11/11/2013

Informatica per la Finanza
Informatica per le Scienze Economiche
Informatica e Programmazione per il Turismo

Tipo String

- Una stringa è un insieme ordinato di caratteri
- In VB esistono
  - Stringhe di lunghezza fissa che possono contenere fino a 2^{16} caratteri
    - Si dichiarano con **`Dim`** ***`nome`*** **`As String *`** ***`N`***
      - N indica la lunghezza massima della stringa, oltre questa dimensione la stringa è troncata
      - Non possono essere assegnate a variabili di tipo **`Variant`**
  - <u>Stringhe di lunghezza variabile</u>
    - Si dichiarano con **`Dim`** ***`nome`*** **`As String`**

Tipo String

```
Sub usoString()
  Dim stFix As String * 5
  Dim stVar As String

  stFix = "ciaociao"
  stVar = "ciaociao"

  MsgBox (stFix)
  MsgBox (stVar)
End Sub
```


Tipo String

- Per unire (concatenare) due stringhe si usa l'operatore &
  - Si potrebbe usare anche l'operatore + ma in certi contesti crea ambiguità con l'operatore di somma

```
Cells(1, 3) = Cells(1, 1) & Cells(1, 2)
```

Tipo String

- Per manipolare le stringhe esistono diverse funzioni:
  - Confrontare (***StrComp***)
  - Convertire (***StrConv***, `Format`, `LCase`, `UCase`)
  - Creare stringhe di caratteri ripetuti (***Space***, ***String***)
  - Trovare la lunghezza di una stringa (***Len***)
  - Allineare una stringa (`LSet`, `RSet`)
  - Manipolare Stringhe (***InStr***, `Left`, `Mid`, `Right`, ***Replace***, `LTrim`, `RTrim`, `Trim`)
  - Manipolare i valori ASCII e ANSI (`Asc`, `Chr`)

Tipo String

Nella modalità `vbTextCompare` non si distingue fra maiuscolo e minuscolo. I caratteri accentati sono considerati diversi da quelli non accentati ed hanno una posizione diversa a seconda del tipo di accento!

`StrComp (`*`str1, str2, modo`*`)`

- Confronta ***str1*** con ***str2*** come specificato da ***modo***
  - ***modo*** (facoltativo) può valere
    - **`vbUseCompareOption`** confronta secondo l'impostazione di **`Option Compare`**.
      - » **`Option Compare`** ***`valore`***
        definita a livello di modulo ***`valore`*** può essere **`Binary`** (default) o **`Text`**
    - **`vbBinaryCompare`** confronto binario.
    - **`vbTextCompare`** confronto di testo.
- Restituisce

```
 -1  str1 < str2
  0  str1 = str2
  1  str1 > str2
```

Null se una delle due stringhe è Null

Tipo String

`StrConv(`*`stringa`*`,`*`conversione`*`)`

- Converte *`stringa`* secondo le impostazioni specificate in *`conversione`* (le opzioni di conversione in alcuni casi possono essere sommate per ottenere più effetti)
- ***`conversione`*** può valere ad esempio:
  - **`vbUpperCase`** converte i caratteri in maiuscolo
    - » Equivale all'uso di **`UCase(`*`stringa`*`)`**
  - **`vbLowerCase`** converte i caratteri in minuscolo
    - » Equivale all'uso di **`LCase(`*`stringa`*`)`**
  - **`vbProperCase`** converte in maiuscolo il primo carattere di ogni parola

Tipo String

- Alcune funzioni permettono di scrivere stringhe con caratteri ripetuti:
  - **`Space(`*`numero`*`)`**: restituisce una stringa con ***`numero`*** spazi
  - **`String(`*`numero`*`,`*`carattere`*`)`**: restituisce una stringa con ***`carattere`*** ripetuto ***`numero`*** volte
    - Carattere può essere una costante o un numero che rappresenta la posizione del carattere nella tabella ASCII
- Per conoscere la lunghezza di una stringa in caratteri si usa
  - **`Len(`*`stringa`*`)`**
    - **`Len(`*`nomeVariabile`*`)`** in generale restituisce il numero di byte con cui è rappresentata la variabile

