3 LE ISTRUZIONI

3.1 Introduzione

Un'istruzione è un elemento eseguibile del programma; quindi un'istruzione specifica un'azione. Il C raggruppa le istruzioni nel seguente modo:

- Istruzioni di selezione (if e switch)
- Istruzioni di iterazione (for, while e do-while)
- Istruzioni di salto (break, continue, return)
- Espressioni (espressione valida in C).

3.2 Le istruzioni di selezione

Il C consente di utilizzare due istruzioni di selezione: if e switch. In alcune circostanze, in alternativa a if si può utilizzare l'operatore ?.

3.2.1 L'istruzione if

La forma generale dell'istruzione if è la seguente:

```
If (espressione) {
   istruzioni
}
else {
   istruzioni
}
```

La clausola else è opzionale. Se *espressione* fornisce un risultato vero, viene eseguita l'istruzione o il blocco relativo all'if; in caso contrario, verrà eseguita, se esiste, l'istruzione (o il blocco) relativa all'else.
Il seguente programma di esempio, che utilizza l'istruzione if, rappresenta una versione molto semplice del gioco "indovina il numero magico". Quando il giocatore indovina il numero, viene visualizzato il messaggio "Corretto!". Per generare il numero viene utilizzata la funzione della libreria standard rand().

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
  int magico;     /* numero magico */
  int giocatore;  /* numero immesso dall'utente */

  magico=rand();  /* genera il numero magico */

  printf("Indovina il numero magico: ");
  scanf("%d",&giocatore);

  if (magico==giocatore) {
    printf("\nCorretto!");
  }
  else {
```

```
    printf("\nErrato! Il numero magico e': %d\n",magico);
  }

  return 0;
}
```

3.2.2 L'istruzione switch

La switch è un'istruzione di selezione che controlla in successione il valore di un'espressione confrontandolo con un elenco di costanti intere o caratteri. Quando viene trovata una corrispondenza, vengono eseguite le istruzioni associate alla costante. La forma generale di una istruzione switch è la seguente:

```
switch (espressione) {
   case costante1:
      sequenza istruzioni
      break;
   case costante2:
      sequenza istruzioni
      break;
   case costante3:
      sequenza istruzioni
      break;
   …
   default:
      sequenza istruzioni
}
```

L'espressione deve fornire un valore costituito da un carattere o da un intero.
Il valore di *espressione* viene confrontato, nell'ordine, con i valori delle costanti specificate nelle istruzioni case. Quando viene trovata una corrispondenza, viene eseguita la sequenza di istruzioni associata al case e ciò fino alla successiva istruzione break o alla fine dello switch. L'istruzione default è opzionale e viene eseguita solo se non viene trovata alcuna corrispondenza.
L'istruzione break è una delle istruzioni di salto del C. Quando viene raggiunto un break in uno switch, l'esecuzione del programma "salta" alla riga di codice che segue l'istruzione switch.
Nell'esempio seguente la funzione menu() visualizza un menu per la scelta della lingua e richiama le procedure appropriate.

```
void menu(void) {
  char scelta;

  printf("Scegli la lingua del corso\n");
  printf("1. Italiano\n");
  printf("2. Inglese\n");
  printf("3. Francese\n");
  printf("Altro tasto per uscire.\n");

  scelta=getchar(); /* legge il tasto premuto */

  switch(scelta) {
    case '1':
```

```
      corso_It();
      break;
    case '2':
      corso_En();
      break;
    case '3':
      corso_Fr();
      break;
    default:
      printf("Non è stata selezionata alcuna opzione\n");
  }
}
```

3.3 Le istruzioni di iterazione

Le istruzioni di iterazione consentono la ripetizione di un gruppo di istruzioni fino al verificarsi di una determinata condizione.

3.3.1 Il ciclo for

La forma generale dell'istruzione for è la seguente:

for (*inizializzazione*; *condizione*; *incremento*) {
   istruzioni
}

L'*inizializzazione* è una istruzione di assegnamento che serve ad impostare la variabile di controllo del ciclo; la *condizione* è un'espressione relazionale che determina l'uscita dal ciclo; l' *incremento* determina come la variabile di controllo debba modificarsi ad ogni iterazione del ciclo.
Il ciclo for ripete l'esecuzione del blocco di istruzioni finché la *condizione* è vera. Quando la *condizione* diventa falsa, il ciclo viene terminato e l'esecuzione riprende dall'istruzione successiva al for.
Nel seguente esempio il ciclo for viene utilizzato per calcolare il fattoriale di un numero. Inizialmente alla variabile x viene assegnato il valore 1; fintantoché x è minore o uguale alla variabile Numero, che contiene il numero di cui si vuole calcolare il fattoriale, viene ripetuto il ciclo.

