Prof. Mariacarla Staffa
a.a. 2022/2023

Laboratorio di Architettura Degli Elaboratori

Rappresentazione dei Numeri

Codifica ottimale

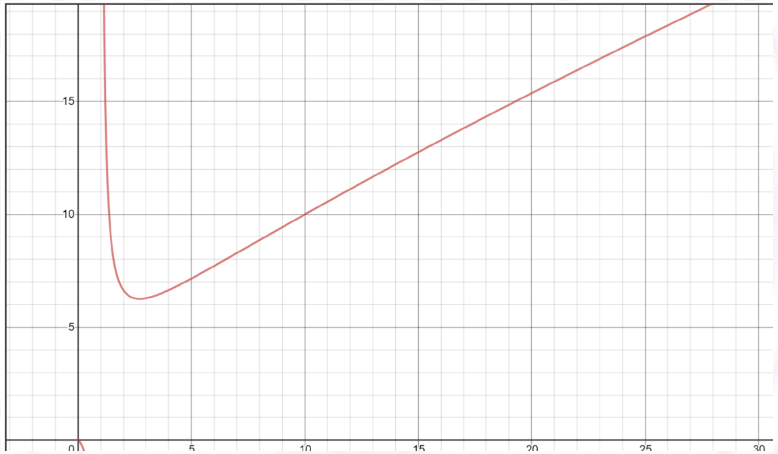
- Ritorniamo alle basi ammissibili ($b \geq 2$)
- Abbiamo visto che più grande è la base b, minore è il numero di cifre che occorrono per rappresentare un numero
- Quindi il codice binario è il sistema di codifica meno economico fra quelli ammissibili
  - Per esempio $\log_2 10 \cong 3.32$ questo vuol dire che per rappresentare un numero *n* in binario occorrono *circa il triplo* delle cifre che occorrono per rappresentare lo stesso *n* nel sistema decimale
  - *Perché i computer adottano la codifica binaria?*

Codifica ottimale

- Non bisogna tener presente solo la lunghezza di codifica ma anche il fatto che un calcolatore per operare in una certa base *b* deve poter rappresentare tutte le cifre di quella base, *quindi gli servono b differenti stati*.
- Una nozione di costo di una codifica che tiene conto anche di questo fattore è data dal prodotto

$$b \cdot m_b \simeq b \cdot \log_b n$$

Si può mostrare che il valore di b che minimizza tale costo è il numero di Nepero e ≅ 2.7, quindi le codifiche che si avvicinano di più a tale valore ottimale sono quelle in base 2 e 3



Digital Discipline: Binary Values

Two discrete values:

1's and 0's
1, TRUE, HIGH
0, FALSE, LOW

1 and 0: voltage levels, rotating gears, fluid levels, etc.

