Operatori di confronto, logici e matematici

Per combinare o confrontare specifici valori in un'espressione si usano gli **operatori**. Il loro nome deriva dal fatto che essi sono i simboli che indicano specifiche operazioni matematiche o di altro genere da eseguire su vari valori in un'espressione. Quando in un'espressione si usa un operatore, i dati, che siano variabili o costanti su cui l'operatore agisce vengono detti operandi. Nell'espressione **2+1**, per esempio, i numeri **2** e **1** sono gli operandi dell'operatore di somma (+). Adesso vediamo i vari tipi di operatori e come si usano

Operatori di confronto

Gli operatori di confronto, chiamati a volte operatori relazionali, vengono utilizzati soprattutto per stabilire il criterio, in base al quale prendere delle decisioni. Il risultato di un'operazione di confronto è sempre di tipo *Boolean*: *True* o *False* e sono usati per comparare valori letterali, costanti o variabili di ogni tipo. La tabella rappresentata in **figura 1** elenca gli operatori di confronto disponibili nel VBA

Operatore	Significato
=	Uguaglianza
>	Maggiore di
<	Minore di
>=	Maggiore o uguale
<=	Minore o uguale
<>	Diverso
In	All'interno di un insieme
Is Null	Il campo è vuoto
Is Not Null	Il campo non è vuoto
Between	Tra due valori

Fig. 1

Se entrambi gli operandi di un'espressione di confronto sono dello stesso tipo di dati, il VBA esegue il confronto diretto per quel tipo di dati, per esempio se entrambi sono stringhe, VBA compara le due stringhe, se sono date VBA compara le date e così via. Utilizzando gli operatori, è possibile impostare delle condizioni più sofisticate alle istruzioni che andremo a scrivere. Aiutiamoci con un esempio per comprendere l'utilizzo degli operatori di confronto

4 <> 5 => il risultato è True, 4 è diverso da 5
Alex > Carlo => il risultato è False, "Alex" è più corto di "Carlo"
Abc = abc => il risultato è False, Abc è diverso da abc

Possiamo dire, In maniera molto sintetica, che questo tipo di operatore altro non fa che confrontare due variabili e appurare se la condizione che chiediamo tramite il simbolo dell'operatore, sia vera o falsa e il risultato del confronto è sempre un valore di tipo Boolean. Le variabili booleane riportano o accettano solo due valori, Vero(True) e Falso(False) e possiamo impostarle in questo modo

```
Dim alex As Boolean
alex = True
```

Dalle due righe di codice abbiamo dedotto che è stata dichiarata la variabile alex con tipo di dati Boolean invece di impostarla come tipo di dati con Integer o Stringa. La seconda linea di codice inserisce un valore nella variabile, ma non è un dato che possiamo manipolare, mettiamo solo una "condizione" cioè diciamo che la variabile alex è uguale a Vero. Vediamo qualche riga di codice per comprendere meglio come usarla

Codice:

```
If alex = True Then
MsgBox "E 'vero"
Else
MsgBox "E 'falso"
End If
```

La prima linea del ciclo **If** verifica se la variabile alex corrisponde a un valore Vero, se lo è, allora visualizza un messaggio, dato che ci sono solo due opzioni da testare (True e False) possiamo avere una parte del ciclo If per testare un valore (Che può essere True) e se non viene soddisfatto sarà sicuramente il contrario (False). Possiamo approfondire con il codice seguente

Codice:

```
Sub operatori()
Dim alex As Integer
alex = 10
If alex < 20 Then
MsgBox alex & " " & "è inferiore a 20"
End If
End Sub
```

Se osserviamo il codice sopra riportato notiamo che abbiamo creato una variabile (alex), l'abbiamo dichiarata di tipo *Integer* (Dim alex As Integer) e memorizzato il valore 10 in essa (alex=10), ma se osservate la prima riga della dichiarazione **If**, viene posta la condizione If alex < 20 Then, se si consulta la tabella di **figura 1** vedrete che il simbolo < significa Minore di, quindi la condizione posta è "Se alex è inferiore a 20" la condizione restituisce TRUE e il codice tra If e End If viene eseguito, se invece la condizione restituisce FALSE allora VBA salterà tutte le righe di codice e passerà alla prima istruzione che trova dopo End If. Se eseguiamo la macro otteniamo una finestra come la seguente

