Dipartimento di Informatica e Sistemistica
Antonio Ruberti

“Sapienza” Università di Roma

Rappresentazione dell’Informazione

Corso di Fondamenti di Informatica

Laurea in Ingegneria Informatica

(Canale di Ingegneria delle Reti e dei Sistemi Informatici - Polo di Rieti)

Anno Accademico 2007/2008

Prof. Paolo Romano

Si ringrazia il Prof. Alberto Finzi per aver reso
disponibile il materiale didattico sul quale si basano queste slides

Rappresentazione binaria dell'informazione

Per informazione intendiamo tutto ciò che è manipolabile dal calcolatore:

- Numeri (naturali, interi, reali, …)
- Caratteri
- Immagini, Suoni etc.
- Programmi

Nel calcolatore, al livello macchina, le informazioni sono rappresentate in forma binaria: sequenze di 1 e 0.

Per motivi tecnologici: distinguere tra due valori di una grandezza fisica è più semplice.

Rappresentazione numeri naturali

Numero naturale è un oggetto matematico che può essere rappresentato come una sequenza di simboli a partire da un alfabeto (l'alfabeto dei numeri sono le cifre: 1,2,3, …)

Il numero è l'entità astratta, il numerale è il suo rappresentante: 123 è un numero rappresentato dal numerale "123" in base 10, "CXXIII" in cifre romane, "1111011" in base 2, "7B" in base 16

Confrontiamo due rappresentazioni:

1. Additiva (ad es. cifre romane)
2. Posizionale: il valore di ogni cifra dipende dalla posizione che occupa nella sequenza

Rappresentazione addizionale dei numeri

Alfabeto: cifre romane

I V X L C D M

I, II, III, IV, V, VI, VII …

X, XX, XXX, XL, L, LX, LXX, LXXX, XC

C, CC, CCC, CD, D, DC, DCC, DCCC, CM

M, MM, MMM

Ruolo posizione:

cifra crescente (da sin. a des.) $\Rightarrow$ sommata

cifra decrescente $\Rightarrow$ sottratta

Rappresentazione posizionale dei numeri

Il numero è rappresentato da una sequenza finita di cifre:

$$c_{n-1}\ c_{n-2}\ \ldots\ c_1\ c_0 = N_b$$

c_{n-1} cifra più significativa

c_0 cifra meno significativa

Il numero b delle cifre disponibili (dimensioni dell'alfabeto) è detto base del sistema di numerazione

Base	Alfabeto	Sistema
2	0,1	binario
8	0,1,3,…,7	ottale
10	0,1,3,…,9	decimale
16	0,1,3,…,9,A, …, F	esadecimale

Rappresentazione posizionale dei numeri

Il significato del numerale N_b:

