LE ISTRUZIONI CONDIZIONALI IN LINGUAGGIO C

In questo Workshop trattiamo un nuovo argomento legato alla programmazione in C, sospendendo temporaneamente la costruzione di RZB-1, che riprenderemo nei prossimi fascicoli.

Finora abbiamo studiato le basi del linguaggio C utilizzando esempi caratterizzati da flussi di istruzioni che possiamo definire 'lineari', ossia eseguite l'una di seguito all'altra senza ricorrere a 'salti' da un punto all'altro del programma. Come è intuibile, tuttavia, programmi di questo tipo sono di pochissima utilità in situazioni reali in cui è indispensabile offrire all'utente (o al programma stesso) la possibilità di scegliere tra più alternative possibili o di ripetere operazioni (o blocchi di codice) in maniera arbitraria. Per questo motivo praticamente tutti i linguaggi di programmazione più evoluti (tra questi, ovviamente, anche il C) offrono strutture condizionali e strutture cicliche (o iterative).

SE... ALLORA... ›››

In questo paragrafo iniziamo ad affrontare il primo dei due argomenti che verranno trattati in questo fascicolo, relativamente alle strutture condizionali. Definire una struttura condizionale significa, innanzitutto, definire un costrutto che consente di eseguire istruzioni nel caso in cui il programma si trovi a verificare una o più precise condizioni. Ma cos'è una condizione? Una condizione è una semplice composizione di istruzioni logiche utilizzanti operatori booleani, matematici o di confronto. Per introdurre semplicemente tali concetti, ricorriamo a un esempio pratico considerando la frase 'se piove prendo l'ombrello, altrimenti esco in maglietta'. 'Prendo l'ombrello' e 'esco in maglietta' possono essere considerate le due istruzioni operative (le chiameremo rispettivamente I_1 e I_2) di un ipotetico programma. Si intuisce, però, che tali operazioni sono mutuamente esclusive e legate al verificarsi di un preciso evento, ossia 'piove'. Esso è la condizione di esecuzione dell'istruzione I_1, mentre I_2 costituisce l'alternativa. Ma in che modo è possibile esprimere questa frase in C? Attraverso un semplice costrutto algoritmico costituito dalle due parole chiave 'if' e 'else' (scritte in minuscolo). Nell'esempio sottostante puoi osservare la frase precedente tradotta in pseudocodice C (ossia in un linguaggio intermedio tra la programmazione e la lingua comune, che viene spesso usato per rendere più 'informali' gli algoritmi):

Esempio 1

```
if (piove)
{
      Prendo l'ombrello;
}
else
{
      esco in maglietta;
}
```

'if' e 'else', quindi, permettono di creare due blocchi distinti di codice nei quali è possibile racchiudere istruzioni da eseguire nel caso in cui venga verificata o meno la condizione iniziale. A livello più generico, la definizione del costrutto 'if...else..' è la seguente:

definizione if... else...

```
if ( condizione )
{
    istruzioni da eseguire se la condizione è vera;
}
else
{
    istruzioni da eseguire in alternativa;
}
```

È importante tenere presente che il blocco 'else' è facoltativo e, di conseguenza, può non essere presente o necessario all'interno di un programma. Se la frase iniziale fosse stata, ad esempio: 'se piove prendo l'ombrello', non sarebbe stata data nessuna istruzione da eseguire in alternativa al 'prendere l'ombrello'; ne consegue che anche nella sua 'traduzione' in C tale frase vedrebbe omesso il blocco 'else' di codice. È inoltre possibile concatenare più istruzioni 'if', in modo da creare più alternative. Consideriamo, ad esempio, il piccolo programma in pseudocodice presentato qui di seguito.

Esempio 2

```
if(piove)
{
 esco prendendo l'ombrello;
}
else if (il clima è caldo)
{
  esco indossando una maglietta;
}
else
{
  esco indossando un maglione;
}
```

In questo esempio puoi osservare una sequenza concatenata di due strutture if/else. Vediamo come verrebbe eseguito tale codice da un eventuale computer. Per prima cosa il sistema effettuerebbe il test per verificare la presenza o meno di pioggia (istruzione if(piove)). Nel caso in cui piovesse verrebbe, ovviamente, eseguita l'operazione 'esco prendendo l'ombrello' e in alternativa (e solo in alternativa) il sistema passerebbe a effettuare il secondo test sulla temperatura (istruzioni else if (il clima è caldo)) decidendo, di conseguenza, se indossare una maglietta o un maglione. In questo caso, quindi, l'esecuzione del secondo test condizionale vincolata al fallimento della prima condizione. Vediamo un altro esempio, all'apparenza simile al precedente, nel quale viene omessa la parola chiave 'else' dalla quinta riga.

