Relazione laboratorio sull’errore

Calcolo numerico: Aritmetica di macchina e stabilità numerica

---

Pezzano Enrico (S4825087)

ESERCIZIO 1
Per il primo esercizio prendo in considerazione il mio numero di matricola e utilizzo due variabili d_0 e d_1, che rappresentano gli ultimi due decimali (rispettivamente 8 e 7). Successivamente, eseguo i calcoli di macchina a doppia precisione utilizzando variabili di tipo double:

- (a+b)+c;
- a+(b+c);

Output programma:

labo1 — -zsh — 80×60

```
Iterazione n° 0
a = 9
b = 8e+20
c = -8e+20

(a+b)+c = 0
a+(b+c) = 9

erorre assoluto (a + b) + c -> 9
erorre relativo (a + b) + c -> 1

erorre assoluto a + (b + c) -> 9
erorre relativo a + (b + c) -> 1
------------------------------------------
Iterazione n° 1
a = 90
b = 8e+20
c = -8e+20

(a+b)+c = 0
a+(b+c) = 90

erorre assoluto (a + b) + c -> 90
erorre relativo (a + b) + c -> 1

erorre assoluto a + (b + c) -> 90
erorre relativo a + (b + c) -> 1
------------------------------------------
Iterazione n° 2
a = 900
b = 8e+20
c = -8e+20

(a+b)+c = 0
a+(b+c) = 900

erorre assoluto (a + b) + c -> 900
erorre relativo (a + b) + c -> 1

erorre assoluto a + (b + c) -> 900
erorre relativo a + (b + c) -> 1
------------------------------------------
Iterazione n° 3
a = 9000
b = 8e+20
c = -8e+20

(a+b)+c = 0
a+(b+c) = 9000

erorre assoluto (a + b) + c -> 9000
erorre relativo (a + b) + c -> 1

erorre assoluto a + (b + c) -> 9000
erorre relativo a + (b + c) -> 1
------------------------------------------
```

labo1 — -zsh — 80×43

```
------------------------------------------
Iterazione n° 4
a = 90000
b = 8e+20
c = -8e+20

(a+b)+c = 131072
a+(b+c) = 90000

erorre assoluto (a + b) + c -> 41072
erorre relativo (a + b) + c -> 0.456356

erorre assoluto a + (b + c) -> 41072
erorre relativo a + (b + c) -> 0.456356
------------------------------------------
Iterazione n° 5
a = 900000
b = 8e+20
c = -8e+20

(a+b)+c = 917504
a+(b+c) = 900000

erorre assoluto (a + b) + c -> 17504
erorre relativo (a + b) + c -> 0.0194489

erorre assoluto a + (b + c) -> 17504
erorre relativo a + (b + c) -> 0.0194489
------------------------------------------
Iterazione n° 6
a = 9e+06
b = 8e+20
c = -8e+20

(a+b)+c = 9.04397e+06
a+(b+c) = 9e+06

erorre assoluto (a + b) + c -> 43968
erorre relativo (a + b) + c -> 0.00488533

erorre assoluto a + (b + c) -> 43968
erorre relativo a + (b + c) -> 0.00488533
enrico@MacBook-Air-di-Enrico labo1 %
```

OSSERVAZIONE OUTPUT:

- Durante le iterazioni <4, notiamo che il risultato di (a+b)+c non cambia per via delle cancellazioni. Al contrario, il risultato ottenuto dal calcolo a+(b+c) non rimane invariato.

- Durante le iterazioni >=4, dati valori alti, per via delle cancellazioni, notiamo che i risultati di (a+b)+c e a+(b+c) aumentano fino ad "esplodere".

In **conclusione**, il risultato corretto tra i due calcoli è quello di a+(b+c).

ESERCIZIO 2
Per svolgere il secondo esercizio invece, è richiesta l'implementazione di un programma che calcoli $f_N(x)$ per il punto *x* e il grado *N* presi in input, considerando la funzione "*my_taylor*" e la funzione *exp* della libreria ANSI math.h e infine confrontando i risultati ottenuti per $f_N(x)$ tramite errore relativo e assoluto.

Il **polinomio di Taylor**, di cui abbiamo bisogno, ha il compito di approssimare una funzione difficile da trattare. In altre parole, questa formula stima il comportamento di una funzione *f(x)* con un opportuno polinomio di grado *N*, i cui coefficienti dipenderanno da f.
Inoltre, l'approssimazione della funzione dipenderà dal grado N del polinomio: *più sarà alto il grado N del polinomio, tanto più precisa sarà l'approssimazione.*
Definita la formula, possiamo svolgere l'esercizio che richiede di fissare un intero positivo *N*, calcolare $f_N(x)$ per il punto *x* e il grado *N* dati in input, implementando due algoritmi differenti (*Alg1() e Alg2()*, come sviluppato nel codice).

