7.2 Sottoprogrammi C

In C i sottoprogrammi sono detti *funzioni*: a partire da uno o più valori presi in ingresso, esse *ritornano* (o *restituiscono*) un valore al programma chiamante. Come indicato in Figura 7.1, una funzione può essere pensata come una *scatola nera* che a determinati valori in ingresso fa corrispondere un determinato valore in uscita.

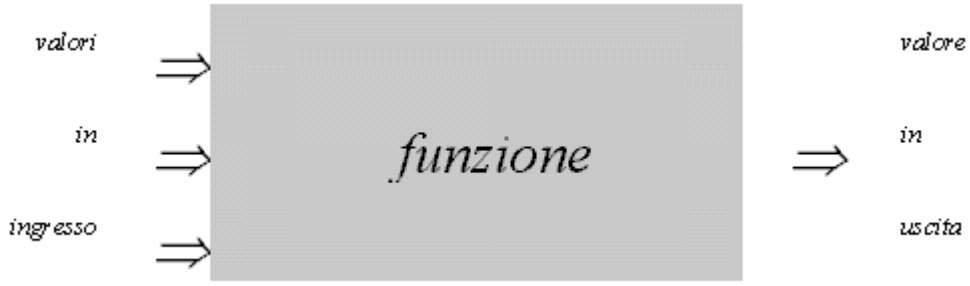

Figura 7.1 La funzione come scatola nera

Un esempio di funzione C è abs(i), già utilizzata più volte. Considerando la funzione abs come una scatola nera, tutto quello che dobbiamo sapere – e in effetti già sappiamo – è che inserendo come argomento i di tale funzione un numero intero essa ne ritorna il valore assoluto (Figura 7.2).

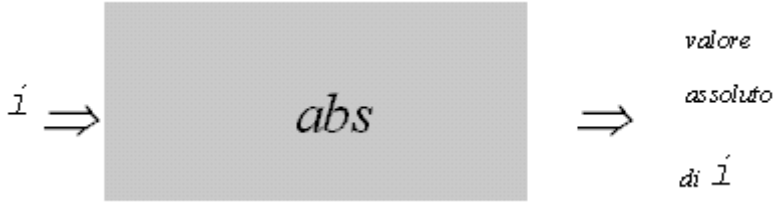

Figura 7.2 La funzione abs()come scatola nera che produce il valore assoluto di i

Questo discorso è valido in generale: come programmatori possiamo utilizzare una qualsiasi funzione di libreria considerandola una scatola nera, cioè senza conoscere niente del suo funzionamento interno e interessandoci solo di cosa passarle in ingresso e di cosa viene restituito in uscita. Se però vogliamo creare noi stessi una nostra specifica funzione dobbiamo anche occuparci di come questa possa svolgere il compito affidatole.

Prima di esaminare la sintassi di dichiarazione e definizione di una funzione si consideri l'esempio riportato nel Listato 7.1.

```
#include <stdio.h>

double cubo(float);
```

```
main()
{
  float  a;
  double b;

  printf("Inserisci un numero: ");
  scanf("%f", &a);

  b = cubo(a);
  printf("%f elevato al cubo è uguale a %f", a, b);
}


double cubo(float c)
{
  return (c*c*c);
}
```

Listato 7.1 Dichiarazione, definizione e invocazione di una funzione

Per poter usare un identificatore occorre innanzitutto dichiararlo. La dichiarazione:

```
double cubo(float);
```

che precede main introduce l'identificatore cubo. Per mezzo di questa dichiarazione si specifica che cubo è il nome di una funzione che restituisce al programma chiamante un valore di tipo double. Inoltre si dichiara che la funzione cubo accetta in ingresso un solo valore come argomento, il cui tipo è float.
Si noti come con questa dichiarazione non si sia ancora definita la funzione cubo, cioè ancora non vengano specificate le istruzioni che caratterizzano la funzione; semplicemente abbiamo dichiarato un nuovo *nome*, cubo, e abbiamo detto a quale categoria appartiene questo nome. La definizione della funzione cubo avviene più tardi, dopo la fine del blocco di istruzioni di main:

```
double cubo(float c)
     {
        return (c*c*c);
     }
```

Oltre al nome della funzione viene definito il numero, il tipo e il nome dei suoi parametri, cioè le variabili su cui essa agisce. Nel nostro esempio è presente un solo parametro, il cui tipo è float e il cui nome è c. Il compito svolto da cubo è molto semplice: il valore passato nel parametro c è moltiplicato per se stesso tre volte (c*c*c) e il risultato di questa espressione è convertito in double e restituito (con return) al programma chiamante.
Il programma chiamante non ha da fare altro che passare alla funzione cubo un valore. Nell'esempio lo fa passando a cubo il valore contenuto nella variabile a: cubo(a). Successivamente il valore calcolato da cubo viene assegnato a una variabile di tipo double. Nell'esempio tale variabile è b e l'assegnazione è

```
b = cubo(a);
```

Un esempio di esecuzione è il seguente:

```
Inserisci il numero: 5
     5.000000 elevato al cubo è uguale a 125.000000
```

Con questo semplice esempio abbiamo messo in luce diversi aspetti della sintassi delle funzioni:

- la dichiarazione di una funzione: double cubo(float);
- la definizione di una funzione: double cubo(float c) {...};
- il ritorno di un valore: return(c*c*c);
- l'invocazione di funzione: b = cubo(a).

Passiamo ora a considerare in dettaglio ciascuno dei punti evidenziati.