

Corso di Laboratorio di programmazione e calcolo

Docente: **Stefano Finzi Vita**

Pagina del corso: http://www1.mat.uniroma1.it/mat_cms/pres_corso.php?corso_da_presentare=1851&lingua=italiano
oppure link da <http://www1.mat.uniroma1.it/people/finzi/>

Presentazione del corso

(tutto quello che serve sapere prima di cominciare)

Orario:

- **Lezioni:** Aula II, Lunedì/Mercoledì 11:15-13:00 - Venerdì 9:15-11:00
- **Esercitazioni pratiche:** Lab1-Lab2, Martedì 9:15-11:00

Ricevimento: **Studio 17** (piano terra), Mercoledì 14:30-16:00

Tutore: **Alessandro ALLA** (in Laboratorio e nel suo orario di ricevimento)

Libri di testo:

- dispense in rete distribuite dal docente (v. pagina del corso)
- un manuale di C++ (**Aguilar** o testo equivalente)
- un testo di analisi numerica
(**Quarteroni-Sacco-Saleri** o testo equivalente)

Struttura del corso

- 1 Introduzione alla programmazione
- 2 Basi elementari del linguaggio C++ (I/O, cicli, alternative, array)
- 3 Costruzione di semplici programmi di "manipolazione di numeri"

Primo esonero (11-15 novembre 2013)

- 4 Funzioni in C++, uso di files di dati, grafica con Gnuplot
- 5 Nozioni di analisi numerica elementare:
 - ▶ Risoluzione di sistemi lineari
 - ▶ Metodi per la ricerca degli zeri
 - ▶ Approssimazione di funzioni (interpolazione, Taylor)
 - ▶ Calcolo approssimato di derivate e integrali

Secondo esonero (13-17 gennaio 2014)

Modalità degli esami

Form per la registrazione al canale entro il 31 ottobre:

<http://www.mat.uniroma1.it/didattica/registrazione/>

- 1 **Prova scritta** (o **esoneri**):
scrittura di un programma in C++ in grado di risolvere alcuni problemi matematici proposti (in aula, carta e penna)

Modalità degli esami

Form per la registrazione al canale entro il 31 ottobre:

<http://www.mat.uniroma1.it/didattica/registrazione/>

- 1 **Prova scritta** (o **esoneri**):
scrittura di un programma in C++ in grado di risolvere alcuni problemi matematici proposti (in aula, carta e penna)
- 2 **Valutazione dei programmi**:
esame dei programmi in C++ scritti per risolvere alcuni degli esercizi proposti dai fogli settimanali (in laboratorio)

Modalità degli esami

Form per la registrazione al canale entro il 31 ottobre:

<http://www.mat.uniroma1.it/didattica/registrazione/>

- 1 **Prova scritta** (o **esoneri**):
scrittura di un programma in C++ in grado di risolvere alcuni problemi matematici proposti (in aula, carta e penna)
- 2 **Valutazione dei programmi**:
esame dei programmi in C++ scritti per risolvere alcuni degli esercizi proposti dai fogli settimanali (in laboratorio)
- 3 **Prova orale**:
comprensione dei risultati di analisi numerica introdotti a lezione

Il Laboratorio

- 30 postazioni doppie (Lab1) + 17 postazioni singole (Lab2)
- PC con sistema operativo *Linux*
- Editor per la scrittura dei programmi: *Emacs*
- Compilatore C++: *g++*

Istruzioni: <http://www.mat.uniroma1.it/people/seghini/LPC/>

Ogni settimana (giovedì o venerdì) vi verrà proposto in rete un nuovo foglio di esercizi da risolvere scrivendo programmi. E' fondamentale che ci lavoriate per conto vostro prima di venire in laboratorio.

Il martedì si discuteranno i risultati ottenuti, le difficoltà incontrate, si proporranno possibili soluzioni alternative, ecc.