Esempio 3

```
if(piove)
{
 esco prendendo l'ombrello;
}
if (il clima è caldo)
{
  esco indossando una maglietta;
}
else
{
  esco indossando un maglione;
}
```

In tal caso la condizione 'il clima è caldo' non dipende dal fatto che piova o meno. Al contrario essa rappresenta un test ulteriore che viene svolto in ogni caso per stabilire cosa indossare in funzione del clima (a prescindere dalla necessità di dotarsi di ombrello).

QUICK REFERENCE ›››

Data l'importanza e il frequente utilizzo del costrutto if/else, riassumiamo il suo funzionamento:

```
if (espressione condizionale)
{
  'blocco vero';
}
else
{
  'blocco falso';
}
```

Permette di eseguire istruzioni in modo subordinato al verificarsi o meno di un'espressione condizionale passata come parametro. Se l'espressione è 'vera' allora vengono eseguite le istruzioni contenute nel 'blocco vero'. In alternativa viene eseguito il 'blocco falso'. L'uso della parola chiave 'else' e del relativo blocco di istruzioni alternative è facoltativo e necessario in funzione dei casi e del programma che si vuole sviluppare.

LE CONDIZIONI ›››

Le espressioni condizionali per i test possono essere dei tipi più disparati, in funzione della necessità. Possono essere, ad esempio, condizioni molto semplici basate sugli operatori di test (fascicolo 24, pag. 10) oppure condizioni composte, ottenute legando più elementi primitivi attraverso gli operatori logici (fascicolo 24, pag. 11). Una condizione primitiva può essere quella di confronto numerico, come: 'se la variabile 'n' è maggiore di 10...':

```
if ( n>10 )
```

oppure 'se il carattere 'c' è uguale a 't'...':

```
if ( c == 't' )
```

Un esempio di condizione complessa può essere, invece, l'espressione: 'se la variabile 'n' è uguale a '5' oppure a '23'...':

```
if ( n == 5 || n == 23 )
```

oppure l'espressione 'se la variabile q è un numero compreso tra 10 e 100 (inclusi) ed è pari...':

```
if ( q >= 10 && q <= 100
  && ( q%2 == 0 ) )
```

In quest'ultimo esempio, in particolare, si può vedere l'utilizzo dell'operatore 'modulo' (%) che viene usato per stabilire se un numero è pari o meno. Ricordiamo, infatti, che l'operatore '%' restituisce il resto intero della divisione tra i suoi operandi e che un numero pari diviso per due ha resto uguale a 0. Le espressioni condizionali possono includere quindi anche calcoli complessi e strutturati. Vediamo qualche esempio pratico di codice che potrai sperimentare sul tuo computer utilizzando l'ambiente di sviluppo Dev-C++ che hai installato nel fascicolo 17.

SWITCH... CASE... Oltre al costrutto if/else esiste un'altra struttura condizionale molto usata, quella nota con il nome **'switch/case'**, che in certe condizioni può dimostrarsi molto vantaggiosa. Il costrutto switch/case, infatti, **permette di scrivere in modo molto rapido le operazioni condizionali multiblocco la cui condizione decisionale si basa sul valore di una singola variabile.** Pensiamo, ad esempio, ai vecchi programmi privi di interfaccia grafica, il cui funzionamento era per lo più basato su menu a selezione numerica (ad esempio scegli '1' per la prima operazione di menu, '2' per la seconda ecc.). In questi applicativi il codice sorgente doveva prevedere dapprima la lettura di un valore numerico da tastiera, per poi passare all'attivazione della funzione richiesta sulla base del numero inserito dall'utente. Utilizzare il costrutto if/else in una situazione come questa significherebbe creare un lungo concatenamento di test e verifiche del valore della variabile di lettura. Tale soluzione, pur funzionando ugualmente, rende difficile la lettura e la manutenzione del codice. La struttura switch/case è, invece, molto più semplice per gestire questi casi. Alla base del suo funzionamento vi è la possibilità di **indicare il nome di una variabile (o un'intera espressione)** su cui verranno eseguiti in sequenza una **serie di test di uguaglianza**, in modo da creare un insieme di casi di funzionamento legati al valore dell'espressione stessa. Vediamo la sua sintassi. Nella struttura d'esempio presentata qui sotto puoi vedere come va impiegato correttamente il costrutto switch/case. Bisogna iniziare, innanzitutto, con la **parola chiave 'switch'**, seguita dall'espressione di test su cui si vuole operare. In seguito si definisce un sottoblocco di codice che racchiude i singoli casi di funzionamento, indicati con la **parola chiave 'case'**, attraverso la quale viene indicato anche il **valore di attivazione** (valore1, valore2...). L'ultimo caso è quello indicato con la **parola chiave 'default'**, che contiene le operazioni da compiere nel caso in cui l'espressione assuma un valore non citato esplicitamente nei singoli 'case'. Una menzione particolare deve essere fatta relativamente all'istruzione **'break'**, che deve essere usata ogniqualvolta si voglia concludere e uscire da un blocco 'case' per riprendere l'esecuzione del programma.