Digital circuits use **voltage** levels to represent 1 and 0

Bit: *Binary digit*

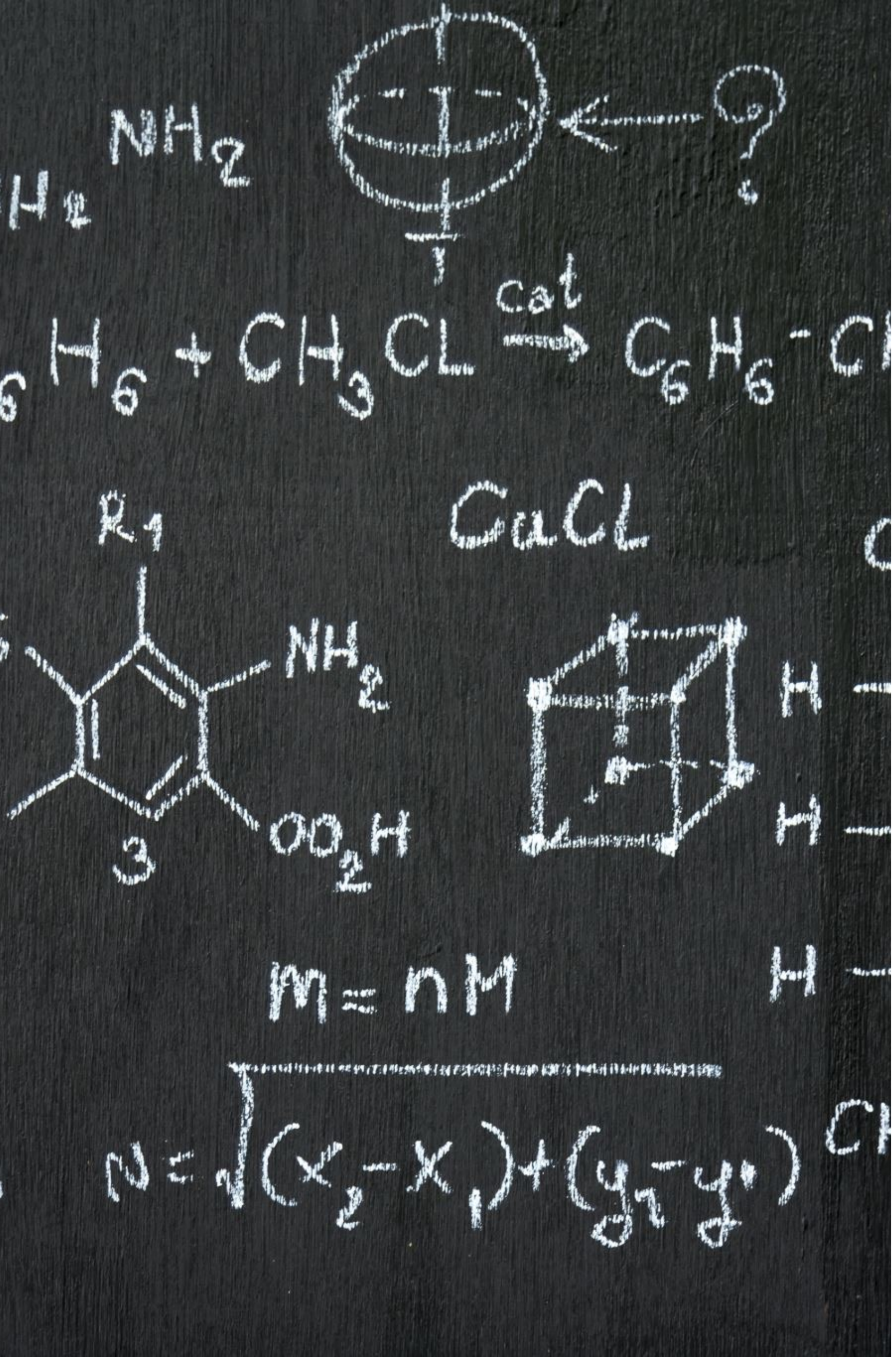
Cambiamento di base

- Problema: dato n rappresentato in base a, quale è la sua rappresentazione in base b?
- Supponiamo di saper fare le quattro operazioni in base a
- L'algoritmo di **cambiamento di base** consiste nel:
  - dividere ripetutamente *n* (espresso in base a) per b finché il quoziente non risulti uguale a zero.
  - La sequenza di resti ottenuti (compresi tra 0 e b-1) è la codifica (dalla cifra meno significativa a quella più significativa) di n in base b

Esempio:
codificare 251_{10} in base 3

$$251/3 = 83 \quad \text{resto: } 2$$
$$83/3 = 27 \quad \text{resto: } 2$$
$$27/3 = 9 \quad \text{resto: } 0$$
$$9/3 = 3 \quad \text{resto: } 0$$
$$3/3 = 1 \quad \text{resto: } 0$$
$$1/3 = 0 \quad \text{resto: } 1$$

$$251_{10} = 100022_3$$



Esercizio

- 43 : 2 = 21 con resto di 1
- 21 : 2 = 10 con resto di 1
- 10 : 2 = 5 con resto di 0
- 5 : 2 = 2 con resto di 1
- 2 : 2 = 1 con resto di 0
- 1 : 2 = 0 con resto di 1

