TI-Nspire™
Guida di riferimento

La presente Guida è relativa alla versione 4.3 del software TI-Nspire™. Per ottenere la versione più aggiornata della documentazione, visitare il sito education.ti.com/guides.

Informazioni importanti

Salvo per quanto dichiarato espressamente nella licenza che accompagna un programma, Texas Instruments non rilascia alcuna garanzia, esplicita o implicita, incluse fra le altre le garanzie implicite di commerciabilità e di idoneità ad uno scopo particolare, per quanto riguarda programmi o materiali cartacei e rende disponibili tali materiali esclusivamente su base "tal quale." In nessun caso Texas Instruments è responsabile verso chicchessia di danni speciali, collaterali, incidentali o consequenziali in relazione con o derivanti dall'acquisto o dall'uso di questi materiali, e la sola ed esclusiva responsabilità di Texas Instruments, indipendentemente dalla forma dell'azione, non supera l'importo indicato nella licenza per il programma. Inoltre Texas Instruments non è responsabile di rivendicazioni di alcun genere contro l'uso di questi materiali da parte di chiunque.

Licenza

Leggere per intero la licenza installata in **C:\Program Files\TI Education\<TI-Nspire™ Product Name>\license**.

© 2006 - 2016 Texas Instruments Incorporated

Sommario

Informazioni importanti 2
Sommario 3

Modelli di espressione 5

Elenco alfabetico 11

A 11
B 20
C 24
D 41
E 49
F 57
G 65
I 73
L 80
M 96
N 105
O 114
P 117
Q 124
R 127
S 141
T 162
U 175
V 176
W 177
X 179
Z 180

Simboli 187

Elementi vuoti (nulli) 212

Scelte rapide per l'inserimento di espressioni matematiche 214

EOS™ (Equation Operating System) gerarchia 216

Codici di errore e messaggi 218

Codici di avvertenza e messaggi 227

Informazioni generali 229

Informazioni sul servizio di manutenzione e riparazione del prodotto TI e sulla 229

garanzia .....

Indice **231**

Modelli di espressione

I modelli di espressione rappresentano un metodo veloce per introdurre espressioni matematiche in notazione matematica standard. Un modello, quando inserito, viene visualizzato nella riga di introduzione come tanti quadratini al posto degli elementi che si possono inserire. Un cursore indica quale elemento si può inserire.

Utilizzare i tasti freccia o premere [tab] per spostare il cursore su ciascun elemento e digitare un valore o un'espressione per esso. Premere [enter] o [ctrl] [enter] per calcolare l'espressione.

Modello di frazione

Tasti [ctrl] [÷]

$\frac{\square}{\square}$ **Nota:** vedere anche **/ (divisione)**, pagina 189.

Esempio:

$\frac{12}{8\cdot 2}$ $\quad$ $\frac{3}{4}$

Modello di esponente

Tasto [^]

$\square^{\square}$

Nota: digitare il primo valore, premere [^], quindi digitare l'esponente. Per riportare il cursore sulla linea di base, premere la freccia a destra (▶).

Nota: vedere anche **^ (potenza)**, pagina 190.

Esempio:

2^3 $\quad$ 8

Modello di radice quadrata

Tasti [ctrl] [x²]

$\sqrt{\square}$ **Nota:** vedere anche **√() (radice quadrata)**, pagina 199.

Esempio:

$\sqrt{4}$ $\quad$ 2

$\sqrt{\{9,16,4\}}$ $\quad$ $\{3,4,2\}$

Modello di radice ennesima

Tasti [ctrl] [^]

$\sqrt[\square]{\square}$ **Nota:** vedere anche **root()**, pagina 138.

Esempio:

Modello di radice ennesima

Tasti ctrl ^

$\sqrt[3]{8}$	2
$\sqrt[3]{\{8,27,15\}}$	$\{2,3,2.46621\}$

Modello di funzione esponenziale e

Tasti e^x

$e^{\square}$

Esponenziale naturale e elevato a potenza

Nota: vedere anche **e^()**, pagina 49.

Esempio:

e^1	2.71828182846

Modello di log

Tasto ctrl

$\log_{\square}(\square)$

Calcola il logaritmo nella base specificata. Per la base 10 predefinita, omettere la base.

Nota: vedere anche **log()**, pagina 91.

Esempio:

$\log_4(2.)$	0.5

Modello di funzione piecewise a 2 tratti

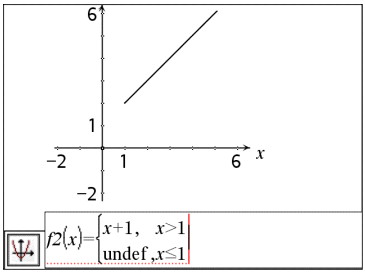
Catalogo >

$\begin{cases}\square, & \square \\ \square, & \square\end{cases}$

Consente di creare espressioni e condizioni per funzioni definite a due-tratti. Per aggiungere un tratto, fare clic sul modello e ripeterlo.

Nota: vedere anche **piecewise()**, pagina 118.

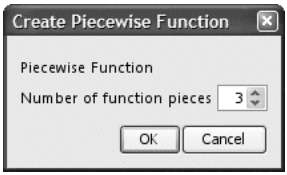
Esempio:



Modello di funzione piecewise a N tratti

Catalogo >

Consente di creare espressioni e condizioni per una funzione definita a N-tratti. Richiede l'introduzione di N.



Nota: vedere anche **piecewise()**, pagina 118.

Esempio:

vedere l'esempio del Modello di funzione piecewise a 2 tratti.

Modello di sistema di 2 equazioni

Catalogo >

Crea un sistema di due equazioni lineari. Per aggiungere una riga a un sistema esistente, fare clic sul modello e ripeterlo.

Nota: vedere anche **system()**, pagina 161.

Esempio:

$$\text{solve}\left(\begin{cases} x+y=0 \\ x-y=5 \end{cases}, x, y\right) \qquad x=\frac{5}{2} \text{ and } y=\frac{-5}{2}$$

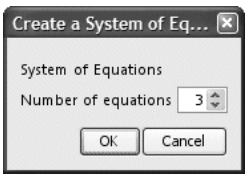
$$\text{solve}\left(\begin{cases} y=x^2-2 \\ x+2\cdot y=-1 \end{cases}, x, y\right)$$

$$x=\frac{-3}{2} \text{ and } y=\frac{1}{4} \text{ or } x=1 \text{ and } y=-1$$

Modello di sistema di N equazioni

Catalogo >

Consente di creare un sistema di N equazioni lineari. Richiede l'introduzione di N.



Nota: vedere anche **system()**, pagina 161.

Esempio:

vedere l'esempio del Modello di sistema di equazioni (2 equazioni).

Modello di valore assoluto

Catalogo >

$|\square|$ **Nota:** vedere anche **abs()**, pagina 11.

Esempio:

$|\{2,-3,4,-4^3\}|$ $\{2,3,4,64\}$

Modello di gg°pp'ss.ss''

Catalogo >

$\square°\square'\square''$

Consente di inserire angoli nel formato **gg°pp'ss.ss''**, dove **gg** è il numero di gradi decimali, **pp** è il numero di primi e **ss.ss** è il numero di secondi.

Esempio:

30°15'10" 0.528011

Modello di matrice (2 x 2)

Catalogo >

$\begin{bmatrix}\square & \square\\ \square & \square\end{bmatrix}$

Crea una matrice 2 x 2.

Esempio:

$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}\cdot 5$ $\begin{bmatrix}5 & 10\\ 15 & 20\end{bmatrix}$

Modello di matrice (1 x 2)

Catalogo >

$\begin{bmatrix}\square & \square\end{bmatrix}$.

Esempio:

crossP$(\begin{bmatrix}1 & 2\end{bmatrix},\begin{bmatrix}3 & 4\end{bmatrix})$ $\begin{bmatrix}0 & 0 & -2\end{bmatrix}$

Modello di matrice (2 x 1)

Catalogo >

$\begin{bmatrix}\square\\ \square\end{bmatrix}$

Esempio:

$\begin{bmatrix}5\\ 8\end{bmatrix}\cdot 0.01$ $\begin{bmatrix}0.05\\ 0.08\end{bmatrix}$

Modello di matrice (m x n)

Catalogo >

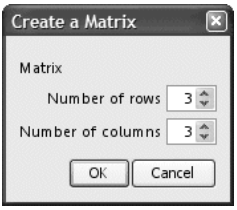
Il modello appare dopo la richiesta di specificare il numero di righe e colonne.

Esempio:

diag$\left(\begin{bmatrix}4 & 2 & 6\\ 1 & 2 & 3\\ 5 & 7 & 9\end{bmatrix}\right)$ $\begin{bmatrix}4 & 2 & 9\end{bmatrix}$

Modello di matrice (m x n)

Catalogo >



Nota: se si crea una matrice con un numero elevato di righe e colonne, è possibile che la visualizzazione richieda un po' di tempo.

Modello di sommatoria (Σ)

Catalogo >

$$\sum_{\square=\square}^{\square}(\square)$$

Nota: vedere anche **Σ() (sumSeq)**, pagina 201.

Esempio:

$$\sum_{n=3}^{7}(n) \qquad 25$$

Modello di prodotto (Π)

Catalogo >

$$\prod_{\square=\square}^{\square}(\square)$$

Nota: vedere anche **Π() (prodSeq)**, pagina 200.

Esempio:

$$\prod_{n=1}^{5}\left(\frac{1}{n}\right) \qquad \frac{1}{120}$$

Modello di derivata prima

Catalogo >

$$\frac{d}{d\square}(\square)$$

Può essere utilizzato inoltre per calcolare numericamente la derivata prima in un punto con metodi di differenziazione automatica.

Esempio:

$$\frac{d}{dx}(|x|)|x=0 \qquad \text{undef}$$

Modello di derivata prima

Catalogo >

Nota: vedere anche **d() (derivata)**, pagina 198.

Modello di derivata seconda

Catalogo >

$$\frac{d^2}{d\square^2}(\square)$$

Può essere utilizzato inoltre per calcolare numericamente la derivata seconda in un punto con metodi di differenziazione automatica.

Nota: vedere anche **d() (derivata)**, pagina 198.

Esempio:

$$\frac{d^2}{dx^2}\left(x^3\right)\Big|_{x=3} \qquad 18$$

Modello di integrale definito

Catalogo >

$$\int_{\square}^{\square} \square \, d\square$$

Il modello di integrale definito può essere utilizzato per calcolare numericamente l'integrale definito, con lo stesso metodo di nInt().

Nota: vedere anche **nInt(), pagina 109.**

Esempio:

$$\int_0^{10} x^2 \, dx \qquad 333.333$$

Elenco alfabetico

Gli elementi i cui nomi sono composti da caratteri non alfabetici (come ad esempio +, !, >) sono elencati alla fine della presente sezione, pagina 187. Se non diversamente specificato, tutti gli esempi della presente sezione sono stati eseguiti in modalità reset predefinita e tutte le variabili sono intese come non definite.

A

abs() (Valore assoluto) — Catalogo >

abs(*Valore1***)⇒***valore*

abs(*Lista1***)⇒***lista*

abs(*Matrice1***)⇒***matrice*

Restituisce il valore assoluto dell'argomento.

Nota: vedere anche **Modello di valore assoluto**, pagina 8.

Se l'argomento è un numero complesso, restituisce il modulo del numero.

Nota: tutte le variabili non definite vengono considerate come variabili reali.

$$\left|\left\{\frac{\pi}{2},\frac{-\pi}{3}\right\}\right| \qquad \{1.5708,1.0472\}$$

$$|2-3\cdot i| \qquad 3.60555$$

amortTbl() — Catalogo >

amortTbl(*NPmt*,*N*,*I*,*PV*, [*Pmt*], [*FV*], [*PpY*], [*CpY*], [*PmtAt*], [*valoreArrotondato*]**)⇒***matrice*

Funzione di ammortamento che restituisce una matrice come una tabella di ammortamento per un set di argomenti TVM.

NPmt è il numero di rate da includere nella tabella. La tabella inizia con la prima rata.

N, *I*, *PV*, *Pmt*, *FV*, *PpY*, *CpY* e *PmtAt* sono descritti nella tabella degli argomenti TVM, pagina 173.

- Se si omette *Pmt*, viene utilizzata l'impostazione predefinita *Pmt*=**tvmPmt(***N,I,PV,FV,PpY,CpY,PmtAt***)**.
- Se si omette *FV*, viene utilizzata

amortTbl(12,60,10,5000,,,12,12)

0	0.	0.	5000.
1	-41.67	-64.57	4935.43
2	-41.13	-65.11	4870.32
3	-40.59	-65.65	4804.67
4	-40.04	-66.2	4738.47
5	-39.49	-66.75	4671.72
6	-38.93	-67.31	4604.41
7	-38.37	-67.87	4536.54
8	-37.8	-68.44	4468.1
9	-37.23	-69.01	4399.09
10	-36.66	-69.58	4329.51
11	-36.08	-70.16	4259.35
12	-35.49	-70.75	4188.6

l'impostazione predefinita FV=0.

- Le impostazioni predefinite di PpY, CpY e $PmtAt$ sono le stesse delle funzioni TVM.

valoreArrotondato specifica il numero di cifre decimali di arrotondamento. Impostazione predefinita=2.

Le colonne nella matrice risultante appaiono nel seguente ordine: numero di rate, interesse pagato, capitale versato e saldo.

Il saldo visualizzato nella riga n è il saldo dopo la rata n.

È possibile utilizzare la matrice di output come input per le altre funzioni di ammortamento **ΣInt()** e **ΣPrn()**, pagina 201, e **bal()**, pagina 20.

and

Espressione booleana1 **and** *Espressione booleana2* ⇒*Espressione booleana*

Lista booleana1 **and** *Lista booleana2*⇒*Lista booleana*

Matrice booleana1 **and** *Matrice booleana2*⇒*Matrice booleana*

Restituisce vero o falso o una forma semplificata dell'espressione immessa originariamente.

*Intero1***and***Intero2*⇒*intero*

Confronta due interi reali bit per bit utilizzando un'operazione **and**. Internamente, entrambi gli interi vengono convertiti in numeri binari a 64 bit con segno. Quando vengono confrontati bit corrispondenti, il risultato sarà 1 se entrambi sono uguali a 1; in caso contrario il risultato sarà 0. Il valore restituito rappresenta il risultato dei bit e viene visualizzato nella modalità base che è stata impostata.

In modalità base Esadecimale:

0h7AC36 and 0h3D5F	0h2C16

Importante: è zero, non la lettera O.

In modalità base Bin:

0b100101 and 0b100	0b100

In modalità base Dec:

and

Catalogo >

È possibile inserire gli interi in qualsiasi base numerica. Se si tratta di un numero binario o esadecimale, utilizzare rispettivamente il prefisso 0b o 0h. Senza prefisso, gli interi vengono considerati decimali (base 10).

Se viene indicato un intero decimale troppo grande per una forma binaria con segno a 64 bit, verrà utilizzata un'operazione a modulo simmetrico per portare il valore nell'intervallo appropriato.

37 and 0b100	4

Nota: un numero binario può contenere fino a 64 cifre (oltre al prefisso 0b). Un numero esadecimale può contenere fino ad 16 cifre.

angle() (Angolo)

Catalogo >

angle(*Valore1***)⇒***valore*

Restituisce l'angolo dell'argomento, interpretando l'argomento come numero complesso.

In modalità angolo in gradi:

$\text{angle}(0+2\cdot i)$	90

In modalità angolo in gradianti (gradi centesimali):

$\text{angle}(0+3\cdot i)$	100

In modalità angolo in radianti:

$\text{angle}(1+\boldsymbol{i})$	0.785398
$\text{angle}(\{1+2\cdot\boldsymbol{i},3+0\cdot\boldsymbol{i},0-4\cdot\boldsymbol{i}\})$	$\{1.10715,0.,-1.5708\}$

angle(*Lista1***)⇒***lista*

angle(*Matrice1***)⇒***matrice*

Restituisce una lista o una matrice di angoli degli elementi contenuti in *Lista1* o *Matrice1*, interpretando ciascun elemento come un numero complesso che rappresenta un punto di coordinate rettangolari bidimensionali.