Sintassi switch/case

```
switch (espressione)
{
        case valore1:
                        istruzione;
                        istruzione;
                        istruzione;
                        ...
                        break;

        case valore2:
                        istruzione;
                        istruzione;
                        istruzione;
                        ...
                        break;

-----------.....

                default:
                        istruzione;
                        istruzione;
                        istruzione;
                        ...
                        break;
}
```

STEPbySTEP

IL CALCOLO DEL VALORE ASSOLUTO DI UN NUMERO INTERO

Il calcolo del valore assoluto è un'operazione che trova un vasto uso in matematica. Il suo effetto è, fondamentalmente, quello di operare su un numero passato come parametro rendendolo positivo (ossia 'negandolo') nel caso in cui esso sia minore di zero. Vediamo in questo esempio un semplice programma che permette di leggere un numero da tastiera, stampando a video il suo valore assoluto.

Il valore assoluto di un numero

```
/* calcolo del valore assoluto */

#include <stdio.h>

int main()
{
     int numero_letto;    /* variabile intera che conterrà
                             il numero letto */

     /* stampa a video il messaggio di richiesta */
     printf("\nScrivi un numero intero, positivo o negativo: ");

     /* legge il numero intero */
     scanf("%d", &numero_letto);

     /* se il numero letto è minore di zero */
     if(numero_letto<0)
     {
        /* reinizializza la variabile stessa con il suo opposto
           in modo da renderlo positivo*/
        numero_letto = -numero_letto;
      }

     /* se numero_letto è maggiore di 0 (positivo) non c'è bisogno
        di reinizializzare la variabile. */

     /* stampa a video il valore assoluto del numero */
    printf("\nIl valore assoluto del numero che hai scritto è:  %d",
numero_letto);

     /* termina il programma */
    return 1;
}
```

UN SEMPLICE GIOCO CON SOMMA E MOLTIPLICAZIONE

Il secondo esempio che analizzeremo sarà un semplice gioco basato sul calcolo della somma e della moltiplicazione tra due numeri inseriti dall'utente. Il programma chiederà in un primo momento di inserire i due numeri di partenza e proporrà in seguito di calcolare la loro somma e la loro moltiplicazione, verificando l'abilità matematica del giocatore.

Somma e moltiplicazione

```
/* gioco somma e moltiplicazione */

#include <stdio.h>

int main()
{
   /* variabili che conterranno i due numeri che letti */
   int numero1, numero2;

   /* variabili che archivieranno i risultati dell'utente */
   int somma, moltiplicazione;

  /* stampa la richiesta di inserimento del primo numero */
  printf("\nScrivi il primo numero: ");

  /* acquisisce il primo numero */
  scanf("%d", &numero1);

  /* stampa la richiesta di inserimento del secondo numero */
  printf("\nScrivi il secondo numero: ");

  /* acquisisce il secondo numero */
  scanf("%d", &numero2);

  /* stampa di un messaggio */
  printf("\nOra calcola la loro somma e scrivi il risultato: ");

  /* acquisisce la somma proposta dal giocatore */
  scanf("%d", &somma);

  /* stampa di un messaggio */
  printf("\nOra calcola il loro prodotto e scrivi il risultato: ");

  /* acquisisce il prodotto proposto dal giocatore */
  scanf("%d", & moltiplicazione);

  /* verifica i risultati */

  /* se somma e prodotto inseriti corrispondono a quelli
     calcolati */
  if(somma == (numero1*numero2) &&
     moltiplicazione == (numero1*numero2) )
      {

          /* Stampa il messaggio di congratulazioni */
          printf("\nBravo! Hai indovinato sia la somma, sia il
                  prodotto.");
       }
```

```
    /* se la somma inserita dall'utente corrisponde a quella
         calcolata dall'utente, ma il risultato del prodotto
        è differente */

    else if (somma == (numero1+numero2) &&
                moltiplicazione!=(numero1*numero2))
        {
           /* stampa il messaggio associato */
           printf("\Hai indovinato la somma, ma hai sbagliato la
                       moltiplicazione.");
         }

    /* se la somma è errata, ma il prodotto corrisponde a
       quello calcolato */
    else if (somma != (numero1+numero2) &&
             moltiplicazione==(numero1*numero2))
        {
            /* stampa il messaggio associato all'esito */
            printf("\Hai indovinato il prodotto,
                    ma hai sbagliato la somma.");
         }

    /* nessuno dei tre casi è stato verificato. */
    else
          {
            printf("\Hai sbagliato entrambe le operazioni,
                     devi esercitarti di più.");
           }

     return 1; /* termina il programma */
}
```