- 43 = 101011

Cambiamento di base

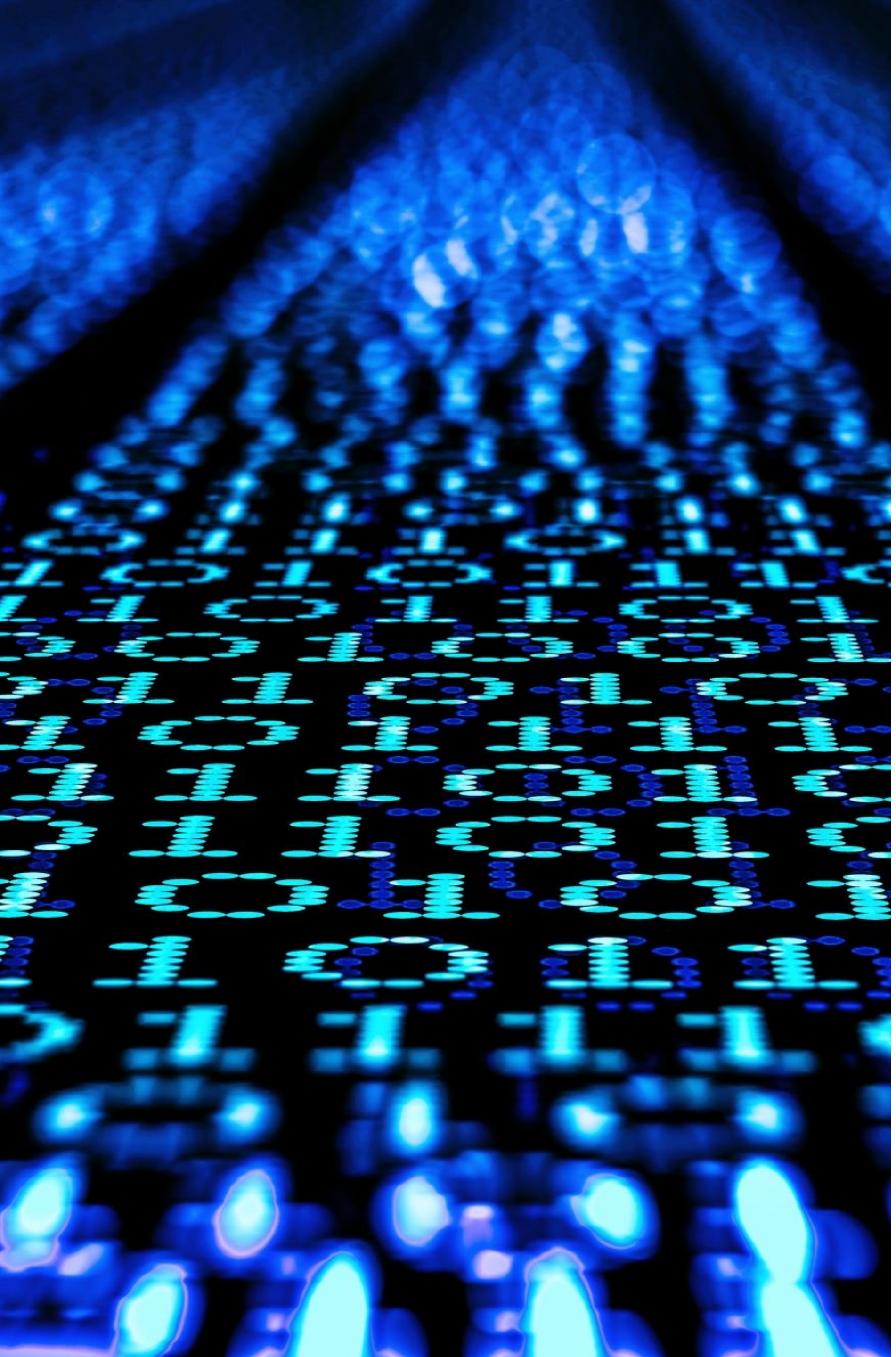
- Esempio: codificare 333_7 in base 9
- Problema: non siamo molto allenati con la divisione in base 7. Meglio affrontare il problema in due passi:
  - Codifico 333_7 in base 10

