3.9 Variabili di tipo virgola mobile

I numeri che hanno una parte frazionaria sono detti in virgola mobile (*floating point*). Per esempio:

```
152.23
-91.64
0.867
```

non possono essere memorizzati nelle variabili di tipo int. Le variabili che contengono tali valori sono di tipo float:

```
float x, y, z;
```

La parola chiave float specifica che gli identificatori x, y e z che la seguono si riferiscono a variabili in virgola mobile. La definizione fa sì che venga riservato uno spazio in memoria la cui dimensione può variare rispetto all'implementazione, ma che spesso è di 4 byte (32 bit), sufficiente per contenere numeri che vanno da 3.4E-38 a 3.4E+38, cioè valori positivi e negativi che in modulo sono compresi approssimativamente tra 10 elevato alla -38 e 10 alla +38. Le seguenti istruzioni assegnano valori a variabili float:

```
x = 152.23;

y = 0.00008;

z = 7E+20;
```

La seconda istruzione poteva essere scritta anche come y=.00008, dove lo zero prima del punto decimale viene sottinteso. Il valore assegnato a z è in notazione esponenziale e va letto come $7*10^{20}$. La lettera E, che può essere anche minuscola, indica che il numero che la precede deve essere moltiplicato per 10 elevato al numero che la segue. L'uso della notazione esponenziale da parte del programmatore risulta comoda quando il numero da rappresentare è o molto grande o molto piccolo; infatti avremmo anche potuto scrivere

Per visualizzare una variabile float all'interno del formato della printf si deve specificare dopo il solito simbolo % il carattere f se si desidera il numero decimale in virgola mobile, e il carattere e se lo si desidera in forma esponenziale (detta anche notazione scientifica). Come per i numeri interi, si può far seguire il simbolo di percentuale da un numero che specifica la lunghezza del campo in cui dovrà essere posto il valore opportunamente allineato a destra. Per esempio, printf ("%15f", x), riserva quindici caratteri per la stampa di 152.23, che verrà posto negli ultimi sette caratteri della maschera. La lunghezza del campo può essere seguita da un punto e dal numero di caratteri del campo che devono essere riservati alla parte decimale. Per esempio:

```
printf("%15.5f", x);
```

riserva 5 caratteri per la parte decimale dei 15 totali.

1	2	3 4	4 5	6			10 11						
					1	5	2	2	3	0	0	0	

Se la parte decimale non entra completamente nel sottocampo a lei riservato, le cifre meno significative vengono perdute. Al contrario, se la parte intera è più grande, il campo viene esteso fino a contenerla tutta. Se si scrive %.0 o %0.0 la parte decimale non viene visualizzata.

Come per gli int, un carattere -, dopo il simbolo di percentuale e prima della specifica del campo, indica che il valore deve essere allineato a sinistra.

Le istruzioni:

```
printf("%15.5f", x); printf("%f", x); printf("%e", x);
    printf("%15.5f", y); printf("%f", y); printf("%e", y);
```

restituiscono rispettivamente

```
.....152.23000 152.229996 1.522300e+002
......0.00008 0.000080 8.000000e-005
```

Esiste anche una terza possibilità, data dal %g, che stampa la rappresentazione più breve tra %f e %e, eliminando eventualmente gli zeri superflui , per cui

```
printf("%g", x);
printf("%g", y);
```

visualizzano

```
152.23
8e-005
```

In memoria le variabili float vengono comunque rappresentate in una particolare notazione esponenziale, in modo da risparmiare spazio. Naturalmente i numeri reali sono infiniti mentre i sistemi di elaborazione devono fare i conti con le limitazioni fisiche proprie della macchina; dunque dobbiamo fare attenzione: stiamo lavorando con delle approssimazioni che calcoli successivi possono rendere inaccettabili. Per far in modo che la rappresentazione dei reali sia ulteriormente più precisa il C ha un altro tipo di dato in virgola mobile, detto double, che occupa uno spazio generalmente di 8 byte (64 bit) e che quindi permette di lavorare con numeri positivi e negativi nell'intervallo da 1.7E-308 a 1.7E+308.

Il tipo double comunque non garantisce automaticamente di poter rappresentare un numero doppio di cifre significative, ma certamente migliora l'accuratezza delle operazioni aritmetiche e riduce l'effetto degli errori di arrotondamento.

Per poter visualizzare una variabile double con la printf si può utilizzare nel formato la solita notazione %f o la sequenza %lf (*long float*), con le stesse convenzioni viste per le variabili float. L'istruzione

```
scanf("%f", &x);
```

memorizza il valore passato dall'utente nella variabile float x. Al posto di %f si può utilizzare indifferentemente %e. Analogamente per le variabili double si usa %f o %lf...

Ogni definizione di costante che includa un punto decimale fa sì che venga creata una costante di tipo double:

```
#define PI 3.14159
```

definisce la costante PI che può essere utilizzata all'interno del programma al posto del valore 3.14159; naturalmente il valore di PI non può essere modificato ...