$$N_b = c_{n-1}\, c_{n-2} \ldots c_1\, c_0$$

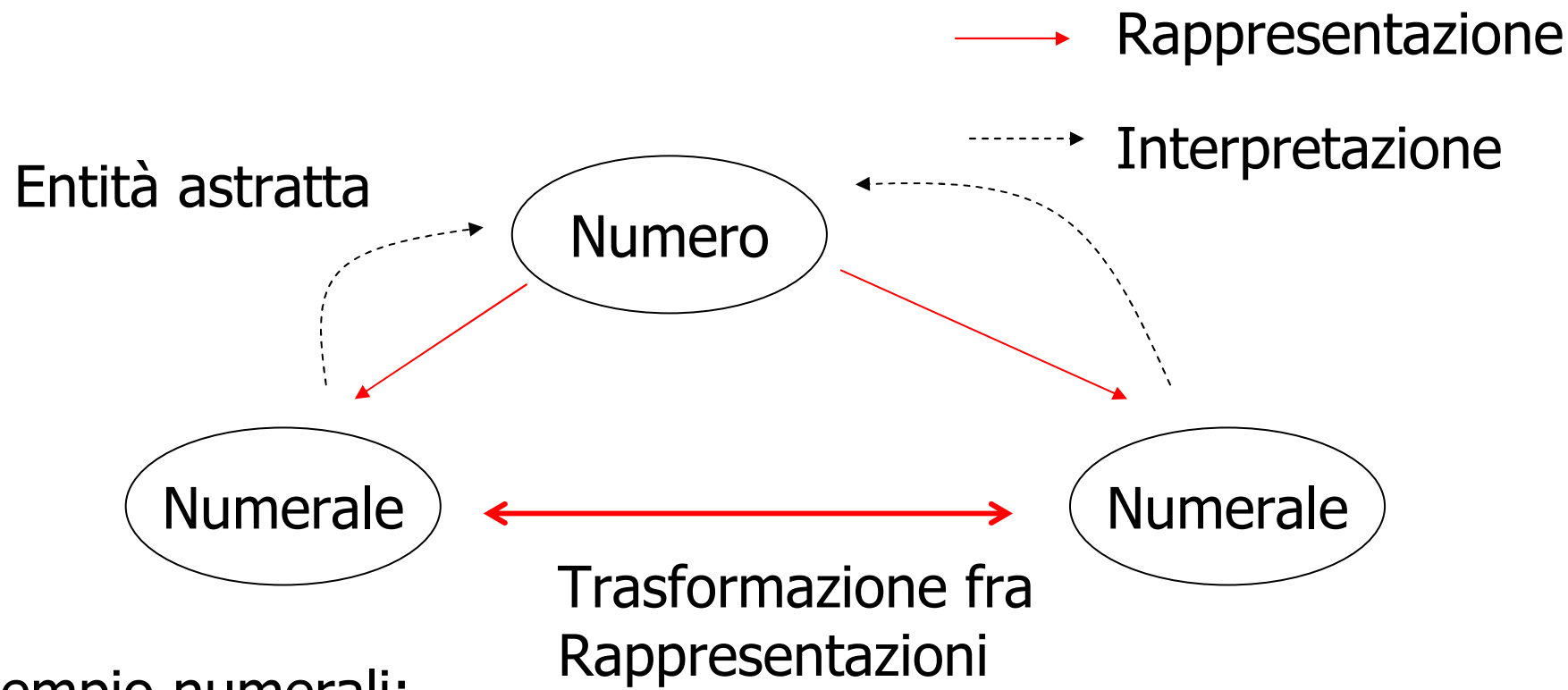
è il numero N:

$$N = c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b + c_0 = \Sigma_{i=0}^{n-1} c_i \cdot b^i$$

Esempio: il numerale 101 rappresenta numeri diversi a seconda del sistema usato.

Sistema	Base b	$(101)_b$	Valore (in decimale)
binario	2	$(101)_2$	5
ottale	8	$(101)_8$	65
decimale	10	$(101)_{10}$	101
esadecimale	16	$(101)_{16}$	257

Numeri e numerali



Esempio numerali:
"dodici", 12, XII,

Analogia: *gatto* e *cat* denotano la stessa "entità"
in due lingue differenti

Conversione di Base

Conversione da base b a base 10:

$$c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b + c_0 = \Sigma_{i=0}^{n-1} c_i \cdot b^i = N$$

Esprimendo cifre e base in base 10 e facendo i conti in base 10

Esempio: $(203)_4$ base 10?

$$2*4^2 + 0*4 + 3 = 35$$

Conversione di Base

Conversione da base 10 a base b:

$$N = c_0 + c_1 \cdot b + c_2 \cdot b^2 + \ldots + c_{n-2} \cdot b^{n-2} + c_{n-1} \cdot b^{n-1} =$$
$$c_0 + b \cdot (c_1 + b \cdot (c_2 + \ldots + b \cdot (c_{n-2} + c_{n-1} \cdot b) \ldots))$$

Si vogliono determinare le cifre $c_{n-1} \ldots c_0$ del numero in base b

Consideriamo la divisione di N per b e denotiamo con R il resto e con Q il quoziente:

$$N = R + b \cdot Q = c_0 + b \cdot (c_1 + b \cdot (\ldots))$$

a. Il resto R coincide con la cifra meno significativa c_0
b. A partire dal quoziente Q si può iterare il procedimento …

$$Q = R_1 + b \cdot Q_1 = c_1 + b \cdot (c_2 + b \cdot (\ldots))$$

… fino ad ottenere $Q_{n-1} = 0$

Algoritmo di conversione da base 10 a B (N intero)