A casa:

- Sistemi operativi Windows: scaricare *DEV-C++*
- Sistemi operativi Mac: *g++* in genere preinstallato

- Questo NON E' un corso di Informatica

- Questo NON E' un corso di Informatica
- Questo NON E' un corso di Programmazione in C o C++

- Questo NON E' un corso di Informatica
- Questo NON E' un corso di Programmazione in C o C++
- Questo NON E' un corso di Analisi numerica

- Questo NON E' un corso di Informatica
- Questo NON E' un corso di Programmazione in C o C++
- Questo NON E' un corso di Analisi numerica
- Questo E' un corso che attraverso l'apprendimento delle strutture di base di un linguaggio di programmazione (nel nostro caso il C++) e l'introduzione di alcune nozioni elementari di calcolo numerico, vi può aiutare a risolvere semplici problemi di matematica, e a capire meglio alcune nozioni introdotte parallelamente nei corsi di Algebra lineare e Calcolo

Dal problema ai risultati

- **Analisi del problema** *Formulare e capire a fondo il problema, scomporlo in parti elementari di facile gestione*

Dal problema ai risultati

- **Analisi del problema** *Formulare e capire a fondo il problema, scomporlo in parti elementari di facile gestione*
- **Algoritmo** *Scrittura di una sequenza di passi discreti, di lunghezza finita, deterministici, ripetibili, che traducono l'analisi fatta*
[diagramma a blocchi o pseudocodice]

Dal problema ai risultati

- **Analisi del problema** *Formulare e capire a fondo il problema, scomporlo in parti elementari di facile gestione*
- **Algoritmo** *Scrittura di una sequenza di passi discreti, di lunghezza finita, deterministici, ripetibili, che traducono l'analisi fatta [diagramma a blocchi o pseudocodice]*
- **Programma sorgente** *Traduzione dell'algoritmo in un linguaggio di programmazione, cioè in un insieme di parole, simboli e regole rigide per combinarli insieme [tramite un **editor**]*

Dal problema ai risultati

- **Analisi del problema** *Formulare e capire a fondo il problema, scomporlo in parti elementari di facile gestione*
- **Algoritmo** *Scrittura di una sequenza di passi discreti, di lunghezza finita, deterministici, ripetibili, che traducono l'analisi fatta*
[diagramma a blocchi o pseudocodice]
- **Programma sorgente** *Traduzione dell'algoritmo in un linguaggio di programmazione, cioè in un insieme di parole, simboli e regole rigide per combinarli insieme* [tramite un **editor**]
- **Compilazione**
 - ▶ Preprocessore (*rimuove i commenti e interpreta le direttive speciali*)
 - ▶ Compilatore (*traduce il sorgente in codice macchina*)
 - ▶ Assembler (*crea il codice oggetto*)
 - ▶ Link (*combina le funzioni di libreria col main per creare l'eseguibile*)

Dal problema ai risultati

- **Analisi del problema** *Formulare e capire a fondo il problema, scomporlo in parti elementari di facile gestione*
- **Algoritmo** *Scrittura di una sequenza di passi discreti, di lunghezza finita, deterministici, ripetibili, che traducono l'analisi fatta*
[diagramma a blocchi o pseudocodice]
- **Programma sorgente** *Traduzione dell'algoritmo in un linguaggio di programmazione, cioè in un insieme di parole, simboli e regole rigide per combinarli insieme* [tramite un **editor**]
- **Compilazione**
 - ▶ Preprocessore (*rimuove i commenti e interpreta le direttive speciali*)
 - ▶ Compilatore (*traduce il sorgente in codice macchina*)
 - ▶ Assembler (*crea il codice oggetto*)
 - ▶ Link (*combina le funzioni di libreria col main per creare l'eseguibile*)
- **Esecuzione Input** (*dati da tastiera*) → **Output** (*risultati su schermo*)

Dal problema ai risultati

- **Analisi del problema** *Formulare e capire a fondo il problema, scomporlo in parti elementari di facile gestione*
- **Algoritmo** *Scrittura di una sequenza di passi discreti, di lunghezza finita, deterministici, ripetibili, che traducono l'analisi fatta [diagramma a blocchi o pseudocodice]*
- **Programma sorgente** *Traduzione dell'algoritmo in un linguaggio di programmazione, cioè in un insieme di parole, simboli e regole rigide per combinarli insieme [tramite un **editor**]*
- **Compilazione**
 - ▶ Preprocessore (*rimuove i commenti e interpreta le direttive speciali*)
 - ▶ Compilatore (*traduce il sorgente in codice macchina*)
 - ▶ Assembler (*crea il codice oggetto*)
 - ▶ Link (*combina le funzioni di libreria col main per creare l'eseguibile*)
- **Esecuzione Input** (*dati da tastiera*) → **Output** (*risultati su schermo*)
- **Post-processing** *Analisi dei risultati, uso di librerie grafiche, eventuale correzione del programma*

