# Corso di Visual Basic (Parte 4) *di Maurizio Crespi*

*Oggetto di studio della quarta puntata del corso dedicato alla programmazione in Visual Basic sono i cicli e in particolare la struttura For, mediante la quale è possibile far eseguire al calcolatore delle operazioni ripetitive*

Uno dei punti di forza dei calcolatori è rappresentato dalla possibilità di eseguire in modo estremamente veloce delle operazioni ripetitive. A tal fine, i linguaggi di programmazione mettono a disposizione delle strutture che prendono il nome di cicli. Naturalmente, Visual Basic non si sottrae a questa regola. La prima struttura ad essere presa in esame è rappresentata dal ciclo For, a cui è dedicata questa lezione.

## Le soluzioni degli esercizi dello scorso numero

Come sempre, prima di procedere allo studio dei nuovi argomenti, saranno illustrate le soluzioni degli esercizi proposti nello scorso numero.

### *Primo esercizio*

Il primo esercizio prevede la realizzazione di un programma in grado di leggere un numero composto da una sola cifra e di offrire come risultato la sua indicazione in lettere. Si tratta di una tipica applicazione della struttura *Select Case*, che opera una scelta fra dieci valori. Come interfaccia utente si fa uso di un form costituito da una textbox, denominata *txtCifra*, a cui è affidata la ricezione dei dati in ingresso. Il risultato è restituito per mezzo della label *lblMessaggio*, che deve essere aggiornata ogniqualvolta la cifra in ingresso subisca una variazione, ovvero in corrispondenza del verificarsi dell'evento *change* sulla textbox. La gestione di quest'ultimo è possibile per mezzo della seguente procedura:

```
Private Sub txtCifra_Change()
Dim Cifra As Integer
Dim DatoIngresso As String

DatoIngresso = txtCifra.Text

If IsNumeric(DatoIngresso) Then
Cifra = Val(DatoIngresso)
Select Case Cifra
Case 0
lblMessaggio.Caption = "ZERO"
Case 1
lblMessaggio.Caption = "UNO"
Case 2
lblMessaggio.Caption = "DUE"
Case 3
lblMessaggio.Caption = "TRE"
Case 4
lblMessaggio.Caption = "QUATTRO"
Case 5
lblMessaggio.Caption = "CINQUE"
```

```
Case 6
lblMessaggio.Caption = "SEI"
Case 7
lblMessaggio.Caption = "SETTE"
Case 8
lblMessaggio.Caption = "OTTO"
Case 9
lblMessaggio.Caption = "NOVE"
Case Else
lblMessaggio.Caption = "Il numero non è composto da una sola cifra"
End Select
Else
lblMessaggio.Caption = "Dato non numerico"
End If

End Sub
```

La prima operazione effettuata consiste nella lettura del dato in ingresso attraverso la textbox. Se esso risulta convertibile in un numero, ovvero se la funzione *IsNumeric* restituisce il valore logico *True*, il dato è copiato nella variabile intera *Cifra* ed è utilizzato per controllare la struttura *Select Case*, che provvede ad effettuare i confronti necessari per verificare l'uguaglianza con i numeri compresi fra 0 e 9 ed a visualizzare la stringa corretta mediante la label.

*Secondo esercizio*

Il secondo esercizio prevede la creazione di un programma che, ricevendo in ingresso il nome di un mese, sia in grado di indicarne il numero dei giorni in un anno specificato. La soluzione comporta nuovamente l'uso della struttura Select Case. Per l'acquisizione dei dati si provvede, come di consueto, ad utilizzare delle caselle di testo, denominate *txtMese* e *txtAnno*. Il risultato è visualizzato in seguito alla pressione di un pulsante per mezzo della label *lblMessaggio*. Dopo aver creato un form ed aver posto su di essi gli oggetti grafici, è possibile implementare la procedura destinata al calcolo. Supponendo che al tasto sia assegnato il nome btnCalcola, si può scrivere:

```
Private Sub btnCalcola_Click()

Dim Anno As Integer
Dim Giorni As Integer
Dim Mese As String
Anno = Val(txtAnno.Text)
Mese = UCase(txtMese.Text)

Select Case Mese
Case "GENNAIO", "MARZO", "MAGGIO", "LUGLIO", "AGOSTO", "OTTOBRE", "DICEMBRE":
Giorni = 31
Case "APRILE", "GIUGNO", "SETTEMBRE", "NOVEMBRE":
Giorni = 30
Case "FEBBRAIO":
If ((Anno Mod 4) = 0) And ((Anno Mod 100) <> 0) Then
```

```
Giorni = 29
Else
Giorni = 28
End If
Case Else
Giorni = 0
End Select
If Giorni > 0 Then
lblMessaggio.Caption = "Numero giorni :" & Giorni
Else
lblMessaggio.Caption = "Mese non corretto"
End If

End Sub
```