N intero da convertire,
B base di arrivo

```
i←0;
 while  (N<>0) {
        c_i ← N mod  B;
        N←N div B;
         i=i+1;
}
```

Divisione intera: ritorna il quoziente della divisione, ovvero arrotonda il risultato esatto della divisione *ris* (in generale un numero reale) al massimo numero naturale minore di *ris* (anche detta parte inferiore di *ris,* ed indicata con $\lfloor ris \rfloor$):

es. 3 div 2= $\lfloor 3/2 \rfloor = \lfloor 1.5 \rfloor = 1$

definizione alternativa: quoziente = parte inferiore di *ris*

Conversione di Base

Conversione da base 10 a base b:

Esempio: $(25)_{10} = (???)_2$

N : b	Q	R	cifra
25 : 2	12	1	c_0
12 : 2	6	0	c_1
6 : 2	3	0	c_2
3 : 2	1	1	c_3
1 : 2	0	1	c_4

$(25)_{10} = (11001)_2$

Esempio: $(25)_{10} = (???)_4$

25 : 4	6	1	c_0
6 : 4	1	2	c_1
1 : 4	0	1	c_2

$(25)_{10} = (121)_4$

Conversione di Base

Conversione quando una base è potenza di un'altra:

Da base b^k a base b (e viceversa) (es. da base 8 a base 2)

$$c_{n-1} \cdot (b^k)^{n-1} + c_{n-2} \cdot (b^k)^{n-2} + \ldots + c_1 \cdot (b^k) + c_0 = N$$

Ogni c_i compresa tra 0 e b^k-1 quindi $c_i = c_{i,k-1}\, b^{k-1} + \ldots + c_{i,1}\, b + c_{i,0}$

Sostituendo c_i si ottiene:

$$(c_{n-1,k-1} \cdot b^{k-1} + \ldots + c_{n-1,1} \cdot b + c_{n-1,0})\ (b^k)^{n-1} +$$
$$(c_{n-2,k-1} \cdot b^{k-1} + \ldots + c_{n-2,1} \cdot b + c_{n-2,0})\ (b^k)^{n-2} +$$
$$\ldots +$$
$$(c_{0,k-1} \cdot b^{k-1} + \ldots + c_{i,1} \cdot b + c_{0,0}) = N$$

Coefficienti base b: $c_{n-1,k-1} \ldots c_{n-1,0}\ c_{n-2,k-1} \ldots c_{n-2,0}\ c_{n-2,k-1} \ldots c_{n-2,0}$

Conversione di Base

Conversione quando una base è potenza di un'altra:

Esempio: $(547)_8 = (???)_2$

$$(101)_2 (2^3)^2 + (100)_2 (2^3) + (111)_2 (2^3)^0$$

$$(547)_8 = (101\ 100\ 111)_2$$

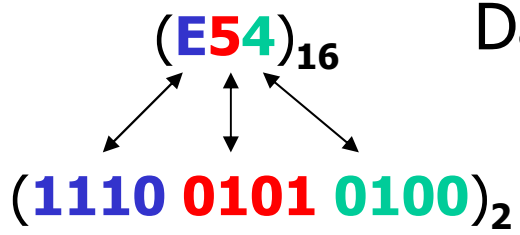
Da b a b^k si procede in modo duale raccogliendo le cifre

Esempio: $(12321)_4 = (???)_{16}$

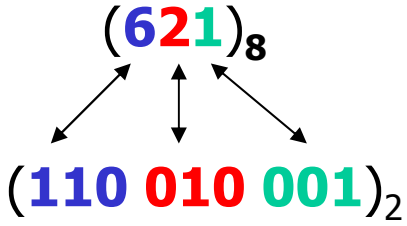
$$(1)(4^2)^2 + (2\ (4) + 3)\ (4^2) + (2\ (4) + 1)$$

$$(12321)_4 = (1B9)_{16}$$

Relazione fra le basi 2/8/16



Da base 16(2) a 2(16)



Da base 8(2) a 2(8)

$(E54)_{16} \leftrightarrow (1110\ 0101\ 0100)_2 \leftrightarrow (111\ 001\ 010\ 100)_2 \leftrightarrow (7124)_8$

Da base 16(8) a 8(16)

Numeri Interi

Occorre rappresentare numero e segno.

