21/10/2013

Informatica per la Finanza

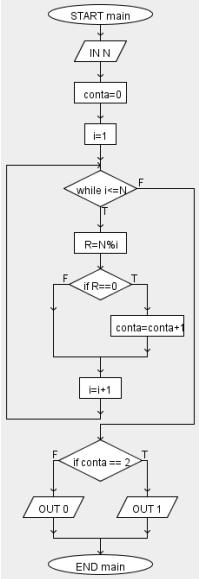
Informatica per le Scienze Economiche

Informatica e Programmazione per il Turismo

Algoritmo: esempi

- Dato un numero Naturale diverso da 0 N restituire 1 se il numero è primo 0 se non è primo
  - Un numero naturale N è primo sse è divisibile solo per 1 o per N
    - 1 non è un numero primo
  - Per verificare la condizione basta contare quante volte il numero N è divisibile per tutti i numeri da 1 ad N

Algoritmo: esempi



N	conta	i	R
7	0	1	

```
PROG main
    IN N
    ASSIGN conta=0
    ASSIGN i=1
    WHILE i<=N
        ASSIGN R=N%i
        IF R==0
            ASSIGN conta=conta+1
        ELSE //if R==0
        END IF //R==0
        ASSIGN i=i+1
    END WHILE //i<=N
    IF conta == 2
        OUT 1
    ELSE //if conta == 2
        OUT 0
    END IF //conta == 2
END PROG //main
```


Algoritmo: esempi

- Dato un numero N
  - Calcolare il suo valore assoluto e scriverlo in A
  - restituire 1 se il numero è intero, 0 altrimenti
- Se un numero N è positivo si può provare, usando una variabile X partendo da 0, ad incrementarla finché raggiunge o supera il numero N; se X è uguale ad N allora N è intero, altrimenti non lo è

Algoritmo: esempi

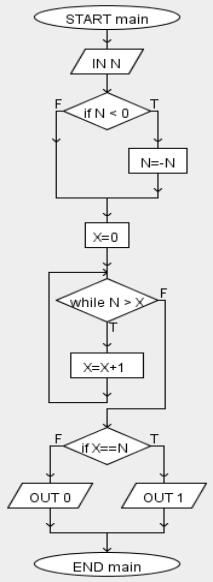
N	X
6,3	0
6,3	1
6,3	2
6,3	3
6,3	4
6,3	5
6,3	6
6,3	7

=> 0

N	X
6	0
6	1
6	2
6	3
6	4
6	5
6	6

=> 1

Algoritmo: esempi



N	X
-4.3	
4.3	
4.3	0
4.3	1
4.3	2
4.3	3
4.3	4
4.3	5

=> 0

```
PROG main
    IN N
    IF N < 0
        ASSIGN N=-N
    ELSE //if N < 0
    END IF //N < 0
    ASSIGN X=0
    WHILE N > X
        ASSIGN X=X+1
    END WHILE //N > X
    IF X==N
        OUT 1
    ELSE //if X==N
        OUT 0
    END IF //X==N
END PROG //main
```

VBA

Bibliografia

- **S. Chapra, Introduction to VBA for Excel (2nd Edition), Pearson**
- Guccini, Excel 2007 Macro, Apogeo
  - Riferimento per il linguaggio
- Rota, Visual Basic, Hoepli (ed. per le scuole superiori)
  - Specifico per Visual Basic non per VBA ma per i principianti della programmazione può essere utile per apprendere i fondamenti della programmazione.

Caratteristiche del linguaggio

- Visual Basic for Applications
  - linguaggio di programmazione derivato da Visual Basic per offrire ai prodotti Microsoft Office un ambiente di programmazione
  - linguaggio di programmazione
    - **Imperativo** (come C/C++, Pascal)
    - **guidato dagli eventi** (*event driven*)
    - **orientato agli oggetti** (*object oriented*)
  - Il codice scritto in VBA funziona solo all'interno del programma Office in cui è utilizzato

VBA: caratteristiche

- Orientato agli eventi:
  - L'interazione avviene quando si compie un evento (come avviene in una pagina WEB)
    - Schiacciare un tasto/bottone,
    - Entrare/uscire da una casella di testo
    - ...
- Orientato agli oggetti:
  - Il programma si compone di tante parti (oggetti) collegate fra loro
  - Ogni oggetto ha delle proprietà che sono peculiari
  - Gli oggetti sono inseriti in una gerarchia

