Fondamenti di Informatica

Modulo 2

Secondo Semestre

Anno 2025/26 • Prof. GRASSI VINCENZO

Diego Marini
diego@marini.work | marini.work

Ultimo aggiornamento: 8 marzo 2026

Nota: appunti personali in continuo aggiornamento. Possibili errori: segnalazioni benvenute.

Indice

Come sono strutturati questi appunti

- **Concetti** con esempi minimi e chiari.
- **Schemi** riassuntivi per ripasso veloce.
- **Esercizi** e **tracce** con ragionamento.
- **Codice** in blocchi monospazio (leggibile e replicabile).

1 Rappresentazioni

1.1 Idea generale di rappresentazione

Una **rappresentazione** è un modo per associare a un oggetto astratto una scrittura concreta mediante simboli.
Nel caso dei numeri, rappresentare significa associare a ciascun numero una stringa di simboli che ne consenta la codifica e, successivamente, l'interpretazione.

Nel seguito ci interesserà in particolare la rappresentazione dei numeri.

Osservazione 1.1. Una stessa informazione può essere rappresentata in modi diversi.
Ad esempio, il numero tre può essere scritto come

$$3, \qquad \text{tre}, \qquad \text{III}.$$

Queste scritture sono diverse, ma si riferiscono tutte allo stesso valore numerico.

1.2 Rappresentazione di numeri naturali

Indichiamo con

$$\mathbb{N}$$

l'insieme dei numeri naturali.
Sia poi S un insieme finito di simboli, che chiameremo *alfabeto*.
Con

$$S^*$$

indichiamo l'insieme di tutte le stringhe finite costruibili mediante i simboli di S.

Osservazione 1.2. Esempi:

$$S = \{a\} \quad \Longrightarrow \quad S^* = \{\lambda, a, aa, aaa, aaaa, \ldots\}$$

$$S = \{a, b\} \quad \Longrightarrow \quad S^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, \ldots\}$$

dove λ indica la stringa vuota.

Una **rappresentazione** dei numeri naturali è una funzione iniettiva

$$r : \mathbb{N} \to S^*.$$

L'iniettività garantisce che numeri distinti abbiano rappresentazioni distinte.

1.3 Interpretazione o decodifica

Una volta fissata una rappresentazione r, si può definire il processo inverso, cioè l'**interpretazione** o **decodifica**.

La funzione di interpretazione associata a r è, in generale, una funzione parziale

$$\mathcal{I} : S^* \to \mathbb{N},$$

definita da

$$\mathcal{I}(\sigma) = \begin{cases} n & \text{se } r(n) = \sigma, \\ \text{indefinita} & \text{altrimenti.} \end{cases}$$

Quindi la rappresentazione trasforma un numero in una stringa, mentre l'interpretazione trasforma una stringa nel numero rappresentato, quando ciò è possibile.

1.4 Rappresentazioni additive

Una rappresentazione si dice **additiva** se ogni simbolo ha un valore numerico prefissato che non dipende dalla posizione che occupa nella stringa.

Esempio 1: rappresentazione unaria

Poniamo

$$S = \{I\}.$$

Allora si può definire:

$$r(1) = I, \qquad r(2) = II, \qquad r(3) = III, \qquad r(4) = IIII, \ldots$$

In questo caso ogni simbolo I vale sempre 1.

Esempio 2: rappresentazione romana

Nella numerazione romana i simboli principali sono

$$S = \{I, V, X, L, C, D, M\}.$$

Anche qui ogni simbolo ha un valore prefissato:

$$I = 1, \quad V = 5, \quad X = 10, \quad L = 50, \quad C = 100, \quad D = 500, \quad M = 1000.$$

Osservazione 1.3. Nelle rappresentazioni additive i simboli ripetuti mantengono sempre lo stesso valore, indipendentemente dalla posizione.

Ad esempio, nella scrittura `XXVIII`, il simbolo X rappresenta sempre una decina.

Osservazione 1.4. Il limite principale delle rappresentazioni additive è la scarsa efficienza spaziale: la lunghezza della scrittura cresce quasi linearmente con il numero rappresentato.

Questo rende meno efficienti anche molti algoritmi di calcolo.

1.5 Rappresentazioni posizionali dei numeri interi

Una rappresentazione si dice **posizionale** se il valore di un simbolo dipende dalla posizione che occupa nella stringa.

