

# Strutture di Controllo

Le strutture di controllo consentono di intervenire sul flusso di esecuzione del programma. Senza le istruzioni per il controllo del flusso, il programma seguirebbe una logica unidirezionale, eseguendo le istruzioni da sinistra a destra e dall'alto al basso. La validità e l'utilità di un linguaggio di programmazione dipendono comunque dalla possibilità di modificare l'ordine delle istruzioni utilizzando strutture e cicli.

## Strutture decisionali

Visual Basic supporta le seguenti strutture decisionali:

- If...Then
- If...Then...Else
- Select Case

### If...Then

La struttura **If...Then** consente di eseguire una o più istruzioni in base alle condizioni specificate; la sintassi può essere a riga singola o a blocco:

```
If condizione Then istruzione
                        If condizione Then
                          istruzioni
                        End If
```

La *condizione* è in genere un confronto, ma può essere anche un'espressione che restituisce un valore numerico interpretato automaticamente come True o False. I valori numerici uguali a zero sono considerati False, mentre quelli diversi da zero sono considerati True. Se la condizione risulta True, vengono eseguite tutte le istruzioni successive alla parola chiave Then. Per eseguire una sola istruzione è possibile utilizzare la sintassi a riga singola.

Nella struttura If...Then a riga singola non viene utilizzata alcuna istruzione End If. Se si desidera eseguire più righe di codice quando la condizione risulta True, è necessario utilizzare la sintassi a blocco If...Then...End If.

### If...Then...Else

La struttura If...Then...Else consente di definire più blocchi di istruzioni di cui uno solo viene eseguito:

```
If condizione1 Then
  [bloccoistruzioni-1]
[Else] f condizione2 Then
  [bloccoistruzioni-2] ...
[Else]
  [bloccoistruzioni-n]
End If
```

Viene innanzitutto verificata la **condizione1**. Se risulta False, viene verificata la **condizione2** e così via fino a quando non viene individuata una condizione che risulti **True**. Verrà quindi eseguito il blocco di istruzioni corrispondente ed il codice successivo all'istruzione End If. E' anche possibile includere un blocco di istruzioni Else, il quale verrà eseguito se nessuna delle condizioni impostate risulta True.

Il blocco If...Then...Elseif rappresenta una delle possibilità di utilizzo della struttura If...Then...Else. È possibile includere qualsiasi numero di proposizioni Elseif oppure nessuna. La proposizione Else può essere inclusa indipendentemente dalla presenza di proposizioni Elseif.

Alla struttura **If...Then** è possibile aggiungere in qualsiasi momento altre parti **Elseif**. Se le istruzioni Elseif vengono utilizzate per confrontare la stessa espressione con valori diversi, la sintassi potrebbe risultare estremamente laboriosa, per cui è consigliabile utilizzare una struttura decisionale **Select Case**.

## Select Case

In Visual Basic è disponibile la struttura **Select Case** che può essere utilizzata in alternativa alla struttura If...Then...Else per eseguire in modo selettivo solo uno dei blocchi di istruzioni specificati. Select Case è analoga a If...Then...Else, ma migliora la leggibilità del codice nel caso di più scelte.

La struttura Select Case si basa su un'unica espressione di prova che viene valutata una sola volta all'inizio della struttura stessa. Il risultato di tale espressione viene confrontato con il valore di ciascuna istruzione Case della struttura e in caso di corrispondenza viene eseguito il blocco di istruzioni associato all'istruzione Case:

```
Select Case espressione di prova
    [Case elencoespressioni1
        [bloccoistruzioni-1]]
    [Case elencoespressioni2
        [bloccoistruzioni-2]]
    .
    .
    .
    [Case Else
        [bloccoistruzioni-n]]
End Select
```

Ciascun argomento **elencoespressioni** rappresenta un elenco di valori separati da virgole. Ciascun argomento **bloccoistruzioni** contiene da zero a più istruzioni. Se più istruzioni Case corrispondono all'espressione di prova, viene eseguito solo il blocco di istruzioni associato alla prima istruzione Case. Le istruzioni della proposizione Case Else, che è facoltativa, vengono eseguite se nessuno dei valori dell'elenco di espressioni corrisponde all'espressione di prova.

