2.2 Le istruzioni composte

Nell'istruzione `if` soltanto un'istruzione semplice viene controllata dalla valutazione di *`espressione`*. Scrivendo

```
if(i<100)
  printf("minore di 100 ");
printf("istruzione successiva");
```

la seconda `printf` viene infatti eseguita indipendentemente dal risultato della valutazione di `i<100`. Ciò non ha niente a che vedere, ovviamente, con l'aspetto grafico della sequenza: lo stesso risultato si ottiene scrivendo

```
if(i<100)
  printf("minore di 100 ");
  printf("istruzione successiva");
```

Lo stesso dicasi per la clausola `else`: nel frammento di codice

```
if(i<100)
  printf("minore di 100 ");
else
  printf("maggiore o uguale a 100 ");
printf("istruzione successiva alla clausola else");
```

l'ultima `printf` viene eseguita indipendentemente dal fatto che `i` risulti minore oppure maggiore o uguale a 100.

Supponiamo ora di voler aggiungere nel Listato 2.1 un'ulteriore istruzione, oltre alla `printf`, ai due rami del costrutto `if-else`. Inizializziamo due variabili intere, `mag_100` e `min_100`, a zero. Nel caso in cui `i` risulti essere minore di 100, assegniamo il valore 1 a `min_100`, altrimenti lo assegniamo a `mag_100`. In termini tecnici diciamo che alziamo il *flag* `mag_100` o il *flag* `min_100` in base al risultato del controllo effettuato dall'istruzione `if` (vedi Listato 2.2).

```
/* Esempio istruzioni composte */

#include <stdio.h>

int i;
int mag_100;
int min_100;

main()
{
mag_100 = 0;
min_100 = 0;

printf("Dammi un intero: ");
scanf("%d", &i);

if(i<100) {
  printf("minore di 100\n");
  min_100 = 1;
}
else {
  printf("maggiore o uguale a 100\n");
  mag_100 = 1;
}
}
```

Listato 2.2 Esempio di utilizzo di istruzioni composte

A tale scopo utilizziamo *l'istruzione composta*, detta anche *blocco*, costituita da un insieme di istruzioni inserite tra parentesi graffe che il compilatore tratta come un'istruzione unica. Quindi la scrittura

```
{
```

```
  printf("minore di 100\n");
  min_100 = 1;
}
```

è un'istruzione composta costituita da due istruzioni. Nel listato completo la parentesi graffa aperta è stata inserita nella stessa linea dell'istruzione `if`, dopo la chiusura della parentesi tonda; ovviamente il significato non cambia: l'importante è saper riconoscere l'inizio e la fine dell'istruzione composta. Analoga considerazione vale per la clausola `else`.

Se nella sintassi assumiamo che un'istruzione possa essere semplice o composta, l'esempio del paragrafo precedente e quello appena visto sono riconducibili alla stessa forma del comando:

`if(`*espressione*`)` *istruzione1* `[else` *istruzione2*`]`

dove per entrambi gli esempi *espressione* corrisponde a `i<100`, mentre *istruzione1* e *istruzione2* corrispondono rispettivamente a:

Istruzione semplice	*Istruzione composta*

`printf("minore di 100\n");`	`{`
	`printf("minore di 100\n");`
	`min_100 = 1;`
	`}`

`printf("maggiore o uguale a 100\n");`	`{`
	`printf("maggiore o uguale a 100\n");`
	`mag_100 = 1;`
	`}`

Se non si fossero utilizzate le parentesi graffe il significato del programma sarebbe stato ben diverso:

```
if(i<100)
  printf("minore di 100\n");
  min_100 = 1;
else
  printf("maggiore o uguale a 100\n");
  mag_100 = 1;
```

Ciò che può fuorviare è l'aspetto grafico del codice, ma dato che l'indentazione non viene considerata la compilazione rivelerà un errore trovando un'istruzione `else` spuria, cioè non ricollegabile a un'istruzione `if`. Non essendo stata aperta la parentesi graffa, l'`if` regge la sola istruzione `printf("è minore di 100\n")`, dopo di che, se non trova la clausola `else`, si chiude.

Un'istruzione composta può essere immessa nel programma dovunque possa comparire un'istruzione semplice e quindi indipendentemente dal costrutto `if-else`; al suo interno, dopo la parentesi graffa aperta e prima delle altre istruzioni, possono essere inserite delle dichiarazioni di *variabili locali* a quel blocco, variabili – cioè – che possono essere utilizzate fino alla chiusura del blocco stesso:

```
{
  dichiarazione di variabili
  istruzione1
  istruzione2
  ...
  istruzioneN
}
```