$\text{angle}(\{1+2\cdot i,3+0\cdot i,0-4\cdot i\})$	$\left\{\frac{\pi}{2}-\tan^{-1}\left(\frac{1}{2}\right),0,\frac{-\pi}{2}\right\}$

ANOVA (Analisi della varianza)

Catalogo >

ANOVA *Lista1*,*Lista2*[,*Lista3*,...,*Lista20*] [,*Flag*]

ANOVA (Analisi della varianza) Catalogo >

Esegue l'analisi della varianza a una dimensione per confrontare le medie di un numero di popolazioni compreso tra due e venti. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Flag=0 per Dati, *Flag*=1 per Statistiche

Variabile di output	Descrizione
stat.F	Valore della statistica F
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.df	Grado di liberà dei gruppi
stat.SS	Somma dei quadrati dei gruppi
stat.MS	Quadrati medi dei gruppi
stat.dfError	Gradi di libertà degli errori
stat.SSError	Somma dei quadrati degli errori
stat.MSError	Quadrato medio degli errori
stat.sp	Deviazione standard aggregata
stat.xbarlist	Media dell'input delle liste
stat.CLowerList	Intervalli di confidenza al 95% per la media di ogni lista di input
stat.CUpperList	Intervalli di confidenza al 95% per la media di ogni lista di input

ANOVA2way (Analisi della varianza a due dimensioni) Catalogo >

ANOVA2way *Lista1*,*Lista2* [,*Lista3*,...,*Lista10*][,*RigaLiv*]

Esegue l'analisi a due dimensioni della varianza per confrontare le medie di un numero di popolazioni compreso tra due e dieci. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

LevRow=0 per Blocco

LevRow=2,3,...,*Len*-1, per Due Fattori, dove *Len*=lunghezza(*Lista1*)=lunghezza(*Lista2*) = ... = lunghezza(*Lista10*) e *Len* / *LevRow* ∈ {2,3,...}

`Output: design blocco`

Variabile di output	Descrizione
stat.F	Statistica F del fattore colonna
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.df	Gradi di libertà del fattore colonna
stat.SS	Somma dei quadrati del fattore colonna
stat.MS	Quadrati medi del fattore colonna
stat.FBlock	Statistica F per fattore
stat.PValBlock	Probabilità minima in cui l'ipotesi nulla può essere rifiutata
stat.dfBlock	Gradi di libertà per fattore
stat.SSBlock	Somma dei quadrati per fattore
stat.MSBlock	Quadrati medi per fattore
stat.dfError	Gradi di libertà degli errori
stat.SSError	Somma dei quadrati degli errori
stat.MSError	Quadrati medi degli errori
stat.s	Deviazione standard dell'errore

Output FATTORE COLONNA

Variabile di output	Descrizione
stat.Fcol	Statistica F del fattore colonna
stat.PValCol	Valore di probabilità del fattore colonna
stat.dfCol	Gradi di libertà del fattore colonna
stat.SSCol	Somma dei quadrati del fattore colonna
stat.MSCol	Quadrati medi del fattore colonna

Output FATTORE RIGA

Variabile di output	Descrizione
stat.Frow	Statistica F del fattore riga
stat.PValRow	Valore di probabilità del fattore riga
stat.dfRow	Gradi di libertà del fattore riga
stat.SSRow	Somma dei quadrati del fattore riga
stat.MSRow	Quadrati medi del fattore riga

Output di INTERAZIONE

Variabile di output	Descrizione
stat.FInteract	F dell'interazione
stat.PValInteract	Valore di probabilità dell'interazione
stat.dfInteract	Gradi di liberà dell'interazione
stat.SSInteract	Somma dei quadrati dell'interazione
stat.MSInteract	Quadrati medi dell'interazione

Output di ERRORE

Variabile di output	Descrizione
stat.dfError	Gradi di libertà degli errori
stat.SSError	Somma dei quadrati degli errori
stat.MSError	Quadrati medi degli errori
s	Deviazione standard dell'errore

Ans (Ultimo risultato)

Tasti ctrl (–)

Ans⇒*valore*

Restituisce il risultato dell'ultima espressione calcolata.

56	56
56+4	60
60+4	64

approx() Approssima

Catalogo >

approx(*Valore1***)⇒***numero*

Restituisce il calcolo dell'argomento come espressione contenente valori decimali, ove possibile, indipendentemente dalla modalità corrente **Auto o Approssimato**.

Equivale a inserire l'argomento e a premere ctrl enter.

$\text{approx}\left(\frac{1}{3}\right)$	0.333333
$\text{approx}\left(\left\{\frac{1}{3},\frac{1}{9}\right\}\right)$	$\{0.333333,0.111111\}$
$\text{approx}(\{\sin(\pi),\cos(\pi)\})$	$\{0.,-1.\}$
$\text{approx}(\left[\sqrt{2}\quad\sqrt{3}\right])$	$[1.41421\quad 1.73205]$
$\text{approx}\left(\left[\frac{1}{3}\quad\frac{1}{9}\right]\right)$	$[0.333333\quad 0.111111]$

approx(*Lista1***)⇒***lista*

approx(*Matrice1***)⇒***matrice*

Restituisce una lista o una *matrice* nella quale ciascun elemento è stato calcolato con valori decimali, ove possibile.

$\text{approx}(\{\sin(\pi),\cos(\pi)\})$	$\{0.,-1.\}$
$\text{approx}(\left[\sqrt{2}\quad\sqrt{3}\right])$	$[1.41421\quad 1.73205]$

▸approxFraction()

Catalogo >

Valore ▸*approxFraction(*[*Tol*]*)⇒valore*

Lista▸**approxFraction(**[*Tol*]**)⇒***lista*

Matrice ▸**approxFraction(**[*Tol*]**)⇒***matrice*

Restituisce l'argomento come frazione utilizzando una tolleranza *Tol*. Se *tol* è omesso, viene utilizzata una tolleranza di 5.E-14.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `@>approxFraction(…)`.

$\frac{1}{2}+\frac{1}{3}+\tan(\pi)$	0.833333
$0.83333333333333\blacktriangleright\text{approxFraction}(5.\text{E}^{-}14)$	$\frac{5}{6}$
$\{\pi,1.5\}\blacktriangleright\text{approxFraction}(5.\text{E}^{-}14)$	$\left\{\frac{5419351}{1725033},\frac{3}{2}\right\}$

approxRational()

Catalogo >

approxRational(*Valore***[, *Tol*]**)⇒***valore***

approxRational(*Lista***[, *tol*]**)⇒***lista***

approxRational(*Matrice***[, *tol*]**)⇒***matrice***

Restituisce l'argomento come frazione utilizzando una tolleranza *tol*. Se *tol* è omesso, viene utilizzata una tolleranza di 5.E-14.

$\text{approxRational}(0.333,5\cdot10^{-5})$	$\frac{333}{1000}$
$\text{approxRational}(\{0.2,0.33,4.125\},5.\text{E}^{-}14)$	$\left\{\frac{1}{5},\frac{33}{100},\frac{33}{8}\right\}$

arccos() **Vedere $\cos^{-1}()$, pagina 32.**

arccosh() **Vedere $\cosh^{-1}()$, pagina 34.**

arccot() **Vedere $\cot^{-1}()$, pagina 35.**

arccoth() **Vedere $\coth^{-1}()$, pagina 35.**

arccsc() **Vedere $\csc^{-1}()$, pagina 38.**

arccsch() **Vedere $\mathrm{csch}^{-1}()$, pagina 39.**

arcsec() **Vedere $\sec^{-1}()$, pagina 142.**

arcsech() **Vedere $\mathrm{sech}^{-1}()$, pagina 142.**

arcsin() **Vedere $\sin^{-1}()$, pagina 151.**

arcsinh() **Vedere $\sinh^{-1}()$, pagina 152.**

arctan() Vedere $\tan^{-1}$(), pagina 163.

arctanh() Vedere $\tanh^{-1}$(), pagina 164.

augment() (Affianca/concatena) Catalogo >

augment(*Lista1, Lista2***)⇒***lista*

Restituisce una nuova lista in cui *Lista2* viene aggiunta (accostata) alla fine di *Lista1*.

augment({1,-3,2},{5,4}) {1,-3,2,5,4}

augment(*Matrice1***, *Matrice2***)⇒***matrice*

Restituisce una nuova matrice in cui *Matrice2* viene aggiunta alla fine di *Matrice1*. Se si usa il carattere "," le matrici devono avere uguale numero di righe; *Matrice2* viene aggiunta a *Matrice1* come nuove colonne. Non modifica *Matrice1* o *Matrice2*.

$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\rightarrow m1$ $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$

$\begin{bmatrix}5\\6\end{bmatrix}\rightarrow m2$ $\begin{bmatrix}5\\6\end{bmatrix}$

augment(m1,m2) $\begin{bmatrix}1 & 2 & 5\\3 & 4 & 6\end{bmatrix}$

avgRC() (Tasso di variazione media) Catalogo >

avgRC(*Espr1***, *Var* [=*Valore*] [, *Incr*])⇒***espressione***

avgRC(*Espr1***, *Var* [=*Valore*] [, *Lista1*])⇒***lista***

avgRC(*Lista1***, *Var* [=*Valore*] [, *Incr*])⇒***lista***

avgRC(*Matrice1***, *Var* [=*Valore*] [, *Incr*])⇒***matrice***

Restituisce il rapporto incrementale (tasso di variazione media).

Espr1 può essere un nome di funzione definito dall'utente (vedere **Func**).

Se il *Valore* è specificato, lo stesso sostituisce qualsiasi assegnazione di variabile precedente o qualsiasi sovrapposizione corrente "|" della variabile.

x:=2 2

avgRC(x^2-x+2,x) 3.001

avgRC(x^2-x+2,x,.1) 3.1

avgRC(x^2-x+2,x,3) 6

avgRC() (Tasso di variazione media) Catalogo >

Incr è il valore di incremento. Se *Incr* è omesso, viene impostato per default su 0.001.

Si noti che la funzione simile **centralDiff()** utilizza la formula del rapporto incrementale bilaterale.

B

bal() Catalogo >

bal(*NPmt***,***N***,***I***,***PV***,[***Pmt***], [***FV***], [***PpY***], [***CpY***], [***PmtAt***], [***valoreArrotondato***])⇒***valore***

bal(*NPmt***,***tabellaAmmortamento***)⇒***valore***

Funzione di ammortamento che calcola il saldo del piano di rientro dopo una rata specificata.

N, *I*, *PV*, *Pmt*, *FV*, *PpY*, *CpY* e *PmtAt* sono descritti nella tabella degli argomenti TVM, pagina 173.

NPmt specifica il numero della rata a partire dalla quale deve essere calcolato il saldo.

N, *I*, *PV*, *Pmt*, *FV*, *PpY*, *CpY* e *PmtAt* sono descritti nella tabella degli argomenti TVM, pagina 173.

- Se si omette *Pmt*, viene utilizzata l'impostazione predefinita *Pmt*=**tvmPmt (***N,I,PV,FV,PpY,CpY,PmtAt***)**.
- Se si omette *FV*, viene utilizzata l'impostazione predefinita *FV*=0.
- Le impostazioni predefinite di *PpY*, *CpY* e *PmtAt* sono le stesse delle funzioni TVM.

valoreArrotondato specifica il numero di cifre decimali di arrotondamento. Impostazione predefinita=2.

bal(5,6,5.75,5000,,12,12) 833.11

tbl:=amortTbl(6,6,5.75,5000,,12,12)

0	0.	0.	5000.
1	-23.35	-825.63	4174.37
2	-19.49	-829.49	3344.88
3	-15.62	-833.36	2511.52
4	-11.73	-837.25	1674.27
5	-7.82	-841.16	833.11
6	-3.89	-845.09	-11.98

bal(4,*tbl*) 1674.27

bal() Catalogo >

bal(*NPmt*,*tabellaAmmortamento***)** calcola il saldo dopo la rata numero *NPmt* sulla base della tabella di ammortamento *tabellaAmmortamento*. L'argomento *tabellaAmmortamento* deve essere una matrice avente la forma descritta in **amortTbl()**, pagina 11.

Nota: vedere anche **ΣInt()** e **ΣPrn()**, pagina 201.

▸Base2 Catalogo >

Intero1 **▸Base2**⇒*intero*

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@>Base2`.

Converte *Intero1* in un numero binario. I numeri binari o esadecimali hanno sempre, rispettivamente, il prefisso 0b o 0h. Zero, non la lettera O, seguito da b o h.

0b *numeroBinario*

0h *numeroEsadecimale*

Un numero binario può contenere fino a 64 cifre. Un numero esadecimale può contenere fino ad 16 cifre.

Senza prefisso, *Intero1* viene considerato decimale (base 10). Il risultato viene visualizzato in modalità binaria, indipendentemente dalla modalità Base impostata.

I numeri negativi sono visualizzati nella forma "a complemento di due". Ad esempio,

−1 è visualizzato come

0hFFFFFFFFFFFFFFFF in modalità base Esadecimale 0b111...111 (la cifra 1 ripetuta 64 volte) in modalità base Binaria

256▸Base2	0b100000000
0h1F▸Base2	0b11111

▶Base2 Catalogo >

-2^{63} è visualizzato come 0h8000000000000000 in modalità base Esadecimale e 0b100...000 (63 zeri) in modalità base Binaria

Se viene indicato un intero decimale esterno alla gamma di una forma binaria con segno a 64 bit, verrà utilizzata un'operazione a modulo simmetrico per portare il valore nell'intervallo appropriato. Considerare i seguenti esempi di valori esterni alla gamma.

2^{63} diventa -2^{63} ed è visualizzato come 0h8000000000000000 in modalità base Esadecimale e 0b100...000 (63 zeri) in modalità base Binaria

2^{64} diventa 0 ed è visualizzato come 0h0 in modalità base Esadecimale e 0b0 in modalità base Binaria

$-2^{63} - 1$ diventa $2^{63} - 1$ ed è visualizzato come 0h7FFFFFFFFFFFFFFF in modalità base Esadecimale e 0b111...111 (64 1) in modalità base Binaria

▶Base10 Catalogo >

Intero1 ▶**Base10**⇒*intero*

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@>Base10`.

Converte *Intero1* in numero decimale (base 10). Le voci binarie o esadecimali devono sempre avere, rispettivamente, il prefisso 0b o 0h.

0b *numeroBinario*

0h *numeroEsadecimale*

Zero, non la lettera O, seguito da b o h.

0b10011▶Base10	19
0h1F▶Base10	31

▶Base10 Catalogo >

Un numero binario può contenere fino a 64 cifre. Un numero esadecimale può contenere fino ad 8 cifre.

Senza prefisso, *Intero1* viene considerato decimale (base10). Il risultato viene visualizzato in modalità decimale, indipendentemente dalla modalità Base impostata.

▶Base16 Catalogo >

Intero1 ▶**Base16**⇒*intero*

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **`@>Base16`**.

Converte *Intero1* in un numero esadecimale. I numeri binari o esadecimali hanno sempre, rispettivamente, il prefisso 0b o 0h.

0b *numeroBinario*

0h *numeroEsadecimale*

Zero, non la lettera O, seguito da b o h.

Un numero binario può contenere fino a 64 cifre. Un numero esadecimale può contenere fino ad 16 cifre.

Senza prefisso, *Intero1* viene considerato decimale (base 10). Il risultato viene visualizzato in modalità esadecimale, indipendentemente dalla modalità Base impostata.

Se viene indicato un intero decimale troppo grande per una forma binaria con segno a 64 bit, verrà utilizzata un'operazione a modulo simmetrico per portare il valore nell'intervallo appropriato. Per ulteriori informazioni, vedere ▶**Base2**, pagina 21.

256▶Base16	0h100
0b111100001111▶Base16	0hF0F

binomCdf() (Funzione della probabilità cumulativa per la distribuzione binomiale)

Catalogo >

binomCdf(*n***,***p***)**⇒*lista*

binomCdf(*n***,***p***,***valoreInferiore***,** *valoreSuperiore***)** ⇒ *numero* se *valoreInferiore* e *valoreSuperiore* sono numeri, *lista* se *valoreInferiore* e *valoreSuperiore* sono liste

binomCdf(*n***,***p***,***valoreSuperiore***)**per P(0≤X ≤*valoreSuperiore*)⇒*numero* se *valoreSuperiore* è un numero, *lista* se *valoreSuperiore* è una lista

Calcola la probabilità cumulativa per la distribuzione binomiale discreta con il numero di prove *n* e le probabilità di successo *p* per ciascuna prova.

Per P(X ≤ *valoreSuperiore*), impostare *valoreInferiore*=0

binomPdf()

Catalogo >

binomPdf(*n***,***p***)**⇒*lista*

binomPdf(*n***,***p***,***ValX***)**⇒*numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola una probabilità in corrispondenza di *valX* per la distribuzione binomiale discreta con il numero *n* di prove e la probabilità *p* di successo per ogni prova.

C

ceiling() (Arrotondato per eccesso)

Catalogo >

ceiling(*Valore1***)**⇒*valore*

```
ceiling(.456)                    1.
```

Restituisce il più vicino numero intero ≤ all'argomento.

L'argomento può essere un numero reale o complesso.

Nota: vedere anche **floor()**.

ceiling() (Arrotondato per eccesso) Catalogo >

ceiling(*Lista1***)⇒***lista*

ceiling(*Matrice1***)⇒***matrice*

Restituisce una lista o matrice del valore arrotondato per eccesso di ciascun elemento.

