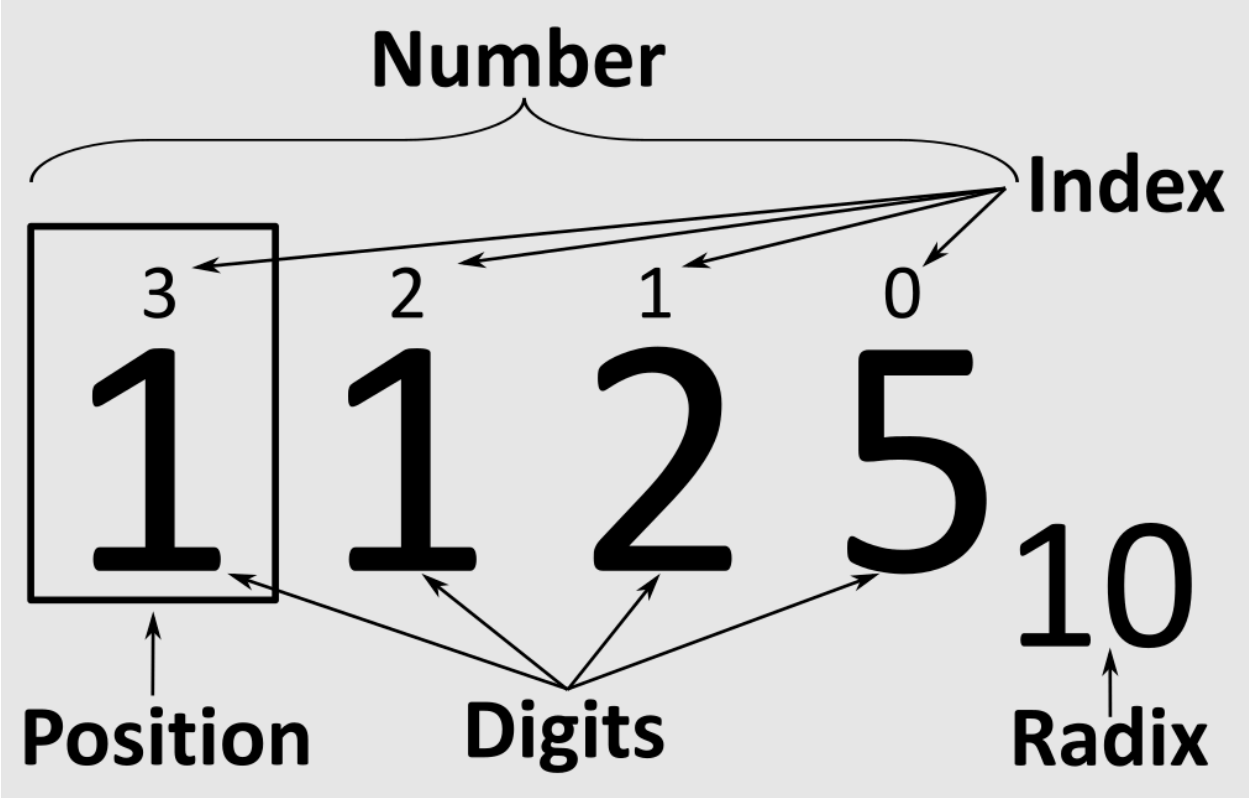
Chiang Rai International School

Computer Science

Number Systems in Computer Science

Standard mathematics notation uses a ***Positional Number System***. See the labeled parts:



"In a *positional numeral system*, the *radix* or ***base*** is the number of unique *digits*, including the digit zero, used to represent numbers." (Wikipedia: Radix)

Base 10 (decimal)

Decimal numbers use digits 0-9. Consider the the number 155_{10}:

$10^3 = 1000$	$10^2 = 100$	$10^1 = 10$	$10^0 = 1$
0	1	5	5

Calculate using the place values:
$(1 * 100) + (5 * 10) + (5 * 1) = 155_{10}$

Base 2 (binary)

Binary numbers use digits 0 and 1. In many programming languages, a *literal* binary number is prefixed with a `0b`. Each binary digit is known as a ***bit***. The smallest addressable unit of memory has a size of eight bits, known as a ***byte***. A byte has a maximum value of 255_{10}.

Let's convert 155_{10} to binary:

$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
1	0	0	1	1	0	1	1

Decimal to Binary Algorithm
(continuously divide by 2 and record the remainder):

$155 \div 2 = 77\ rem\ 1$
$77 \div 2 = 38\ rem\ 1$
$38 \div 2 = 19\ rem\ 0$
$19 \div 2 = 9\ rem\ 1$
$9 \div 2 = 4\ rem\ 1$
$4 \div 2 = 2\ rem\ 0$
$2 \div 2 = 1\ rem\ 0$
$1 \div 2 = 0\ rem\ 1$

Now, read the remainders from top-to-bottom. These are your bits from right-to-left (least significant digit to most significant). If the number of bits is less than 8, zero-fill the leading digits. (example: $0b10101 \Rightarrow 0b00010101$)

Check your answer by adding the place values that have a bit set (to 1).

Binary to Decimal
Count the place values:

$0b10011011 \Rightarrow 128 + 16 + 8 + 2 + 1 = 155_{10}$

Base 16 (hexadecimal)

Hexadecimal numbers use digits [0-9A-F] representing values 0-15 (decimal). In most programming languages, a *literal* hexadecimal number is prefixed with a `0x`.

Hexadecimal Digits

with Equivalent Decimal & Binary Values

Each **byte** is **eight bits**, which can be represented using two hexadecimal characters.
Half of a byte (four bits) is known as a ***nibble*** and can be represented as a single hexadecimal digit. The maximum value that can be stored in a nibble is $2^4 - 1, or\ 16 - 1 = 15.$

Decimal	**Hex**adecimal	**Bin**ary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Let's convert 155_{10} to hexadecimal:

$16^3 = 4096$	$16^2 = 256$	$16^1 = 16$	$16^0 = 1$
		9	B

Decimal to Hexadecimal Algorithm
(continuously divide by 16 and record the remainder):

$155 \div 16 = 9\ rem\ 11\ (or\ hex\ B;\ see\ "Hexadecimal\ Digits \ldots Equivalent\ Values")$
$9 \div 16 = 0\ rem\ 9$

Hexadecimal to Decimal
Take a sum of each digit multiplied by its place value.

$0x9B \Rightarrow (9 * 16) + (11 * 1) = 144 + 11 = 155_{10}$

Hexadecimal to Binary
Converting from hexadecimal to binary is easy. Just append the binary nibble values for each hexadecimal digit like a string. (The '+' [plus] operator here means concatenation, not addition.)

$0x9B \Rightarrow 0b????????$
$0x9 = 0b1001$
$0xB = 0b1011$
$0x9B \Rightarrow '1001' + '1011' = 0b10011011$

Binary to Hexadecimal
Likewise, use the lookup table to translate each nibble into a *hex* digit and string append each hex digit.

$0b10011011 \Rightarrow 0x??$
$0b1001 = 0x9$
$0b1011 = 0xB$
$0b10011011 \Rightarrow '9' + 'B' = 0x9B$

Bits and Bytes

A byte is made up of 8 bits (binary digits). In our example, one byte of storage can be represented using the number `0b10011011` or `0x9B`. Bytes are the smallest addressable unit of memory and the most common size units used by computers.

Storage is usually measured in one of the ***byte prefixes*** below:

Binary			
1024	2^{10}	**kB**	**kilo**byte
1024^2	2^{20}	**MB**	**mega**byte

1024^3	2^{30}	**GB**	**giga**byte

Byte Unit Symbols