$$333_7 = 3 \cdot 7^2 + 3 \cdot 7^1 + 3 \cdot 7^0 = 171$$

  - Codifico 171_{10} in base 9

$$171/9 = 19 \quad \text{resto: } 0$$
$$19/9 = 2 \quad \text{resto: } 1$$
$$2/9 = 0 \quad \text{resto: } 2$$

$$333_7 = 210_9$$



Decimal to Binary Conversion

- Two methods:
  - **Method 1:** Find the largest power of 2 that fits, subtract and repeat
  - **Method 2:** Repeatedly divide by 2, remainder goes in next most significant bit

Decimal to Binary Conversion

Method 1: Find the largest power of 2 that fits, subtract and repeat

53_{10}

Method 2: Repeatedly divide by 2, remainder goes in next most significant bit

53_{10}

Decimal to Binary Conversion

Method 1: Find the largest power of 2 that fits, subtract and repeat

53_{10}	32×1
53-32 = 21	16×1
21-16 = 5	4×1
5-4 = 1	1×1

= 110101_2

Method 2: Repeatedly divide by 2, remainder goes in next most significant bit

53_{10} =

53/2 = 26 R=1
26/2 = 13 R=0
13/2 = 6 R=1
6/2 = 3 R=0
3/2 = 1 R=1
1/2 = 0 R=1

= 110101_2

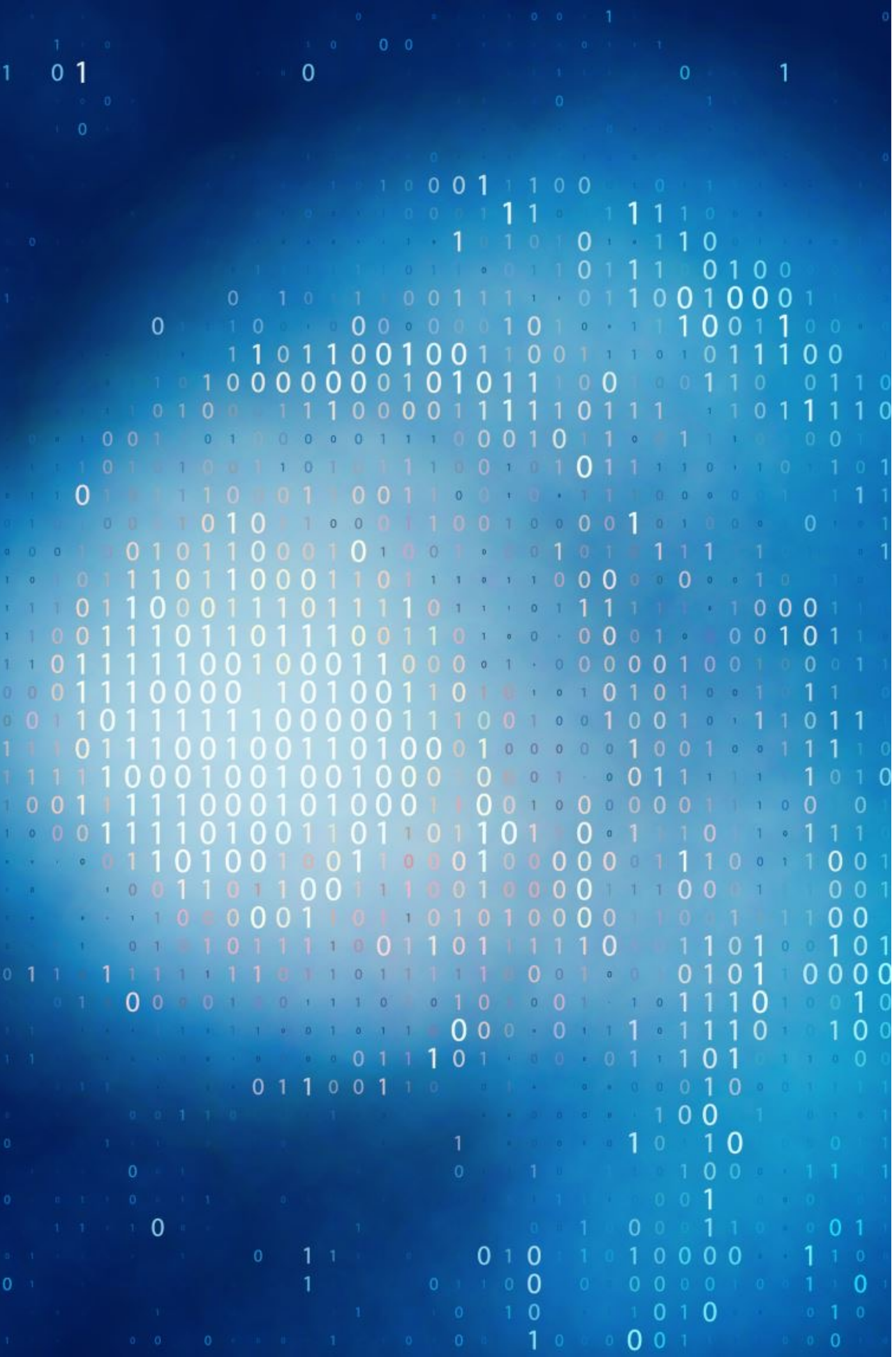
Esercizi: Number Conversion

- Binary to decimal conversion:
  - Convert 10011_2 to decimal

- Decimal to binary conversion:
  - Convert 47_{10} to binary

Number Conversion

- Binary to decimal conversion:
  - Convert 10011_2 to decimal
  - $16\times1 + 8\times0 + 4\times0 + 2\times1 + 1\times1 = 19_{10}$
- Decimal to binary conversion:
  - Convert 47_{10} to binary