Fissiamo un intero

$$B \geq 2,$$

detto **base** della rappresentazione.
L'insieme dei simboli utilizzati sarà

$$S_B = \{0, 1, \dots, B-1\}.$$

La rappresentazione posizionale in base B di un numero naturale n è una stringa di cifre

$$(c_{p-1}c_{p-2}\dots c_0)_B, \qquad c_i \in S_B,$$

la cui interpretazione è

$$\mathcal{I}_B(c_{p-1}c_{p-2}\dots c_0) = \sum_{i=0}^{p-1} c_i B^i.$$

Osservazione 1.5. Nella rappresentazione posizionale, la cifra più a destra pesa B^0, la successiva pesa B^1, quella ancora successiva pesa B^2, e così via.

Esempio 1.6. La stringa 110 rappresenta valori diversi a seconda della base:

$$(110)_2 = 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 6$$

$$(110)_3 = 1 \cdot 3^2 + 1 \cdot 3^1 + 0 \cdot 3^0 = 12$$

$$(110)_{10} = 1 \cdot 10^2 + 1 \cdot 10^1 + 0 \cdot 10^0 = 110.$$

Osservazione 1.7. La stessa stringa di simboli può rappresentare numeri diversi se cambia la base. Per questo motivo, nella rappresentazione posizionale, la base deve sempre essere specificata.

Osservazione 1.8. Nelle rappresentazioni posizionali la lunghezza della rappresentazione di n cresce come il logaritmo in base B di n.
Questo le rende molto più efficienti delle rappresentazioni additive.

1.6 Prime proprietà delle rappresentazioni posizionali

Rappresentazione delle potenze della base

Per ogni intero $k \geq 0$, si ha

$$r_B(B^k) = 100\dots0$$

cioè un 1 seguito da k zeri.
Ad esempio:

$$(8)_{10} = (1000)_2, \qquad (27)_{10} = (1000)_3.$$

Massimo numero rappresentabile con p cifre

Se si usano al massimo p cifre in base B, il massimo numero rappresentabile è

$$M = B^p - 1.$$

Infatti il valore massimo si ottiene scegliendo tutte le cifre uguali a $B - 1$:

$$M = (B-1)B^{p-1} + (B-1)B^{p-2} + \cdots + (B-1).$$

Raccogliendo $(B - 1)$:

$$M = (B-1)\sum_{i=0}^{p-1} B^i.$$

Raccogliendo il fattore $(B - 1)$ si ottiene

$$M = (B-1)(B^{p-1} + B^{p-2} + \cdots + 1).$$

Per calcolare questa somma osserviamo che

$$B(B^{p-1} + B^{p-2} + \cdots + 1) = B^p + B^{p-1} + \cdots + B.$$

Sottraendo membro a membro le due espressioni otteniamo

$$B(B^{p-1} + B^{p-2} + \cdots + 1) - (B^{p-1} + B^{p-2} + \cdots + 1) = B^p - 1.$$

Il membro sinistro si può raccogliere:

$$(B-1)(B^{p-1} + B^{p-2} + \cdots + 1) = B^p - 1.$$

Quindi

$$M = B^p - 1.$$

Osservazione 1.9. L'idea è che moltiplicando la somma per B tutte le potenze si "spostano" di una posizione.

Sottraendo le due espressioni tutti i termini intermedi si cancellano e rimangono solo B^p e 1.

1.7 Rappresentazione con un numero finito di cifre

Fissati una base B e un intero $p \geq 1$, i numeri rappresentabili con al più p cifre in base B sono precisamente gli elementi dell'insieme

$$\mathbb{N}_{B,p} = \{n \in \mathbb{N} : 0 \leq n \leq B^p - 1\}.$$

Osservazione 1.10. Se $n_1, n_2 \in \mathbb{N}_{B,p}$, non è detto che anche

$$n_1 + n_2, \qquad n_1 - n_2, \qquad n_1 \cdot n_2, \qquad n_1/n_2$$

appartengano ancora a $\mathbb{N}_{B,p}$.

Può infatti verificarsi il fenomeno del **trabocco** (*overflow*).

Esempio 1.11. In base 10, usando al più 2 cifre, il massimo numero rappresentabile è

$$10^2 - 1 = 99.$$

Quindi, ad esempio,

$$70 + 50 = 120$$

non è rappresentabile con sole due cifre decimali.