Fig. 2

Possiamo usare altri operatori esposti in figura 1 in tutti i casi che possono aiutarci a porre delle condizioni come per esempio utilizzando un ciclo If in questo modo

Codice:

```
Sub operatori()
Dim alex As Integer
alex = 19
If alex = 20 Then
MsgBox alex & " " & "è uguale a 20"
ElseIf alex > 20 Then
MsgBox alex & " " & "è maggiore di 20"
Else
MsgBox alex & " " & "è inferiore a 20"
End If
End Sub
```

Fig. 3

Per concludere l'argomento possiamo riassumere affermando che Il risultato di un operatore di confronto è un cosiddetto valore di verità: può essere vero(True) oppure falso(False)

Operatori Matematici

Il VBA è in grado di eseguire tutte le operazioni aritmetiche normali come somma, sottrazione, divisione e moltiplicazione in quanto l'elaborazione di dati numerici è una delle attività principali di un programma. VBA supporta una serie di operatori matematici che possono essere usati negli enunciati di un programma. Queste operazioni, con il relativo simbolo che identifica l'operatore matematico, sono riportate nella tabella di **figura 4**

Operatore	Significato
+	Somma
-	Sottrazione
*	Moltiplicazione
/	Divisione
\	Divisione intera
Mod	Resto della divisione
&	Concatenazione di 2 stringhe

Fig. 4

Avrete certamente notato nelle linee di codice sopra esposte che abbiamo usato un simbolo particolare dopo l'istruzione MsgBox. Abbiamo inserito la variabile alex poi uno spazio e poi una E commerciale (**&**). Questo simbolo in VBA serve per concatenare (unire) due o più stringhe. Infatti dopo la & abbiamo uno spazio seguito da una stringa vuota posta virgolette, poi ancora una &, uno spazio e il testo del messaggio racchiuso tra virgolette. Quindi, con la & abbiamo unito la variabile e il testo separandolo da uno spazio vuoto

Se **NON** avessimo usato la concatenazione tra la variabile e il testo del messaggio avremmo ricevuto un errore di compilazione come il seguente

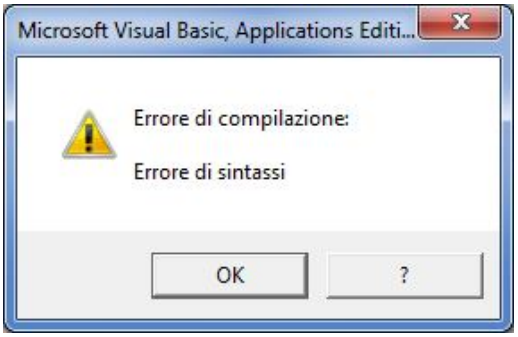

Fig. 5

Al tempo stesso concatenando la variabile col testo del messaggio senza inserire lo spazio vuoto avremmo ottenuto un box come il seguente

Fig. 6

Come si può notare il valore della variabile alex risulta essere attaccato al testo del messaggio, che sicuramente è poco leggibile e interpretabile. Per meglio comprendere possiamo usare il codice sotto riportato

Codice:

```
Sub Prova()
a = "Auguro a tutti"
b = "gli utenti del Pianeta"
c = "Buon Natale e Felice Anno Nuovo" >
MsgBox a & b & c
End Sub
```

Che riporta un messaggio come il seguente

Fig. 7

Nota : *Fate attenzione che se avete all'inizio delle routine la parola chiave Option Explicit vi viene rimandato un errore, in quanto come abbiamo detto poco sopra in presenza di questa parola chiave tutte la variabili vanno dichiarate.*

Se osservate attentamente la **Fig. 9** noterete che ci sono delle frasi attaccate (tuttigli e PianetaBuon), modifichiamo allora il nostro codice utilizzando un *operatore di concatenazione* (**&**) in questo modo

Codice:

```
Sub Prova1()
a = "Auguro a tutti"
b = "gli utenti del Pianeta"
c = "Buon Natale e Felice Anno Nuovo"
MsgBox a & " " & b & " " &  c
End Sub
```

Con questo ulteriore concatenamento abbiamo inserito uno spazio vuoto tra una frase e l'altra, infatti la sintassi " " (uno spazio vuoto racchiuso dalle virgolette) sta ad indicare uno spazio vuoto e di conseguenza la nostra scritta ci appare così