VBA: caratteristiche

- Oggetti fondamentali:
  - File Excel o Cartella di Lavoro (**`Workbook`**, **`Wb`**)
    - Ogni file Excel aperto è un Wb se ve ne è più di uno sono numerati consecutivamente
    - Il Wb attivo si chiama **`ThisWorkbook`**
  - Ogni Wb contiene dei fogli di lavoro (**`Worksheet`**, **`Ws`**)
    - I Ws sono numerati consecutivamente

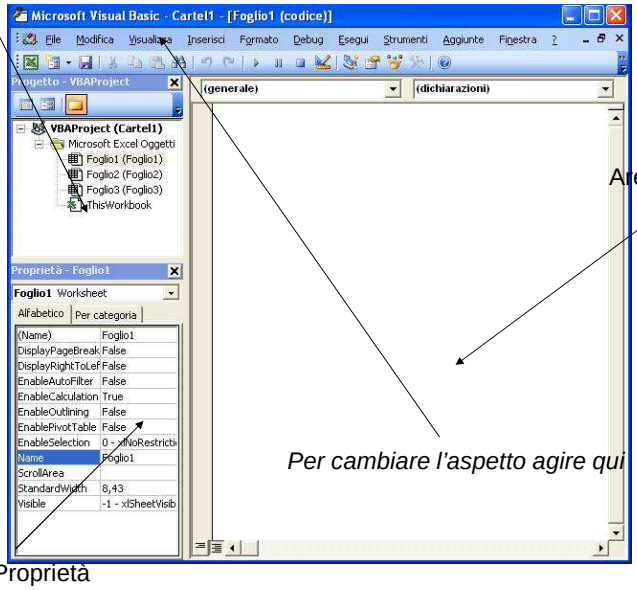
Ambiente di lavoro

- Per attivare l'ambiente di lavoro in qualsiasi versione di Excel:

  `ALT F11`

- compare una schermata simile a quella della pagina successiva (per modificare l'aspetto bisogna selezionare il menù visualizza)

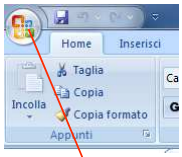
Area Progetto



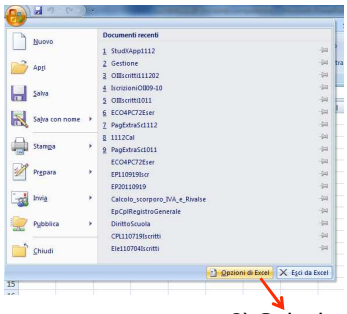
Area Codice

Area Proprietà

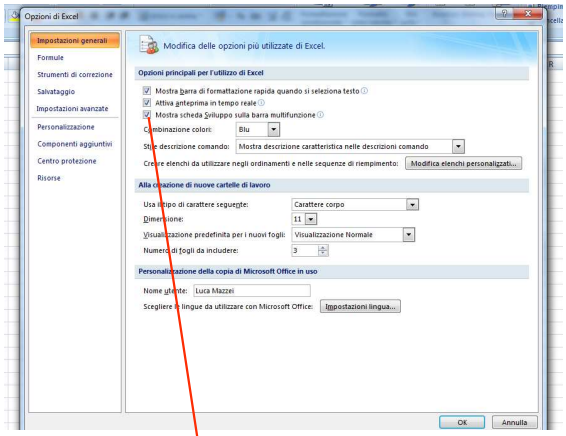
PER LE NUOVE VERSIONI DI EXCEL PER FAR APPARIRE SVILUPPO



1) Selezionare



2) Selezionare



3) Spuntare

VBA: un primo esempio

- Creare un file Excel di nome *primo*
  - Scrivere in A2 la parola ***esempio***
  - Scrivere in B2 il valore ***5***
- Nell'ambiente schiacciare assieme i due tasti **ALT F11**
  - si apre l'ambiente VBA
  - Nell'*Area Progetto* schiacciare due volte sulla voce **Foglio1** dell'elenco **Microsoft Excel Oggetti**

VBA: un primo esempio

- Nell'Area Codice scrivere le seguenti istruzioni:

```
Sub primoEsempio()
 Range("A3") = ThisWorkbook.Name
 Range("A5").Value = Worksheets(1).Name
Range("B2").Characters.Font.Name="Arial Black"
End Sub
```