$\text{ceiling}\left(\left\{-3.1,1,2.5\right\}\right)$	$\left\{-3.,1,3.\right\}$
$\text{ceiling}\left(\begin{bmatrix} 0 & -3.2\cdot i \\ 1.3 & 4 \end{bmatrix}\right)$	$\begin{bmatrix} 0 & -3.\cdot i \\ 2. & 4 \end{bmatrix}$

centralDiff() Catalogo >

centralDiff(*Espr1***,***Var* [=*Valore*] [,*Incr*])⇒***espressione***

centralDiff(*Espr1***,***Var* [,*Incr*])|***Var*=*Valore*⇒*espressione***

centralDiff(*Espr1***,***Var* [=*Valore*] [,*Lista*])⇒***lista***

centralDiff(*Lista1***,***Var* [=*Valore*] [,*Incr*])⇒***lista***

centralDiff(*Matrice1***,***Var* [=*Valore*] [,*Incr*])⇒***matrice***

Restituisce la derivata numerica utilizzando la formula del rapporto incrementale bilaterale.

Se il *Valore* è specificato, lo stesso sostituisce qualsiasi assegnazione di variabile precedente o qualsiasi sovrapposizione corrente "|" della variabile.

Incr è il valore di incremento. Se *Incr* è omesso, viene impostato per default su 0.001.

Se si utilizza *Lista1* o *Matrice1*, l'operazione viene mappata sui valori della lista o sugli elementi della matrice.

Nota: vedere anche .

$\text{centralDiff}\left(\cos(x),x\right)\vert x=\frac{\pi}{2}$	-1.

char() (Stringa di caratteri) Catalogo >

char(*Intero***)⇒***carattere***

char(38)	"&"
char(65)	"A"

char() (Stringa di caratteri) Catalogo >

Restituisce un carattere stringa corrispondente al numero *Intero* del set di caratteri del palmare. L'intervallo valido per intero *Intero* è compreso tra 0 e 65535.

$\chi^2$2way Catalogo >

χ^2**2way** *MatriceOss*

chi22way *MatriceOss*

Esegue una verifica c^2 per l'associazione di numeri nella tabella a due variabili nella matrice osservata *MatriceOss*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Per informazioni sull'effetto di elementi vuoti in una matrice, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.χ^2	Statistica Chi quadrato: somma (osservati - attesi)^2/attesi
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.df	Gradi di libertà per le statistiche chi quadrato
stat.ExpMat	Matrice della tabella di numeri elementari attesi, assumendo l'ipotesi nulla
stat.CompMat	Matrice di contributi statistici chi quadrato elementari

χ^2Cdf() Catalogo >

χ^2**Cdf**
(
valoreInferiore
,*valoreSuperiore***,***gl***)**⇒*numero* se *valoreInferiore* e *valoreSuperiore* sono numeri, *lista* se *valoreInferiore* e *valoreSuperiore* sono liste

chi2Cdf
(
valoreInferiore

χ^2Cdf()

Catalogo >

,*valoreSuperiore***,***gl***)⇒***numero* se *valoreInferiore* e *valoreSuperiore* sono numeri, *list* se *valoreInferiore* e *valoreSuperiore* sono liste

Calcola la probabilità della distribuzione χ^2 tra il *valoreInferiore* e il *valoreSuperiore* per i gradi di libertà *gl* specificati.

Per P($X \leq$ *valoreSuperiore*), impostare *valoreInferiore*= 0.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

χ^2GOF

Catalogo >

χ^2**GOF** *listaOss***,***listaAtt***,***gl*

chi2GOF *listaOss***,***listaAtt***,***gl*

Esegue una verifica per confermare che i dati del campione appartengono a una popolazione conforme a una data distribuzione. *listaOss* è una lista di conteggi e deve contenere numeri interi. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	**Descrizione**
stat.χ^2	Statistica Chi quadrato: sum((osservati - attesi)^2/attesi
stat.PVal	Livello minimo di significatività in corrispondenza del quale l’ipotesi nulla può essere rifiutata
stat.df	Gradi di libertà per le statistiche chi quadrato
stat.CompList	Contributi statistici chi quadrato elementari

χ^2Pdf()

Catalogo >

χ^2Pdf(*ValX*,*gl*)⇒*numero* se *ValX* è un numero, *lista* se *ValX* è una lista

chi2Pdf(*ValX*,*gl*)⇒*numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola la funzione della densità di probabilità (pdf) per la distribuzione χ^2 a un dato valore *ValX* per i gradi di libertà *gl* specificati.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

ClearAZ (Cancella AZ)

Catalogo >

ClearAZ

Cancella tutte le variabili con il nome di un solo carattere nello spazio attività corrente.

Se una o più variabili sono bloccate, questo comando visualizza un messaggio di errore ed elimina solo le variabili sbloccate. Vedere **unLock, pagina 176.**

$5 \rightarrow b$	5
b	5
ClearAZ	*Done*
b	"Error: Variable is not defined"

ClrErr

Catalogo >

ClrErr

Cancella lo stato di errore e imposta la variabile di sistema *errCode* su zero.

L’istruzione **Else** del blocco **Try...Else...EndTry** dovrebbe utilizzare **ClrErr** o **PassErr**. Se l’errore deve essere elaborato o ignorato, utilizzare **ClrErr**. Se non si sa quale azione applicare all’errore, utilizzare **PassErr** per inviarlo al successivo blocco di gestione degli errori. Se non ci sono ulteriori blocchi di gestione degli errori **Try...Else...EndTry** in attesa di applicazione, la finestra di dialogo dell’errore viene visualizzata come normale.

Nota: vedere anche **PassErr**, pagina 118 e **Try**, pagina 168.

Per un esempio di **ClrErr**, vedere l’esempio 2 del comando **Try**, pagina 169.

ClrErr

Catalogo >

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

colAugment() (Affianca colonna)

Catalogo >

colAugment(*Matrice1***, *Matrice2*)⇒*matrice***

Restituisce una nuova matrice in cui *Matrice2* viene aggiunta alla fine di *Matrice1*. Le matrici devono avere uguale numero di colonne; *Matrice2* viene aggiunta a *Matrice1* come nuove colonne. Non modifica *Matrice1* o *Matrice2*.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\rightarrow m1 \qquad \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$$

$$\begin{bmatrix}5 & 6\end{bmatrix}\rightarrow m2 \qquad \begin{bmatrix}5 & 6\end{bmatrix}$$

$$\text{colAugment}(m1,m2) \qquad \begin{bmatrix}1 & 2\\3 & 4\\5 & 6\end{bmatrix}$$

colDim() (Dimensione colonna)

Catalogo >

colDim(*Matrice***)⇒*espressione***

Restituisce il numero delle colonne contenute in *Matrice*.

Nota: vedere anche **rowDim()**.

$$\text{colDim}\left(\begin{bmatrix}0 & 1 & 2\\3 & 4 & 5\end{bmatrix}\right) \qquad 3$$

colNorm() (Norma colonna)

Catalogo >

colNorm(*Matrice***)⇒*espressione***

Restituisce il massimo delle somme dei valori assoluti degli elementi nelle colonne di *Matrice*.

Nota: non sono ammessi elementi non definiti di una matrice. Vedere anche **rowNorm()**.

$$\begin{bmatrix}1 & -2 & 3\\4 & 5 & -6\end{bmatrix}\rightarrow mat \qquad \begin{bmatrix}1 & -2 & 3\\4 & 5 & -6\end{bmatrix}$$

$$\text{colNorm}(mat) \qquad 9$$

conj() (Coniugato)

Catalogo >

conj(*Valore1***)⇒*valore***

conj(*Lista1***)⇒*lista***

conj(*Matrice1***)⇒*matrice***

$$\text{conj}(1+2\cdot i) \qquad 1-2\cdot i$$

$$\text{conj}\left(\begin{bmatrix}2 & 1-3\cdot i\\-i & -7\end{bmatrix}\right) \qquad \begin{bmatrix}2 & 1+3\cdot i\\i & -7\end{bmatrix}$$

conj() (Coniugato) Catalogo >

Restituisce il complesso coniugato dell'argomento.

Nota: tutte le variabili non definite vengono considerate come variabili reali.

constructMat() (Costruisci matrice) Catalog >

constructMat
(
Espr
,
Var1
,*Var2*,*numRighe*,*numColonne***)**⇒*matrice*

Restituisce una matrice sulla base degli argomenti.

Espr è un'espressione nelle variabili *Var1* e *Var2*. Gli elementi nella matrice risultante sono formati calcolando *Espr* per ciascun valore incrementato di *Var1* e *Var2*.

Var1 è incrementato automaticamente da **1** a *numRighe*. All'interno di ciascuna riga, *Var2* è incrementato da **1** a *numColonne*.

$$\text{constructMat}\left(\frac{1}{i+j},i,j,3,4\right) \qquad \begin{bmatrix} \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix}$$

CopyVar (Copia variabile) Catalog >

CopyVar *Var1*, *Var2*

CopyVar *Var1*., *Var2*.

CopyVar *Var1*, *Var2* copia il valore della variabile *Var1* nella variabile *Var2*, creando *Var2* se necessario. La variabile *Var1* deve contenere un valore.

Se *Var1* è il nome di una funzione esistente definita dall'utente, copia la definizione di quella funzione nella funzione *Var2*. La funzione *Var1* deve essere definita.

Var1 deve soddisfare i requisiti validi per i nomi di variabile oppure deve essere un'espressione indiretta che viene semplificata in un nome di variabile che soddisfa i suddetti requisiti.

Define $a(x)=\frac{1}{x}$	*Done*
Define $b(x)=x^2$	*Done*
CopyVar a,c: $c(4)$	$\frac{1}{4}$
CopyVar b,c: $c(4)$	16

CopyVar (Copia variabile) Catalog >

CopyVar *Var1.*, *Var2.* copia tutti i membri del gruppo di variabili *Var1.* nel gruppo *Var2.*, creando *Var2.* se necessario.

Var1. deve essere il nome di un gruppo di variabili esistente, come ad esempio i risultati statistici *stat.nn* o le variabili create utilizzando la funzione **LibShortcut()**. Se *Var2.* esiste già, questo comando sostituisce tutti i termini che sono comuni a entrambi i gruppi e aggiunge i termini che non esistono ancora. Se uno o più termini di *Var2.* sono bloccati, tutti i termini di *Var2.* vengono lasciati invariati

$aa.a:=45$	45
$aa.b:=6.78$	6.78
CopyVar $aa.,bb.$	*Done*
getVarInfo()	$\begin{bmatrix} aa.a & \text{"NUM"} & \text{"}\square\text{"} & 0 \\ aa.b & \text{"NUM"} & \text{"}\square\text{"} & 0 \\ bb.a & \text{"NUM"} & \text{"}\square\text{"} & 0 \\ bb.b & \text{"NUM"} & \text{"}\square\text{"} & 0 \end{bmatrix}$

corrMat() (Matrice di correlazione) Catalogo >

corrMat(*Lista1*,*Lista2*[,...[,*Lista20*]]**)**

Calcola la matrice di correlazione per la matrice affiancata [*Lista1 Lista2 . . . Lista20*].

cos() (Coseno) Tasto trig

cos(*Valore1***)⇒***valore*

cos(*Lista1***)⇒***lista*

cos(*Valore1***)** restituisce sotto forma di valore il coseno dell'argomento.

cos(*Lista1***)** restituisce una lista dei coseni di tutti gli elementi di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, l'argomento viene interpretato come angolo in gradi, gradianti o radianti. È possibile utilizzare °, G o r per escludere provvisoriamente la modalità d'angolo selezionata.

In modalità angolo in gradi:

$\cos\left(\left(\frac{\pi}{4}\right)^{r}\right)$	0.707107
$\cos(45)$	0.707107
$\cos(\{0,60,90\})$	$\{1.,0.5,0.\}$

In modalità angolo in gradianti (gradi centesimali):

$\cos(\{0,50,100\})$	$\{1.,0.707107,0.\}$

In modalità angolo in radianti:

$\cos\left(\frac{\pi}{4}\right)$	0.707107
$\cos(45°)$	0.707107

cos

In modalità angolo in radianti:

cos() (Coseno)

Tasto trig

(*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce il coseno della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare il coseno di ogni elemento.

Quando una funzione scalare f(A) opera su *matriceQuadarata1* (A), il risultato viene calcolato dall'algoritmo:

Calcola gli autovalori (λ_i) e gli autovettori (V_i) di A.

matriceQuadrata1 deve essere diagonalizzabile. Inoltre, non può avere variabili simboliche alle quali non sia stato assegnato un valore.

Forma le matrici:

$$B = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \text{ and } X = [V_1, V_2, \dots, V_n]$$

Quindi A = X B X^{-1}e f(A) = X f(B) X^{-1}. Ad esempio, cos(A) = X cos(B) X^{-1} dove:

cos (B) =

$$\begin{bmatrix} \cos(\lambda_1) & 0 & \dots & 0 \\ 0 & \cos(\lambda_2) & \dots & 0 \\ 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & \cos(\lambda_n) \end{bmatrix}$$

Tutti i calcoli vengono eseguiti in virgola mobile.

$$\cos\left(\begin{bmatrix} 1 & 5 & 3 \\ 4 & 2 & 1 \\ 6 & -2 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} 0.212493 & 0.205064 & 0.121389 \\ 0.160871 & 0.259042 & 0.037126 \\ 0.248079 & -0.090153 & 0.218972 \end{bmatrix}$$

$\cos^{-1}$() (Arcocoseno)

Tasto trig

$\cos^{-1}$(*Valore1***)⇒***valore*

$\cos^{-1}$(*Lista1***)⇒***lista*

$\cos^{-1}$(*Valore1***)** restituisce l'angolo il cui coseno è *Valore1*.

$\cos^{-1}$(*Lista1***)** restituisce la lista dell'inversa dei coseni di ciascun elemento di *Lista1*.

In modalità angolo in gradi:

$\cos^{-1}(1)$ 0.

In modalità angolo in gradianti (gradi centesimali):

$\cos^{-1}(0)$ 100.

$\cos^{-1}()$ (Arcocoseno) Tasto trig

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arccos(...)`.

In modalità angolo in radianti:

$$\cos^{-1}(\{0,0.2,0.5\}) \qquad \{1.5708,1.36944,1.0472\}$$

$\cos^{-1}$ (*matriceQuadrata1*)⇒*matriceQuadrata*

Restituisce il coseno inverso della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare il coseno inverso di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$$\cos^{-1}\left(\begin{bmatrix} 1 & 5 & 3 \\ 4 & 2 & 1 \\ 6 & -2 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} 1.73485+0.064606\cdot i & -1.49086+2.10514 \\ -0.725533+1.51594\cdot i & 0.623491+0.77836\cdot \\ -2.08316+2.63205\cdot i & 1.79018-1.27182\cdot \end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

cosh() (Coseno iperbolico) Catalogo >

cosh(*Valore1*)⇒*valore*

cosh(*Lista1*)⇒*lista*

cosh(*Valore1*) restituisce il coseno iperbolico dell'argomento.

cosh(*Lista1*) restituisce una lista dei coseni iperbolici di ciascun elemento di *Lista1*.

In modalità angolo in gradi:

$$\cosh\left(\left(\frac{\pi}{4}\right)^{r}\right) \qquad 1.74671\text{E}19$$

cosh (*matriceQuadrata1*)⇒*matriceQuadrata*

Restituisce il coseno iperbolico della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare il coseno iperbolico di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti:

$$\cosh\left(\begin{bmatrix} 1 & 5 & 3 \\ 4 & 2 & 1 \\ 6 & -2 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} 421.255 & 253.909 & 216.905 \\ 327.635 & 255.301 & 202.958 \\ 226.297 & 216.623 & 167.628 \end{bmatrix}$$

$\cosh^{-1}()$ (Arcocoseno iperbolico) Catalogo >

$\cosh^{-1}$(*Valore1*)⇒*valore*

$\cosh^{-1}$(*Lista1*)⇒*lista*

$\cosh^{-1}$(*Valore1*) restituisce l'inversa del coseno iperbolico dell'argomento.

$\cosh^{-1}$(*Lista1*) restituisce una lista dell'inversa dei coseni iperbolici di ciascun elemento di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`arccosh(...)`**.

$\cosh^{-1}(1)$ → 0

$\cosh^{-1}(\{1,2.1,3\})$ → $\{0,1.37286,\cosh^{-1}(3)\}$

$\cosh^{-1}$ (*matriceQuadrata1*)⇒*matriceQuadrata*

Restituisce l'inversa del coseno iperbolico della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare l'inversa del coseno iperbolico di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$$\cosh^{-1}\left(\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}\right)$$

$$\begin{bmatrix}2.52503+1.73485\cdot i & -0.009241-1.4908\\ 0.486969-0.725533\cdot i & 1.66262+0.623491\\ -0.322354-2.08316\cdot i & 1.26707+1.79018\end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

cot() (Cotangente) Tasto trig

cot(*Valore1*) ⇒ *valore*

cot(*Lista1*) ⇒ *lista*

Restituisce la cotangente di *Valore1* oppure restituisce una lista delle cotangenti di tutti gli elementi di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, l'argomento viene interpretato come angolo in gradi, gradianti o radianti. È possibile utilizzare °, G o r per escludere provvisoriamente la modalità d'angolo selezionata.