La struttura di selezione è controllata dalla variabile Mese, il cui contenuto è opportunamente convertito in maiuscolo per mezzo della funzione *UCase* e confrontato con i nomi dei mesi, anch'essi composti solo da lettere maiuscole. Ciò fa sì che il confronto non sia case sensitive. Si noti, che nel caso in cui il mese selezionato sia febbraio, è eseguito un ulteriore controllo per verificare se l'anno è bisestile. Tale verifica è effettuata testando la contemporanea validità di due condizioni, ovvero la divisibilità dell'anno per quattro e la non divisibilità per cento. La condizione risulta vera se entrambi i resti delle divisioni, calcolati per mezzo della funzione *Mod*, sono nulli. Si ricorre pertanto all'operatore logico And, in grado di restituire il valore True solo in caso di contemporanea verifica delle condizioni che ne costituiscono gli operandi.

## Il ciclo For

Si supponga di voler realizzare un programma in grado di calcolare il fattoriale di un numero, ovvero il prodotto di tutti i valori interi positivi minori o uguali ad esso. Gli strumenti illustrati nelle puntate precedenti di questo corso non si rivelano sufficienti a tal fine, in quanto è necessaria la capacità di ripetere per un numero variabile di volte l'operazione di moltiplicazione. Si impone pertanto il ricorso alle strutture di iterazione. Nel caso dell'esempio, occorre applicare la moltiplicazione a tutti i numeri naturali minori o uguali a quello di cui si desidera calcolare il fattoriale. Appare così evidente la necessità di disporre di una struttura in grado di permettere la ripetizione di una porzione di codice per un numero finito di volte; si tratta della classica struttura *For*, che nel caso di Visual Basic è caratterizzata dalla seguente sintassi:

```
For <contatore> = <valore iniziale> To <valore finale> [Step <passo>]
<istruzione 1>
...
<istruzione n>
Next [<contatore>]
```

Dopo la parola chiave *For* è necessario far seguire una variabile intera, che funge da contatore. Il suo valore è incrementato ad ogni ripetizione di un numero di unità pari a quello specificato dopo la parola chiave *Step*, fino al raggiungimento del valore finale; questa condizione determina la fine delle iterazioni e il passaggio all'istruzione seguente la parola riservata *Next*.

Volendo quindi realizzare un'applicazione in grado di calcolare il fattoriale di un numero, occorre creare un form ed inserire su di esso una textbox, usata per leggere il dato in ingresso, una label, destinata ad accogliere il risultato e un pulsante, a cui è affidato il compito di avviare il calcolo. Alla pressione di quest'ultimo deve essere associata la seguente procedura:

```
Private Sub btnCalcola_Click()

Dim Contatore As Integer
Dim Fattoriale As Integer
Dim Numero As Integer

If IsNumeric(txtNumero.Text) Then
Numero = Val(txtNumero.Text)

     If Numero >= 0 Then
     Fattoriale = 1
         For Contatore = 1 To Numero
         Fattoriale = Fattoriale * Contatore
     Next Contatore
     lblRisultato.Caption = Fattoriale
     Else
     lblRisultato.Caption = "Errore: valore negativo"
     End If

Else
lblRisultato.Caption = "Errore: dato non numerico"
End If

End Sub
```