Prima soluzione: riservare un bit per il segno
1. bit più significativo è il segno (0 = + , 1 = -)
2. n-1 bit cifre

> Problemi:
> a. doppio numero per lo zero (0000, 1000)
> b. operazioni complicate (occorre distinguere i casi ++, +-,--)

Soluzione alternativa:
rappresentazione in complemento

Rappresentazione in Complemento

Rappresentazione in complemento alla base b con n cifre

$$\text{Comp}_{b^n}(X) = \begin{cases} X & \text{se } X \text{ in } [0,\ b^n/2\ ) \\ b^n - X & \text{se } X \text{ in } [-b^n/2\ ,\ 0) \end{cases}$$

Utilizzata per rapp. in complemento alla base (b con n cifre)

Rappresentazione di interi relativi nell'intervallo $[-b^n/2,\ b^n/2)$ mediante il residuo modulo b^n

- $X > 0$: Complemento di X in $[0,\ b^n/2\ )$
- $X < 0$: Complemento di X in $(b^n/2\ ,\ b^n]$

Nota: lo 0 ha unica rappresentazione

Approfondimento: definizione matematica alternativa

La rapp. in complemento di un intero X è esprimibile equivalentemente tramite la nozione di residuo modulo b^n :

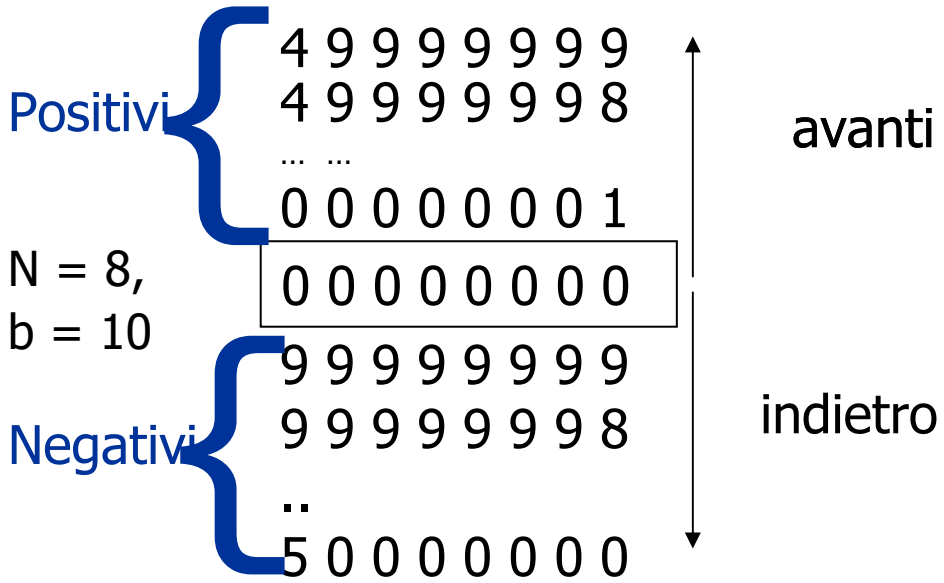
$| X |_{b^n} = X - \lfloor X / b^n \rfloor \cdot b^n$ dove $\lfloor X \rfloor$ è la parte inferiore di X

$| 25 |_{10^2} = 25 - \lfloor 25 / 10^2 \rfloor \cdot 10^2 = 25$
$| - 25 |_{10^2} = - 25 - \lfloor - 25 / 10^2 \rfloor \cdot 10^2 = - 25 - (-1)\ 10^2 = 75$

a. Ad un numero corrisponde un solo residuo
b. Ad un residuo:
   - corrispondono due numeri (pos. e neg.)
   - unico numero in $[-b^n/2 , b^n /2)$