In modalità angolo in gradi:

$\cot(45)$ → 1.

In modalità angolo in gradianti (gradi centesimali):

$\cot(50)$ → 1.

In modalità angolo in radianti:

$\cot(\{1,2.1,3\})$ → $\{0.642093,-0.584848,-7.01525\}$

$\cot^{-1}()$ (Arcocotangente) — Tasto trig

$\cot^{-1}$(*Valore1*)⇒*valore*

$\cot^{-1}$(*Lista1*)⇒*lista*

Restituisce l'angolo la cui cotangente è *Valore1* oppure restituisce una lista contenente l'inversa delle cotangenti di ciascun elemento di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arccot(...)`.

In modalità angolo in gradi:

$\cot^{-1}(1)$	45

In modalità angolo in gradianti (gradi centesimali):

$\cot^{-1}(1)$	50

In modalità angolo in radianti:

$\cot^{-1}(1)$	.785398

coth() (Cotangente iperbolica) — Catalogo >

coth(*Valore1*)⇒*valore*

coth(*Lista1*)⇒*lista*

Restituisce la cotangente iperbolica di *Valore1* o restituisce una lista delle cotangenti iperboliche di tutti gli elementi di *Lista1*.

$\coth(1.2)$	1.19954
$\coth(\{1,3.2\})$	$\{1.31304,1.00333\}$

$\coth^{-1}()$ Arcocotangente iperbolica — Catalogo >

$\coth^{-1}$(*Valore1*)⇒*valore*

$\coth^{-1}$(*Lista1*)⇒*lista*

Restituisce la cotangente iperbolica inversa di *Valore1* oppure restituisce una lista contenente le cotangenti iperboliche inverse di ciascun elemento di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arccoth(...)`.

$\coth^{-1}(3.5)$	0.293893
$\coth^{-1}(\{-2,2.1,6\})$	$\{-0.549306,0.518046,0.168236\}$

count() Catalogo >

count(*Valore1oLista1* [,*Valore2oLista2* [,...]])⇒*valore*

Restituisce il totale accumulato di tutti gli elementi negli argomenti che danno come risultato valori numerici.

Gli argomenti possono essere un'espressione, un valore, una lista o una matrice. È possibile mischiare tipi di dati e utilizzare argomenti di varie dimensioni.

Per una lista, una matrice o un intervallo di celle, viene calcolato ciascun elemento per determinare se dovrebbe essere incluso nel conteggio.

In Foglio elettronico, è possibile utilizzare un intervallo di celle al posto di qualsiasi argomento.

Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere, pagina 212.

$\text{count}(2,4,6)$ 3

$\text{count}(\{2,4,6\})$ 3

$\text{count}\left(2,\{4,6\},\begin{bmatrix}8 & 10\\ 12 & 14\end{bmatrix}\right)$ 7

countif() Catalogo >

countif(*Lista*,*Criteri*)⇒*valore*

Restituisce il totale accumulato di tutti gli elementi di *Lista* che soddisfano i *Criteri* specificati.

Criteri può essere:

- Un valore, un'espressione o una stringa. Ad esempio, **3** conta solo quegli elementi di *Lista* che sono semplificati nel numero 3.
- Un'espressione booleana contenente il simbolo **?** come segnaposto di ciascun elemento. Ad esempio, **?<5** conta solo quegli elementi di *Lista* che sono minori di 5.

In Foglio elettronico, è possibile utilizzare un intervallo di celle al posto di *Lista*.

Gli elementi vuoti (nulli) della lista vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere, pagina 212.

$\text{countIf}(\{1,3,\text{"abc"},\text{undef},3,1\},3)$ 2

Conta il numero di elementi uguali a 3.

$\text{countIf}(\{\text{"abc"},\text{"def"},\text{"abc"},3\},\text{"def"})$ 1

Conta il numero di elementi uguali a "def".

$\text{countIf}(\{1,3,5,7,9\},?<5)$ 2

Conta 1 e 3.

$\text{countIf}(\{1,3,5,7,9\},2<?<8)$ 3

Conta 3, 5 e 7.

$\text{countIf}(\{1,3,5,7,9\},?<4 \text{ or } ?>6)$ 4

countif() Catalogo >

Nota: vedere anche **sumIf()**, pagina 161 e **frequency()**, pagina 62.

Conta 1, 3, 7 e 9.

cPolyRoots() Catalogo >

cPolyRoots(*Poli*,*Var***)⇒***lista*

cPolyRoots(*ListaDiCoeff***)⇒***lista*

La prima sintassi, **cPolyRoots(***Poli*,*Var***)**, restituisce una lista di radici complesse del polinomio *Poli* in funzione della variabile specificata *Var*.

Poli deve essere un polinomio in forma normale in una variabile. Non utilizzare polinomi non ridotti a forma normale, quali $y^2 \cdot y+1$ o $x \cdot x+2 \cdot x+1$

La seconda sintassi, **cPolyRoots (***ListaDiCoeff***)**, restituisce una lista di radici complesse per i coefficienti di *ListaDiCoeff*.

Nota: vedere anche **polyRoots()**, pagina 120.

$\text{polyRoots}(y^3+1,y)$ $\{-1\}$

$\text{cPolyRoots}(y^3+1,y)$ $\{-1,0.5-0.866025 \cdot i,0.5+0.866025 \cdot i\}$

$\text{polyRoots}(x^2+2 \cdot x+1,x)$ $\{-1,-1\}$

$\text{cPolyRoots}(\{1,2,1\})$ $\{-1,-1\}$

crossP() (Prodotto vettoriale) Catalogo >

crossP(*Lista1***,** *Lista2***)⇒***lista*

Restituisce sotto forma di lista il prodotto vettoriale di *Lista1* e *Lista2*.

Lista1 e *Lista2* devono essere uguali, 2 o 3.

crossP(*Vettore1***,** *Vettore2***)⇒***vettore*

Restituisce un vettore riga o colonna (a seconda degli argomenti) corrispondente al prodotto vettoriale di *Vettore1* per *Vettore2*.

Vettore1 e *Vettore2* devono essere entrambi vettori riga o vettori colonna. Le dimensioni di entrambi devono essere uguali, 2 o 3.

$\text{crossP}(\{0.1,2.2,-5\},\{1,-0.5,0\})$ $\{-2.5,-5.,-2.25\}$

$\text{crossP}([1\ \ 2\ \ 3],[4\ \ 5\ \ 6])$ $[-3\ \ 6\ \ -3]$

$\text{crossP}([1\ \ 2],[3\ \ 4])$ $[0\ \ 0\ \ -2]$

csc() (Cosecante) Tasto trig

csc(*Valore1***)⇒***valore*

csc(*Lista1***)⇒***lista*

Restituisce la cosecante di *Valore1* oppure restituisce una lista contenente le cosecanti di tutti gli elementi di *Lista1*.

In modalità angolo in gradi:

$\csc(45)$	1.41421

In modalità angolo in gradianti (gradi centesimali):

$\csc(50)$	1.41421

In modalità angolo in radianti:

$\csc\left(\left\{1,\frac{\pi}{2},\frac{\pi}{3}\right\}\right)$	$\{1.1884,1.,1.1547\}$

csc^{-1}() (Cosecante inversa) Tasto trig

csc^{-1}(*Valore1***)** ⇒ *valore*

csc^{-1}(*Lista1***)** ⇒ *lista*

Restituisce l'angolo la cui cosecante è *Valore1* oppure restituisce una lista contenente le cosecanti inverse di ciascun elemento di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arccsc(...)`.

In modalità angolo in gradi:

$\csc^{-1}(1)$	90

In modalità angolo in gradianti (gradi centesimali):

$\csc^{-1}(1)$	100

In modalità angolo in radianti:

$\csc^{-1}(\{1,4,6\})$	$\{1.5708,0.25268,0.167448\}$

csech (Cosecante iperbolica) Catalogo >

csch(*Valore1***)** ⇒ *valore*

csch(*Lista1***)** ⇒ *lista*

Restituisce la cosecante iperbolica di *Valore1* oppure restituisce una lista di cosecanti iperboliche di tutti gli elementi di *Lista1*.

$\mathrm{csch}(3)$	0.099822
$\mathrm{csch}(\{1,2.1,4\})$	$\{0.850918,0.248641,0.036644\}$

csch⁻¹() (Cosecante iperbolica inversa) Catalogo >

csch⁻¹(*Valore1***)** ⇒ *valore*

csch⁻¹(*Lista1***)** ⇒ *lista*

Restituisce la cosecante iperbolica inversa di *Valore1* oppure restituisce una lista contenente le cosecanti iperboliche inverse di ciascun elemento di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`arccsch(…)`**.

$\text{csch}^{-1}(1)$	0.881374
$\text{csch}^{-1}(\{1,2.1,3\})$	$\{0.881374,0.459815,0.32745\}$

CubicReg (Regressione cubica) Catalogo >

CubicReg *X*, *Y*[, [*Freq*] [, *Categoria*, *Includi*]]

Calcola la regressione polinomiale cubicay = a ·x3+b · x2+c ·x+dsulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a ·x^3+b ·x^2+c ·x+d
stat.a, stat.b, stat.c, stat.d	Coefficienti di regressione
stat.R^2	Coefficiente di determinazione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

cumulativeSum() Catalogo >

cumulativeSum(*Lista1***)⇒***lista*

Restituisce una lista delle somme cumulative degli elementi in *Lista1*, incominciando dall'elemento 1.

cumulativeSum({1,2,3,4}) {1,3,6,10}

cumulativeSum(*Matrice1***)⇒***matrice*

Restituisce una matrice delle somme cumulative degli elementi di *Matrice1*. Ciascun elemento è la somma cumulativa della colonna, dall'alto al basso.

Un elemento vuoto (nullo) in *Lista1* o *Matrice1* produce un elemento vuoto nella lista o matrice risultante. Per ulteriori informazioni sugli elementi vuoti, vedere, pagina 212.

$$\begin{bmatrix}1&2\\3&4\\5&6\end{bmatrix}\rightarrow m1 \qquad \begin{bmatrix}1&2\\3&4\\5&6\end{bmatrix}$$

$$\text{cumulativeSum}(m1) \qquad \begin{bmatrix}1&2\\4&6\\9&12\end{bmatrix}$$

Cycle (Ripeti) Catalogo >

Ripeti

Trasferisce il controllo della funzione alla iterazione immediatamente successiva del ciclo corrente (**For**, **While** o **Loop**).

Cycle non è ammesso al di fuori delle tre strutture di ciclo (**For**, **While** o **Loop**).

Funzione che elenca le somme degli interi da 1 a 100 saltando 50.

Cycle (Ripeti) Catalogo >

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define g()=Func                    Done
          Local temp,i
          0→temp
          For i,1,100,1
          If i=50
          Cycle
          temp+i→temp
          EndFor
          Return temp
          EndFunc
g()                                5000
```

▸Cylind (Forma cilindrica) Catalogo >

Vettore **▸Cylind**

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **`@>Cylind`**.

Visualizza il vettore riga o colonna nella forma cilindrica [r,∠θ, z].

Vettore deve avere esattamente tre elementi. Può essere una riga o una colonna.

```
[2  2  3]▸Cylind
                [2.82843  ∠0.785398  3.]
```

D

dbd() Catalogo >

dbd(*data1*,*data2***)⇒***valore*

Restituisce il numero di giorni tra la *data1* e la *data2* usando il metodo di conteggio dei giorni effettivi.

data1 e *data2* possono essere numeri o liste di numeri all'interno di un intervallo di date del calendario normale. Se sia *data1* che *data2* sono liste, esse devono contenere lo stesso numero di elementi.

data1 e *data2* devono essere comprese tra gli anni 1950 e 2049.

```
dbd(12.3103,1.0104)                1
dbd(1.0107,6.0107)               151
dbd(3112.03,101.04)                1
dbd(101.07,106.07)               151
```

dbd() Catalogo >

È possibile inserire le date in uno dei due formati, che differiscono esclusivamente per la posizione del punto decimale.

MM.GGAA (formata usato generalmente negli Stati Uniti)

GGMM.AA (formato usato generalmente in Europa)

▸DD (Visualizza angolo decimale) Catalogo >

Valore1 ▸**DD**⇒*valore*

Lista1 ▸**DD**⇒*lista*

Matrice1 ▸**DD**⇒*matrice*

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **@>DD**.

Restituisce l'equivalente decimale dell'argomento espresso in gradi. L'argomento è un numero, una lista o una matrice interpretata in gradianti, radianti o gradi dall'impostazione della modalità Angolo.

In modalità angolo in gradi:

$(1.5°)$▸DD	$1.5°$
$(45°22'14.3")$▸DD	$45.3706°$
$\{\{45°22'14.3",60°0'0"\}\}$▸DD	$\{45.3706°,60°\}$

In modalità angolo in gradianti (gradi centesimali):

1▸DD	$\frac{9}{10}°$

In modalità angolo in radianti:

(1.5)▸DD	$85.9437°$

▸Decimal (Decimale) Catalogo >

Valore1 ▸*Decimal*⇒*valore*

Lista1 ▸**Decimal**⇒*valore*

Matrice1 ▸**Decimal**⇒*valore*

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **@>Decimal**.

Visualizza l'argomento nella forma decimale. Questo operatore può essere utilizzato solo alla fine della riga di introduzione.

$\frac{1}{3}$▸Decimal	0.333333

Define *Var* **=** *Espressione*

Define *Funzione*(*Param1*, *Param2*, ...) **=** *Espressione*

Definisce la variabile *Var* o la funzione *Funzione* definita dall'utente.

Parametri, quali *Param1*, sono segnaposto per il passaggio di argomenti alla funzione. Quando si chiama una funzione definita dall'utente, occorre fornire argomenti (ad esempio, valori o variabili) corrispondenti ai parametri. Una volta chiamata, la funzione calcola *Espressione* utilizzando gli argomenti forniti.

Var e *Funzione* non possono essere il nome di una variabile di sistema né una funzione o un comando predefiniti.

Nota: questa forma di **Define** equivale all'esecuzione dell'espressione: *espressione* ➞ *Funzione* (*Param1,Param2*).

Define $g(x,y)=2\cdot x-3\cdot y$	*Done*
$g(1,2)$	-4
$1\rightarrow a$: $2\rightarrow b$: $g(a,b)$	-4
Define $h(x)=\text{when}(x<2,2\cdot x-3,-2\cdot x+3)$	*Done*
$h(-3)$	-9
$h(4)$	-5

Define *Funzione*(*Param1*, *Param2*, ...) **= Func**
Blocco
EndFunc

Define *Programma*(*Param1*, *Param2*, ...) **= Prgm**
Blocco
EndPrgm

In questa forma, il programma o la funzione definita dall'utente può eseguire un blocco di istruzioni multiple.

Blocco può essere una singola istruzione o una serie di istruzioni su righe separate. *Inoltre Blocco* può includere espressioni e istruzioni (quali **If**, **Then**, **Else** e **For**).

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Define $g(x,y)$=Func	*Done*
If $x>y$ Then	
Return x	
Else	
Return y	
EndIf	
EndFunc	
$g(3,-7)$	3

Define (Definisci) Catalogo >

Nota: vedere anche **Define LibPriv**, pagina 44 e **Define LibPub**, pagina 44.