  - $32\times1 + 16\times0 + 8\times1 + 4\times1 + 2\times1 + 1\times1 = 101111_2$



Esercizi: Decimal to Binary Conversion

- **Another example:** Convert 75_{10} to binary.

Decimal to Binary Conversion

Another example: Convert 75_{10} to binary.

$75_{10} = 64 + 8 + 2 + 1 = 1001011_2$

Decimal to Binary Conversion

Another example: Convert 75_{10} to binary

$75_{10} = 64 + 8 + 2 + 1 = 1001011_2$

or

75/2	= 37	R1
37/2	= 18	R1
18/2	= 9	R0
9/2	= 4	R1
4/2	= 2	R0
2/2	= 1	R0
1/2	= 0	R1

Codifica binaria e esadecimale

- Abbiamo detto che i componenti digitali operano in codice binario.
- Tuttavia la rappresentazione in binario non è molto human-friendly perché genera codici piuttosto lunghi
  - $599 = 1001010111_2$
- Si potrebbe pensare di usare la nostra base naturale (10). Questo comporta usare il precedente algoritmo per codifica/decodifica
  - Non proprio agevole
  - Non proprio velocissimo
  - Il problema è che 10 non è una potenza di 2

Codifica binaria e esadecimale

- La codifica/decodifica in una base m potenza di 2 permette invece di usare qualche trucchetto che migliora i tempi di codifica e decodifica.
- le cifre utilizzate nel sistema esadecimale sono '0',…,'9','a',…,'f'.
- Inoltre, poiché $16=2^4$ è possibile codificare ogni cifra del sistema esadecimale mediante 4 bit
- Viceversa 4 bit del sistema binario corrispondono ad una cifra esadecimale