Esempio

- Scrivere una stringa formata da:
  - 5 volte il carattere A
  - 8 spazi
  - 3 volte il carattere 1

```
Ris=String(5, "A") & Space(8) & String(3, "1")
```


Tipo String

`InStr(inizio,str1,str2,modo)`

- Cerca in ***`str1`*** le occorrenze di ***`str2`*** a partire dalla posizione ***`inizio`*** di ***`str1`*** confrontando in base al valore di ***`modo`*** (facoltativo)
  - **`vbUseCompareOption`**, **`vbBinaryCompare`**, **`vbTextCompare`**
- Restituisce i seguenti valori

0	se non è trovata corrispondenza
>0	punto di inizio della corrispondenza
`Null`	se str1 o str2 valgono Null

Esercizio

- Data una stringa X verificare quante volte compare in essa la stringa Y
- X |rosa bella rosetta bellina|
- Y |ros|

	occorrenze
InStr(1,X,Y) => 1	1
InStr(4,X,Y) => 12	2
InStr(15,X,Y) => 0	2

Esercizio

```
Function contaVolte(s1 As String, s2 As String) _
                                        As Integer
  Dim pos As Integer, lS2 As Integer

  lS2 = Len(s2)
  pos = 1
  contaVolte = 0
  While (InStr(pos, s1, s2) <> 0)
    contaVolte = contaVolte + 1
    pos = InStr(pos, s1, s2)
    pos = pos + lS2
  Wend

End Function
```

Esercizo (sol Alternativa)

```
Option Explicit
Option Base 1
Function volte(X As String, Y As String)
  Dim pos As Integer, ris As Integer

  pos = 1
  Do
    ris = InStr(pos, X, Y)
    pos = ris + Len(Y)
    volte = volte + 1
  Loop While (ris <> 0)
  volte = volte - 1
End Function

Sub usoString()
  ' A1 stringa in cui cercare
  ' A2 cosa cercare

  Range("C3") = volte(Range("a1").Value, Range("a2").Value)

End Sub
```


Tipo String

- Le seguenti funzioni restituiscono porzioni della stringa argomento
  - **Left(*stringa, N*)**
    - Restituisce i primi N Caratteri di stringa
  - **Right(*stringa, N*)**
    - Restituisce gli ultimi N caratteri di stringa
  - **Mid(*stringa, partenza, N*)**
    - Restituisce i primi N caratteri di stringa a partire dalla posizione partenza; se N è omesso restituisce la stringa a partire dalla posizione partenza

Tipo String

Operatore **`Like`**

`risultato` **`=`** *`stringa`* **`Like`** *`criterio`*

`stringa` è confrontata con `criterio`

Il valore in `risultato` è:

- **`True`** se vi è corrispondenza con `criterio`
- **`False`** se non vi è corrispondenza con `criterio`
- **`Null`** se stringa o criterio valgono `Null`.

- Il confronto è fatto sulla base del valore di `Option Compare`
  - **`Binary`** (default)
    - » fatto in base alla posizione del carattere nell'alfabeto in uso
  - **`Text`**
    - » Fatto in base al valore alfabetico dei caratteri (maiuscolo e minuscolo indifferente, ma i caratteri accentati seguono quelli non accentati)

Tipo String

`criterio` può contenere dei metacaratteri:

`?`	Un carattere qualsiasi
`*`	Zero o più caratteri qualsiasi
`#`	una singola cifra
`[`*`elencoCar`*`]`	qualsiasi carattere in `elencoCar`
`[!`*`elencoCar`*`]`	qualsiasi carattere non in `elencoCar`

- `elencoCar` contiene un elenco di caratteri fra cui (non) trovare corrispondenza. Può contenere
  - » Intervalli di caratteri consecutivi (il primo e l'ultimo elemento della sequenza son separati dal segno **`-`**)
  - » Il carattere **`–`** deve essere il primo o l'ultimo dell'elenco se non utilizzato in un intervallo (per evitare confusioni)
  - » Il carattere **`!`** Se non è in prima posizione mantiene il suo significato
  - » I metacaratteri **`[ ? * #`** debbono essere racchiusi fra parentesi quadre la parentesi quadra chiusa **`]`** NON può essere usata in un elenco di caratteri