```
Fattoriale=1;
for (x=1; x<=Numero; x++) {
  Fattoriale=Fattoriale*x;
}
```

3.3.2 Il ciclo while

La forma generale del ciclo while è la seguente:

while (*condizione*) {
   istruzioni
}

ove *condizione* è una qualsiasi espressione valida.

Il ciclo while ripete l'esecuzione del blocco di istruzioni finché la *condizione* è vera. Quando la *condizione* diventa falsa, il ciclo viene terminato e l'esecuzione riprende dall'istruzione successiva al while.
Si osservi come la *condizione* venga testata prima dell'inizio del blocco.
Nel seguente esempio viene riproposto il calcolo del fattoriale di un numero mediante l'impiego del ciclo while.

```
Fattoriale=1;
x=1;
while (x<=Numero) {
  Fattoriale=Fattoriale*x;
  x++;
}
```

3.3.3 Il ciclo do-while

La forma generale del ciclo do-while è la seguente:

do {
   istruzioni
} while (*condizione*)

ove *condizione* è una qualsiasi espressione valida.
Il ciclo do-while ripete l'esecuzione del blocco di istruzioni finché la *condizione* è vera. Quando la *condizione* diventa falsa, il ciclo viene terminato e l'esecuzione riprende dall'istruzione successiva al ciclo.
Differentemente dai cicli for e while, che verificano la *condizione* prima dell'esecuzione del blocco di codice, il costrutto do-while testa la condizione al termine del blocco. Ciò implica che il blocco di istruzioni viene eseguito almeno una volta.
Nel seguente esempio l'utente è invitato ripetutamente ad immettere un numero finchè questo non sia maggiore di 50.

```
do {
  printf("\nImmettere un numero maggiore di 50: ");
  scanf("%d",&Numero);
} while (Numero<=50);
```

3.4 Le istruzioni di salto

Le istruzioni di salto presenti nel C sono: return, break e continue.

3.4.1 L'istruzione return

L'istruzione return produce l'uscita da una funzione e la restituzione del valore ad esso a ssociato.
La forma generale dell'istruzione è la seguente:

return *espressione*;

ove *espressione* è il valore restituito dalla funzione.

In una funzione possono esserci più istruzioni return. La funzione termina non appena l'esecuzione incontra il primo return o la parentesi graffa che ne chiude la definizione. Se questo caso si verifica in una funzione non void, il valore restituito sarà indefinito. È bene ricordare che una funzione void, poiché non restituisce alcun valore, non può contenere un'istruzione return che specifica una espressione.

3.4.2 L'istruzione break

L'istruzione break viene utilizzata per concludere un case in una istruzione switch e per terminare l'esecuzione di un ciclo. In entrambi i casi l'esecuzione del programma riprende dall'istruzione che segue, rispettivamente, lo switch o il ciclo.

3.4.3 L'istruzione continue

A differenza della istruzione break che causa la conclusione di un ciclo, l'istruzione continue salta tutto il codice che la segue e produce l'esecuzione la successiva iterazione.
Nella funzione seguente l'istruzione continue viene impiegata per effettuare la somma di tutti i numeri dispari compresi tra 1 ed il valore passato come parametro.

```
unsigned int Somma_Dispari (unsigned int Numero) {
  unsigned int Risultato;

  Risultato=0;
  for (unsigned int i=1; i<=Numero; i++) {
    if (i%2 ==0) continue;
    Risultato=Risultato+i;
  }
  return Risultato;
}
```