1.8 Conversione di un intero dalla base 10 alla base B

Supponiamo di partire da un numero naturale n scritto in base 10 e di voler determinarne la rappresentazione in base B:

$$r_B(n) = (c_p c_{p-1} \dots c_0)_B.$$

Il problema consiste nel determinare le cifre $c_0, c_1, \dots, c_p$.

Idea dell'algoritmo

Dalla formula

$$n = c_0 + c_1 B + c_2 B^2 + \cdots + c_p B^p$$

si vede che:

- c_0 è il resto della divisione di n per B;
- il quoziente della divisione di n per B contiene le cifre successive;
- ripetendo il procedimento si ottengono tutte le cifre.

Algoritmo di conversione di un intero in base B:

ConvInt(B, n)

1. porre $i \leftarrow 0$;
2. finché $n > 0$:
   (a) $c_i \leftarrow$ resto della divisione di n per B;
   (b) $n \leftarrow$ quoziente della divisione di n per B;
   (c) $i \leftarrow i + 1$;
3. il risultato è la stringa

$$(c_{i-1} c_{i-2} \dots c_0)_B.$$

Osservazione 1.12. L'algoritmo calcola prima la cifra meno significativa c_0, poi c_1, poi c_2, e così via.

Per questo motivo le cifre ottenute vanno lette in ordine inverso.

Esempio 1.13. Convertiamo il numero $(13)_{10}$ in base 2.

Eseguiamo divisioni successive per 2:

$$13 = 2 \cdot 6 + 1$$

$$6 = 2 \cdot 3 + 0$$

$$3 = 2 \cdot 1 + 1$$

$$1 = 2 \cdot 0 + 1.$$

Leggendo i resti dal basso verso l'alto si ottiene

$$(13)_{10} = (1101)_2.$$

1.9 Rappresentazione posizionale di numeri reali

La rappresentazione posizionale si estende naturalmente ai numeri reali.
Una scrittura del tipo

$$(c_{h-1}c_{h-2}\dots c_0, c_{-1}c_{-2}\dots c_{-k})_B$$

ha interpretazione

$$\mathcal{I}_B(c_{h-1}\dots c_0, c_{-1}\dots c_{-k}) = \sum_{i=-k}^{h-1} c_i B^i.$$

Esempio 1.14. In base 10:

$$(10,011)_{10} = 1 \cdot 10^1 + 0 \cdot 10^0 + 0 \cdot 10^{-1} + 1 \cdot 10^{-2} + 1 \cdot 10^{-3}$$

$$= 10 + 0{,}01 + 0{,}001 = 10{,}011.$$

In base 2:

$$(10,011)_2 = 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3}$$

$$= 2 + 0{,}25 + 0{,}125 = 2{,}375.$$

1.10 Conversione di un numero reale dalla base 10 alla base B

Sia

$$x = i + f, \qquad i \in \mathbb{N}, \qquad 0 \le f < 1,$$

dove i è la parte intera e f la parte frazionaria.
La conversione si effettua separatamente:

- la parte intera si converte con l'algoritmo già visto;
- la parte frazionaria richiede un procedimento diverso.

1.11 Conversione della parte frazionaria

Supponiamo di voler rappresentare

$$0 < f < 1$$

in base B:

$$r_B(f) = (c_{-1}c_{-2}\dots c_{-k})_B.$$

Allora

$$f = c_{-1}B^{-1} + c_{-2}B^{-2} + \dots + c_{-k}B^{-k}.$$

Moltiplicando per B:

$$Bf = c_{-1} + c_{-2}B^{-1} + \dots + c_{-k}B^{-k+1}.$$

Quindi:

- la parte intera di Bf fornisce c_{-1};
- la parte frazionaria residua permette di calcolare la cifra successiva.

Algoritmo di conversione della parte frazionaria:

$$\text{ConvFraz}(B, f)$$

1. porre $i \leftarrow 1$;
2. finché $f > 0$:
   (a) $c_{-i} \leftarrow$ parte intera di $f \cdot B$;
   (b) $f \leftarrow$ parte frazionaria di $f \cdot B$;
   (c) $i \leftarrow i + 1$;
3. il risultato è

$$(c_{-1}c_{-2}\dots)_B.$$

Esempio 1.15. Convertiamo $0{,}625$ dalla base 10 alla base 2.
Moltiplichiamo successivamente per 2:

$$0{,}625 \cdot 2 = 1{,}25 \qquad \Longrightarrow \qquad c_{-1} = 1$$

$$0{,}25 \cdot 2 = 0{,}5 \qquad \Longrightarrow \qquad c_{-2} = 0$$

$$0{,}5 \cdot 2 = 1{,}0 \qquad \Longrightarrow \qquad c_{-3} = 1.$$

Quindi

$$(0{,}625)_{10} = (0{,}101)_2.$$

1.12 Approssimazione

A differenza dell'algoritmo per gli interi, l'algoritmo per la parte frazionaria può non terminare mai.

