Laboratorio di Calcolo per Fisici, Seconda esercitazione

Canale Pb-Z, Docente: Lilia Boeri

Lo scopo della seconda esercitazione di laboratorio è di scrivere da zero dei semplici programmi in c, usando le funzioni della libreria matematica `math.h` e di familiarizzarsi con i diversi comandi di *formattazione* dell'output per realizzare diversi tipi di grafico.

► **Prima parte:**

1. Scrivere da zero il programma C `matematica.c` che, utilizzando la libreria matematica esterna `math.h`, calcoli correttamente e stampi su schermo i valori delle seguenti espressioni:

$$8+7 \qquad 3.2+8.4 \qquad 2^2+4^2 \qquad \sqrt{36}$$
$$\cos\left(\frac{\pi}{4}\right) \qquad \sin\left(\frac{\pi}{6}\right) \qquad \tanh(1) \qquad \sinh(0)$$

2. **Attenzione!** Per *linkare* la libreria `math.h` bisogna inserire un opportuno comando nell'*header* del programma, e chiamare il compilatore `gcc` con l'opzione -lm, cioè:

```
gcc matematica.c -lm -o matematica.x
```

3. Una volta verificato che il programma calcola esattamente le quantità richieste, utilizzare i comandi di formattazione del C per stampare i risultati su due righe con i seguenti formati: *Per i numeri interi*: massimo tre cifre; *Per i numeri razionali*: quattro cifre dopo la virgola.
Suggerimento: Nella stringa di formattazione oltre ai caratteri relativi al formato delle variabili si possono inserire lettere, numeri e caratteri speciali.

4. Aggiungere al programma un messaggio iniziale, che viene stampato prima della lista di numeri, del tipo `Benvenuto! Questo è un programma dimostrativo delle funzioni della libreria math.h`.

5. Aggiungere una sezione di programma che chieda all'utente di inserire il proprio nome come *stringa* e lo stampi all'interno del messaggio di benvenuto; il risultato finale deve essere qualcosa del tipo:
`Benvenuto Marco! Questo è un programma dimostrativo delle funzioni della libreria math.h`.

► **Seconda parte:** Un moto circolare uniforme nel piano (x, y) è descritto dalle equazioni:

$$\begin{cases} x(t) = R\cos(\omega t) \\ y(t) = R\sin(\omega t)\,, \end{cases}$$

dove R è il raggio della traiettoria e ω la velocità angolare. L'obiettivo della seconda parte dell'esercitazione è quello di scrivere un programma che utilizzi la libreria `math.h` per risolvere le equazioni della traiettoria per $R = 6.2$ m, $\omega = 0.1\ rad/s$ e stampi i risultati in maniera opportuna.

1. Scrivere un programma che calcoli i valori di $x(t)$ e $y(t)$ per i valori: $t = 0$, $t = 0.5$, $t = 10$, $t = 20$ s. Verificare che i risultati ottenuti con il programma riproducano quelli che si otterrebbero risolvendo *a mano* le equazioni del moto.

2. Inserire un blocco di programma che chieda all'utente di inserire il valore del tempo t da tastiera, e usi il valore inserito per calcolare e stampare i valori di x e y nel formato:
`(x,y)= ...,...`

3. Una volta verificato che il programma produce risultati corretti, riformattare l'output in modo da stampare i tre valori t, x, y su tre colonne, separate da spazi vuoti, come nell'esempio sottostante (stampare almeno 4 cifre decimali):
`2.5000 4.3841 4.3841`

4. *Suggerimento:* Per modificare piccole parti di programma senza cancellare quello che si è fatto in precedenza può essere utile inserire un *commento*. In *C* un commento è racchiuso dai delimitatori `/*` e `*/`, come nell'esempio sottostante:

```
/* Questo e' un commento */
```

► Terza parte (obbligatoria)

In questa terza parte dell'esercitazione useremo il programma `circle.c` per studiare l'andamento del moto circolare uniforme, utilizzando `gnuplot`.