```
Define g(x,y)=Prgm
              If x>y Then
              Disp x," greater than ",y
              Else
              Disp x," not greater than ",y
              EndIf
              EndPrgm
                                        Done
g(3,-7)
                          3 greater than -7
                                        Done
```

Define LibPriv (Definisci libreria privata) Catalogo >

Define LibPriv *Var* **=** *Espressione*

Define LibPriv *Funzione*(*Param1*, *Param2*, ...) **=** *Espressione*

Define LibPriv *Funzione*(*Param1*, *Param2*, ...) **= Func**
Blocco
EndFunc

Define LibPriv *Programma*(*Param1*, *Param2*, ...) **= Prgm**
Blocco
EndPrgm

Funziona come **Define**, eccetto che definisce una variabile, una funzione o un programma libreria privata. Funzioni e programmi privati non sono elencati nel Catalogo.

Nota: vedere anche **Define**, pagina 43 e **Define LibPub**, pagina 44.

Define LibPub (Definisci libreria pubblica) Catalogo >

Define LibPub *Var* **=** *Espressione*

Define LibPub *Funzione*(*Param1*, *Param2*, ...) **=** *Espressione*

Define LibPub *Funzione*(*Param1*, *Param2*,

Define LibPub (Definisci libreria pubblica) Catalogo >

...) = Func
Blocco
EndFunc

Define LibPub *Programma*(*Param1*, *Param2*, ...) **= Prgm**
Blocco
EndPrgm

Funziona come **Define**, eccetto che definisce una variabile, una funzione o un programma libreria pubblica. Funzioni e programmi pubblici vengono elencati nel Catalogo dopo che la libreria è stata salvata e aggiornata.

Nota: vedere anche **Define**, pagina 43 e **Define LibPriv**, pagina 44.

deltaList() Vedere ΔList(), pagina 88.

DelVar Catalog >

DelVar *Var1*[, *Var2*] [, *Var3*] ...

DelVar *Var*.

Elimina dalla memoria la variabile o il gruppo di variabili specificato.

Se una o più variabili sono bloccate, questo comando visualizza un messaggio di errore ed elimina solo le variabili sbloccate. Vedere **unLock, pagina 176.**

$2 \rightarrow a$	2
$(a+2)^2$	16
DelVar a	*Done*
$(a+2)^2$	"Error: Variable is not defined"

DelVar

Catalog >

DelVar *Var*. elimina tutti i membri del gruppo di variabili *Var*. (come ad esempio i risultati statistici *stat.nn* o le variabili create utilizzando la funzione **LibShortcut()**). Il punto (.) in questa forma del comando **DelVar** ne limiti l'applicabilità all'eliminazione di un gruppo di variabili; non può essere applicato alla variabile semplice *Var*.

aa.a:=45	45
aa.b:=5.67	5.67
aa.c:=78.9	78.9
getVarInfo()	$\begin{bmatrix} aa.a & \text{"NUM"} & \text{"}\square\text{"} \\ aa.b & \text{"NUM"} & \text{"}\square\text{"} \\ aa.c & \text{"NUM"} & \text{"}\square\text{"} \end{bmatrix}$
DelVar *aa*.	*Done*
getVarInfo()	"NONE"

delVoid()

Catalogo >

delVoid(*Lista1***)⇒***lista*

Restituisce una lista che ha il contenuto di *Lista1* con tutti gli elementi vuoti (nulli) rimossi.

Per ulteriori informazioni sugli elementi vuoti, vedere, pagina 212.

delVoid({1,void,3})	{1,3}

det() (Determinante)

Catalogo >

det(*matriceQuadrata***[,** *Tolleranza***])⇒***espressione*

Restituisce il determinante di *matriceQuadrata*.

In alternativa, un elemento qualsiasi della matrice viene considerato zero se il suo valore assoluto è minore di *Tolleranza*. Tale tolleranza viene utilizzata solo se la matrice contiene elementi a virgola mobile e non contiene variabili simboliche alle quali non sia stato assegnato un valore. In caso contrario, *Tolleranza* viene ignorato.

- Se si usa ctrl enter oppure se si imposta la modalità **Auto o Approssimato** su Approssimato, i calcoli verranno eseguiti in virgola mobile.
- Se *Tolleranza* viene omesso o non è utilizzato, la tolleranza predefinita viene calcolata come:

$\det\left(\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\right)$	-2
$\begin{bmatrix} 1.\text{E}20 & 1 \\ 0 & 1 \end{bmatrix} \rightarrow mat1$	$\begin{bmatrix} 1.\text{E}20 & 1 \\ 0 & 1 \end{bmatrix}$
det(*mat1*)	0
det(*mat1*,.1)	1.E20

det() (Determinante) Catalogo >

5E-14 ·**max(dim**
(*matriceQuadrata***)) ·rowNorm**
(*matriceQuadrata***)**

diag() (Diagonale) Catalogo >

diag(*Lista***)⇒***matrice*

diag(*matriceRiga***)⇒***matrice*

diag(*matriceColonna***)⇒***matrice*

Restituisce una matrice avente i valori dell'argomento lista o matrice nella diagonale principale.

$\text{diag}\left(\begin{bmatrix}2 & 4 & 6\end{bmatrix}\right)$ $\begin{bmatrix}2 & 0 & 0\\0 & 4 & 0\\0 & 0 & 6\end{bmatrix}$

diag(*matriceQuadrata***)⇒***matriceRiga*

Restituisce una matrice riga contenente gli elementi della diagonale principale di *matriceQuadrata*.

matriceQuadrata deve essere quadrata.

$\begin{bmatrix}4 & 6 & 8\\1 & 2 & 3\\5 & 7 & 9\end{bmatrix}$ $\begin{bmatrix}4 & 6 & 8\\1 & 2 & 3\\5 & 7 & 9\end{bmatrix}$

$\text{diag}(Ans)$ $\begin{bmatrix}4 & 2 & 9\end{bmatrix}$

dim() (Dimensione) Catalogo >

dim(*Lista***)⇒***intero*

Restituisce le dimensioni di *Lista*.

$\dim(\{0,1,2\})$ 3

dim(*Matrice***)⇒***lista*

Restituisce le dimensioni di Matrice nella forma di una lista a due elementi {righe, colonne}.

$\dim\left(\begin{bmatrix}1 & -1\\2 & -2\\3 & 5\end{bmatrix}\right)$ $\{3,2\}$

dim(*Stringa***)⇒***intero*

Restituisce il numero di caratteri contenuti nella stringa *Stringa*.

$\dim(\text{"Hello"})$ 5

$\dim(\text{"Hello "\&"there"})$ 11

Disp

Catalogo >

Disp *esprOrString1* [, *esprOrString2*] ...

Visualizza gli argomenti nella cronologia di *Calculator*. Gli argomenti possono essere visualizzati in successione, separati da sottili spazi.

Questo comando è utile soprattutto in programmi e funzioni per assicurare la visualizzazione dei calcoli intermedi.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define chars(start,end)=Prgm
                        For i,start,end
                        Disp i," ",char(i)
                        EndFor
                        EndPrgm
                                        Done
chars(240,243)
                                  240 ð
                                  241 ñ
                                  242 ò
                                  243 ó
                                        Done
```

▸DMS (Gradi/primi/secondi)

Catalogo >

Valore1 ▸*DMS*

Lista ▸**DMS**

Matrice ▸**DMS**

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **@>DMS**.

Interpreta l'argomento come un angolo e visualizza il numero DMS equivalente (GGGGGG°PP'SS.ss''). Per ulteriori informazioni sul formato DMS (gradi, primi, secondi) vedere °, ', '' pagina 206.

Nota: ▸quando DMS viene utilizzato in modalità angolo in radianti, converte i radianti in gradi. Se i dati inseriti sono seguiti dal simbolo dei gradi °, non verrà eseguita alcuna conversione. ▸**DMS** può essere utilizzato solo alla fine di una riga di introduzione.

In modalità angolo in gradi:

```
(45.371)▸DMS                    45°22'15.6"
({45.371,60})▸DMS               {45°22'15.6",60°}
```

dotP() (Prodotto scalare)

Catalogo >

dotP(*Lista1***,** *Lista2***)⇒***espressione*

Restituisce il prodotto scalare di due liste.

```
dotP({1,2},{5,6})                       17
```

dotP() (Prodotto scalare) Catalogo >

dotP(*Vettore1***, *Vettore2***)⇒***espressione*

Restituisce il prodotto scalare di due vettori.

Entrambi devono essere vettori riga o colonna.

dotP([1 2 3],[4 5 6])	32

E

e^() (Funzione esponenziale) Tasto e^x

e^(*Valore1***)⇒***valore*

Restituisce **e** elevato alla potenza di *Valore1*.

e^1	2.71828
e^{3^2}	8103.08

Nota: vedere anche **e modello di funzione esponenziale**, pagina 6.

Nota: premere e^x per visualizzare e^(è diverso da accedere al carattere **E** dalla tastiera.

Un numero complesso può essere inserito nella forma polare $re^{i\theta}$. Usare questa forma solo nella modalità di misurazione degli angoli in radianti; nella modalità in gradi o gradianti causa un errore del dominio.

e^(*Lista1***)⇒***lista*

Restituisce **e** elevato alla potenza di ciascun elemento di *Lista1*.

$e^{\{1,1.,0.5\}}$	$\{2.71828,2.71828,1.64872\}$

e^(*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce l'esponenziale della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare e elevato alla potenza di ciascun elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

$e^{\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}}$	$\begin{bmatrix}782.209 & 559.617 & 456.509\\ 680.546 & 488.795 & 396.521\\ 524.929 & 371.222 & 307.879\end{bmatrix}$

eff() Catalogo >

eff(*tassoNominale,CpY***)⇒***valore***

Funzione finanziaria che converte il tasso di interesse nominale *tassoNominale* in un tasso effettivo annuo, essendo *CpY* il numero di periodi di capitalizzazione per anno.

tassoNominale deve essere un numero reale e *CpY* deve essere un numero reale > 0.

Nota: vedere anche **nom()**, pagina 109.

$\text{eff}(5.75,12)$ 5.90398

eigVc() (Autovettore) Catalogo >

eigVc(*matriceQuadrata***)⇒***matrice***

Restituisce una matrice contenente gli autovettori per una *matriceQuadrata* reale o complessa, in cui ogni colonna del risultato corrisponde ad un autovalore. Tenere presente che un autovettore non è univoco; esso infatti può essere scalato per qualsiasi fattore costante. Gli autovettori vengono normalizzati, cioè se V = [x_1, x_2, ..., x_n], allora:

$x_1^2 + x_2^2 + ... + x_n^2 = 1$

matriceQuadrata viene dapprima equilibrata con similitudini fino a quando le norme di riga e colonna sono il più vicino possibili allo stesso valore. *matriceQuadrata* viene quindi ridotta nella forma superiore di Hessenberg mentre gli autovettori vengono calcolati con una scomposizione in fattori di Schur.

In modalità formato rettangolare complesso:

$$\begin{bmatrix} -1 & 2 & 5 \\ 3 & -6 & 9 \\ 2 & -5 & 7 \end{bmatrix} \rightarrow m1 \qquad \begin{bmatrix} -1 & 2 & 5 \\ 3 & -6 & 9 \\ 2 & -5 & 7 \end{bmatrix}$$

$\text{eigVc}(m1)$

$$\begin{bmatrix} -0.800906 & 0.767947 & 0 \\ 0.484029 & 0.573804+0.052258\cdot i & 0.5738 \\ 0.352512 & 0.262687+0.096286\cdot i & 0.2626 \end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

eigVl() (Autovalore) Catalogo >

eigVl(*matriceQuadrata***)⇒***lista***

Restituisce la lista degli autovalori di una *matriceQuadrata* reale o complessa.

In modalità formato rettangolare complesso:

eigVl() (Autovalore) Catalogo >

matriceQuadrata viene dapprima equilibrata con similitudini fino a quando le norme di riga e colonna sono il più vicino possibili allo stesso valore. *matriceQuadrata* viene quindi ridotta nella forma superiore di Hessenberg mentre gli autovalori vengono calcolati dalla matrice superiore di Hessenberg.

$$\begin{bmatrix} -1 & 2 & 5 \\ 3 & -6 & 9 \\ 2 & -5 & 7 \end{bmatrix} \rightarrow m1 \qquad \begin{bmatrix} -1 & 2 & 5 \\ 3 & -6 & 9 \\ 2 & -5 & 7 \end{bmatrix}$$

$$\text{eigVl}(m1)$$

$$\{-4.40941, 2.20471+0.763006\cdot i, 2.20471-0.$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

Else Vedere If, pagina 73.

ElseIf Catalogo >

If *Espressione booleana1* **Then**
  Blocco1
ElseIf *Espressione booleana2* **Then**
  Blocco2
⋮
ElseIf *Espressione booleanaN* **Then**
  BloccoN
EndIf
⋮

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define g(x)=Func
        If x≤-5 Then
        Return 5
        ElseIf x>-5 and x<0 Then
        Return -x
        ElseIf x≥0 and x≠10 Then
        Return x
        ElseIf x=10 Then
        Return 3
        EndIf
        EndFunc
                                Done
```

EndFor Vedere For, pagina 60.

EndFunc Vedere Func, pagina 64.

EndIf Vedere If, pagina 73.

EndLoop

Vedere Loop, pagina 95.

EndPrgm

Vedere Prgm, pagina 122.

EndTry

Vedere Try, pagina 168.

EndWhile

Vedere While, pagina 179.

euler ()

Catalogo >

euler(*Espr*, *Var*, *varDipendente*, {*Var0*, *VarMax*}, *varDipendente0*, *incrVar* [, *incrEulero*]**)** ⇒*matrice*

euler(*SistemaDiEspr*, *Var*, *ListaDiVarDipendenti*, {*Var0*, *VarMax*}, *ListaDiVarDipendenti0*, *incrVar* [, *incrEulero*]**)** ⇒*matrice*

euler(*ListaDiEspr*, *Var*, *ListaDiVarDipendenti*, {*Var0*, *VarMax*}, *ListaDiVarDipendenti0*, *incrVar* [, *incrEulero*]**)** ⇒*matrice*

Utilizza il metodo di Eulero per risolvere il sistema

$$\frac{d\,depVar}{d\,Var} = Expr(Var, depVar)$$

con *varDipendente* (*Var0*)=*varDipendente0* nell'intervallo [*Var0*,*VarMax*]. Restituisce una matrice la cui prima riga definisce i valori ottenuti di *Var* e la cui seconda riga definisce il valore del primo componente della soluzione per i valori di *Var* corrispondenti, e così via.

Espr è il secondo membro che definisce l'equazione differenziale ordinaria (ODE).

Equazione differenziale:

y'=0.001*y*(100-y) e y(0)=10

$$\text{euler}(0.001\cdot y\cdot(100-y),t,y,\{0,100\},10,1)$$

$$\begin{bmatrix} 0. & 1. & 2. & 3. & 4. \\ 10. & 10.9 & 11.8712 & 12.9174 & 14.042 \end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

Sistema di equazioni

$$\begin{cases} y1' = -y1 + 0.1\cdot y1\cdot y2 \\ y2' = 3\cdot y2 - y1\cdot y2 \end{cases}$$

con *y1*(0)=2 e *y2*(0)=5

$$\text{euler}\left(\begin{cases} -y1+0.1\cdot y1\cdot y2 \\ 3\cdot y2-y1\cdot y2 \end{cases},t,\{y1,y2\},\{0,5\},\{2,5\},1\right)$$

$$\begin{bmatrix} 0. & 1. & 2. & 3. & 4. & 5. \\ 2. & 1. & 1. & 3. & 27. & 243. \\ 5. & 10. & 30. & 90. & 90. & -2070. \end{bmatrix}$$

euler ()

Catalogo >

SistemaDiEspr è il sistema di secondi membri che definiscono il sistema di ODE (corrisponde all'ordine di variabili dipendenti in *ListaDiVarDipendenti*).

ListaDiEspr è una lista di secondi membri che definisce il sistema di ODE (corrisponde all'ordine di variabili dipendenti in *ListaDiVarDipendenti*).

Var è la variabile indipendente.

ListaDiVarDipendenti è una lista di variabili dipendenti.

{*Var0*, *VarMax*} è una lista a due elementi che indica la funzione di integrare da *Var0* a *VarMax*.

ListaDiVarDipendenti0 è una lista di valori iniziali di variabili dipendenti.

incrVar è un numero diverso da zero tale che **sign(***incrVar***)** = **sign(***VarMax-Var0***)** e sono restituite soluzioni a *Var0*+i·*incrVar* per tutti i valori di i=0,1,2,... tali che *Var0*+i·*incrVar* sia in [*var0,VarMax*] (potrebbe non esserci un valore di soluzione a *VarMax*).

incrEulero è un numero intero positivo (predefinito a 1) che definisce il numero di incrementi di Eulero tra i valori ottenuti. La dimensione effettiva dell'incremento utilizzato dal metodo di Eulero è *incrVar/incrEulero*.

eval ()

Menu Hub

eval(*Espr***)** ⇒ *stringa*

eval() è valido solo in TI-Innovator™ Hub Argomento del comando delle istruzioni di programmazione **Get**, **GetStr** e **Send**. Il software calcola l'espressione *Espr* e sostituisce l'istruzione **eval()** con il risultato come stringa di caratteri.

L'argomento *Espr* deve essere semplificato in un numero reale.

Impostare l'elemento blu del LED RGB su metà intensità.

lum:=127	127
Send "SET COLOR.BLUE eval(lum)"	*Done*

Reimpostare l'elemento blu su OFF.

Send "SET COLOR.BLUE OFF"	*Done*

eval () Menu Hub

L'argomento eval() deve essere semplificato in un numero reale.

```
Send "SET LED eval("4") TO ON"
                    "Error: Invalid data type"
```

Programmare per aumentare gradualmente l'elemento rosso

```
Define fadein()=
Prgm
For i,0,255,10
  Send "SET COLOR.RED eval(i)"
    Wait 0.1
  EndFor
  Send "SET COLOR.RED OFF"
EndPrgm
```

Eseguire il programma.