Essa dapprima verifica che nella textbox *txtNumero* sia presente un dato numerico. In tal caso, quest'ultimo è assegnato alla variabile *Numero* dopo la necessaria conversione per mezzo della funzione *Val.* Tale valore è quindi oggetto di un secondo controllo volto a verificarne la positività. La funzione fattoriale, infatti, non è definita per i numeri negativi e vale 1 se il dato in ingresso è nullo. Successivamente, la procedura provvede ad effettuare le moltiplicazioni. A tal fine fa uso di un ciclo *For*, avente come valore iniziale 1 e come valore finale il numero di cui si desidera calcolare il fattoriale. Si noti che è stata omessa la parola chiave *Step. I*n questo caso, il passo adottato è quello predefinito, cioè 1. All'interno della struttura è presente una sola riga di codice, che ha il fine di moltiplicare il contenuto della variabile *Fattoriale* per il valore assunto dal contatore. La sua ripetizione per tutti i valori compresi fra 1 e il numero fornito in ingresso fa sì che alla fine del ciclo la variabile *Fattoriale* contenga il risultato desiderato, che può essere visualizzato per mezzo della label. Si noti ciò che avviene se in ingresso è fornito il numero 0. La variabile *Fattoriale* è comunque inizializzata al valore 1. L'istruzione all'interno del ciclo, invece, non è mai eseguita. Infatti, come si è detto in precedenza, la ripetizione termina quando il contatore raggiunge il valore finale. In questo caso, la condizione è immediatamente soddisfatta, per cui non si verificano iterazioni.

*Esercizio*
Per esercitarsi sull'uso della struttura *For*, si provi a realizzare un programma che, dato in ingresso un numero intero, visualizzi la somma di tutti i numeri naturali pari minori o uguali ad esso.

## Cicli nidificati

Si supponga ora di voler modificare l'applicazione dell'esempio precedente per fare in modo che fornisca all'utente la somma dei fattoriali dei primi *n* numeri interi positivi, dove *n* rappresenta il dato in ingresso acquisito mediante la casella di testo *txtNumero*.

Per il calcolo del fattoriale è necessario ricorrere nuovamente a un ciclo, che è sostanzialmente identico a quello visto in precedenza. Tale operazione deve essere però ripetuta per tutti i numeri naturali minori o uguali al dato *n* fornito dall'utente, ovvero per tutti i valori interi compresi fra 1 e *n.*

Si impone pertanto l'uso di un ulteriore ciclo, che deve risultare più esterno rispetto al precedente. Al pulsante *btnCalcola* si può quindi associare la seguente procedura:

```
Private Sub btnCalcola_Click()

Dim Contatore As Integer
Dim Fattoriale As Integer
Dim n As Integer
Dim Risultato As Long

If IsNumeric(txtNumero.Text) Then
n = Val(txtNumero.Text)

     If n > 0 Then
         Risultato = 0
             For Contatore1 = 1 To n
               Fattoriale = 1
                  For Contatore2 = 1 To Contatore1
                    Fattoriale = Fattoriale * Contatore2
                  Next Contatore2

               Risultato = Risultato + Fattoriale
             Next Contatore1
       lblRisultato.Caption = Risultato
     Else
       lblRisultato.Caption = "Errore: valore negativo o nullo"
     End If

Else
lblRisultato.Caption = "Errore: dato non numerico"
End If

End Sub
```

Anche questa volta è per prima cosa eseguito un controllo della validità del dato digitato dall'utente. Se si tratta di un numero intero positivo, la procedura provvede ad avviare il calcolo, dopo aver posto a zero il valore della variabile *Risultato*. Si noti che tale operazione non è in realtà necessaria, in quanto Visual Basic provvede automaticamente ad azzerare le variabili intere in fase di creazione. La sua presenza, però, contribuisce a rendere leggibile il codice, soprattutto a coloro che sono abituati all'uso di altri linguaggi che non presentano questa caratteristica. Si osservino ora i due cicli nidificati. La variabile usata come contatore in quello più esterno funge da valore finale nel ciclo interno. Ad ogni iterazione è pertanto calcolato il fattoriale del numero contenuto nella variabile *Contatore1*. Tale valore va ad incrementare quello della variabile *Risultato*, che alla fine è visualizzato mediante la label. Allo scopo di mantenere una buona leggibilità del codice, è necessario fare in modo che alle istruzioni che si trovano all'interno di un ciclo sia associato un rientro sinistro maggiore. In tal modo risulta semplice riconoscere l'inizio e la fine della struttura. Tale necessità è ancora più evidente allorquando vi siano più cicli nidificati. Per lo stesso motivo, sebbene sia possibile farne a meno, è preferibile far seguire ogni parola chiave *Next* dal nome della variabile di conteggio che caratterizza il ciclo. In caso di omissione, l'interprete comunque assume che la struttura da delimitare sia quella posta nel blocco più interno.