Output programma:

Algoritmo1:

Approssimazione di *f(x)=e^x* con *x*=0.5: **1.64872**,
approssimazione di *f(x)=e^x* con *x*=30: **1.06865e+013**,
approssimazione di *f(x)*= *e^x* con x=**-0.5: 0.606531**,
approssimazione di f(x)=e^x con x=-30: **9-35762e-14**,
e valuto il polinomio di Taylor per N=3,10,50,100,150.
Successivamente, ripeto l'esercizio con due valori diversi x= -0.5 e x= -30.
Stampo ogni errore assoluto e relativo.

```
labo1 — -zsh — 80×60
enrico@MacBook-Air-di-Enrico labo1 % g++ es2\ ok.cpp
enrico@MacBook-Air-di-Enrico labo1 % ./a.out
Lanciamo il primo algoritmo:

Per il punto x=0.5...

...errore assoluto con N=3      ->      -0.00288794
...errore assoluto con N=10     ->      -1.27627e-11
...errore assoluto con N=50     ->      -4.44089e-16
...errore assoluto con N=100    ->      -4.44089e-16
...errore assoluto con N=150    ->      -4.44089e-16
...errore relativo con N=3      ->      -0.00175162
...errore relativo con N=10     ->      -7.74096e-12
...errore relativo con N=50     ->      -2.69354e-16
...errore relativo con N=100    ->      -2.69354e-16
...errore relativo con N=150    ->      -2.69354e-16


Per il punto x=30...

...errore assoluto con N=3      ->      -1.06865e+13
...errore assoluto con N=10     ->      -1.06862e+13
...errore assoluto con N=50     ->      -3.18471e+09
...errore assoluto con N=100    ->      0.00390625
...errore assoluto con N=150    ->      0.00390625
...errore relativo con N=3      ->      -1
...errore relativo con N=10     ->      -0.999978
...errore relativo con N=50     ->      -0.000298013
...errore relativo con N=100    ->      3.65532e-16
...errore relativo con N=150    ->      3.65532e-16


Per il punto x=-0.5...

...errore assoluto con N=3      ->      -0.00236399
...errore assoluto con N=10     ->      1.17416e-11
...errore assoluto con N=50     ->      -1.11022e-16
...errore assoluto con N=100    ->      -1.11022e-16
...errore assoluto con N=150    ->      -1.11022e-16
...errore relativo con N=3      ->      -0.00389757
...errore relativo con N=10     ->      1.93586e-11
...errore relativo con N=50     ->      -1.83045e-16
...errore relativo con N=100    ->      -1.83045e-16
...errore relativo con N=150    ->      -1.83045e-16


Per il punto x=-30...

...errore assoluto con N=3      ->      -4079
...errore assoluto con N=10     ->      1.21255e+08
...errore assoluto con N=50     ->      8.78229e+08
...errore assoluto con N=100    ->      -4.82085e-06
...errore assoluto con N=150    ->      -4.82086e-06
...errore relativo con N=3      ->      -4.35901e+16
...errore relativo con N=10     ->      1.29579e+21
...errore relativo con N=50     ->      9.38517e+21
...errore relativo con N=100    ->      -5.15179e+07
...errore relativo con N=150    ->      -5.1518e+07
```

Possiamo notare che per i valori >*0*, ovvero *0.5 e 30*, abbiamo un risultato preciso grazie all'utilizzo del *polinomio di Taylor* e che quando *N* vale 100 e 150 si ottiene un valore vicino a quello atteso.

Algoritmo 2:

Calcolo il reciproco dei valori ***-0.5*** e ***-30*** e stampo ogni errore assoluto e relativo.

```
labo1 — -zsh — 80×29
-----------------------------------------------------

Lanciamo il secondo algoritmo:

...errore assoluto con N=3      ->      0.00106428
...errore assoluto con N=10     ->      4.69513e-12
...errore assoluto con N=50     ->      1.11022e-16
...errore assoluto con N=100    ->      1.11022e-16
...errore assoluto con N=150    ->      1.11022e-16

...errore relativo con N=3      ->      0.0017547
...errore relativo con N=10     ->      7.74097e-12
...errore relativo con N=50     ->      1.83045e-16
...errore relativo con N=100    ->      1.83045e-16
...errore relativo con N=150    ->      1.83045e-16

...errore assoluto con N=3      ->      0.000200763
...errore assoluto con N=10     ->      4.18699e-09
...errore assoluto con N=50     ->      2.78952e-17
...errore assoluto con N=100    ->      -3.78653e-29
...errore assoluto con N=150    ->      -3.78653e-29

...errore relativo con N=3      ->      2.14545e+09
...errore relativo con N=10     ->      44744.2
...errore relativo con N=50     ->      0.000298102
...errore relativo con N=100    ->      -4.04647e-16
...errore relativo con N=150    ->      -4.04647e-16

enrico@MacBook-Air-di-Enrico labo1 %
```

In questo caso, si riesce ad osservare che i valori *<0*, ossia *-0.5* e *-30* si distinguono in due modi:

1. nel primo, si verifica un problema di cancellazione dovuto al fatto che nella formula del *polinomio di Taylor*, la somma dei valori negativi, attraverso il modulo, è uguale a quella del valori positivi; inoltre, possiamo notare che l'errore relativo è molto diverso da quello atteso.
2. nel secondo invece, si verifica una buona approssimazione, siccome si utilizza il *polinomio di Taylor* coi valori positivi e si fa il reciproco.

Esercizio 3

L'ultimo esercizio richiede di determinare la precisione di macchina e calcolarne il valore in singola e doppia precisione.

I calcoli in doppia precisione richiedono ovviamente più spazio in memoria (64 bit), ma hanno una maggiore precisione, mentre quelli a precisione singola occupano meno spazio (32 bit), ma sono meno precisi.

Output programma:

```
labo1 — -zsh — 80×5
enrico@MacBook-Air-di-Enrico labo1 % g++ es3\ ok.cpp
enrico@MacBook-Air-di-Enrico labo1 % ./a.out
 La singola precisione di macchina e': 1.19209e-07
 La doppia precisione di macchina e': 2.22045e-16
enrico@MacBook-Air-di-Enrico labo1 %
```