Esame di Laboratorio di Fisica Computazionale

7 ottobre 2015, ore 14.00

shell scripting

1. Si corregga con uno script il testo nel file `brano.tex`, sostituendo alla parola "aumento" la parola "calo".

Mathematica

1. Si risolva il seguente sistema di equazioni differenziali:

$$\begin{cases} x_1'(t) &= 3x_1(t) + t \\ x_2'(t) &= x_2(t) + 1 \\ x_3'(t) &= x_2(t) + x_3(t) \\ x_1(0) &= 0, \quad x_2(0) = 1, \quad x_3(0) = 0 \end{cases}$$

2. Si disegni nel piano (x, y) la curva parametrica $r(\theta) = 2\theta$ con $\theta \in [0, 4\pi]$.

3. Si calcoli il seguente integrale parametrico:

$$\int_{0.1}^{1} dx \, \sin(\log(ax)) \tag{1}$$

   e si disegni il risultato per $a \in [0.1, 2]$.

4. Si studi la mappa logistica $x_{n+1} = r x_n(1 - x_n)$, eseguendo i seguenti passaggi:

   - si ponga $x_0 = 0.5$;
   - si scriva una funzione $f(r)$ che calcola 1000 valori della successione, prende gli ultimi 100 e infine seleziona gli elementi distinti (si utilizzino i comandi `Take` e `Union` e si utilizzi il trucco di salvare in memoria l'elemento $n-$esimo appena calcolato); questa funzione restituisce come risultato una lista di valori;
   - si utilizzi la funzione $f(r)$ del punto precedente per calcolare i valori della mappa logistica con $r \in [2.85, 3.55]$ in step di 0.1;
   - si definisca una funziona $g(r)$ che associa a ciascun valore X della lista calcolata dalla funzione $f(r)$ il corrispondente r, formando la coppia (r, X); questa funzione restituisce come risultato $((r, X_1(r)), (r, X_2(r)), ...)$; (questa nuova funzione può utilizzare quella del punto precedente)
   - si utilizzi la funzione $g(r)$ per generare una lista di coppie con $r \in [2.85, 3.55]$, questa volta in step di 0.01 (si utilizzi il comando `Flatten`) e infine si disegni l'insieme delle coppie (`ListPlot`)

C++

Si risolva l'esercizio proposto. Per facilitare la correzione, se possibile includere tutto in un unico file sorgente.

Esercizio 1

Si vuole scrivere una classe `Complex` che descriva numeri complessi.

1. Si pongano come membri privati due variabili `double`, corrispondenti alle parti reale x e immaginaria y. Tra i membri pubblici, si scriva un opportuno costruttore che richieda come parametri x e y, con valori di default rispettivamente 1 e 0.

2. Si implementino due access function per leggere le due variabili private.

3. Si scriva una funzione membro `print` che stampi il numero complesso su `std::cout` nella forma `x+yi`. (Si faccia in modo che i numeri reali compaiano nella forma usuale, e che i numeri puramente immaginari compaiano come `yi`.)

4. Si scriva l'operatore + (fuori dalla classe, cioè non membro di `Complex`) che restituisca la somma tra due oggetti `Complex`.

5. Si scriva l'operatore * (anch'esso esterno alla classe) che restituisca il prodotto tra un numero reale e un numero complesso (in questo ordine). Si scriva poi un overloading di questo operatore che accetti il numero reale e quello complesso nell'ordine inverso; per fare questo si richiami (con una chiamata a funzione) l'operatore * già scritto.

6. Si verifichi il comportamento del codice con un `main` che esegua le seguenti istruzioni:

   - istanziare un numero complesso z, inizializzandolo con i valori di default (vedi punto 1).
   - istanziare un numero complesso w, inizializzandolo a $1 + 2i$.
   - istanziare un numero complesso v, inizializzandolo a $2 + 2i$.
   - stampare su `std::cout` il valore di z+w e di $z + w + v * (-1)$, e verificare che i risultati siano corretti.

7. Si crei un `std::vector` di puntatori a `Complex`, e lo si riempia prima con i valori 0 e $1+i$, e poi con altri 18 numeri complessi allocati dinamicamente, le cui componenti cartesiane siano generate a caso tra 0 e 1 (si puó usare `drand48()` per generare numeri reali random tra 0 e 1). Ricordarsi di liberare la memoria allocata, alla fine del programma.

8. Si vuole ora ordinare il vettore per moduli crescenti, usando l'algoritmo

   ```
   std::sort(it1, it2, pred)
   ```

che ordina gli elementi compresi tra gli iteratori `it1` (compreso) e `it2` (escluso), confrontandoli attraverso il predicato `pred` (si ricorda che un predicato è una funzione che restituisce un `bool`). Si scriva il predicato opportuno, si ordini il vettore, e si controlli infine (stampandoli su `cout`) che il primo e l'ultimo elemento siano rispettivamente 0 e $1 + i$.