## Le funzioni Len e Mid$

Si supponga di voler scrivere un programma che sia in grado di invertire una stringa fornita in ingresso. Per la sua realizzazione occorre far uso di due funzioni specifiche per il trattamento dei dati alfanumerici. Si tratta della funzione *Len*, che restituisce la lunghezza della stringa passatale come parametro e della più complessa funzione *Mid$*. Quest'ultima permette di estrarre da una stringa una sequenza composta da un dato numero di caratteri consecutivi. La sua sintassi è la seguente:

```
<risultato> = Mid$(<stringa>, <posizione>, [<numero caratteri>])
```

I parametri rappresentano rispettivamente la stringa fornita in ingresso, la posizione del primo carattere da estrarre e il numero dei caratteri oggetto dell'operazione. Si noti che quest'ultimo è facoltativo. In sua assenza, è restituita tutta la porzione della stringa compresa fra la posizione indicata e il carattere posto all'estrema destra. Ad esempio, la riga

```
A$ = Mid$("Developing Software Solutions", 12, 8)
```

Assegna alla variabile *A$* la stringa "Software". Scrivendo invece

```
A$ = Mid$("Developing Software Solutions", 12)
```

si provoca l'assegnamento della sequenza alfanumerica "Software Solutions".

## Cicli con decremento

Ora si dispone dei mezzi necessari per la realizzazione dell'applicazione in grado di invertire una stringa. Supponendo di utilizzare un form analogo a quello degli esempi precedenti, caratterizzato da una textbox destinata all'acquisizione dei dati in ingresso, denominata *txtStringa*, dalla label *lblRisultat*o, utilizzata per visualizzare la stringa risultante e dal pulsante *btnInverti*, che ha lo scopo di provocare l'inversione della sequenza, la procedura da associare ad esso è la seguente:

```
Private Sub btnInverti_Click()

Dim i As Integer
Dim Lunghezza As Integer
Dim Risultato As String
Dim Carattere As String * 1

Lunghezza = Len(txtStringa.Text)
Risultato = ""


For i = Lunghezza To 1 Step -1
    Carattere = Mid$(txtStringa.Text, i, 1)
    Risultato = Risultato + Carattere
Next i

lblRisultato.Caption = Risultato

End Sub
```

Si fa uso della funzione *Len* per calcolare la lunghezza della stringa digitata dall'utente. Il valore restituito determina la posizione del primo carattere che deve essere aggiunto alla variabile *Risultato*. Esso rappresenta pertanto il valore di partenza in un ciclo *For* caratterizzato da un decremento, ovvero da un passo di segno negativo. Questa volta, quindi, il valore iniziale risulta maggiore di quello finale e le iterazioni terminano quando è raggiunto il valore minimo, nella fattispecie 1. La funzione *Mid$* è utilizzata per estrarre il carattere posto nella posizione indicata dal contatore. Si noti che la variabile atta a contenerlo è di tipo stringa ed è stata dichiarata di dimensione pari a un elemento per mezzo della riga

```
Dim Carattere As String*1
```

Visual Basic, infatti, a differenza di molti altri linguaggi di programmazione, non dispone di un tipo di dati idoneo alla memorizzazione dei singoli caratteri. Per ovviare a questo inconveniente, è necessario ricorrere ad una stringa di dimensione pari ad un'unità, come in questo esempio. Per mezzo dell'operatore di concatenazione (+), ogni carattere estratto è aggiunto in coda al contenuto della variabile Risultato, che al termine delle iterazioni contiene la stringa invertita. Quest'ultima può essere quindi assegnata alla proprietà Caption della label *lblRisultat*o per essere visualizzata.

## *Esercizio*

Per esercitarsi sui concetti esposti, si provi a realizzare un programma che, ricevuto in ingresso un numero intero positivo, sia in grado di restituire il maggior numero naturale per cui esso è divisibile.

## Conclusioni

Il ciclo *For* è estremamente usato in ogni applicazione. Esso non rappresenta però l'unica struttura di iterazione. Ne esistono delle altre, che saranno oggetto di studio nella prossima lezione. Per comprendere il loro funzionamento è necessario aver assimilato quanto esposto in questa trattazione. Al lettore è pertanto rivolto il consueto invito ad esercitarsi nell'applicazione dei concetti illustrati.