1. Eseguire il programma `circle.c` un numero di volte sufficiente ad avere una distribuzione di punti su tutta la traiettoria circolare; salvare i risultati su un file `traiettoria.dat` che contenga tre colonne t, x, y, come nel punto 4. del precedente esercizio. *Ricordatevi che se volete inserire un'intestazione in un file di dati letto tramite* `loadtxt` *in* `python`*, dovete aggiungere il parametro* `comments='#'`*, assumendo che i commenti inizino con il simbolo* `#`*.*

2. Utilizzare `python` per graficare la traiettoria, plottando y in funzione di x. *Per plottare solo due colonne di un file che ne contiene molte, la funzione* `loadtext` *va chiamata con il parametro* `usecols(i,j)`*, dove* `i` *e* `j` *sono le colonne da plottare (*0 *è la prima colonna). Nel nostro caso, se volessimo plottare solo la seconda e la terza colonna il comando completo da usare nello script* `python` *diventa:*

```
np.loadtext('traiettoria.dat', comments=['#'], usecols(1:2),
    unpack=True)
```

3. Creare altri due grafici che mostrino l'andamento della coordinata x e della coordinata y in funzione del tempo, rispettivamente.

4. Salvare i tre grafici in tre file separati, chiamati: `traiettoria.png; x.png; y.png`. *Per salvare i grafici con gnuplot, bisogna usare il seguento comando python*

```
    plt.savefig('traiettoria.png')
```

5. Se i grafici sono corretti, dovreste saper rispondere alle seguenti domande:
   (a) quanto vale il raggio della traiettoria?
   (b) Qual è l'equazione della traiettoria? (*Per equazione della traiettoria si intende un'espressione del tipo $f(x, y) = c$ soddisfatta da tutti i punti della traiettoria*).
   (c) Qual è il *periodo* della traiettoria?
   (d) In quali punti sono massimi/minimi i valori di $x(t)$ e $y(t)$? (*Scrivere le risposte su un file di testo* `risposte.txt`).

▶ Quarta parte (facoltativa)

L'andamento delle *componenti* della velocità in un moto circolare uniforme è dato da:

$$\begin{cases} v_x(t) = -\omega R \sin(\omega t) \\ v_y(t) = \omega R \cos(\omega t)\,, \end{cases}$$

1. Modificare il programma `circle.c` per calcolare anche la velocità del punto lungo la traiettoria.
2. Graficare l'andamento della velocità in funzione del tempo lungo la traiettoria: quanto vale il *modulo della velocità*? In che direzione punta il *vettore velocità* per $t = 0$? E per $t = 10$? In quali punti sono massimi/minimi i valori di $v_x(t)$ e $v_y(t)$? Qual è il significato fisico di quello che osservate? (*Scrivere le risposte sul file di testo* `risposte.txt`) creato in precedenza.
3. Disegnare il vettore velocità lungo la traiettoria, come una freccia che punta nella direzione corretta sfruttando il seguente codice *python*:

```
x, y, vx, vy = np.loadtxt('temp.dat', usecols=(0,1,2,3), unpack=True)
for cc in range(0,len(x)):
  xi=x[cc]
  yi=y[cc]
  vxi=vx[cc]
  vyi=vy[cc]
  plt.arrow(xi,yi,vxi,vyi,width=0.2,head_width=0.5,head_length=0.3,
    fc='r', ec='r')
```

che disegna `len(x)` frecce di lunghezza `(vx,vy)` nei punti `(x,y)`.

Funzioni piú comuni della libreria `math.h`:

acos arcocoseno
asin arcoseno
atan arcotangente
atan2 arcotangente di due parametri
ceil il più piccolo intero non minore del parametro
cos coseno
cosh coseno iperbolico
exp(double x) funzione esponenziale, calcola e^x
fabs valore assoluto
floor il più grande intero non maggiore del parametro
fmod resto del numero in virgola mobile
frexp frazione e potenza di due.
ldexp operazione in virgola mobile
log logaritmo naturale
log10 logaritmo in base 10
pow(x,y) eleva un valore dato ad esponente, xy
sin seno
sinh seno iperbolico
sqrt radice quadrata
tan tangente
tanh tangente iperbolica