Esempio

- verificare se la stringa in A1 contiene la sequenza baac

  … = Range("A1") Like "*baac*"

- verificare se la stringa contiene parole che:
  - iniziano con **c** sono seguite da **a o u** quindi da **s** e infine da **a o**

    … Like "*c[aou]s[ao]*"

Esempio

```
Sub prova()
  Dim s1 As String, s2 As String, dove As Integer


  's1 = "*AK"
  'criterio tre caratteri qualunque ma non a b c
  'Range("C1") = s1 Like "[!a-c][!a-c][!a-c]"

  'criterio: tutte le stringhe che iniaziano con ca
  s1 = "cosa di montagna"
  Range("C2") = s1 Like "ca*"

  'criterio: tutte le stringhe che iniziano con ca quindi
  ‘ hanno un carattere qls e quindi una a
  s1 = "casa"
  Range("C2") = s1 Like "ca?a*"

End Sub
```

Esercizio

- Scrivere la funzione che ha come argomento una stringa e restituisce uno dei seguenti valori a seconda del valore del parametro
  1. Corrispondono ad un codice fiscale di una persona fisica
     - 16 caratteri
       - » 6 alfabetici
       - » 2 cifre
       - » 1 lettera
       - » 2 cifre
       - » 1 lettera
       - » 3 cifre
       - » 1 lettera
  2. Ad una partita IVA
     - 11 cifre
  3. A nessuno dei casi precedenti

Tipo String

- La funzione **`Replace()`** permette di sostituire parti di una stringa con un'altra restituisce la stringa con le sostituzioni

`Replace(expr, trova, sost, da, tot, modo)`

- **`expr`** indica dove cercare
- **`trova`** indica cosa cercare
- **`sost`** indica con cosa sostituire
- **`da`** posizione da cui partire (se omessa 1)
- **`tot`** quante sostituzione fare (se omessa -1 ovvero tutte)
- **`modo`** tipo di confronto
  - **`vbUseCompareOption`** usa valore di `Option`
  - **`vbBinaryCompare`** confronto binario
  - **`vbTextCompare`** confronto testuale

Esempio

```
Option Compare Text
Sub rimpiazza()
   Range("A1") = Replace("rosa rossa rosa gialla", _
                                 "os", "CILIEGIA", 1, 1)
   Range("A2") = Replace("rosa rossa rosa gialla", _
                                 "os", "CILIEGIA", 1)
   Range("A3") = Replace("rosa rossa rosa gialla", _
                                 "os", "CILIEGIA", 5)
End Sub
```

	A
1	rCILIEGIAa rossa rosa gialla
2	rCILIEGIAa rCILIEGIAsa rCILIEGIAa gialla
3	rCILIEGIAsa rCILIEGIAa gialla
4	

Tipo String

– La funzione
- **Asc(*stringa*)** restituisce il codice del primo carattere di `stringa` (valore intero)
- **Chr(*intero*)** restituisce la stringa che contiene il carattere corrispondente al valore di intero

```
Sub rimpiazza()
   Range("A1") = Asc("rosa rossa rosa gialla")
   Range("A2") = Chr(115)
End Sub
```

	A
1	114
2	s

Esempio

```
Option Compare Binary
Function myStrComp(s1 As String, _
              s2 As String) As Integer
  Dim i As Integer, lun As Integer

  If Len(s1) > Len(s2) Then
    lun = Len(s2)
  Else
    lun = Len(s1)
  End If

  i = 1
  While i <= lun
    If Asc(Mid(s1, i, i)) <> _
                Asc(Mid(s2, i, i)) Then
       If (Asc(Mid(s1, i, i)) < _
                Asc(Mid(s2, i, i))) Then
          myStrComp = -1
       Else
          myStrComp = 1
       End If
       i = lun + 2
    End If
    i = i + 1
  Wend
```

```
If (i = lun + 1) Then
            'le due stringhe sono diverse
     myStrComp = 0
  End If
End Function

Sub prova()
  Dim s1 As String, s2 As String


  s1 = "Rosa"
  s2 = "Cosa"

  Range("A1") = StrComp(s1, s2)
  Range("A2") = myStrComp(s1, s2)

End Sub
```