La programmazione

- **Algoritmo** (da *Mohammed al-Khowarizmi*, matematico persiano del IX secolo) + **Strutture dati** = **Programmi**
- Senza algoritmo non esistono programmi, ma un algoritmo non dipende dal linguaggio usato o dal computer che lo esegue.
- **Programma** = sequenza di istruzioni che traduce l'algoritmo

La programmazione

- **Algoritmo** (da *Mohammed al-Khowarizmi*, matematico persiano del IX secolo) + **Strutture dati** = **Programmi**
- Senza algoritmo non esistono programmi, ma un algoritmo non dipende dal linguaggio usato o dal computer che lo esegue.
- **Programma** = sequenza di istruzioni che traduce l'algoritmo
- **Programmazione strutturata** (procedurale) (**C**, **Fortran**, **Pascal**)
 - ▶ ogni istruzione indica al compilatore di eseguire un compito preciso
 - ▶ quando i programmi diventano complicati, li si struttura in sottoprogrammi (**funzioni** o subroutine) ciascuno dei quali svolge appunto una funzione precisa e possiede un'interfaccia definita con cui comunica con gli altri sottoprogrammi
 - ▶ Ci sono **dati locali** utilizzati esclusivamente da una singola funzione e **dati globali** accessibili da tutte le funzioni

La programmazione

- **Algoritmo** (da *Mohammed al-Khowarizmi*, matematico persiano del IX secolo) + **Strutture dati** = **Programmi**
- Senza algoritmo non esistono programmi, ma un algoritmo non dipende dal linguaggio usato o dal computer che lo esegue.
- **Programma** = sequenza di istruzioni che traduce l'algoritmo
- **Programmazione strutturata** (procedurale) (**C**, **Fortran**, **Pascal**)
 - ▶ ogni istruzione indica al compilatore di eseguire un compito preciso
 - ▶ quando i programmi diventano complicati, li si struttura in sottoprogrammi (**funzioni** o subroutine) ciascuno dei quali svolge appunto una funzione precisa e possiede un'interfaccia definita con cui comunica con gli altri sottoprogrammi
 - ▶ Ci sono **dati locali** utilizzati esclusivamente da una singola funzione e **dati globali** accessibili da tutte le funzioni
- **Programmazione orientata agli oggetti** (**C++**)
 - ▶ Un programma è un insieme finito di **oggetti** (dati + funzioni che operano su essi) che comunicano tra loro tramite messaggi

Requisiti di un metodo numerico

In genere per risolvere un problema non esiste un solo metodo e quindi più algoritmi sono possibili. Anche se tutti forniscono i risultati voluti, è possibile valutarne la convenienza in funzione di alcuni parametri misurabili.

- **Convergenza**: gli effetti degli errori di troncamento e di arrotondamento possono essere resi arbitrariamente piccoli accrescendo lo sforzo di calcolo.
- **Accuratezza**: gli errori sono piccoli rispetto a una tolleranza fissata; l'ordine di accuratezza indica l'ordine di infinitesimo dell'errore in funzione dei parametri usati
- **Affidabilità**: testato sufficientemente, fornisce i risultati voluti
- **Efficienza**: la complessità di calcolo è la minore possibile
 - ▶ numero di operazioni: influenza il tempo di calcolo (**CPU time**)
 - ▶ quantità di memoria richiesta: va confrontata con quella disponibile

I numeri e la loro rappresentazione: caso continuo

- **Numeri reali** = **numeri decimali illimitati**

$$x = n.a_1a_2 \dots a_k \dots = n + \frac{a_1}{10} + \frac{a_2}{10^2} + \dots + \frac{a_k}{10^k} + \dots$$

con $n \in \mathbb{Z}$, $a_j = 0, 1, \dots, 9$.

- ▶ periodici \rightarrow razionali : $2(.00000\dots)$, $3.5(00000\dots)$, $0.\bar{6}$, $7.02\overline{428}$
- ▶ non periodici \rightarrow irrazionali : $\sqrt{2}$, $\sqrt{3}$, π , e

I numeri e la loro rappresentazione: caso continuo

- **Numeri reali** = **numeri decimali illimitati**

$$x = n.a_1a_2 \dots a_k \dots = n + \frac{a_1}{10} + \frac{a_2}{10^2} + \dots + \frac{a_k}{10^k} + \dots$$

con $n \in \mathbb{Z}$, $a_j = 0, 1, \dots, 9$.