```
fadein()                                  Done
```

Anche se **eval()** non mostra il suo risultato, è possibile visualizzare la stringa del comando Hub risultante dopo l'esecuzione del comando controllando ciascuna delle seguenti variabili speciali.

iostr.SendAns
iostr.GetAns
iostr.GetStrAns

```
n:=0.25                                   0.25
m:=8                                      8
n·m                                       2.
Send "SET COLOR.BLUE ON TIME eval(n·m)"
                                          Done
iostr.SendAns    "SET COLOR.BLUE ON TIME 2"
```

Nota: vedere anche **Get** (pagina 66), **GetStr** (pagina 70) e **Send** (pagina 142).

Exit (Esci) Catalogo >

Exit

Permette di uscire dal blocco corrente **For**, **While** o **Loop**.

Exit non è ammesso al di fuori delle tre strutture iterative (**For**, **While** o **Loop**).

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Descrizione della funzione:

Exit (Esci) Catalogo >

```
Define g()=Func                    Done
          Local temp,i
          0→temp
          For i,1,100,1
          temp+i→temp
          If temp>20 Then
          Exit
          EndIf
          EndFor
          EndFunc
g()                                21
```

exp() (e alla potenza) Tasto e^x

exp(*Valore1***)⇒***valore*

Restituisce **e** elevato alla potenza di *Espr1*.

Restituisce **e** elevato alla potenza di *Valore1*.

Nota: vedere anche **e** modello di funzione esponenziale, pagina 6.

Un numero complesso può essere inserito nella forma polare rei θ. Usare questa forma solo nella modalità di misurazione degli angoli in radianti; nella modalità in gradi o gradianti causa un errore del dominio.

e^1 2.71828

e^{3^2} 8103.08

exp(*Lista1***)⇒***lista*

Restituisce **e** elevato alla potenza di ciascun elemento di *Lista1*.

$e^{\{1,1.,0.5\}}$ $\{2.71828,2.71828,1.64872\}$

exp (*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce l'esponenziale della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare **e** elevato alla potenza di ciascun elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

$e^{\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}}$ $\begin{bmatrix}782.209 & 559.617 & 456.509\\ 680.546 & 488.795 & 396.521\\ 524.929 & 371.222 & 307.879\end{bmatrix}$

expr() (Da stringa a espressione) Catalogo >

expr(*Stringa***)⇒***espressione*

Restituisce la stringa di caratteri contenuta in *Stringa* come espressione e la esegue subito.

"Define cube(x)=x^3" → *funcstr*	
	"Define cube(x)=x^3"
expr(*funcstr*)	*Done*
cube(2)	8

ExpReg (Regressione esponenziale) Catalogo >

ExpReg *X, Y* [, [*Freq*] [, *Categoria, Includi*]]

Calcola la regressione esponenzialey = a · (b)ˣsulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a ·(b)ˣ

Variabile di output	Descrizione
stat.a, stat.b	Coefficienti di regressione
stat.r^2	Coefficiente di determinazione lineare di dati trasformati
stat.r	Coefficiente di correlazione per dati trasformati (x, ln(y))
stat.Resid	Residui associati al modello esponenziale
stat.ResidTrans	Residui associati all'adattamento lineare dei dati trasformati
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

F

factor() (Fattorizza) Catalogo >

factor(*numeroRazionale***)** restituisce il numero razionale scomposto in fattori primi. Per i numeri composti, il tempo di elaborazione cresce in modo esponenziale secondo il numero di cifre del secondo fattore più grande. Ad esempio, la scomposizione in fattori di un intero di 30 cifre può richiedere più di un giorno, mentre la scomposizione di un numero di 100 cifre può richiedere più di un secolo.

factor(152417172689)	123457·1234577
isPrime(152417172689)	false

Per arrestare manualmente un calcolo:

- **Palmare:** Tenere premuto il tasto [on] e premere [enter] più volte.
- **Windows®:** Tenere premuto il tasto **F12** e premere **Invio** più volte.
- **Macintosh®:** Tenere premuto il tasto **F5** e premere **Invio** più volte.
- **iPad®:** L'app mostra un prompt. È possibile continuare ad attendere o annullare.

factor() (Fattorizza) Catalogo >

Se si desidera soltanto determinare se un numero è primo, utilizzare **isPrime()**. Ciò risulta molto più veloce, in particolare se *numeroRazionale* non è primo e se il secondo fattore più grande ha più di cinque cifre.

FCdf() (Probabilità di distribuzione F) Catalogo >

FCdf
(
estremoInf
,*estremoSup*,*glNumer*,*glDenom*)⇒*numero*
se *estremoInf* e *estremoSup* sono numeri,
lista se *estremoInf* e *estremoSup* sono liste

FCdf
(
estremoInf
,*estremoSup*,*glNumer*,*glDenom*)⇒*numero*
se *estremoInf* e *estremoSup* sono numeri,
lista se *estremoInf* e *estremoSup* sono liste

Calcola la probabilità di distribuzione F tra *lowBound* e *upBound* per il *dfNumero specificato* (gradi di libertà) e *dfDenom*.

Per P($X \le upBound$), impostare *lowBound* = 0.

Fill (Riempi) Catalogo >

Fill *Valore, varMatrice*⇒*matrice*

Sostituisce ciascun elemento della variabile *varMatrice* con *Valore*.

varMatrice deve esistere già.

$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} \rightarrow amatrix$ $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$

Fill 1.01,*amatrix* *Done*

amatrix $\begin{bmatrix}1.01 & 1.01\\1.01 & 1.01\end{bmatrix}$

Fill *Valore, varLista*⇒*lista*

Sostituisce ciascun elemento della variabile *varLista* con *Valore*.

varLista deve esistere già.

$\{1,2,3,4,5\} \rightarrow alist$ $\{1,2,3,4,5\}$

Fill 1.01,*alist* *Done*

alist $\{1.01,1.01,1.01,1.01,1.01\}$

FiveNumSummary Catalog >

FiveNumSummary *X*[,[*Freq*] [,*Categoria*,*Includi*]]

Fornisce una versione abbreviata delle statistiche a 1 variabile nella lista *X*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

X rappresenta una lista contenente i dati.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici di categoria dei dati corrispondenti di *X*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Un elemento vuoto (nullo) in qualsiasi lista *X*, *Freq* o *Categoria* produce un elemento vuoto corrispondente in tutte queste liste. Per ulteriori informazioni sugli elementi vuoti, vedere pagina 212

Variabile di output	Descrizione
stat.MinX	Minimo dei valori x
stat.Q_1X	1° quartile di x
stat.MedianX	Mediana di x
stat.Q_3X	3° quartile di x
stat.MaxX	Massimo dei valori x

floor() (Arrotondato per difetto) Catalogo >

floor(*Valore1***)⇒***intero*

Restituisce il numero intero più grande che è ≤ all'argomento. Questa funzione è identica a **int()**.

```
floor(-2.14)                    -3.
```

floor() (Arrotondato per difetto) Catalogo >

L'argomento può essere un numero reale o complesso.

floor(*Lista1***)⇒***lista*

floor(*Matrice1***)⇒***matrice*

Restituisce una lista o matrice del valore arrotondato per difetto di ciascun elemento.

Nota: vedere anche **ceiling()** e **int()**.

$\text{floor}\left(\left\{\frac{3}{2},0,{}^{-}5.3\right\}\right)$	$\{1,0,{}^{-}6.\}$
$\text{floor}\left(\begin{bmatrix}1.2 & 3.4\\ 2.5 & 4.8\end{bmatrix}\right)$	$\begin{bmatrix}1. & 3.\\ 2. & 4.\end{bmatrix}$

For Catalogo >

For *Var*, *Basso*, *Alto* [, *Incr*]
Blocco
EndFor

Esegue iterativamente le istruzioni di *Blocco* per ciascun valore di *Var*, da *Basso* a *Alto*, secondo incrementi pari a *Incr*.

Var non deve essere una variabile di sistema.

Incr può essere un valore positivo o negativo. Il valore predefinito è 1.

Blocco può essere una singola istruzione o una serie di istruzioni separate dal carattere ":".

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define g()=Func                                  Done
            Local tempsum,step,i
            0→tempsum
            1→step
            For i,1,100,step
            tempsum+i→tempsum
            EndFor
            EndFunc
g()                                              5050
```

format() (Formato) Catalogo >

format(*Valore*[, *stringaFormato*]**)⇒***stringa*

Restituisce *Valore* come stringa di caratteri basata sul modello di formato.

format(1.234567,"f3")	"1.235"
format(1.234567,"s2")	"1.23E0"
format(1.234567,"e3")	"1.235E0"
format(1.234567,"g3")	"1.235"
format(1234.567,"g3")	"1,234.567"
format(1.234567,"g3,r:")	"1:235"

format() (Formato) Catalogo >

stringaFormato è una stringa e deve essere espressa nella forma: “F[n]”, “S[n]”, “E[n]”, “G[n][c]”, dove le porzioni racchiuse tra parentesi [ ] sono facoltative.

F[n]: formato fisso. n rappresenta il numero di cifre da visualizzare dopo la virgola decimale.

S[n]: formato scientifico. n rappresenta il numero di cifre da visualizzare dopo la virgola decimale.

E[n]: formato tecnico. n rappresenta il numero di cifre dopo la prima cifra significativa. L’esponente è modificato secondo multipli di tre e la virgola decimale viene spostata verso destra di zero, una o due cifre.

G[n][c]: analogo al formato fisso, separa inoltre le cifre a sinistra del separatore decimale in gruppi di tre. c specifica il carattere separatore dei gruppi; il valore predefinito è la virgola. Se c è un punto, il separatore decimale viene visualizzato come virgola.

[Rc]: tutti gli indicatori precedenti possono essere seguiti dal suffisso di radice Rc, dove c è un singolo carattere che specifica che cosa sostituire al punto della radice.

fPart() Funzione parte frazionaria Catalogo >

fPart(*Espr1***)⇒***espressione*

fPart(*Lista1***)⇒***lista*

fPart(*Matrice1***)⇒***matrice*

Restituisce la parte frazionaria dell’argomento.

Per una lista o una matrice, restituisce le parti frazionarie degli elementi.

L’argomento può essere un numero reale o complesso.

fPart(⁻1.234)	⁻0.234
fPart({1,⁻2.3,7.003})	{0,⁻0.3,0.003}

FPdf() (Probabilità di distribuzione F) Catalogo >

FPdf(*valX***,***glNumer***,***glDenom***)⇒***numero* se *ValX* è un numero, *lista* se *ValX* è una lista

FPdf(*valX***,***glNumer***,***glDenom***)⇒***numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola la probabilità di distribuzione F in *valX* per il *glNumer* (gradi di libertà) e *glDenom* specificati.

freqTable▸list() Catalog >

freqTable▸list(*Lista1***,***Lista ListaInteriFreq***)⇒***lista*

Restituisce una lista contenente gli elementi di *Lista1* espansi secondo le frequenze in *ListaInteriFreq*. Questa funzione può essere utilizzata per costruire una tabella di frequenze per l'applicazione Dati e statistiche.

Lista1 può essere qualsiasi lista valida.

ListaInteriFreq deve avere la stessa dimensione di *Lista1* e deve contenere solo elementi interi non negativi. Ciascun elemento specifica il numero di volte che l'elemento corrispondente di *Lista1* verrà ripetuto nella lista dei risultati. Un valore zero esclude l'elemento corrispondente di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`freqTable@>list(…)`**.

Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

```
freqTable▸list({1,2,3,4},{1,4,3,1})
                    {1,2,2,2,2,3,3,3,4}
freqTable▸list({1,2,3,4},{1,4,0,1})
                    {1,2,2,2,2,4}
```

frequency() Catalogo >

frequency(*Lista1***,***listaContenitori***)⇒***lista*

Restituisce una lista contenente i conteggi degli elementi di *Lista1*. I conteggi sono basati su intervalli (contenitori) definiti nell'argomento *listaContenitori*.

```
datalist:={1,2,e,3,π,4,5,6,"hello",7}
    {1,2,2.71828,3,3.14159,4,5,6,"hello",7}
frequency(datalist,{2.5,4.5})        {2,4,3}
```

frequency() Catalogo >

Se *listaContenitori* è {b(1), b(2), …, b(n)}, gli intervalli specificati sono {**?**≤b(1), b(1)<**?**≤b(2),…,b(n-1)<**?**≤b(n), b(n)>**?**}. La lista risultante è un elemento più lungo di *listaContenitori*.

Ciascun elemento del risultato corrisponde al numero di elementi di *Lista1* che rientrano nell'intervallo di quel contenitore. Espresso nei termini della funzione **countIf ()**, il risultato è { countIf(list, **?**≤b(1)), countIf(list, b(1)<**?**≤b(2)), …, countIf(list, b (n-1)<**?**≤b(n)), countIf(list, b(n)>**?**)}.

Gli elementi di *Lista1* che non possono essere "inseriti in un contenitore" vengono ignorati. Gli elementi vuoti (nulli) vengono ugualmente ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

In Foglio elettronico, è possibile utilizzare un intervallo di celle al posto di entrambi gli argomenti.

Nota: vedere anche **countIf()**, pagina 36.

Spiegazione del risultato:

2 elementi di *Datalist* sono ≤2.5

4 elementi di *Datalist* sono> 2.5 e ≤4.5

3 elementi di *Datalist* sono >4.5

L'elemento "hello" è una stringa e non può essere collocata in alcun contenitore definito.

FTest_2Samp (Verifica F su due campioni) Catalogo >

FTest_2Samp *Lista1*,*Lista2*[,*Freq1*[,*Freq2*[,*Ipotesi*]]]

FTest_2Samp *Lista1*,*Lista2*[,*Freq1*[,*Freq2*[,*Ipotesi*]]]

(Input lista dati)

FTest_2Samp *sx1*,*n1*,*sx2*,*n2*[,*Ipotesi*]

FTest_2Samp *sx1*,*n1*,*sx2*,*n2*[,*Ipotesi*]

(Input statistiche riepilogo)

Consente di eseguire una verifica F su due campioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Per H_a: σ1 > σ2, impostare *Ipotesi*>0

FTest_2Samp (Verifica F su due campioni) Catalogo >

Per H_a: σ1 ≠ σ2 (default), impostare *Ipotesi* =0

Per H_a: σ1 < σ2, impostare *Ipotesi*<0

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.F	Statistica Û calcolata per la sequenza di dati
stat.PVal	Livello minimo di significatività in corrispondenza del quale l’ipotesi nulla può essere rifiutata
stat.dfNumer	Gradi di libertà del numeratore = n1-1
stat.dfDenom	Gradi di libertà del denominatore = n2-1
stat.sx1, stat.sx2	Deviazioni standard dei campioni delle sequenze di dati di *Lista 1* e *Lista 2*
stat.x1_bar stat.x2_bar	Medie dei campioni delle sequenze di dati di *Lista 1* e *Lista 2*
stat.n1, stat.n2	Dimensione dei campioni

Func Catalogo >

Func
Blocco
EndFunc

Modello per la creazione di una funzione definita dall’utente.

Blocco può essere una singola istruzione, una serie di istruzioni separate dal carattere “:” o una serie di istruzioni su righe separate. La funzione può utilizzare l’istruzione **Return** per restituire un dato risultato.

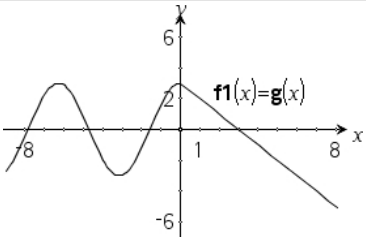
Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Definizione di una funzione Piecewise (definita a tratti):

```
Define g(x)=Func                    Done
        If x<0 Then
        Return 3·cos(x)
        Else
        Return 3−x
        EndIf
        EndFunc
```

Risultato della rappresentazione grafica g(x)

Func — Catalogo >



G

gcd() (Massimo comun divisore) — Catalogo >

gcd(*Numero1, Numero2***)⇒***espressione*

Restituisce il massimo comune divisore (gcd) dei due argomenti. Il **gcd** di due frazioni è il **massimo comune divisore** dei rispettivi numeratori diviso per il **minimo comune multiplo (lcm)** dei loro denominatori.

In modalità Auto o Approssimato, il **gcd** di numeri decimali in virgola mobile è 1.0.

$\gcd(18,33)$ → 3

gcd(*Lista1, Lista2***)⇒***lista*

Restituisce i massimi comuni divisori degli elementi corrispondenti in *Lista1* e *Lista2*.

$\gcd(\{12,14,16\},\{9,7,5\})$ → $\{3,7,1\}$

gcd(*Matrice1, Matrice2***)⇒***matrice*

Restituisce i massimi comuni divisori degli elementi corrispondenti in Matrice1 e *Matrice2*.