Mentre l'istruzione Select Case valuta un'espressione solo una volta all'inizio della struttura, la struttura If...Then...Else può valutare un'espressione diversa per ogni istruzione Elseif. È possibile sostituire la struttura If...Then...Else con una struttura Select Case solo se ogni istruzione If ed Elseif valuta la stessa espressione.

## Strutture iterative

Le strutture iterative consentono di eseguire ripetutamente una o più righe di codice. Visual Basic supporta le seguenti strutture iterative:

- Do...Loop
- For...Next
- For Each...Next

### Do...Loop

La struttura **Do...Loop** consente di eseguire un blocco di istruzioni per un numero di volte imprecisato. L'istruzione Do...Loop prevede numerose varianti, ciascuna delle quali valuta una condizione numerica che consente di determinare se continuare o meno l'esecuzione. Come per la struttura If...Then, la *condizione* deve essere rappresentata da un valore o da un'espressione che risulti False (uguale a zero) o True (diversa da zero).

Nella seguente struttura Do...Loop, le **istruzioni** vengono eseguite fino a quando la *condizione* risulta True:

```
Do While condizione
    istruzioni
Loop
```

L'esecuzione della struttura Do...Loop prevede innanzitutto la verifica della *condizione*. Se risulta False (uguale a zero), tutte le istruzioni vengono ignorate, se invece risulta True (diversa da zero), le istruzioni vengono eseguite e, dopo la riesecuzione dell'istruzione Do While, la condizione viene nuovamente verificata.

Il ciclo può pertanto essere eseguito un numero di volte imprecisato fino a quando la *condizione* non risulta True, ovvero diversa da zero. **Se la condizione risulta subito False, le istruzioni non verranno mai eseguite.**

L'istruzione Do...Loop prevede un'altra variante che consente di eseguire le istruzioni e di verificare la *condizione* al termine di ciascuna esecuzione. In questo modo le **istruzioni** vengono eseguite almeno una volta:

```
Do
    istruzioni
Loop While condizione
```

Le altre due varianti sono analoghe a quelle descritte, con la sola differenza che il ciclo continua fino a quando la *condizione* risulta False anziché True.

Ciclo eseguito zero o più volte

```
Do Until condizione
    Istruzioni
Loop
```

Ciclo eseguito almeno una volta

```
Do
    Istruzioni
Loop Until condizione
```

## For...Next

Le strutture iterative Do risultano utili quando non si conosce il numero di cicli di esecuzione delle istruzioni richiesti. Se è necessario eseguire le istruzioni per un numero di volte specifico, è consigliabile utilizzare un ciclo For...Next. A differenza dei cicli Do, i cicli For utilizzano una variabile contatore il cui valore aumenta o diminuisce durante ciascuna iterazione del ciclo. La sintassi è la seguente:

```
For contatore = inizio To fine [Step incremento]
    istruzioni
Next [contatore]
```

Gli argomenti *contatore*, *inizio*, *fine* e *incremento* sono numerici.

L'argomento *incremento* può essere positivo o negativo. Se è positivo, le istruzioni del ciclo vengono eseguite solo se l'argomento *inizio* è minore o uguale all'argomento *fine*. Se è negativo, il corpo centrale del ciclo viene eseguito solo se l'argomento *inizio* è maggiore o uguale all'argomento *fine*. Se non si imposta l'istruzione Step, l'argomento *incremento* viene automaticamente impostato su 1.

Durante l'esecuzione del ciclo For:

1. L'argomento *contatore* viene impostato sul valore dell'argomento *inizio*.
2. Viene verificato se l'argomento *contatore* è maggiore dell'argomento *fine*. In tal caso il ciclo termina.
3. Se l'argomento *incremento* è negativo, viene verificato se l'argomento *contatore* è minore dell'argomento *fine*.
4. Vengono eseguite le *istruzioni*.
5. L'argomento *contatore* viene incrementato di 1 oppure in base all'argomento *incremento*, se specificato.
6. Vengono ripetuti i passaggi da 2 a 4.

## For Each...Next

Il ciclo For Each...Next è simile al ciclo For...Next, ma anziché ripetere le istruzioni il numero di volte specificato, ripete un gruppo di istruzioni per ciascun elemento di un insieme di oggetti o di una matrice. Ciò risulta particolarmente utile se non si conosce il numero di elementi di un insieme.

La sintassi del ciclo For Each...Next è la seguente:

```
For Each elemento In gruppo
    istruzioni
Next elemento
```