LO PSEUDOCODICE›››

Con pseudocodice (o pseudolinguaggio) si indica normalmente un linguaggio di programmazione che ha il puro scopo di 'descrivere' il funzionamento di programmi e algoritmi. Esso, quindi, non è un reale linguaggio di programmazione e non può, di conseguenza, essere compilato e convertito in un vero file eseguibile. Gli pseudolinguaggi nascono, solitamente, a discrezione dei programmatori che fondono sintassi e paradigmi della tecnologia di sviluppo utilizzata al linguaggio informale comune. Proprio per tale ragione, non si può parlare di standardizzazione di tali linguaggi, ma piuttosto di una creazione soggettiva che dipende dai singoli programmatori. A destra puoi vedere un semplice stralcio di pseudolinguaggio ispirato al C, utilizzato per descrivere il processo di navigazione di un robot all'interno di un ambiente esplorato per mezzo di sensori.

```
.......

if (il sensore destro indica
   un ostacolo)

{
   motore_destro=avanti;
   motore_sinistro=indietro;
}


else if (il sensore sinistro
   indica un ostacolo)
{
   motore_destro=indietro;
   motore_sinistro=avanti;
}
else if (entrambi i sensori
   indicano un ostacolo)
{
.......
```

UN SEMPLICE ESEMPIO DI MENU

In questo terzo esempio potrai sperimentare un semplice esempio di menu testuale basato sul costrutto condizionale switch/case.

Un esempio di menu

```
/* un semplice menu */
#include <stdio.h>

int main()
{
  /* variabile che conterrà la scelta dell'utente */
  int sceltautente;

  /* stampa il menu */

  /* stampa le quattro righe di menu */
  printf("\n\n1) voce di menu 1");
  printf("\n\n2) voce di menu 2");
  printf("\n\n3) voce di menu 3");
  printf("\n\n4) voce di menu 4");

  /* stampa messaggio di richiesta del valore */
  printf("\n\nInserisci la tua scelta:  ");

  /* acquisisce il valore numerico corrispondente
     alla scelta dell'utente */
  scanf("%d", &sceltautente);


  /* applica lo switch alla variabile 'sceltautente' */
  switch(sceltautente)
  {
        /* se sceltautente è uguale a '1' */
         /* stampa il messaggio corrispondente */
        case 1:    printf("\n\nHai selezionato l'opzione 1");
                   break; /* uscita dalla swich */
        /* se sceltautente è uguale a '2' */
         /* stampa il messaggio corrispondente */
        case 2:    printf("\n\nHai selezionato l'opzione 2");
                   break; /* uscita dalla swich */
        /* se sceltautente è uguale a '3' */
         /* stampa il messaggio corrispondente */
        case 3:    printf("\n\nHai selezionato l'opzione 3");
                   break; /* uscita dalla swich */
        /* se sceltautente è uguale a '4' */
         /* stampa il messaggio corrispondente */
        case 4:    printf("\n\nHai selezionato l'opzione 4");
                    break; /* uscita dalla swich */
         /* la scelta non corrisponde a nessuna delle opzioni*/
        default:    printf("\n\nNon hai scelto alcuna delle opzioni
      indicate");
                   break; /* uscita dalla swich */
  }
   return 1; /* fine del programma */
}
```