Rappresentazione in Complemento

Esempio Contachilometri:



Positivi: $[0, 10^8 / 2 - 1] = [0, 49999999]$

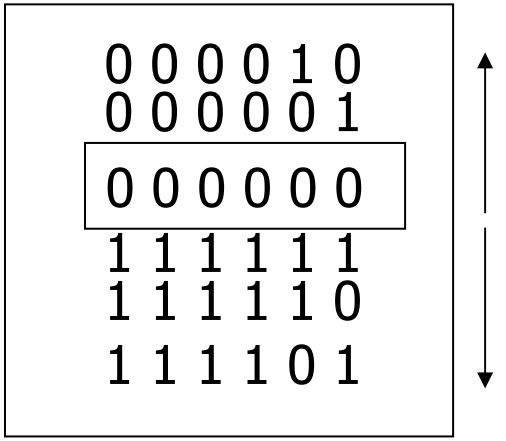
Negativi: $[-10^8 /2 , -1] = [-50000000, -1]$

Complemento a 2

Numeri positivi: cifra più significativa 0
Numeri negativi: cifra più significativa 1

Esempio: b = 2, n = 6, X = -9

$b^6 - |X| = 1000000 - 1001 = 110111$
$= (55)_{10} = 64 - 9$



Nota: $2^n - |X| = (2^n - 1) - X + 1$
dove $2^n - 1$ rapp. con 111…1 (n uno)

Metodo Rapido per calcolare il complemento: inverto i bit di |X|, sommo 1

Esempio: $9 = 001001$; $2^6 - 1 - (9) = 110110$; $110110 + 1 = 110111$

Metodo ancora più rapido:

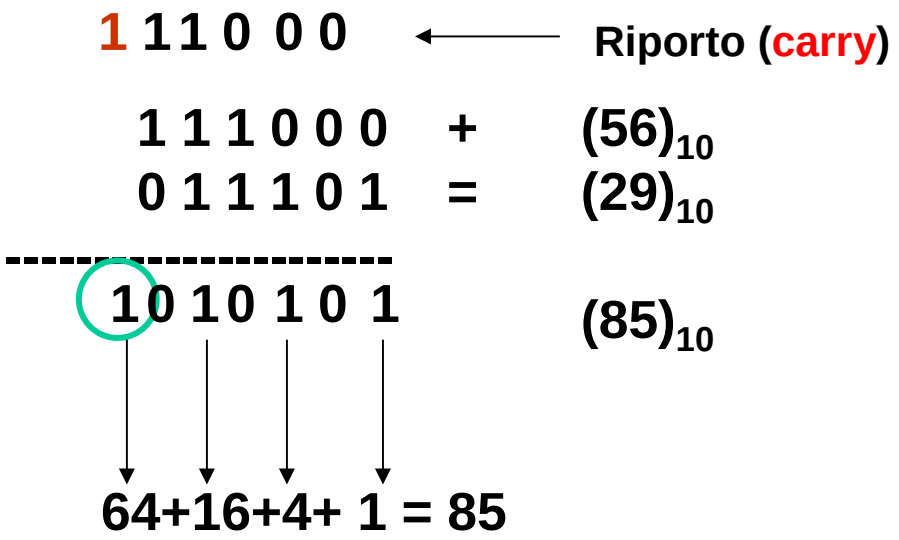
|X| rapp. con n bit;
a partire da destra si cerca il primo 1;
si invertono tutti i bit successivi.

Operazioni aritmetiche binarie

- Somma
- Sottrazione
- Estensione del Segno
- Prodotto e Disione per potenze di 2

Somma binaria

- BASE B=2
- 0+0=0
- 0+1=1
- 1+0=1
- 1+1=10 $=(2)_{10}$
- 1+1+1=11 $=(3)_{10}$



La somma di due numeri a k bit e' rappresentabile al piu' con k+1 bit
Se abbiamo a disposizione k bit ed il risultato richiede k+1 bit si ha overflow

La somma binaria come funzione logica

Somma di due bit A e B

Cin	A	B	Cout	Si
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Cout Cin +
A +
B =