- Per eseguire tasto **F5**

VBA: esempio commento

- **`Range("A3") = ThisWorkbook.Name`**
  - Inserisce nella cella A3 il nome della cartella di lavoro (`ThisWorkbook.Name`)
    - `Name` è una proprietà dell'oggetto `ThisWorkbook`
- **`Range("A5").Value = Worksheets(1).Name`**
  - Inserisce nella cella A5 il nome del foglio di lavoro in cui è contenuta
    - `Value` indica la proprietà dell'oggetto `Range("A5")` che contiene il valore della cella (può essere omessa come nella riga precedente)
    - `Name` è una proprietà dell'oggetto `Worksheets(1)`
      - Si noti un foglio di lavoro si indica con `Worksheets` seguito fra parentesi tonde dalla sua posizione
      - `Name` è una proprietà anche di Worksheets ma non si confonde con l'omonima di Workbook perché è preceduta dall'indicazione dell'oggetto cui si riferisce

VBA: esempio commento

- **`Range("B2").Characters.Font.Name="Arial Black"`**
  - Modifica il tipo di fonte della cella B2
- Gerarchia degli oggetti dell'esempio

Workbook
  Worksheet
    Celle (riferita con: `Range( )`)
      Characters (il carattere nella cella)
        Font (la fonte usata)
          Name (il nome della fonte)

VBA

- Ricapitolando
  - Per riferirsi ad una cella si usa
    - **`Range("`*`nomeCella`*`")`**
      - In alternativa si può usare anche la notazione

        `Cells(`*`riga,colonna`*`)`
        - » dove *`riga`* e *`colonna`* sono coordinate numeriche
        - » In questo caso può essere utile visualizzare le coordinate delle colonne con i numeri e non con le lettere (*Strumenti->Opzioni->Generale* spuntare la casella *Stile di Riferimento R1C1*)
  - Ogni cella ha diverse proprietà che possono essere modificate

VBA Esempio con Cells

```
Sub primoEsempio()
 Worksheets(2).Cells(3, 1) = ThisWorkbook.Name
 Worksheets(2).Cells(5, 1) = Worksheets(1).Range("A5").Value
 Worksheets(2).Cells(2, 2) = Worksheets(1).Range("B2").Value
 Worksheets(2).Cells(2, 2).Characters.Font.Name = "Courier New"
End Sub
```

- Si noti che si lavora sul secondo foglio, prendendo alcuni dati dal primo

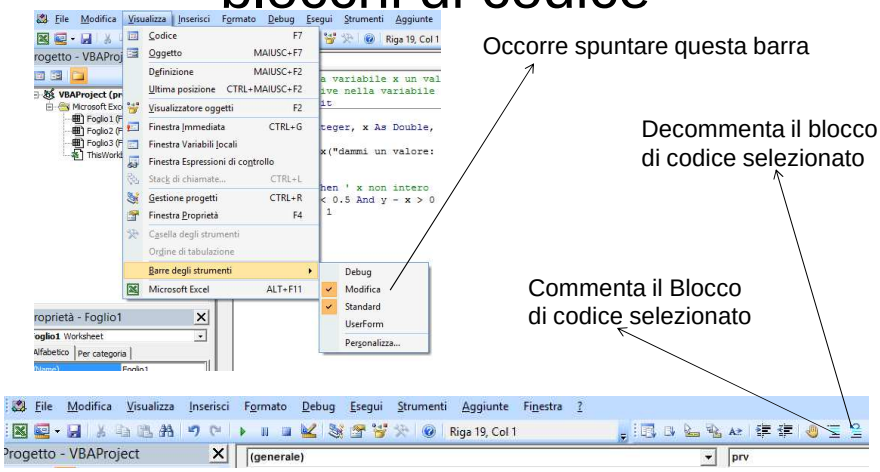
Altre osservazioni

- Si noti che abbiamo sempre racchiuso il codice in questo modo:

```
Private Sub Nome()
  :
End Sub
```

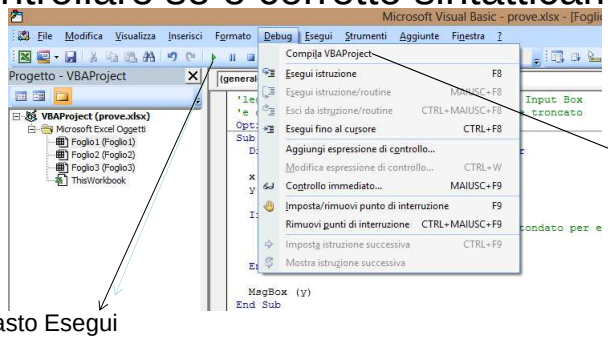
  - Abbiamo creato una procedura (***Sub***routine)
  - Denominata *Nome*
  - La parola chiave ***Private*** non è obbligatoria e non è stata usata
- I commenti si creano premettendo il carattere **'**
  - I commenti terminano con la fine della riga