- ▶ periodici \rightarrow razionali : $2(.00000\dots)$, $3.5(00000\dots)$, $0.\bar{6}$, $7.02\bar{428}$
 - ▶ non periodici \rightarrow irrazionali : $\sqrt{2}$, $\sqrt{3}$, π , e
- Ogni numero reale può essere approssimato (con precisione arbitraria) con un numero razionale [**densità dei razionali in \mathbb{R}**]
 - **Continuità dei numeri reali**: la retta reale "*non ha buchi*"

Approssimazione di un numero reale

- Due numeri reali con la stessa parte intera e le prime k cifre decimali uguali differiscono per meno di 10^{-k} :

$$x = n.a_1a_2 \dots a_k a_{k+1} \dots, \quad y = n.a_1a_2 \dots a_k b_{k+1} \dots$$

$$\Rightarrow |x - y| \leq |a_{k+1} - b_{k+1}|10^{-(k+1)} < 10^{-k}$$

Approssimazione di un numero reale

- Due numeri reali con la stessa parte intera e le prime k cifre decimali uguali differiscono per meno di 10^{-k} :

$$x = n.a_1a_2 \dots a_k a_{k+1} \dots, \quad y = n.a_1a_2 \dots a_k b_{k+1} \dots$$

$$\Rightarrow |x - y| \leq |a_{k+1} - b_{k+1}| 10^{-(k+1)} < 10^{-k}$$

- non vale il viceversa: $x = 0.29999$, $y = 0.3 \Rightarrow |x - y| = 10^{-5}$

Approssimazione di un numero reale

- Due numeri reali con la stessa parte intera e le prime k cifre decimali uguali differiscono per meno di 10^{-k} :

$$x = n.a_1a_2 \dots a_k a_{k+1} \dots, \quad y = n.a_1a_2 \dots a_k b_{k+1} \dots$$

$$\Rightarrow |x - y| \leq |a_{k+1} - b_{k+1}| 10^{-(k+1)} < 10^{-k}$$

- non vale il viceversa: $x = 0.29999$, $y = 0.3 \Rightarrow |x - y| = 10^{-5}$

- Troncamento** (arrotondamento per difetto) di x a k cifre:

$$x_{(k)} = n.a_1a_2 \dots a_k$$

Approssimazione di un numero reale

- Due numeri reali con la stessa parte intera e le prime k cifre decimali uguali differiscono per meno di 10^{-k} :

$$x = n.a_1a_2 \dots a_k a_{k+1} \dots, \quad y = n.a_1a_2 \dots a_k b_{k+1} \dots$$

$$\Rightarrow |x - y| \leq |a_{k+1} - b_{k+1}| 10^{-(k+1)} < 10^{-k}$$

- non vale il viceversa: $x = 0.29999$, $y = 0.3 \Rightarrow |x - y| = 10^{-5}$
- **Troncamento** (arrotondamento per difetto) di x a k cifre:

$$x_{(k)} = n.a_1a_2 \dots a_k$$

- **Arrotondamento per eccesso** di x a k cifre:

$$x^{(k)} = n.a_1a_2 \dots (a_k + 1) = x_{(k)} + 10^{-k}$$

- $x_{(k)} \leq x < x^{(k)}$, $|x - x_{(k)}| < 10^{-k}$, $|x - x^{(k)}| < 10^{-k}$

Approssimazione di un numero reale

- Due numeri reali con la stessa parte intera e le prime k cifre decimali uguali differiscono per meno di 10^{-k} :

$$x = n.a_1a_2 \dots a_k a_{k+1} \dots, \quad y = n.a_1a_2 \dots a_k b_{k+1} \dots$$

$$\Rightarrow |x - y| \leq |a_{k+1} - b_{k+1}| 10^{-(k+1)} < 10^{-k}$$

- non vale il viceversa: $x = 0.29999$, $y = 0.3 \Rightarrow |x - y| = 10^{-5}$
- **Troncamento** (arrotondamento per difetto) di x a k cifre:

$$x_{(k)} = n.a_1a_2 \dots a_k$$

- **Arrotondamento per eccesso** di x a k cifre:

$$x^{(k)} = n.a_1a_2 \dots (a_k + 1) = x_{(k)} + 10^{-k}$$

- $x_{(k)} \leq x < x^{(k)}$, $|x - x_{(k)}| < 10^{-k}$, $|x - x^{(k)}| < 10^{-k}$

- Definiamo **arrotondamento** alla k -ma cifra il numero

$$fl_k(x) = x_{(k)} \text{ se } a_{k+1} < 5; \quad fl_k(x) = x^{(k)} \text{ se } a_{k+1} \geq 5$$

Rappresentazione in virgola mobile

Per confrontare (o sommare) tra loro numeri decimali è spesso comodo rappresentarli in una forma normalizzata che ne mette in risalto l'ordine di grandezza:

$$x = \pm m \cdot 10^z, \quad 0.1 \leq m < 1, \quad z \in \mathbb{Z}$$

con $m =$ mantissa, $z =$ esponente. La corrispondenza $x \rightarrow (m, z)$ è unica.