Si

la coppia di valori (Cout,Si) indica il numero di “uno”, espresso in base 2

← attenzione a due queste configurazioni …. sono le uniche in cui Cin<>Cout

In generale:
Cin e’ il il riporto (carry) generato dalla somma dei bit di peso i-1
Cout è il riporto generato dalla somma dei 2 bit A,B di peso i

Somma algebrica in complemento

- Esprimere gli operandi in complemento alla base
  - La rappresentazione in complemento differisce solo per i valori negativi
- Eseguire la somma
- Trascurare l'eventuale riporto
- Se non si è verificato overflow, allora la somma rappresenta il risultato espresso in complemento
- Si verifica overflow quando gli operandi hanno lo stesso segno ed il risultato ha segno opposto

Overflow, esempio

- Eseguire su k=4 bit la differenza: –3-6

  |-3| → 2+1 →
  0011 →
  1101

  |-6| → 4+2 →
  0110 →
  1010

1000	(riporti)
1101 +	(-3)
1010 =	(-6)
(1)0111	(7!)

Rilevazione overflow

Si e' verificato OVERFLOW se:

1) i due operandi hanno lo stesso segno
2) Il risultato ha il segno diverso dagli operandi

ma…….

```
        1000                          0100
-3 +    1101          +5 +            0101
-6 =    1010          +6 =            0110
------- => ----------  ------- => ----------
-9      10111 => 7    +11           01011 => -5
```

……il verificarsi dell'overflow implica la disuguaglianza del riporto in ingresso e quello in uscita dalla posizione MSB (Cin<>Cout)

L'overflow si può rilevare testando la condizione "Cin<>Cout" di MSB

Estensione del segno

- Problema:
  - Sia dato un intero N, rappresentato in complemento mediante k bit
  - Rappresentare N usando k+q bit (q>0)
- Soluzione:
  - Fare q copie di MSB

- Esempio
  - -2 = $(110)_2$ con 3 bit diventa $(111110)_2$ su 6 bit

Prodotto e divisione per 2k

- Il prodotto di N per 2^k si ottiene postando di k posizioni le cifre a sinistra ed inserendo k bit pari a zero
- La divisione di N per 2^k si ottiene postando di k posizioni le cifre a destra ed inserendo k bit pari al valore di MSB (shift aritmetico)

- Esempio : -128/8 = -16 ($8=2^3$)

  1000 0000 → (3 posizioni a destra)

  1111 0000 = $(-16)_{10}$
- Esempio : -128/8 = -16 ($8=2^3$)

Prodotto e divisione per 2k

- Se N è un numero senza segno, allora il prodotto (divisione) per 2^k si ottiene spostando (shift) le cifre a sinistra (destra) di k posizioni ed introducendo 0 nelle posizioni lasciate libere

Esempio: 15 x 4= 60 ($4=2^2$,shift 2 posizioni)

0000 1111 ←

0011 1100

Esempio: 128 / 2= 64 ($2=2^1$, shift 1 posizione)

1000 0000 →

0100 0000

Attenzione: nel caso di rappresentazioni con segno questa regola non vale..

Aritmetica reale

Reali non enumerabili, rappresentazione approssimata.

Rappresentazione in virgola fissa.

Si rappresentano separatamente parte intera e parte frazionaria, alle parti è riservato un numero fissato di cifre

$$N_b = c_{n-1}\, c_{n-2} \ldots c_1\ \ c_0\, ,\ \ c_{-1}\, c_{-2} \ldots c_{-m}$$

rappresenta il numero

$$N = c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b + c_0 + c_{-1} \cdot b^{-1}\, c_{-2} \cdot b^{-2} \ldots c_{-m} \cdot b^{-m}$$

Conversione da base 10 a B (0<N<1)

Parte frazionaria

$N = \quad c_{-1}B^{-1} + c_{-2}B^{-2} + .. + c_{-m}B^{-m} + .. + \ldots$

$NB = \quad c_{-1}B^{0} + c_{-2}B^{-1} .. + c_{-m}B^{-m+1} + \ldots + \ldots =$

$\mathbf{c_{-1}} + (c_{-2}B^{-1} + .. + c_{-m}B^{-m+1} + .. + ..) = \mathbf{I} + N'$