Fig. 8

Con questi operatori possiamo anche fare operazioni matematiche usando le variabili, se prendiamo questo codice, ricordando che le variabili Alex, x e y sono state dichiarate come *Integer* nella barra delle dichiarazioni quindi condivisibili in tutto il modulo

Codice:

```
Sub somma()
Dim operator As Integer
Alex = 10
x = 20
y = 30
operator = y - x + Alex
MsgBox "Il risultato della tua operazione è" & " " & operator
End Sub
```

Questo codice equivale a: Prendi il valore della variabile **y**, sottrai il valore della variabile **x** e somma il valore della variabile *Alex* eseguendo questa macro ci verrà riportato questo avviso

Fig. 9

Operatori Logici

Il più delle volte gli operatori logici forniti dal VBA vengono usati per combinare i risultati di singole espressioni di confronto al fine di costruire criteri complessi per prendere una decisione all'interno di una procedura, oppure per stabilire le condizioni in base alle quali un enunciato o un gruppo di istruzioni possa essere replicato. Come operando di un operatore logico si può usare qualsiasi espressione valida che dia un risultato Boolean, o un numero che il VBA possa convertire in valore booleano. Il VBA considera 0 come equivalente a False e ogni altro valore numerico equivalente a True. Nella tabella di Figura 12 viene fornito l'elenco degli operatori logici presenti nel VBA

Operatore	Significato
And	Entrambe le espressioni sono vere
Or	E' vera una o l'altra espressione
Not	L'espressione non è vera

Fig. 10

Per poter comprendere al meglio la struttura degli operatori logici è importante saper leggere la tabella delle verità Booleana, essa non è altro che una tabella che mostra tutte le possibili combinazioni di valori per una particolare espressione logica e il risultato di ognuna. Questa tabella ha 3 colonne, la prima contiene il valore del primo operando, la seconda il valore del secondo operando e la terza il valore del risultato dell'espressione. Guardando la riga sotto esposta

False True False

In essa il primo operando è False, il secondo True e il risultato dell'operatore **And** con questi valori è *False*. La riga in questione, perciò, insegna che il risultato dell'espressione *False And True è False*. La sintassi dell'operatore And è la seguente:

Operando1 And Operando2

Vediamo ora la tabella della verità booleana per l'operatore And

- True True True
- True False False
- False True False
- False False False

Dove la prima colonna rappresenta il valore booleano del primo valore. La seconda il valore booleano del secondo valore e la terza colonna il risultato dell'espressione.

L'operatore **And** effettua una congiunzione logica (detta anche somma logica) e il risultato di un'operazione And è True solo se ambedue gli operatori sono True altrimenti è False. L'operatore And viene utilizzato per determinare se due diverse condizioni sono vere contemporaneamente. Per esempio:

(ricavo < 5000) And (profitto < 1000)

L'espressione precedente risulta vera (True) se il valore di ricavo è minore di 5.000 e al tempo stesso il valore del profitto è minore di 1.000 (e solo in quel caso). Si noterà che i due operandi in questa espressione sono espressioni di confronto e anche che sono poste tra parentesi per renderli operandi dell'operatore And. Le parentesi indicano al VBA di calcolare il risultato dell'espressione tra parentesi prima di valutare altre parti dell'espressione completa. Le parentesi, inoltre, rendono l'espressione più leggibile raggruppando le parti correlate di un'espressione. L'uso in questo modo delle parentesi, per raggruppare parti di un'espressione in una sotto-espressione è molto comune con espressioni di ogni tipo, numeriche, di stringa, di data e di confronto.

L'operatore Or effettua una disgiunzione logica, spesso specificata anche come or inclusivo. Il risultato di un'operazione Or è True solo se uno o entrambi gli operandi è True, altrimenti è False. L'operatore *Or* ha questa sintassi:

Operando1 Or Operando2

La tabella della verità booleana per l'operatore Or è la seguente:

- True True True
- True False True
- False True True
- False False False

Si userà l'operatore *Or* per determinare se una o l'altra di due condizioni sia vera. Per esempio

(ricavo < 5000) Or (profitto < 1000)

Il risultato sarà True se il valore di ricavo è minore di 5.000 oppure il valore di profitto è minore di 1.000

L'operatore **Not** effettua la negazione logica, cioè inverte il valore dell'unico operando. Il risultato quindi è True se l'operando è False e False se l'operando è True.

Chiudi questa finestra