- $9 \rightarrow (0.9, 1)$
- $-123.81471 \rightarrow (-0.12381471, 3)$
- $-0.0000715 \rightarrow (0.715, -4)$

Per stimare un'approssimazione si usano **errore assoluto** $e_A = |x - \bar{x}|$ ed **errore relativo** $e_R = \left| \frac{x - \bar{x}}{\bar{x}} \right| = \left| \frac{e_A}{\bar{x}} \right|$. Occhio alla differenza!

- $\bar{x} = 0.000000000234$, $e_A \simeq 10^{-6}$
- $\bar{y} = 2345678912$, $e_A \simeq 10^6$

Rappresentazione in virgola mobile

Per confrontare (o sommare) tra loro numeri decimali è spesso comodo rappresentarli in una forma normalizzata che ne mette in risalto l'ordine di grandezza:

$$x = \pm m \cdot 10^z, \quad 0.1 \leq m < 1, \quad z \in \mathbb{Z}$$

con $m =$ mantissa, $z =$ esponente. La corrispondenza $x \rightarrow (m, z)$ è unica.

- $9 \rightarrow (0.9, 1)$
- $-123.81471 \rightarrow (-0.12381471, 3)$
- $-0.0000715 \rightarrow (0.715, -4)$

Per stimare un'approssimazione si usano **errore assoluto** $e_A = |x - \bar{x}|$ ed **errore relativo** $e_R = \left| \frac{x - \bar{x}}{\bar{x}} \right| = \left| \frac{e_A}{\bar{x}} \right|$. Occhio alla differenza!

- $\bar{x} = 0.000000000234$, $e_A \simeq 10^{-6}$
- $\bar{y} = 2345678912$, $e_A \simeq 10^6$
- $e_R(\bar{x}) \simeq 0.5 \cdot 10^4$ (ordine di grandezza?), $e_R(\bar{y}) \simeq 0.5 \cdot 10^{-3}$ (3 cifre esatte)

La memoria del computer

- I computer elaborano le informazioni in **formato digitale**. Istruzioni e dati sono memorizzati in formato binario.
- L'unità di informazione binaria è il **bit** (**b**inary **d**igit) che può assumere solo i valori **0** o **1**.
- I mattoni fondamentali (le celle di memoria) sono i **byte**, sequenze di 8 bit scritte o lette per intero con un'unica operazione.
- Per ogni carattere in genere si usa un byte (codice ASCII, *American Standard Code for Information Interchange*) $\Rightarrow 2^8 = 256$ caratteri
- Per ogni numero si usano **2**, **4** o **8** byte (dipende dal **tipo**)

La memoria del computer

- Ogni cella di memoria è caratterizzata da
 - ▶ un **indirizzo** (che indica la posizione fisica della cella)
 - ▶ un **contenuto** (che indica l'informazione presente nella cella)
- La memoria principale è costituita in genere da:
 - ▶ **RAM** (*Random Access Memory*): volatile
 - ▶ **ROM** (*Read Only Memory*): permanente
- La capacità di memoria si misura in byte (dipende dal computer):
 - ▶ **Kilobyte** = 2^{10} byte = 1024 byte
 - ▶ **Megabyte** = 2^{20} byte = 1024 Kilobyte
 - ▶ **Gigabyte** = 2^{30} byte = 1024 Megabyte
 - ▶ **Terabyte** = 2^{40} byte = 1024 Gigabyte

Numeri macchina

Con quali numeri lavoriamo davvero in un computer?

I numeri sono memorizzati in **formato binario**, in **virgola mobile**, con un **numero fissato di cifre rappresentative per la mantissa e l'esponente**, che dipendono dalla macchina utilizzata.

