of that **1**

More info: two's complement

TWO'S COMPLEMENT

$72_{10} \leftrightarrow 0100\ 1000_2$

$-72_{10} \leftrightarrow 1011\ 1000_2$

$-1_{10} \leftrightarrow 1111\ 1111_2$

$0_{10} \leftrightarrow 0000\ 0000_2$

More info: two's complement

MAKE SCHOOL