

Corso di Laboratorio di Calcolo - Prof. G. Organtini

Laboratorio 2008.1 *Uso del compilatore su Linux*



Scrivete il testo del Listato riportato sotto in un file cui darete il nome `programma.c`. Per questo usate un editor come `emacs` o `nedit`. Al momento non importa che ne capiate il contenuto, limitatevi a copiare carattere per carattere (attenzione a non dimenticarne, le espressioni potrebbero non essere più valide!). Invocando il comando

```
cpp programma.c
```

vedrete scorrere sullo schermo una lunga serie di istruzioni in C tra cui saranno presenti anche quelle contenute nel file `programma.c`. Si tratta dell'output del preprocessore che esegue le direttive (istruzioni che iniziano con `#`) e dirige il risultato sullo *standard output* (`stdout`), la periferica di output predefinita, che di norma è lo schermo. Osservate, in particolare, cosa accade al simbolo `G` e ai caratteri compresi tra `/*` e `*/`. Il compilatore `gcc` assume sempre che il file in input debba essere passato prima dal preprocessore (un file già preprocessato uscirà invariato dal preprocessore) e dunque un comando del tipo

```
gcc programma.c
```

provoca l'automatica chiamata del preprocessore che invia il suo output in ingresso al compilatore. Questi, a sua volta, invoca direttamente il *linker* per aggiungere al codice oggetto le librerie standard caratteristiche del sistema per produrre l'eseguibile. L'eseguibile è un file il cui nome, in assenza di specifiche opzioni, è `a.out` (facendo la lista dei file presenti nella directory di lavoro dovreste trovare un file con questo nome). Tale nome non suggerisce in alcun modo né il nome del file sorgente a partire dal quale è stato generato, né lo scopo del programma stesso. È sempre consigliabile attribuire al programma eseguibile un nome che ne ricordi la funzione. L'opzione¹ `-o` provoca la modifica del nome del file eseguibile e fa in modo di assegnare a questo il nome che segue l'opzione stessa :

```
gcc -o programma.exe programma.c
```

Ora sul disco è presente un file di nome `programma.exe`. Nel caso in cui si desideri evitare di eseguire il collegamento basta invocare il compilatore con un'apposita opzione. L'opzione che evita l'invocazione automatica del *linker* è `-c` :

```
gcc -c programma.c
```

In questo caso il file oggetto prodotto ha un nome costituito dalla parte a sinistra del punto seguito dall'estensione `o` : `programma.o`.

Il *linker* può essere invocato a parte o, più comunemente, chiamato attraverso lo stesso comando `gcc`. Se si invoca il comando di compilazione su un file oggetto, infatti, questo è passato al *linker* per l'ultima fase di compilazione :

```
gcc -o programma.exe programma.o
```

In alcuni casi è necessario indicare al compilatore le librerie da usare nella fase di collegamento. Se, ad esempio, copiate il programma riportato nel Listato 2008.1, il comando di compilazione non produrrà alcun eseguibile, ma mostrerà alcuni messaggi per informarvi che nel programma sono presenti simboli indefiniti (si tratta dei nomi delle funzioni trigonometriche, definite nella libreria matematica). Allo scopo di collegare una libreria

¹Le opzioni dei comandi sotto UNIX sono specificate nella stessa linea di comando attraverso caratteri preceduti da uno o due segni meno `-`.

all'oggetto si usa l'opzione `-l`, seguita dal nome di questa. Il nome della libreria matematica è semplicemente `m`, per cui il comando di compilazione apparirà come segue :

```
gcc -o programma.exe programma.o -lm
```

È spesso utile usare le opzioni `-Wall` `-pedantic` che visualizzano dei messaggi di attenzione (*warning*) in tutti i casi in cui le espressioni usate dal programmatore possono potenzialmente generare errori in fase di esecuzione (`-Wall`) o quando il formato scelto per le istruzioni non sia compatibile con lo standard ANSI (`-pedantic`). Per maggiori informazioni si rimanda al manuale in linea richiamabile con il comando `man gcc`.

```
1 #include <stdio.h>
2 #include <math.h>
3 #include <stdlib.h>
4
5 #define G 9.82
6 /* questo programma calcola le coordinate x,y di un peso
7 lanciato ad angolo theta con l'asse x, e v iniziale
8 v0. Tutte le grandezze vanno considerate nel sistema MKS */
9 main() {
10     double t = 0;           /* tempo */
11     double theta;           /* angolo di lancio */
12     double v0, v0x, v0y;    /* v iniziale */
13     double xt, yt, vx, vy;
14
15     printf(" Inserisci l'angolo di lancio in radianti: ");
16     scanf("%lf", &theta);
17     printf("\n Inserisci la velocita' iniziale in m/sec: ");
18     scanf("%lf", &v0);
19     printf("\n Inserisci il valore del tempo t: ");
20     scanf("%lf", &t);
21     /* Calcola le componenti di v */
22     v0x = v0 * cos(theta);
23     v0y = v0 * sin(theta);
24     vx = v0x;
25     vy = v0y - G * t;
26     /* ora calcola le coordinate */
27     xt = vx * t;
28     yt = vy * t - 0.5 * G * pow(t, 2.0);
29     printf("\nLe coordinate al tempo %lf sono x = %lf, y = %lf\n",
30           t, xt, yt);
31     exit(0);
32 }
```