$\gcd\left(\begin{bmatrix}2 & 4\\6 & 8\end{bmatrix},\begin{bmatrix}4 & 8\\12 & 16\end{bmatrix}\right)$ → $\begin{bmatrix}2 & 4\\6 & 8\end{bmatrix}$

geomCdf() (Probabilità geometrica cumulata) — Catalogo >

geomCdf(*p***,***valoreInferiore***,***valoreSuperiore***)⇒***numero*

se *valoreInferiore* e *valoreSuperiore* sono numeri, *lista* se *valoreInferiore* e *valoreSuperiore* sono liste

geomCdf() (Probabilità geometrica cumulata) Catalogo >

geomCdf(*p*,*valoreSuperiore***)**per P(1≤X ≤*valoreSuperiore*) ⇒*numero* se *valoreSuperiore* è un numero, *lista* se *valoreSuperiore* è una lista

Calcola una probabilità geometrica cumulata da *valoreInferiore* a *valoreSuperiore* con la probabilità di esiti favorevoli *p* specificata.

Per P(X ≤ *valoreSuperiore*), impostare *valoreSuperiore* = 1.

geomPdf() (Probabilità per la distribuzione geometrica discreta) Catalogo >

geomPdf(*p*,*valX***)**⇒*numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola una probabilità in corrispondenza di *ValX*, il numero della prova in cui si è verificato il primo caso favorevole, per la distribuzione geometrica discreta con la probabilità di esiti favorevoli p specificata.

Get Menu Hub

Get[*promptString*,]*var*[, *statusVar*]

Get[*promptString*,] *funz*(*arg1*, ...*argn*) [, *statusVar*]

Istruzione di programmazione: recupera un valore da uno collegato TI-Innovator™ Hub e assegna il valore alla variabile *var*.

Il valore deve essere obbligatorio:

- In anticipo, attraverso un comando **Send "LEGGI ..."** .

  — oppure —

- Incorporando una richiesta **"LEGGI ..."** come argomento *promptString* opzionale. Questo metodo consente di utilizzare un solo comando per richiedere il valore e recuperarlo.

Esempio: richiedere il valore attuale del sensore di livello luminosità integrato dell'hub. Utilizzare **Get** per recuperare il valore e assegnarlo alla variabile *lightval*.

Send "READ BRIGHTNESS"	*Done*
Get *lightval*	*Done*
lightval	0.347922

Incorporare la richiesta LEGGI all'interno del comando **Get**.

Get "READ BRIGHTNESS",*lightval*	*Done*
lightval	0.378441

Get — Menu Hub

Avviene una semplificazione implicita. Per esempio, una stringa ricevuta di "123" viene interpretata come valore numerico. Per preservare la stringa, utilizzare **GetStr** invece di **Get**.

Se si include l'argomento opzionale *statusVar*, viene assegnato un valore basato sul successo dell'operazione. Un valore di zero significa che non è stato ricevuto nessun dato.

Nella seconda sintassi, l'argomento *funz*() consente a un programma di memorizzare la stringa ricevuta come definizione di una funzione. Questa sintassi opera come se il programma avesse eseguito il comando:

Definire *funz*(*arg1*, ...*argn*) = *stringa ricevuta*

Il programma può quindi usare la funzione definita *funz*().

Nota: è possibile utilizzare il comando **Get** all'interno di un programma definito dall'utente, ma non di una funzione.

Nota: vedere anche **GetStr**, pagina 70 e **Send**, pagina 142.

getDenom() (Ottieni/restituisci denominatore) — Catalogo >

getDenom(*Frazione1***)⇒***valore*

Trasforma l'argomento in un'espressione con denominatore comune ridotto e successivamente restituisce il denominatore.

$x:=5: y:=6$	6
$\text{getDenom}\left(\frac{x+2}{y-3}\right)$	3
$\text{getDenom}\left(\frac{2}{7}\right)$	7
$\text{getDenom}\left(\frac{1}{x}+\frac{y^2+y}{y^2}\right)$	30

getLangInfo() (Ottieni informazioni sulla lingua) — Catalog >

getLangInfo()⇒*stringa*

getLangInfo()	"en"

getLangInfo() (Ottieni informazioni sulla lingua)

Catalog >

Restituisce una stringa che corrisponde all'abbreviazione della lingua attiva corrente. Può essere utilizzato, ad esempio, in un programma o in una funzione per determinare la lingua corrente.

Inglese = "en"

Danese = "da"

Tedesco = "de"

Finlandese = "fi"

Francese = "fr"

Italiano = "it"

Olandese = "nl"

Olandese - Belgio = "nl_BE"

Norvegese = "no"

Portoghese = "pt"

Spagnolo = "es"

Svedese = "sv"

getLockInfo()

Catalogo >

getLockInfo(*Var*)⇒*valore*

Restituisce lo stato bloccato/sbloccato corrente della variabile *Var*.

valore =**0**: *Var* è sbloccata o non esiste.

valore =**1**: *Var* è bloccata e non può essere modificata o eliminata.

Vedere **Lock**, pagina 91 e **unLock**, pagina 176.

a:=65	65
Lock *a*	*Done*
getLockInfo(*a*)	1
a:=75	"Error: Variable is locked."
DelVar *a*	"Error: Variable is locked."
Unlock *a*	*Done*
a:=75	75
DelVar *a*	*Done*

getMode() Catalogo >

getMode(*interoNomeModo***)⇒***valore***

getMode(0)⇒*lista***

getMode(*interoNomeModo***)** restituisce un valore che rappresenta l'impostazione corrente del modo *interoNomeModo*.

getMode(0) restituisce una lista contenente coppie di numeri. Ciascuna coppia è costituita da un numero intero per il modo e da un numero intero per l'impostazione.

Per un elenco dei modi e delle relative impostazioni, vedere la tabella seguente.

Se si salvano le impostazioni con **getMode (0) →** *var*, è possibile utilizzare **setMode (***var***)** in una funzione o in un programma per ripristinare temporaneamente le impostazioni solo all'interno dell'esecuzione della funzione o del programma. Vedere **setMode()**, pagina 145.

getMode(0)	{1,7,2,1,3,1,4,1,5,1,6,1,7,1}
getMode(1)	7
getMode(7)	1

Nome modo	Intero modo	Interi impostazioni
Mostra cifre (Display digits)	1	**1**=Mobile, **2**=Mobile1, **3**=Mobile2, **4**=Mobile3, **5**=Mobile4, **6**=Mobile5, **7**=Mobile6, **8**=Mobile7, **9**=Mobile8, **10**=Mobile9, **11**=Mobile10, **12**=Mobile11, **13**=Mobile12, **14**=Fissa0, **15**=Fissa1, **16**=Fissa2, **17**=Fissa3, **18**=Fissa4, **19**=Fissa5, **20**=Fissa6, **21**=Fissa7, **22**=Fissa8, **23**=Fissa9, **24**=Fissa10, **25**=Fissa11, **26**=Fissa12
Angolo (Angle)	2	**1**=Radianti, **2**=Gradi, **3**=Gradianti
Formato esponenziale (Exponential Format)	3	**1**=Normale, **2**=Scientifico, **3**=Tecnico
Reale o Complesso (Real or Complex)	4	**1**=Reale, **2**=Rettangolare, **3**=Polare
Auto o Approssimato (Auto or Approx.)	5	**1**=Auto, **2**=Approssimato

Nome modo	Intero modo	Interi impostazioni
Formato vettoriale (Vector Format)	6	**1**=Rettangolare, **2**=Cilindrico, **3**=Sferico
Base	7	**1**=Decimale, **2**=Esadecimale, **3**=Binario

getNum() (Ottieni/restituisci numeratore)

Catalogo >

getNum(*Frazione1***)⇒***valore*

Trasforma l'argomento in un'espressione con denominatore comune ridotto e successivamente restituisce il numeratore.

$x:=5: y:=6$	6
$\text{getNum}\left(\frac{x+2}{y-3}\right)$	7
$\text{getNum}\left(\frac{2}{7}\right)$	2
$\text{getNum}\left(\frac{1}{x}+\frac{1}{y}\right)$	11

GetStr

Menu Hub

GetStr[*promptString*,] *var*[**,** *statusVar*]

GetStr[*promptString*,] *funz*(*arg1*, ...*argn*) [**,** *statusVar*]

Istruzione di programmazione: funziona allo stesso modo del comando **Get**, tranne per il fatto che il valore ricevuto viene sempre interpretato come una stringa. A differenza di questo, il comando **Get** interpreta la risposta come espressione a meno che non sia racchiusa tra virgolette ("").

Nota: vedere anche **Get**, pagina 66 e **Send**, pagina 142.

Per degli esempi vedere **Get**.

getType() Catalogo >

getType(*var***)⇒***stringa*

Restituisce una stringa che indica il tipo di dati della variabile *var*.

Se *var* non è stata definita, restituisce la stringa "NONE" (NESSUNA).

$\{1,2,3\}\rightarrow temp$	$\{1,2,3\}$
getType(*temp*)	"LIST"
$3\cdot i\rightarrow temp$	$3\cdot i$
getType(*temp*)	"EXPR"
DelVar *temp*	*Done*
getType(*temp*)	"NONE"

getVarInfo() (Ottieni informazioni variabile) Catalog >

getVarInfo()⇒*matrice* o *stringa*

getVarInfo (*StringaNomeLibreria***)⇒***matrice* o *stringa*

getVarInfo() restituisce una matrice di informazioni (nome di variabile, tipo, accessibilità della libreria e stato bloccato/sbloccato) per tutte le variabili e gli oggetti libreria definiti nell'attività corrente.

Se non ci sono variabili definite, **getVarInfo ()** restituisce la stringa "NONE".

getVarInfo (*StringaNomeLibreria***)**restituisce una matrice di informazioni per tutti gli oggetti libreria definiti nella libreria *StringaNomeLibreria*. *StringaNomeLibreria* deve essere una stringa (testo racchiuso tra virgolette) o una variabile stringa.

Se la libreria *StringaNomeLibreria* non esiste, si produce un errore.

getVarInfo()	"NONE"
Define x=5	*Done*
Lock x	*Done*
Define LibPriv $y=\{1,2,3\}$	*Done*
Define LibPub $z(x)=3\cdot x^2-x$	*Done*
getVarInfo()	$\begin{bmatrix} x & \text{"NUM"} & \text{"[ ]"} & 1 \\ y & \text{"LIST"} & \text{"LibPriv "} & 0 \\ z & \text{"FUNC"} & \text{"LibPub "} & 0 \end{bmatrix}$
getVarInfo(*tmp3*)	"Error: Argument must be a string"
getVarInfo("tmp3")	$\begin{bmatrix} volcyl2 & \text{"NONE"} & \text{"LibPub "} & 0 \end{bmatrix}$

getVarInfo() (Ottieni informazioni variabile)

Catalog >

Notare l'esempio sulla sinistra, in cui il risultato di **getVarInfo()** è assegnato alla variabile *vs*. Se si tenta di visualizzare la riga 2 o la riga 3 di *vs* viene restituito un errore "Invalid list or matrix (Lista o matrice non valida)" perché almeno uno degli elementi di queste righe (ad esempio, variabile *b*) viene ricalcolato in una matrice.

Questo errore potrebbe ripresentarsi quando si utilizza *Ans* per ricalcolare un risultato di **getVarInfo()**.

Questo errore viene generato perché la versione corrente del software non supporta una struttura di matrice generalizzata quando un elemento di una matrice può essere o una matrice o una lista.

```
a:=1                                   1
b:=[1 2]                             [1 2]
c:=[1 3 7]                         [1 3 7]
vs:=getVarInfo()   [a "NUM" "[ ]" 0
                    b "MAT" "[ ]" 0
                    c "MAT" "[ ]" 0]
vs[1]                [1 "NUM" "[ ]" 0]
vs[1,1]                                1
vs[2]        "Error: Invalid list or matrix"
vs[2,1]                              [1 2]
```

Goto (Vai a)

Catalogo >

Goto *nomeEtichetta*

Trasferisce il controllo all'etichetta *nomeEtichetta*.

nomeEtichetta deve essere definito nella stessa funzione mediante un'istruzione **Lbl**.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define g()=Func                        Done
          Local temp,i
          0→temp
          1→i
          Lbl top
          temp+i→temp
          If i<10 Then
          i+1→i
          Goto top
          EndIf
          Return temp
          EndFunc
g()                                      55
```

▸Grad (Gradianti o Gradi centesimali)

Catalogo >

Espr1 ▸ **Grad**⇒*espressione*

Converte *Espr1* in una misura di angolo in gradianti.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@>Grad`.

In modalità angolo in gradi:

```
(1.5)▸Grad                    (1.66667)g
```

In modalità angolo in radianti:

▸Grad (Gradianti o Gradi centesimali) Catalogo >

(1.5)▸Grad	(95.493)^g

I

identity() (Identità) Catalogo >

identity(*Intero***)⇒***matrice*

Restituisce la matrice identità con la dimensione di *Intero*.

Intero deve essere un numero intero positivo.

identity(4)	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

If Catalogo >

If *Espressione booleana*
Istruzione

If *Espressione booleana* **Then**
Blocco
EndIf

Se il valore di *Espressione booleana* è vero, viene eseguita la singola istruzione *Istruzione* o il blocco di istruzioni *Blocco* prima di procedere con l'esecuzione.

Se il valore di *Espressione booleana* è falso, la funzione continua senza eseguire l'istruzione o il blocco di istruzioni.

Blocco può essere una singola espressione o una serie di espressioni separate dal carattere ":".

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Define $g(x)$=Func
If $x<0$ Then
Return x^2
EndIf
EndFunc
	Done

$g(-2)$	4

If

Catalogo >

If *Espressione booleana* **Then**
Blocco1
Else
Blocco2
EndIf

Se il valore di *Espressione booleana* è vero, viene eseguito *Blocco1* e successivamente viene saltato *Blocco2*.

Se il valore di *Espressione booleana* è falso, viene saltato *Blocco1*, ma viene eseguito *Blocco2*.

Blocco1 e *Blocco2* possono essere una singola istruzione.

```
Define g(x)=Func                    Done
            If x<0 Then
            Return ⁻x
            Else
            Return x
            EndIf
            EndFunc
g(12)                               12
g(⁻12)                              12
```

If *Espressione booleana1* **Then**
Blocco1
ElseIf *Espressione booleana2* **Then**
Blocco2
⋮
ElseIf *Espressione booleanaN* **Then**
BloccoN
EndIf

Permette il passaggio a un'altra funzione. Se il valore di *Espressione booleana1* è vero, viene eseguito *Blocco1*. Se il valore di *Espressione booleana1* è falso, viene calcolata *Espressione boolenana2*, ecc.

```
Define g(x)=Func
            If x<⁻5 Then
            Return 5
            ElseIf x>⁻5 and x<0 Then
            Return ⁻x
            ElseIf x≥0 and x≠10 Then
            Return x
            ElseIf x=10 Then
            Return 3
            EndIf
            EndFunc
                                    Done
g(⁻4)                               4
g(10)                               3
```

ifFn()

Catalogo >

ifFn(*EsprBooleana***,***Valore_se_vero* **[,***Valore_se_falso* **[,***Valore_se_sconosciuto***]])⇒***espressione, lista o matrice*

Calcola l'espressione booleana *EsprBooleana* (o ciascun elemento di *EsprBooleana*) e produce un risultato sulla base delle seguenti regole:

- *EsprBooleana* può verificare un singolo valore, una lista o una matrice.
- Se un elemento di *EsprBooleana* dà come risultato una condizione vera,

```
ifFn({1,2,3}<2.5,{5,6,7},{8,9,10})
                                    {5,6,10}
```

Il valore di verifica di **1** è minore di 2.5, così l'elemento

Valore_se_vero corrispondente di **5** viene copiato nella lista del risultato.

Il valore di verifica di **2** è minore di 2.5, così l'elemento

viene restituito l'elemento corrispondente di *Valore_se_vero*.

- Se un elemento di *EsprBooleana* dà come risultato una condizione falsa, viene restituito l'elemento corrispondente di *Valore_se_falso*. Se *Valore_se_falso* è omesso, viene restituito undef.
- Se un elemento di *EsprBooleana* non dà come risultato una condizione vera né una condizione falsa, viene restituito l'elemento corrispondente di *Valore_se_sconosciuto*. Se *Valore_se_sconosciuto* è omesso, viene restituito undef.
- Se il secondo, terzo o quarto argomento della funzione **ifFn()** è una singola espressione, la verifica booleana viene applicata a tutte le posizioni in *EsprBooleana*.

Nota: se l'istruzione *EsprBooleana* semplificata implica una lista o una matrice, tutti gli altri argomenti della lista o della matrice devono avere uguali dimensioni e il risultato avrà uguali dimensioni.

Valore_se_vero corrispondente di **6** viene copiato nella lista del risultato.

Il valore di verifica di **3** è minore di 2.5, così l'elemento *Valore_se_falso* corrispondente di **10** viene copiato nella lista del risultato.

$\text{ifFn}(\{1,2,3\}<2.5,4,\{8,9,10\})$ $\{4,4,10\}$

Valore_se_vero è un singolo valore e corrisponde a qualsiasi posizione selezionata.

$\text{ifFn}(\{1,2,3\}<2.5,\{5,6,7\})$ $\{5,6,\text{undef}\}$

Valore_se_falso non è specificato. Viene utilizzato Undef.

$\text{ifFn}(\{2,\text{"a"}\}<2.5,\{6,7\},\{9,10\},\text{"err"})$ $\{6,\text{"err"}\}$

Un elemento selezionato da *Valore_se_vero*. Un elemento selezionato da *Valore_se_sconosciuto*.

imag() (Parte immaginaria) Catalogo >

imag(*Valore1***)⇒***valore*

Restituisce la parte immaginaria dell'argomento.

Nota: tutte le variabili non definite vengono considerate come variabili reali. vedere anche **real()**, pagina 130

$\text{imag}(1+2\cdot i)$ 2

imag(*Lista1***)⇒***lista*

Restituisce una lista delle parti immaginarie degli elementi.

$\text{imag}(\{-3,4-i,i\})$ $\{0,-1,1\}$

imag(*Matrice1***)⇒***matrice*

$\text{imag}\left(\begin{bmatrix} 1 & 2 \\ i\cdot 3 & i\cdot 4 \end{bmatrix}\right)$ $\begin{bmatrix} 0 & 0 \\ 3 & 4 \end{bmatrix}$

imag() (Parte immaginaria) Catalogo >

Restituisce una matrice delle parti immaginarie degli elementi.

Indirection (Conversione indiretta) Vedere #(), pagina 203.

inString() (All'interno della stringa) Catalogo >

inString(*stringaRicerca***,** *sottoStringa***[,** *Inizio***])⇒***intero*

Restituisce la posizione del carattere nella stringa *stringaRicerca* dal quale inizia per la prima vola la stringa *sottoStringa*.

Inizio, se incluso, specifica la posizione del carattere all'interno di *stringaRicerca* in cui comincia la stringa. L'impostazione predefinita è 1 (il primo carattere di *stringaRicerca*).

Se *stringaRicerca* non contiene *sottoStringa* o *Inizio* è > della lunghezza di *stringaRicerca*, viene restituito zero.