Commentare/Decommentare blocchi di codice



Compilare in VBA

- Prima di eseguire un programma è meglio controllare se è corretto sintatticamente



Selezionare questa voce

Tasto Esegui
Corrisponde a schiacciare F5

Compilare in VBA

– Con questa operazione si può verificare se il codice è corretto sintatticamente
  - Selezionando il tasto Esegui prima di eseguire il programma questa operazione è effettuata automaticamente
  - Se nel codice associato ad un foglio c'è un errore di sintassi allora non è possibile compilare nessun codice in nessun foglio

Interazione con l'utente: MsgBox

- Serve per inviare un messaggio all'utente
- Ha due forme:
  - Funzione:

    MsgBox(*testo, tipoCasella, titolo*)
    - ***testo:*** stringa con il messaggio (fra **""**)
    - ***tipoCasella:*** tipo della acsella (vedi poi; facoltativo)
    - ***titolo***: stringa con il titolo della casella (facoltativo)
    - Restituisce un valore che va messo in una variabile
  - Istruzione

    MsgBox *testo, tipoIcona, titolo*
    - Non restituisce alcun valore

Interazione con l'utente: MsgBox

- ***tipoCasella*** può avere diversi valori fra cui:

  vbOKOnly
  vbCritical
  vbQuestion
  vbExclamation
  vbInformation
- Quando si salta un valore nella lista dei parametri va comunque messa la virgola.

  MsgBox(*testo,, titolo*)

  MsgBox *testo,, titolo*

Interazione con l'utente: MsgBox

- Studiare il comportamento del seguente codice sostituendo i diversi valori per ***TipoCasella*** (osservare come varia il valore restituito):

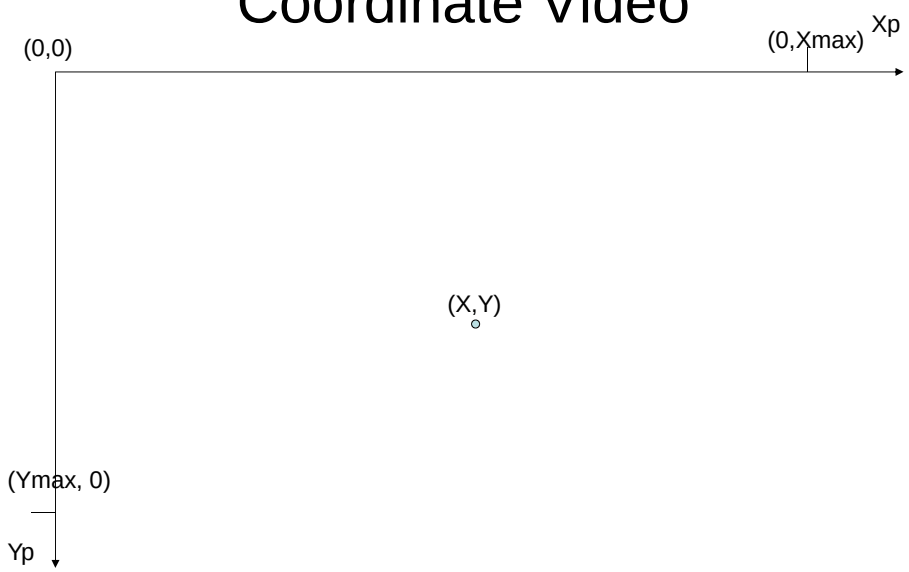
```
Sub interazione()
  ris = MsgBox("Ciao!", , "prova")
  Range("A1") = ris
  MsgBox "Bye!", vbCritical, "prova"
  Range("A1") = ""
End Sub
```


Interazione con l'utente: InputBox

- Serve per richiedere dei dati all'utente
- E' una funzione che restituisce una stringa con il valore inserito dall'utente

  InputBox(*Testo, Titolo, Default, Xp, Yp*)
  - **Testo**: indicazione della richiesta
  - **Titolo**: intitolazione casella (facoltativo)
  - **Default**: valore restituito in caso di nessuna scelta (facoltativo)
  - **Xp, Yp**: coordinate della casella sul video espressa in punti tipografici (1/72 di pollice = 0,35 mm) rispetto all'angolo superiore sx dello schermo (facoltative)