Tipo String

- Le seguenti funzioni eliminano spazi
  - **`LTrim(`*`stringa`*`)`**
    - Elimina gli spazi iniziali di `stringa`
  - **`RTrim(`*`stringa`*`)`**
    - Elimina gli spazi finali di `stringa`
  - **`Trim(`*`stringa`*`)`**
    - Elimina gli spazi iniziali e finali di `stringa`

Tipo Date

- Serve per gestire le date che vengono memorizzate come numeri decimali di 8 byte
  - Le date fra
    - 01/01/100 e 30/12/1899 sono numeri negativi
    - 31/12/1899 e 31/12/9999 sono numeri positivi
    - 0 rappresenta il giorno 0.0.0000
    - Se si usano i numeri decimali la parte decimale rappresenta l'ora
  - Possono essere dei *valori letterali* compresi fra # che somigliano ad una data
    - **`#15 Apr 2009#`** **`#12/5/2007#`**

Tipo Date

- Le principali funzioni:
  - Dai valori dell'orologio di sistema:
    - **`Date()`**: restituisce gg/mm/aaaa
    - **`Time()`**: restituisce hh:mm:ss
    - **`Now()`**: gg/mm/aaa hh:mm:ss
  - Restituire date:
    - **`DateSerial(`*`aaaa,mm,gg`*`)`**
    - **`DateValue(`*`stringaConData`*`)`**
  - Restituire orari:
    - **`TimeSerial(`*`hh,mm,ss`*`)`**
    - **`TimeValue(`*`stringaConOrario`*`)`**

Tipo Date

DatePart(*interval, data, primoGset,primaSanno*)

- Restituisce una parte delle informazioni contenute in data
- ***interval*** indica quale parte interessa (valore fra **"**):
  - **yyyy** Anno **q** Trimestre **m** Mese
  - **y** Giorno dell'anno **d** Giorno
  - **w** Giorno della settimana **ww** Settimana
  - **h** Ora **n** Minuti **s** Secondi
- ***primoGset*** indica quale è il primo giorno della settimana
  - **vbUseSystem** impostazione API **vbSunday** Do (predefinita)
  - **vbMonday** Lu **vbTuesday** Ma **vbWednesday** Me
  - **vbThursday** Gi **vbFriday** Ve **vbSaturday** Sa
- ***primaSanno*** indica da quando si iniziano a contare le settimane di un anno
  - **vbUseSystem** impostazione API
  - **vbFirstJan1** settimana 1 gennaio (predefinita)
  - **vbFirstFourDays** prima settimana di almeno quattro giorni
  - **vbFirstFullWeek** Inizia con la prima settimana completa dell'anno.

Esercizio

- Le celle da A1 a F1 contengono delle date. Scrive il codice necessario per
  - Verificare se la cella contiene effettivamente una data
  - Scrivere nella riga sottostante il giorno della settimana o nulla se non ha la forma di una data

Esercizio

```
Sub giornoSettimana()
   Dim gS(7)  As String
   Dim el As Variant, i As Integer, g As Integer

   gS(1) = "lunedì"
   gS(2) = "martedì"
   gS(3) = "mercoledì"
   gS(4) = "giovedì"
   gS(5) = "venerdì"
   gS(6) = "sabato"
   gS(7) = "domenica"

   i = 1
   For Each el In Range("A1:F1")
     If IsDate(el.Value) Then
       g = DatePart("w", el.Value, vbMonday)
       Cells(2, i).Value = gS(g)
     End If
     i = i + 1
   Next
End Sub
```