(I=intero, N'<1)

Quindi: $\mathbf{c_{-1}} = \lfloor NB \rfloor$ (parte inferiore di NB)

$N' = NB - c_{-1} = (c_{-2}B^{-1} .. + c_{-m}B^{-m+1} + .. + ..)$

- Le altre cifre si isolano in modo analogo:

  c_{-2} = parte intera di N'B

- Finché precisione voluta *oppure* N=0

Algoritmo di conversione
da base 10 a B (0<N<1)

Esempio: $(0.8125)_{10} = (??)_2$

N<1 valore frazionario
da convertire,
B base di arrivo,
m cifre (precisione)

i←1;
while N<>0 **and** i≤m **do**
1. $c_{-i} \leftarrow \lfloor NB \rfloor$;
2. N←NB - c_{-i};
3. I←i+1

endwhile

N	2N	Trunc(2N)	Cifra
0.8125	1.625	1	$d_{-1}=1$
0.625	1.25	1	$d_{-2}=1$
0.25	0.5	0	$d_{-3}=0$
0. 5	1.0	1	$d_{-4}=1$
0			

$(0.8125)_{10} = (0.1101)_2$

Esempio: $(12.25)_{10} \rightarrow (..)_2$

- 12/2 = 6 resto 0 $\rightarrow d_0=0$
- 6/2 = 3 resto 0 $\rightarrow d_1=0$
- 3/2 = 1 resto 1 $\rightarrow d_2=1$
- 1/2 = 0 resto 1 $\rightarrow d_3=1$

- 0.25 x 2 = 0.50, parte intera 0 $\rightarrow c_{-1}=0$
- 0.50 x 2 = 1.0, parte intera 1 $\rightarrow c_{-2}=1$

$$(12.25)_{10} \rightarrow (1100.01)_2$$

Aritmetica reale

Rappresentazione in virgola mobile

Rappresentazione in forma normalizzata in base b

$$X = m \cdot b^e$$

- *e* è la caratteristica in base b di X: intero relativo
- *m* è la mantissa in base b di X: numero frazionario tale che $1/b \leq m < 1$

Se rappresentata dalla sequenza: $c_1\ c_2\ c_3\ \ldots$
allora rappresenta il numero: $c_1 \cdot b^{-1} + c_2 \cdot b^{-2} + c_3 \cdot b^{-2} + \ldots$

Esempio: $(5)_{10} = (101)_2$

$m = (0.1010000\ldots0)_2$
$e = (11)_2$

Rappresentazione in virgola mobile

Fissati:

- *k* bit per la mantissa
- *h* bit per la caratteristica
- 1 bit per il segno

l'insieme dei numeri rappresentabili è limitato:

$$1/2 \leq |m| \leq \Sigma_{i=1}^{k} 2^{-i}$$
$$|e| \leq 2^{h-1} - 1$$

Assunzione realistica: 32 bit
- 24 bit per la *mantissa*
- 7 bit per la *caratteristica* (in complemento)
- 1 bit per il *segno*

Rappresentazione in virgola mobile

Insieme *F* dei numeri rappresentabili:

- *F* insieme finito di razionali
- *F* simmetrico rispetto allo zero
- elementi non uniformemente distribuiti sull'asse (vicino 0 densi, radi intorno al massimo rappresentabile)
- molti razionali non appartengono all'insieme
- *F* non è chiuso rispetto alle operazioni
- reale X rappresentato dall'elemento in *F* più vicino
- la funzione che associa un reale X all'elemento in *F* più vicino è la funzione di arrotondamento

Limitazioni aritmetiche

Il numero di bit utilizzato per la rappresentazione è limitato:

- perdita di precisione
- arrotondamento: la mantissa non è sufficiente per rapp. tutte cifre significative del numero
- overflow: caratteristica non è sufficiente (numero troppo grande)
- underflow: numeri troppo piccolo rapp. con 0

Formati standard proposti dalla IEEE (Institute of Electrical and Electronics Engineers)
- singola precisione: 32 bit
- doppia precisione: 64 bit
- quadrupla precisione: 128 bit