Coordinate Video



Interazione con l'utente: InputBox

- Provare ad eseguire il seguente esempio con diversi valori:

```
Sub interagisci()
ris = InputBox("valore: ", "Casella", "nulla", 10, 20)
   Range("A1") = ris
End Sub
```

Tipi di Dati

- Il tipo di dato indica il modo di codificare i dati e le operazioni che si possono compiere su questi
- Il nome del tipo di dato può richiamare nomi di entità matematiche ma non è detto che gli intervalli e le proprietà dell'ente matematico coincidano con quelle del tipo di dato
  - Finitezza delle rappresentazioni che deriva dalla memoria limitata di un elaboratore:
    - Interi: complemento a due a n bit
    - Reali: virgola mobile ad n bit

Variabili

- Una variabile è una zona di memoria atta a contenere dei valori
- Ad ogni variabile è associato un Tipo (se non dichiarato è **`Variant`**)
- Le variabili si dichiarano in questo modo:

  `Dim` *`nomeVariabile`* **`As`** *`TipoDiDato`*

  - le variabili possono non essere dichiarate prima di essere usate

Principali Tipi di VBA

- **Byte** 0-255
- **Boolean** True False
- **Integer** -32.768 ... 32.767
- **Long** -2.147.483.648... 2.147.483.647
- **Single**
  -3,403 10^{38} ... -1.401 10^{-324}
  1,401 10^{-324} ... 3,403 10^{38}
- **Double**
  -1,798 10^{308} ... -4,941 10^{-324}
  4,941 10^{-324} ...1,798 10^{308}

Principali Tipi di VBA

- **Currency**
  -922.337.203.685.477,5808 ... 922.337.203.685.477,5807
- **Date** 01/01/100 ... 31/12/9999
- **String** (sequenza ordinata di caratteri)
  - Possono essere:
    - Lunghezza variabile da **0** a 2^{31} caratteri.
    - Lunghezza fissa da **1** a 2^{16} caratteri.

Tipo Variant

- quando non viene dichiarato il tipo si assume che sia `Variant`
  - Per dati numerici come `Double`
  - Per dati non numerici come `String` di lunghezza variabile

Variabili

- Inserendo l'opzione **`Option Explicit`** all'inizio del modulo al di fuori di tutte le routine si obbliga il programmatore a dichiarare tutte le variabili
  - Nel seguito dichiareremo esplicitamente tutte le variabili
- Dopo ogni nome di variabile va indicato il tipo (se manca è assunto il tipo **`Variant`**), una dichiarazione simile:

```
Dim v1, v2 As Integer
```

  - Viene interpretata da VB come
    - **`v1`** di tipo **`Variant`** (si è omesso il tipo)
    - **`v2`** di tipo **`Integer`**

Costanti

- Sono variabili cui viene assegnato un valore che non può essere modificato
- Le costanti si dichiarano in questo modo:
  `Const` *`nomeVariabile`* **`As`** *`TipoDiDato`* **`=`** *`Valore`*
- In VBA esistono anche delle costanti predefinite

Espressione

- L'espressione è un insieme ordinato di variabili ed operatori
- Ogni espressione ha
  - Un valore dato dal risultato dell'espressione
  - Un tipo determinato dal tipo delle variabili e dagli operatori
- La forma generale è:

```
Risultato = Operando1 Operatore Operando2
```

  - L'operando (o termine) può essere a sua volta una espressione

Operatori

- Gli operatori sono dei simboli che rappresentano una operazione
- Ogni tipo di dato possiede un insieme di operatori che si possono applicare su di esso
- Gli operatori possono essere
  - Unari: si applicano solo ad un termine (-5)
  - Binari: si applicano su due termini (a + 6)
  - Ternari: si applicano su tre termini

Operatori (in ordine precedenza)

`^`	elevamento a potenza
`-`	negazione (operatore ad 1 operando)
`* /`	prodotto e divisione decimale
`\`	divisione intera (operandi decimali arrotondati)
`Mod`	resto divisione (operandi decimali arrotondati)
`+ -`	addizione e sottrazione
`&`	concatenazione di stringhe
`= <> > >= < <= Like Is IsNot`	operatori confronto
`Not`	negazione logica e bit a bit
`And`	and logico e bit a bit
`Or XOr`	or logico e bit a bit, Xor logico e bit a bit
`Imp`	implica