Osservazione 1.16. Ad esempio, la conversione di $0{,}8$ dalla base 10 alla base 2 produce una rappresentazione infinita.

Per questo motivo, in pratica, si tronca la rappresentazione a un numero fissato di cifre frazionarie.

Versione troncata dell'algoritmo:

$$\text{ConvFraz}(B, f, k)$$

1. porre $i \leftarrow 1$;
2. finché $f > 0$ e $i \leq k$:
   (a) $c_{-i} \leftarrow$ parte intera di $f \cdot B$;
   (b) $f \leftarrow$ parte frazionaria di $f \cdot B$;
   (c) $i \leftarrow i + 1$;
3. il risultato è la rappresentazione troncata

$$(c_{-1}c_{-2}\ldots c_{-k})_B.$$

Osservazione 1.17. Se si tronca a k cifre frazionarie in base B, l'errore assoluto soddisfa

$$|\,\text{valore esatto} - \text{valore troncato}\,| \leq B^{-k}.$$

Quindi l'errore diminuisce all'aumentare di k.

Osservazione 1.18. Queste osservazioni mostrano che la rappresentazione dei numeri in un calcolatore è sempre il risultato di un compromesso tra precisione, spazio occupato e semplicità degli algoritmi di calcolo.

1.13 Confronto tra rappresentazioni additive e posizionali

Caratteristica	Rappresentazione additiva	Rappresentazione posizionale
Valore dei simboli	Fisso	Dipende dalla posizione
Esempi	Numeri romani	Sistema decimale, binario
Efficienza spaziale	Bassa	Alta
Algoritmi di calcolo	Poco efficienti	Molto efficienti
Uso nei calcolatori	No	Sì

Osservazione 1.19. In un sistema di rappresentazione posizionale, la scelta della base determina il numero di simboli disponibili per rappresentare i numeri.

Ad esempio:

$$\text{base } 2 \Rightarrow \{0, 1\}$$

$$\text{base } 10 \Rightarrow \{0, 1, 2, \ldots, 9\}$$

$$\text{base } 16 \Rightarrow \{0, 1, \ldots, 9, A, B, C, D, E, F\}$$

All'aumentare della base diminuisce la lunghezza delle rappresentazioni, ma aumenta il numero di simboli necessari.

1.14 Perché i calcolatori utilizzano la base binaria

I calcolatori rappresentano i numeri utilizzando la **base 2**, cioè il sistema binario.

Nel sistema binario le cifre possibili sono

$$\{0, 1\}.$$

Ogni numero viene quindi rappresentato come una sequenza di bit.

La scelta della base binaria dipende da motivi fisici ed elettronici.

- I circuiti elettronici possono assumere facilmente due stati distinti (ad esempio acceso/-spento oppure alta/bassa tensione).
- Questi due stati possono essere naturalmente associati alle cifre

$$0 \quad \text{e} \quad 1.$$

- Utilizzare solo due simboli rende i circuiti più semplici, più affidabili e meno sensibili al rumore elettrico.

Osservazione 1.20. Anche se i calcolatori utilizzano internamente la base 2, gli utenti lavorano normalmente in base 10.

Per questo motivo i sistemi informatici devono continuamente convertire numeri tra base 10 e base 2.

1.15 Riassunto della sezione

In questa sezione abbiamo introdotto il concetto di rappresentazione dei numeri.
In particolare:

- una rappresentazione associa a ogni numero una stringa di simboli;
- le rappresentazioni additive assegnano ai simboli un valore indipendente dalla posizione;
- le rappresentazioni posizionali assegnano ai simboli un valore dipendente dalla posizione;
- nelle rappresentazioni posizionali un numero si interpreta mediante la formula

$$\sum c_i B^i;$$

- la conversione tra basi può essere effettuata mediante algoritmi basati su divisioni o moltiplicazioni successive.

Le rappresentazioni posizionali sono quelle utilizzate nei sistemi di calcolo.