Codifica binaria e esadecimale

- Codifica delle cifre esadecimali in binario

$$
\begin{array}{ccc}
0 \longrightarrow 0000 & \quad & 8 \longrightarrow 1000 \\
1 \longrightarrow 0001 & & 9 \longrightarrow 1001 \\
2 \longrightarrow 0010 & & a \longrightarrow 1010 \\
3 \longrightarrow 0011 & & b \longrightarrow 1011 \\
4 \longrightarrow 0100 & & c \longrightarrow 1100 \\
5 \longrightarrow 0101 & & d \longrightarrow 1101 \\
6 \longrightarrow 0110 & & e \longrightarrow 1110 \\
7 \longrightarrow 0111 & & f \longrightarrow 1111
\end{array}
$$

- Notate che questa codifica corrisponde alla codifica dei rispettivi numeri con possibili 0 non significativi

$$0111_2 = 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 7$$

Da esadecimale a binario

- Dato un numerale esadecimale per codificarlo in binario è sufficiente giustapporre le codifiche delle singole cifre (eliminando eventualmente zeri non significativi)
- Esempi:
  - $4c9f_{16}$ → 0100 1100 1001 1111 → 100110010011111_2
  - $b2a_{16}$ → 1011 0010 1010 → 101100101010_2
  - 157_{16} → 0001 0101 0111 → 101010111_2
- Nota bene che

$$157_{16} = 1 \cdot 16^2 + 5 \cdot 16^1 + 7 \cdot 16^0 = 343$$

Da binario a esadecimale

- Per il passaggio di base da binario a esadecimale si effettua la codifica inversa avendo cura di aggiungere eventuali zeri non significativi in modo che la lunghezza del numerale in binario sia multipla di 4
- Esempi:
  - $10_2 \rightarrow 0010 \rightarrow 2_{16}$
  - $10011_2 \rightarrow 0001\ 0011 \rightarrow 13_{16}$
  - $111110010101100_2 \rightarrow 1111\ 1001\ 0101\ 1100 \rightarrow f95c_{16}$

Nota: comunemente un numerale esadecimale viene indicato con il prefisso 0x

$f95c_{16} \rightarrow 0xf95c$

Rappresentazione registri

- In un computer le grandezze numeriche sono elaborate mediante sequenze di simboli di lunghezza fissa dette parole
- Poichè una cifra esadecimale codifica 4 bit:
  - 1 byte (8 bit) → 2 cifre esadecimali
  - 4 byte (32 bit) → 8 cifre esadecimali
  - 8 byte (64 bit) → 16 cifre esadecimali

Bits, Bytes, Nibbles…

- Bits

$$\underbrace{1}_{\text{most significant bit}}001011\underbrace{0}_{\text{least significant bit}}$$