Tipo Date

- Alle date si possono aggiungere dei giorni con l'operatore +
- Più corretto è usare le funzioni
  - **DateAdd(*interval*, periodi, *data*)**
    - Somma il numero *periodi* a *data* aumentando del valore di *interval* specificato
      - *interval:* ha gli stessi formati visti in precedenza
  - **DateDiff(*interval*, *data1*, _ *data2*,*primoGset*,*primaSanno*)**
    - Restituisce i periodi trascorsi fra due date specificate

Tipo Date

– Nel foglio di lavoro la colonna A contiene una data, la colonna B l'intervallo da modificare la colonna C di quanto modificare. Scrivere nella colonna D la nuova data

	A	B	C	D
1	31/12/2004	ww	5	04/02/2005
2	23/10/2009	q	3	23/07/2010
3	12/10/2013	d	30	11/11/2013
4	10/10/2000	yyyy	12	10/10/2012

```
Sub prova()
  Dim d1 As Date, riga As Integer
  riga = 1
  Do While Not IsEmpty(Cells(riga, 1))
    Cells(riga, 4) = DateAdd(Cells(riga, 2), _
                  Cells(riga, 3), Cells(riga, 1))
    riga = riga + 1
  Loop
End Sub
```


Lettura e scrittura file

- VB presenta diverse possibilità di accesso ai file ma si illustrerà solo quella per la gestione dei file di testo sequenziali
- Si ricorda che è possibile esportare o importare in Excel file separati da virgola (**C**omma **S**eparated **V**alues) direttamente

Lettura e scrittura file sequenziali

- Questo genere di file sono utili come meccanismo di deposito e scambio di dati
- Le fasi in cui si suddividono le operazioni sono:
  - Apertura (**`Open`**)
  - Lettura (**`Input`**) o scrittura (**`Write`**)
  - Chiusura (**`Close`**)

Lettura e scrittura file sequenziali

Apertura file

`Open` *`percorso`* **`For`** *`modalità`* **`As`** *`#nfile`*

- *`percorso`* indica il percorso compreso il nome del file da aprire
  - Per ottenere il percorso della directory dove si trova il foglio di lavoro usare:
    - » **`ThisWorkbook.Path`**
- *`modalità`* indica come si accederà al file:
  - **`Input`**, **`Output`**, **`Append`**
- *`nfile`* numero fra 1 e 511 che identificherà il file aperto

Lettura e scrittura file sequenziali

Scrittura

`Write #nfile, elencoDati`

- *`Nfile`*: numero del file da aprire
- *`elencoDati:`* elenco dei dati da scrivere separato da virgola
  - I dati decimali sono sempre scritti separando col punto la parte decimale
  - Per i tipi **`Boolean`** viene scritto **`#True#`** o **`#False#`**
  - Per i tipi **`Date`** si usa il formato **`#aaaa-mm-gg#`**
  - Se omesso scrive il carattere di a capo
- Scrive *`elencoDati`* e poi va a capo

Lettura e scrittura file sequenziali

```
Sub usoFileScritt()
  Dim i As Integer, j As Integer
  Dim b As Boolean
  Dim d As Date

  Open ThisWorkbook.Path & "\p.txt" For Output As #1

  For i = 1 To 6
       Write #1, Cells(i, 1).Value, "  ", Cells(i, 2).Value
  Next

  b = True
  Write #1, b

  d = Now()
  Write #1, d

  Close #1
End Sub
```

Esempio

```
Option Explicit
'ricopia dalle celle non vuote in a1:c3
'sul file dati.txt
Sub x()
   Dim r As Range

   Open ThisWorkbook.Path & "\dati.txt" For Output As #1

   For Each r In Range("A1:C3")
      If Not IsEmpty(r) Then
        If IsNumeric(r) Then
          Write #1, r.Value
        End If
      End If
   Next

   Close (1)
End Sub
```


Esempio

- Scrivere il codice che dato il foglio di lavoro corrente che contiene dalla posizione A1 alla posizione D2 dei numeri interi scrive sul file pari.txt i valori pari, sul file dispari.txt i valori dispari
- Attenzione x Mod 2 restituisce -1 se X è un numero dispari negativo (fare il test su 0)

