REPRESENTACIÓN DE INFORMACIÓN EN EL ORDENADOR

- La información se almacena en el ordenador en su memoria interna, conocida como memoria RAM, en posiciones de 8 bits, o potencia de esta. O sea 8, 16, 32, 64, etc.
- La información internamente se almacena en código binario.
- Tenemos por un lado la representación de números y por otro la de caracteres
- Los códigos de caracteres más usados son el ASCII (Antiguo), UNICODE de 16 bits.
- En estos códigos se asigna un número a cada posible carácter del alfabeto.
    - A es el 65, B – 66
    - a – 97, b – 98
    - 0 - 48, 1 - 49
    - É - 201
- Cuando los datos son numéricos se almacena el número correspondiente.

DATOS NUMÉRICOS

- Módulo y signo
- Complemento a 1
- Complemento a 2
- Representación de números en coma flotante

MÓDULO Y SIGNO

Número 10 0 0 0 0 1 0 1 0 en un byte

Número -10 1 0 0 0 1 0 1 0

Número 10 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 en una palabra

Número -10 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0

- El primero es el signo y el resto de los bits son el módulo, el número siempre positivo
- Tenemos dos representaciones del 0
- 2^{n-1} Positivos y 2^{n-1} Negativos
- En 8 bits de -127 hasta +127

COMPLEMENTO A 1

Número 10 0 0 0 0 1 0 1 0 en un byte

Número -10 1 1 1 1 0 1 0 1

- El primero es el signo
    - Si es 0, el número tal cual en positivo
    - Si es 1 el número es negativo pero está cambiado ceros por unos y unos por ceros.
- Tenemos dos representaciones del 0

- 2^{n-1} Positivos y 2^{n-1} Negativos
- En 8 bits de -127 hasta +127

COMPLEMENTO A 2

Número 10 0 0 0 0 1 0 1 0 en un byte

Número -10 C1 1 1 1 1 0 1 0 1

Sumar 1 0 0 0 0 0 0 0 1

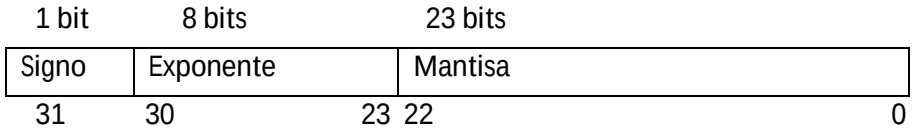
Resultado 1 1 1 1 0 1 1 0

- El primero es el signo
  - Si es 0, el número tal cual en positivo
  - Si es 1 el número es negativo pero está cambiado ceros por unos y unos por ceros, sumado 1.
- Tenemos una representación del 0, se toma como positivo
- 2^{n-1} Positivos y 2^{n-1} Negativos
- En 8 bits de -128 hasta +127

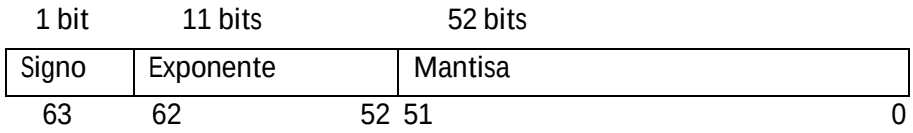
REPRESENTACIÓN DE NÚMEROS EN COMA FLOTANTE

Permite almacenar números grandes y pequeños dependiendo de la precisión, en un tamaño de 16 o 32 bits. En los lenguajes de programación tenemos diversos tipos de datos numéricos, gracias a este tipo de almacenamiento.

Simple Precisión 32 bits



Doble Precisión 64 bits



El número es positivo si el bit de signo es 0, y negativo si el bit de signo es 1

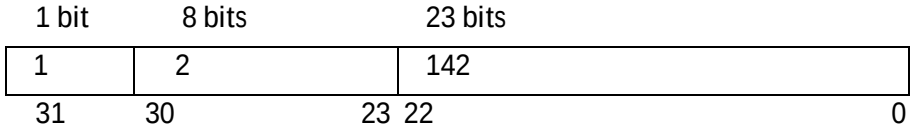
El valor del módulo es Mantisa multiplicado por la Base Elevado al Exponente

Ejemplo: -14,2

- Simple Precisión
- Signo 1
- Base: 10

- Mantisa.
  - Poner el número en formato todo decimal, e sea 0,142
  - Guardamos 142
- Exponente.
  - La base es 10
  - Para que 0,142 se 14,2 se debe multiplicar 0,142 por 100.
  - Base 10. 100 es igual a 10^2
  - El exponente es 2.

Resultado:



Los valores aparecen en decimal por claridad, internamente lo que iría sería la codificación en binario.

La base será 2, 8 o 16 dependiendo del fabricante.