- Bytes & Nibbles

$$\overbrace{1001\underbrace{0110}_{\text{nibble}}}^{\text{byte}}$$

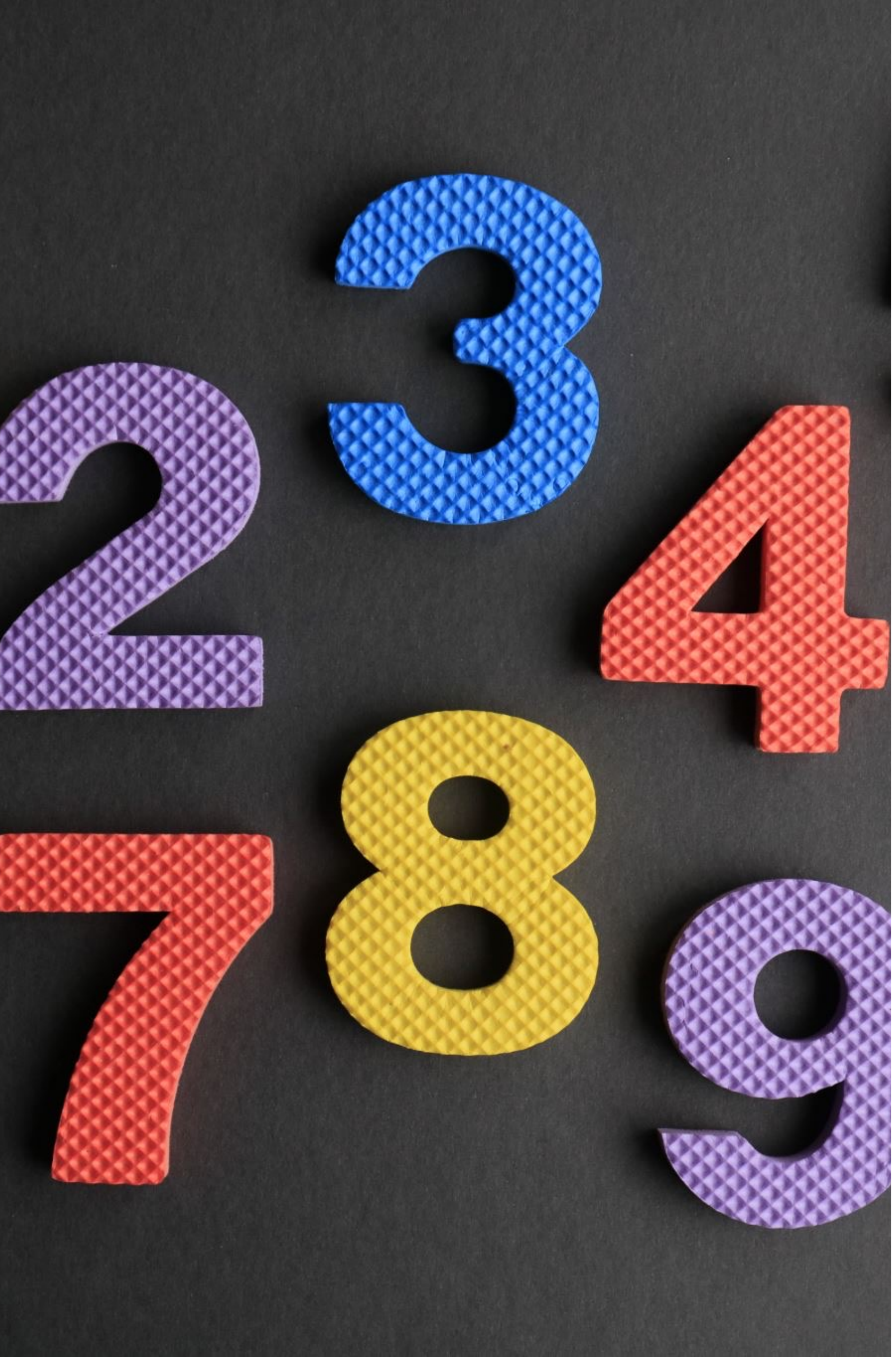
- Bytes

$$\underbrace{\text{CE}}_{\text{most significant byte}}\text{BF9A}\underbrace{\text{D7}}_{\text{least significant byte}}$$

Word

- I microprocessori gestiscono gruppi di bit chiamati word
  - La grandezza dipende dall’architettura del microprocessore
  - 64 bit (o 32)

  - 10011 -> 0001 0011



Esercizio: Quale è il maggiore?

A) 1 0 1 0 1 0 1 0

B) 1 0 0 1 0 1 0 0

C) 1 0 1 0 1 0 1 1

Esercizio: Hexadecimal to Binary Conversion

- Hexadecimal to binary conversion:
  - Convert $4AF_{16}$ (also written 0x4AF) to binary

- Hexadecimal to decimal conversion:
  - Convert 0x4AF to decimal