Esempio

```
Option Explicit
Sub usaFile()
  Dim Y   As Variant      'deve essere per forza variant o Range

  Open ThisWorkbook.Path & "\pari.txt" For Output As #1
  Open ThisWorkbook.Path & "\dispari.txt" For Output As #2

  For Each Y In Range("A1:D2")
     If Y Mod 2 = 0 Then
       Write #1, CInt(Y)
     Else
       Write #2, CInt(Y)
     End If
  Next

  Close #1
  Close #2
End Sub
```


Esempio

```
Function isInteger(val As Double) As Boolean
  If CInt(val) = val Then
    isInteger = True
  Else
    isInteger = False
  End If
End Function

Sub x()
   Dim r As Range
   Open ThisWorkbook.Path & "\pari.txt" For Output As #1
   Open ThisWorkbook.Path & "\dispari.txt" For Output As #2

   For Each r In Range("A1:D2")
     If Not IsEmpty(r) And IsNumeric(r) Then
       If isInteger(r.Value) Then
         If r.Value Mod 2 = 0 Then
           Write #1, r.Value
         Else
           Write #2, r.Value
         End If
       End If
     End If
   Next

   Close (1)
   Close (2)
End Sub
```

In questa versione ricopia solo i numeri interi

Lettura e scrittura file sequenziali

Lettura

`Input #nfile, elencovariabili`

- *`nfile`*: numero identificativo del file
- *`elencovariabili`*: elenco delle variabili in cui scrivere i valori (usare solo variabili)
- Per verificare la fine del file si usa la funzione

  `EOF(`*`nfile`*`)`

  – **`True`** se fine file, **`False`** altrimenti

Lettura e scrittura file sequenziali

```
Sub usoFileLett()
  Dim i As Integer, j, k, h

  Open ThisWorkbook.Path & "\p.txt" For Input As #1

  i = 0
  Do While Not EOF(1)
       Input #1, k, j, h
       Cells(i + 10, 10) = k
       Cells(i + 10, 11) = h
       i = i + 1
  Loop

  Close #1
End Sub
```

Esempio

- leggere i valori contenuti nel file `document.txt` e scrivere nella colonna A i valori interi in quella B i decimali

```
Sub usoFileScritt()
  Dim el As Double
  Dim rgA As Integer, rgB As Integer

  Open ThisWorkbook.Path & "\Document.txt"  _
                          For Input As #1

  rgA = 1:  rgB = 1
  While Not EOF(1)
     Input #1, el
     If CInt(el) = el Then 'intero
       Cells(rgA, 1) = el
       rgA = rgA + 1
     Else
       Cells(rgB, 2) = el
       rgB = rgB + 1
     End If
  Wend

  Close (1)
End Sub
```


Esempio

```
' il file dati.txt contiene dei numeri.
' scrivere nella colonna A i valori positivi, nella colonna B quelli negativi.
‘ Le colonne non devono contenere celle vuote fra un valore ed un altro
Option Explicit
Sub q()
  Dim colA As Integer, colB As Integer, val As Double

  Open ThisWorkbook.Path & "\dati.txt" For Input As #1
  colA = 1: colB = 1
  While Not EOF(1)
    Input #1, val
    If val > 0 Then
      Cells(colA, 1) = val
      colA = colA + 1
    Else
      Cells(colB, 2) = val
      colB = colB + 1
    End If
  Wend
  Close (1)
End Sub
```

Esempio

- Scrivere il programma che aggiunge in coda al file dati.txt il contenuto della colonna A del foglio di lavoro

```
Sub usoFileScritt()
  Dim val As Variant
  Dim rg As Integer
  Dim fr As String, st As String

  Open ThisWorkbook.Path & _
          "\dati.txt" For Append As #1

  rg = 1
  While Not IsEmpty(Cells(rg, 1))
    Write #1, Cells(rg, 1).Value
    rg = rg + 1
  Wend

  Close (1)
End Sub
```