Operatori Booleani

A parità di precedenza: valutazione da sinistra a destra

Esistono altri operatori che non sono stati indicati

Ordine di valutazione

- 2^4+5*3*-1
- 2^4
- -1
- 5*3
- *
- +

Esempio

- Scrivere il codice VBA che dati i due lati A e B di un rettangolo ne calcola l'area. I valori sono letti dopo averli richiesti all'utente:

```
Option Explicit
Sub prova()
  Dim a As Double, b As Double, c As Double

  a = InputBox("dammi il primo lato: ")
  b = InputBox("dammi il secondo lato: ")
  c = a * b
  MsgBox ("area vale: " & c)
End Sub
```

Esempio

- Scrivere il codice VBA che dati i due lati A e B di un rettangolo ne calcola l'area. I valori sono letti dalle celle A1 ed A2 ed il risultato va scritto nella cella C1:

```
Option Explicit
Sub prova()
  Range("C1") = Range("A1") * Range("A2")
End Sub
```

In questo caso non è necessario usare delle variabili; le celle stesse fungono da variabili.

Operatori Booleani: tavole verità

X	Y	NOT X	X AND Y	X OR Y	X XOR Y	X Imp Y
T	T	F	T	T	F	T
T	F	F	F	T	T	F
F	T	T	F	T	T	T
F	F	T	F	F	F	T
						=>

T: True (Vero, valore numerico -1)

F: False (Falso, valore numerico 0)

X Imp Y = NOT X OR Y (Non è commutativo)

X XOR Y = (NOT X AND Y) OR (X AND NOT Y)

Operatori: note

Operatori Booleani:

- su espressioni di tipo booleano si comportano come operatori logici (vedi tavole verità)
  - Valutano sempre tutta l'espressione (a differenza di C/C++, Java, …)
- se hanno come argomento espressioni numeriche eseguono l'operazioni sui singoli bit che rappresentano il numero

Operatori booleani Bit a Bit

- Nell'uso come operatori bit a bit (quando sono applicate a dei numeri) il valore binario degli operandi viene confrontato bit a bit per produrre un nuovo valore in base all'operatore applicato:

```
A = 5        00000101
B = 16       00010000
A And B      00000000                0
A OR B       00010101                21
A Xor B      00010101                21
A Imp B      11111010                -6
```

- Si ricorda che i numeri interi sono rappresentati in complemento a 2, nel caso in esame supponendo A e B di tipo Byte, il complemento a due è a 8 bit
- In caso di una espressione che contenga un termine di tipo numerico ed uno di tipo booleano prevale il tipo numerico
  - Valore tipo booleano: 0 per F, -1 per T

Esempio

- 8 0000000000000100
- true 1111111111111111
- 1111111111111111
- -1

- 11111010 250 (-6)
- 00000110 6

Operatori booleani bit a bit

```
-1      11111…111
           AND
2       00000…010
        00000…010
```

Viene visualizzato 2

Operatori booleani bit a bit

```
-1      11111…111
            OR
 2      00000…010
        11111…111
```

Viene visualizzato -1

Operatori booleani bit a bit

```
-1      11111…111
           XOr
 2      00000…010
        11111…101
```

Viene visualizzato -2

Problemi con la lettura

- La lettura dei dati dal foglio elettronico impone il tipo della cella
- La lettura dei dati da InputBox restituisce un tipo String
- Esistono delle funzioni specifiche per convertire un'espressione in uno dei tipi di VB
  - In caso di impossibilità è generato un errore

Funzioni di conversione

`CBool(espr)`	**converte in `Boolean`**
`CByte(espr)`	**converte in `Byte`**
`CDate(espr)`	**converte in `Date`**
`CDbl(espr)`	**converte in `Double`**
`CInt(espr)`	**converte in `Int`**
`CLong(espr)`	**converte in `Long`**
`CSing(espr)`	**converte in `Single`**
`CStr(espr)`	**converte in `String`**
`CVar(espr)`	**converte in `Variant`**

Conversioni di Tipo

- Alcune funzioni che permettono di evitare errori di conversione

  `IsNumeric(`*`espr`*`)`

  > **`True`** se ***`espr`*** è compatibile con un numero (decimale o intero)

  `IsDate(`*`espr`*`)`

  > **`True`** se ***`espr`*** è compatibile con un formato data