Numeri Razionali in C

Tipi di dato: float, double, long double

```
float n;
double m;
long double g;
```

Specificatori di formato:

```
printf("%f %lf %Lf",n, m, g);
scanf("%f %lf %Lf",&n, &m, &g);
```

Altri formati:
%e, %E, %g, %G

Funzione sizeof: printf("%d %d %d",sizeof(n), sizeof(m), sizeof(g));

(caso DevC++) stampa: 4 8 8

Caratteri in C

Tipo char

char ch;

sizeof(ch) = 1

Rappresentati in codice ASCII

ASCII (American Standard Code For Interchange Information) ideato nel 1965. In 8 bit codificati 256 caratteri divisi in "standard" ed "esteso".
Lo Standard rappresenta 128 caratteri (da 00 a 7F nel sistema esadecimale). I primi 32 sono caratteri di controllo, gli altri 96 segni di interpunzione, cifre da 0 a 9 e alle lettere dell'alfabeto latino, maiuscole e minuscole.

Specificatore formato: %c

scanf("%c", &ch);

printf("%c", ch);

Codice ASCII

valore decim. →	0	16	32	48	64	80	96	112
↓ valore esadecimale	0	1	2	3	4	5	6	7
0	0	vuoto (NULL)	►	vuoto (spazio)	0	@	P	'	p
1	1	○	◄	!	1	A	Q	a	q
2	2	●	↕	"	2	B	R	b	r
3	3	♥	‼	#	3	C	S	c	s
4	4	♦	¶	$	4	D	T	d	t
5	5	♣	§	%	5	E	U	e	u
6	6	♠	▬	&	6	F	V	f	v
7	7	•	↨	'	7	G	W	g	w
8	8	◘	↑	(	8	H	X	h	x
9	9	○	↓	)	9	I	Y	i	y
10	A	◙	→	*	:	J	Z	j	z
11	B	♂	←	+	;	K	[	k	{
12	C	♀	∟	,	<	L	\	l	¦
13	D	♪	↔	—	=	M	]	m	}
14	E	♫	▲	.	>	N	^	n	~
15	F	☼	▼	/	?	O	_	o	⌂

valore decim. →	128	144	160	176	192	208	224	240
↓ valore esadecimale	8	9	A	B	C	D	E	F
0	Ç	É	á	▒	└	╨	α	≡
1	ü	æ	í	▒	┴	╤	ß	±
2	é	Æ	ó	▒	┬	╥	Γ	≥
3	â	ô	ú	│	├	╙	π	≤
4	ä	ö	ñ	┤	─	╘	Σ	⌠
5	à	ò	Ñ	╡	┼	╒	σ	⌡
6	å	û	ª	╢	╞	╓	µ	÷
7	ç	ù	º	╖	╟	╫	τ	≈
8	ê	ÿ	¿	╕	╚	╪	Φ	°
9	ë	Ö	⌐	╣	╔	┘	Θ	•
10	è	Ü	¬	║	╩	┌	Ω	·
11	ï	¢	½	╗	╦	■	δ	√
12	î	£	¼	╝	╠	▄	∞	n
13	ì	¥	¡	╜	═	▌	φ	2
14	Ä	₧	«	╛	╬	▐	∈	■
15	Å	ƒ	»	┐	╧	▀	∩	vuoto (FF)

Funzioni della libreria matematica

Per utilizzarle occorre: `#include<math.h>`

Sia argomenti che valore di ritorno reali in doppia precisione, ovvero tipo `double` (non `float`)

Funzioni disponibili:

`sqrt(x)`	radice quadrata
`exp(x)`	e^x
`log(x)`	logaritmo naturale
`log10(x)`	logaritmo base 10
`fabs(x)`	valore assoluto
`ceils(x)`	arrotonda all'intero più piccolo > x
`floor(x)`	arrotonda all'intero più grande < x
`pow(x,y)`	x^y
`fmod(x,y)`	resto di x/y (in virgola mobile)
`sin(x), cos(x), tan(x)`	trigonometriche (x in radianti)