Quindi non tutti i numeri reali sono rappresentabili. Se chiamiamo \mathcal{A} l'insieme di tali numeri, vediamo di capire con un esempio chi c'è e chi no.

ESEMPIO

Supponiamo di avere a disposizione TRE cifre per la mantissa e UNA per l'esponente, e per comodità ragioniamo in base 10 con solo numeri positivi. Allora se $x \in \mathcal{A}$:

$$x = 0.a_1a_2a_3 \cdot 10^{\pm e}, \quad a_1 \neq 0$$

- Appartengono ad \mathcal{A} : 0.000715, 12.7, 10^{-8} , 9miliardi
- Non appartengono ad \mathcal{A} : $\sqrt{2}$, $1/3$, 39210, 217.5, 10^9 , 10^{-11}

Numeri macchina

Se abbiamo a disposizione k cifre per la mantissa e j per l'esponente, **non appartengono** ad \mathcal{A} :

- tutti i numeri irrazionali (decimali illimitati non periodici)
- tutti i numeri decimali periodici (periodo diverso da zero)
- tutti i decimali finiti con più di k cifre significative per la mantissa
- i numeri in modulo maggiori di M (massimo numero macchina)
[overflow]
- i numeri in modulo minori di m (minimo numero macchina positivo)
[underflow]

Nell'esempio precedente: $M = 0.999 \cdot 10^9$, $m = 10^{-10} = 0.1 \cdot 10^{-9}$

In altre parole il nostro insieme \mathcal{A} è un colabrodo, e tutti i numeri che non sono in \mathcal{A} vanno approssimati con il più vicino numero macchina.

La precisione macchina

Definizione. Definiamo **precisione macchina** ε_M la distanza tra 1 e il numero di \mathcal{A} più vicino ad esso, cioè il più piccolo numero x tale che $1 + x > 1$.

Vale $\varepsilon_M = b^{1-k}$, se b indica la base e k sono le cifre a disposizione per la mantissa.

Nell'esempio precedente: $\varepsilon_M = 10^{1-3} = 10^{-2} = 0.01$. Infatti:

$$1 + 0.01 = 1.01$$

mentre

$$1 + 0.009 = 1$$

L'algebra dei numeri macchina

Le conseguenze di quanto visto sono molte e spesso **drammatiche!**

- **L'insieme \mathcal{A} non è chiuso** rispetto alle operazioni elementari:
 $x, y \in \mathcal{A}$ non implica $x + y, x - y, xy, x/y \in \mathcal{A}$

L'algebra dei numeri macchina

Le conseguenze di quanto visto sono molte e spesso **drammatiche!**

- **L'insieme \mathcal{A} non è chiuso** rispetto alle operazioni elementari:
 $x, y \in \mathcal{A}$ non implica $x + y, x - y, xy, x/y \in \mathcal{A}$
- Anche quando posso eseguirle, **le operazioni non rispettano** necessariamente **le proprietà** di cui godono tra i reali (proprietà associativa, commutativa, ecc.)

L'algebra dei numeri macchina

Le conseguenze di quanto visto sono molte e spesso **drammatiche!**

- **L'insieme \mathcal{A} non è chiuso** rispetto alle operazioni elementari:
 $x, y \in \mathcal{A}$ non implica $x + y, x - y, xy, x/y \in \mathcal{A}$
- Anche quando posso eseguirle, **le operazioni non rispettano** necessariamente **le proprietà** di cui godono tra i reali (proprietà associativa, commutativa, ecc.)
- **L'elemento neutro della somma non è unico**: $x + y = x$ non implica $y = 0$.

L'algebra dei numeri macchina

Le conseguenze di quanto visto sono molte e spesso **drammatiche!**

- **L'insieme \mathcal{A} non è chiuso** rispetto alle operazioni elementari: $x, y \in \mathcal{A}$ non implica $x + y, x - y, xy, x/y \in \mathcal{A}$
- Anche quando posso eseguirle, **le operazioni non rispettano** necessariamente **le proprietà** di cui godono tra i reali (proprietà associativa, commutativa, ecc.)
- **L'elemento neutro della somma non è unico**: $x + y = x$ non implica $y = 0$.
- Avvengono **fenomeni di propagazione dell'errore**: il risultato delle operazioni tra numeri approssimati può essere un'approssimazione molto peggiore del risultato esatto: ad es. posso avere $e_R(x), e_R(y)$ piccoli ma $e_R(x - y)$ molto grande se x e y sono due numeri positivi molto vicini tra loro.