Prerequisiti: conversioni base_10 ↔ base_2; numeri interi con/senza segno in base 2; complemento a 2

RAPPRESENTAZIONE STANDARD E NORMALIZZATA DEI NUMERI

Per **notazione standard** di un numero reale si intende la rappresentazione che ben conosciamo in cui il numero è composto dal segno, da una parte intera **i**, da una parte decimale **m** (chiamata **mantissa**) e dalla virgola in mezzo (oggi si vede più spesso il punto '.' in mezzo). In sintesi:

$$x = \pm i.m$$

Un esempio è -43.856_{10} oppure $6DF.5_{16}$ oppure 110101.101_2 (esempi con basi diverse)

Per **notazione esponenziale** di un numero reale si intende il numero scritto come segue:

$$x = \pm n \cdot b^e$$

n sono le cifre del numero, **e** è l'esponente, **±** il segno, **b** la base scelta (normalmente è 10).

Del numero -43.856 abbiamo molte possibili rappresentazioni esponenziali:

$$x = -43.856 \cdot 10^0 = -4.3856 \cdot 10^1 = -43856 \cdot 10^{-3} = -0.43856 \cdot 10^2 = -0.0043856 \cdot 10^4$$

Scrivere altre 3 rappresentazioni esponenziali di -43.856

Nelle calcolatrici si vedono spesso numeri in notazione esponenziale; per esempio 3.734675e12, che si deve leggere come $3.734675 \cdot 10^{12}$; quindi la "e" significa "moltiplicato per 10 elevato a". Ritornando alla notazione standard si ha che: $3.734675 \cdot 10^{12} = 3734675000000$

Si nota che esiste una precisa relazione tra l'esponente e "di quanti posti si sposta la virgola". Il fatto che la virgola ("point" in inglese) possa essere collocata dove vogliamo pur rappresentando lo stesso numero porta alla denominazione di "**floating point**" per tale rappresentazione; potremmo tradurla con "virgola fluttuante".

ⓘ Perché le calcolatrici usano (anche) questa rappresentazione? _____

Scrivere in notazione standard il numero $-3.734675 \cdot 10^2$

Scrivere in notazione esponenziale (una delle tante) il numero +56678.9012; come apparirebbe il display di una calcolatrice con 5 cifre per il numero e 1 cifra per l'esponente, tipo ±1.2345E±8 ?

Sul display di una calcolatrice come sopra è rappresentato il numero -3.7346e-8; scriverlo in notazione standard;

Qual è il numero più grande e quello più piccolo rappresentabile in tale calcolatrice? E il più piccolo numero positivo?

Tra tutte le rappresentazioni esponenziali prende il nome di **rappresentazione normalizzata** quella che a sinistra della virgola ha solo la cifra più significativa del numero; nel nostro esempio:

$$x = -4.3856 \cdot 10^1$$

ⓘ La vostra calcolatrice usa la rappresentazione esponenziale o quella normalizzata? _____

RAPPRESENTAZIONE DEI NUMERI NEI PC con n° di cifre limitato

Nei computer si pone sempre il problema di come rappresentare i numeri, e anche il segno di tali numeri. Il problema fondamentale è che le cifre con cui si rappresentano i numeri sono limitate, non infinite.

Le varie rappresentazioni hanno vantaggi e svantaggi, e sono vincolate a come i numeri sono trattati dalle **ALU** (Arithmetic Logic Unit) e dalle **FPU** (Floating Point Unit); e viceversa. La convenzione adottata nella maggior parte dei computer è lo **standard IEEE 754** del 1985, a 32 bit, che si basa sulla rappresentazione normalizzata dei numeri, in base 2 (ovviamente, all'interno di un computer ci sono solo bit).

Un numero reale in base 2 e in forma normalizzata si può sempre scrivere così:

$$x = \pm 1.m \cdot 10^{e}$$

con l'unica eccezione dello zero infatti tutti i numeri in binario hanno come cifra più significativa 1. Come base della potenza trovate scritto 10 che infatti è 2 espresso in binario; "m" costituisce la mantissa mentre "e" è l'esponente. Per esempio il numero -9 in base 2 diventa -1001 e si può scrivere così:

$$-1.001 \cdot 10^{11}$$

ⓘ convertire l'espressione precedente in decimale e verificare se si ottiene il numero 9

Nel caso che all'esponente siano assegnati 8 bit e alla mantissa 23 bit (coma fa IEEE 754) -9 diventa:

$$-1.00100000000000000000000 \cdot 10^{00000011}$$

Abbiamo colorato in nero le parti che non cambiano mai, cioè l' 1 prima della virgola e la base 10. IEEE 754 le considera "scontate" e… non le rappresenta neppure! (vedremo meglio tra poco cosa comporta).

IEEE 754 assegna un bit al segno, e precisamente 0 per il segno '+' e 1 per il segno '-' Potremo scrivere così:

$$(-1)^{s} \cdot 1.00100000000000000000000 \cdot 10^{00000011}$$

in cui "s" rappresenta il bit del segno, infatti: $(-1)^{s}$ vale +1 se s=0 e vale -1 se s=1. Generalizzando:

$$x = (-1)^{s} \cdot 10^{e} \cdot 1.m$$

dove **m** è la mantissa, **e** è l'esponente, **s** il segno. Tutte queste quantità sono espresse in binario, cioè con 0 o 1. Finalmente il numero -9 sarà memorizzato con 32 bit nel formato floating point IEEE 754, così:

00000011001000000000000000000000

- Convertire **10000000000000000000100000010000** in notazione esponenziale e poi in notazione standard
- Come sopra con **10000110001000000100000000000000**
- Come sopra con **00000001010011001000000000000000**
- Come sopra con **10010000000010010000001000010000**
- Come sopra con **10000100001000000100110000000000**
- Scrivere in notazione standard tutti i numeri degli esercizi precedenti, prima in base 2 e poi in base 10.