```
inString("Hello there","the")        7
inString("ABCEFG","D")               0
```

int() (Funzione parte intera) Catalogo >

int(*Valore***)⇒***intero*

int(*Lista1***)⇒***lista*

int(*Matrice1***)⇒***matrice*

Restituisce il più grande dei numeri interi che è minore o uguale all'argomento. Questa funzione è identica a **floor()**.

L'argomento può essere un numero reale o complesso.

Con una lista o con una matrice, restituisce il numero intero maggiore di ciascuno degli elementi.

```
int(-2.5)                            -3.
int([-1.234  0  0.37])               [-2.  0  0.]
```

intDiv() Divisione intera

Catalogo >

intDiv(*Numero1***, *Numero2*)⇒*intero***

intDiv(*Lista1***, *Lista2*)⇒*lista***

intDiv(*Matrice1***, *Matrice2*)⇒*matrice***

Restituisce la parte intera di (*Numero1* ÷ *Numero2*).

Con liste e matrici, restituisce la parte intera di (argomento 1 ÷ argomento 2) per ciascuna coppia di elementi.

```
intDiv(-7,2)                                  -3
intDiv(4,5)                                    0
intDiv({12,-14,-16},{5,4,-3})          {2,-3,5}
```

∫➜ παγ α ♦▲▲-

interpolate ()

Catalogo >

interpolate(*valoreX***, *listaX***, *listaY***, *listaYprime***)⇒*lista***

Questa funzione consente di fare quanto segue:

Dati *listaX*, *listaY*=**f(***listaX***)** e *listaYprime*=**f'(***listaX***)** per una funzione **f** non nota, viene utilizzato una interpolante cubica per approssimare la funzione **f** al *valoreX*. Si assume che *listaX* è una lista di numeri monotonicamente crescenti o decrescenti, ma questa funzione può restituire un valore anche quando non c'è. Questa funzione scorre *listaX* cercando un intervallo [*listaX*[i], *listaX*[i+1]] che contenga *valoreX*. Se trova un intervallo di questo tipo, restituisce un valore interpolato per **f(***valoreX***)**; altrimenti, restituisce **undef.**

listaX, *listaY* e *listaPrimiY* devono avere la stessa dimensione ≥ 2 e contenere espressioni che vengono semplificate in numeri.

valoreX può essere un numero o una lista di numeri.

Equazione differenziale:

$y'=-3\cdot y+6\cdot t+5$ e $y(0)=5$

```
rk:=rk23(-3·y+6·t+5,t,y,{0,10},5,1)
[0.      1.        2.        3.        4.        ▸
 5.  3.19499  5.00394  6.99957  9.00593  10
```

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

Utilizzare la funzione interpolate() per calcolare i valori della funzione per listavalorix:

```
xvaluelist:=seq(i,i,0,10,0.5)
{0,0.5,1.,1.5,2.,2.5,3.,3.5,4.,4.5,5.,5.5,6.,6.5,▸
xlist:=mat▸list(rk[1])
                    {0.,1.,2.,3.,4.,5.,6.,7.,8.,9.,10.}
ylist:=mat▸list(rk[2])
{5.,3.19499,5.00394,6.99957,9.00593,10.9978▸
yprimelist:=-3·y+6·t+5|y=ylist and t=xlist
{-10.,1.41503,1.98819,2.00129,1.98221,2.006▸
interpolate(xvaluelist,xlist,ylist,yprimelist)
{5.,2.67062,3.19499,4.02782,5.00394,6.00011▸
```

invχ2() Catalogo >

invχ2(*Area*,*gl***)**

invChi2(*Area*,*gl***)**

Calcola la funzione della probabilità χ^2 (chi quadrato) cumulativa inversa specificata dal grado di libertà, *gl*, per una data *Area* sotto la curva.

invF() (Distribuzione F cumulativa inversa) Catalogo >

invF(*Area*,*glNumer*,*glDenom***)**

invF(*Area*,*glNumer*,*glDenom***)**

Calcola la funzione della distribuzione F cumulativa inversa specificata da *glNumer* e *glDenom* per una data *Area* sotto la curva.

invNorm() (Distribuzione normale cumulativa inversa) Catalogo >

invNorm(*area*[,μ[,σ]]**)**

Calcola la funzione della distribuzione normale cumulativa inversa per una data *Area* sotto la curva della distribuzione normale specificata da μ e σ.

invt() (Funzione della probabilità t di Student) Catalogo >

invt(*Area*,*gl***)**

Calcola la funzione della probabilità t di Student inversa, specificata dal grado di libertà, *gl* per una data *Area* sotto la curva.

iPart() (Parte intera) Catalogo >

iPart(*Numero***)⇒***intero*

iPart(*Lista1***)⇒***lista*

iPart(*Matrice1***)⇒***matrice*

$\text{iPart}(-1.234)$ $\quad -1.$

$\text{iPart}\left(\left\{\frac{3}{2},-2.3,7.003\right\}\right)$ $\quad \{1.,-2.,7.\}$

iPart() (Parte intera) Catalogo >

Restituisce la parte intera dell'argomento.

Per una lista o per una matrice, restituisce la parte intera di ciascun argomento.

L'argomento può essere un numero reale o complesso.

irr() Catalogo >

irr(*CF0,CFLista* [,*CFFreq*]**)**⇒*valore*

Funzione finanziaria che calcola l'indice di rendimento interno di un investimento (Internal Rate of Return).

CF0 è il cash flow iniziale al tempo 0; deve essere un numero reale.

CFLista è una lista di importi di cash flow dopo il cash flow iniziale CF0.

CFFreq è una lista opzionale in cui ciascun elemento specifica la frequenza di occorrenza di un importo di cash flow raggruppato (consecutivo), che è l'elemento corrispondente di *CFLista*. L'impostazione predefinita è 1; è possibile inserire valori che siano numeri interi positivi < 10000.

Nota: vedere anche **mirr()**, pagina 100.

list1:={6000,⁻8000,2000,⁻3000}	{6000,⁻8000,2000,⁻3000}
list2:={2,2,2,1}	{2,2,2,1}
irr(5000,*list1*,*list2*)	⁻4.64484

isPrime() (Numero primo) Catalogo >

isPrime(*Numero***)**⇒*espressione costante booleana*

Restituisce vero o falso per indicare se *numero* è un numero intero ≥ 2 divisibile solo per se stesso e per 1.

Se *Numero* ha più di 306 cifre e non ha fattori ≤1021, **isPrime(***Numero***)** visualizza un messaggio di errore.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

isPrime(5)	true
isPrime(6)	false

Funzione per trovare il numero primo successivo dopo il numero specificato:

isPrime() (Numero primo) Catalogo >

```
Define nextprim(n)=Func                    Done
                   Loop
                   n+1→n
                   If isPrime(n)
                   Return n
                   EndLoop
                   EndFunc
nextprim(7)                                11
```

isVoid() Catalogo >

isVoid(*Var***)⇒***espressione costante booleana*

isVoid(*Espr***)⇒***espressione costante booleana*

isVoid(*Lista***)⇒***lista di espressioni costanti booleane*

Restituisce vero o falso per indicare se l'argomento è un tipo di dati vuoto.

Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

```
a:=_                                       _
isVoid(a)                                  true
isVoid({1,_,3})                            {false,true,false}
```

L

Lbl (Etichetta) Catalogo >

Lbl *nomeEtichetta*

Definisce un'etichetta chiamata *nomeEtichetta* in una funzione.

È possibile utilizzare un'istruzione **Goto** *nomeEtichetta* per trasferire il controllo del programma all'istruzione immediatamente successiva all'etichetta.

nomeEtichetta deve soddisfare gli stessi requisiti validi per i nomi delle variabili.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define g()=Func                            Done
           Local temp,i
           0→temp
           1→i
           Lbl top
           temp+i→temp
           If i<10 Then
           i+1→i
           Goto top
           EndIf
           Return temp
           EndFunc
g()                                        55
```

lcm() (Minimo comune multiplo) Catalogo >

lcm(*Numero1***, *Numero2***)⇒***espressione*

lcm(*List1***, *Lista2***)⇒***lista*

lcm(*Matrice1***, *Matrice2***)⇒***matrice*

Restituisce il minimo comune multiplo (lcm) di due argomenti. Il **lcm** di due frazioni è il **minimo comune multiplo** dei loro numeratori diviso per il **massimo comune divisore (gcd)** dei loro denominatori. Il **lcm** dei numeri frazionari a virgola mobile è il loro prodotto.

Per due liste o matrici, restituisce i minimi comuni multipli dei corrispondenti elementi.

$\text{lcm}(6,9)$ 18

$\text{lcm}\left(\left\{\frac{1}{3},-14,16\right\},\left\{\frac{2}{15},7,5\right\}\right)$ $\left\{\frac{2}{3},14,80\right\}$

left() (Sinistra) Catalogo >

left(*stringaOrigine***[, *Num***])⇒***stringa*

Restituisce i caratteri *Num* più a sinistra contenuti nella stringa di caratteri *stringaOrigine*.

Se si omette *Num*, restituisce l'intera *stringaOrigine*.

left(*Lista1***[, *Num***])⇒***lista*

Restituisce gli elementi *Num* più a sinistra contenuti in *Lista1*.

Se si omette *Num*, restituisce l'intera *Lista1*.

left(*Confronto***)⇒***espressione*

Restituisce il primo membro di un'equazione o di una disequazione.

left("Hello",2) "He"

$\text{left}(\{1,3,-2,4\},3)$ $\{1,3,-2\}$

libShortcut() (Collegamento a libreria) Catalogo >

libShortcut(*stringaNomeLibr***, *stringaNomeCollegamento***[, *LibPrivFlag***])⇒***lista di variabili*

Questo esempio presuppone un documento libreria memorizzato e aggiornato, denominato **linalg2**, che contiene oggetti definiti come *clearmat*, *gauss1* e *gauss2*.

libShortcut() (Collegamento a libreria) Catalogo >

Crea una gruppo di variabili nell'attività corrente che contiene i riferimenti a tutti gli oggetti nel documento libreria specificato *stringaNomeLibr*. Aggiunge inoltre i membri del gruppo al menu Variables (Variabili). È quindi possibile fare riferimento a ciascun oggetto utilizzando la relativa *stringaNomeCollegamento*.

Impostare *LibPrivFlag*=**0** per escludere oggetti libreria privata (default)

Impostare *LibPrivFlag*=**1** per includere oggetti libreria privata

Per copiare un gruppo di variabili, vedere **CopyVar**, pagina 30.

Per eliminare un gruppo di variabili, vedere **DelVar**, pagina 45.

```
getVarInfo("linalg2")
        [clearmat  "FUNC"  "LibPub "
         gauss1    "PRGM"  "LibPriv "
         gauss2    "FUNC"  "LibPub "]
libShortcut("linalg2","la")
                {la.clearmat,la.gauss2}
libShortcut("linalg2","la",1)
        {la.clearmat,la.gauss1,la.gauss2}
```

LinRegBx (Regressione lineare) Catalogo >

LinRegBx X,Y**[,***Freq***[,***Categoria***,***Includi***]]**

Calcola la regressione linearey = a+b ·xsulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

LinRegBx (Regressione lineare) Catalogo >

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a+b ·x
stat.a, stat.b	Coefficienti di regressione
stat.r^2	coefficiente di determinazione
stat.r	Coefficiente di correlazione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

LinRegMx (Regressione lineare) Catalogo >

LinRegMx X,Y**[**,*Freq*[**,***Categoria***,***Includi*]]

Calcola la regressione lineare y = m ·x+b sulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

LinRegMx (Regressione lineare) Catalogo >

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: m ·x+b
stat.m, stat.b	Coefficienti di regressione
stat.r^2	coefficiente di determinazione
stat.r	Coefficiente di correlazione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

LinRegtIntervals (Regressione lineare) Catalogo >

LinRegtIntervals *X*,*Y*[,*F*[,**0**[,*livelloConfidenza*]]]

Per pendenza Calcola un intervallo di confidenza di livello C per la pendenza.

LinRegtIntervals *X*,*Y*[,*F*[,**1**,*ValX*[,*livelloConfidenza*]]]

Per risposta. Calcola un valore y previsto, un intervallo di previsione del livello C per una singola osservazione e un intervallo di confidenza del livello C per la risposta media.

Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere la stessa dimensione.

X e Y sono liste di variabili indipendenti e dipendenti.

F è una lista opzionale di valori di frequenza. Ciascun elemento di F specifica la frequenza di occorrenza di ogni dato corrispondente di X e Y. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a+b ·x
stat.a, stat.b	Coefficienti di regressione
stat.df	Gradi di libertà
stat.r^2	Coefficiente di determinazione
stat.r	Coefficiente di correlazione
stat.Resid	Residui della regressione

Solo tipo per pendenza

Variabile di output	Descrizione
[stat.CLower, stat.CUpper]	Intervallo di confidenza per la pendenza
stat.ME	Margine di errore dell’intervallo di confidenza
stat.SESlope	Errore standard della pendenza
stat.s	Errore standard sulla linea

Solo tipo per risposta

Variabile di output	Descrizione
[stat.CLower, stat.CUpper]	Intervallo di confidenza per la risposta media
stat.ME	Margine di errore dell’intervallo di confidenza
stat.SE	Errore standard della risposta media
[stat.LowerPred,	Intervallo di previsione per una singola osservazione

Variabile di output	Descrizione
stat.UpperPred]	
stat.MEPred	Margine di errore dell'intervallo di previsione
stat.SEPred	Errore standard per la previsione
stat.$\hat{y}$	a + b · ValX

LinRegtTest (t Test regressione lineare) Catalogo >

LinRegtTest X,Y[,$Freq$[,$Ipotesi$]]

Calcola una regressione lineare sulle liste X e Y e un t test sul valore della pendenza β e il coefficiente di correlazione ρ per l'equazione y=α+βx. Viene verificata l'ipotesi nulla H_0:β=0 (in modo equivalente, ρ=0) in relazione a una di tre ipotesi alternative.

Tutte le liste devono avere la stessa dimensione.

X e Y sono liste di variabili indipendenti e dipendenti.

$Freq$ è una lista opzionale di valori di frequenza. Ciascun elemento di $Freq$ specifica la frequenza di occorrenza di ogni dato corrispondente di X e Y. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

$Ipotesi$ è un valore opzionale che specifica una di tre ipotesi alternative rispetto alla quale verrà testata l'ipotesi nulla (H_0:β=ρ=0).

Per H_a: β≠0 e ρ≠0 (default), impostare $Ipotesi$=0

Per H_a: β<0 e ρ<0, impostare $Ipotesi$<0

Per H_a: β>0 e ρ>0, impostare $Ipotesi$>0

Il riepilogo dei risultati è memorizzato nella variabile $stat.results$. (pagina 156).

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a + b ·x
stat.t	*Statistica* T per il test di significatività
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.df	Gradi di libertà
stat.a, stat.b	Coefficienti di regressione
stat.s	Errore standard sulla linea
stat.SESlope	Errore standard della pendenza
stat.r^2	Coefficiente di determinazione
stat.r	Coefficiente di correlazione
stat.Resid	Residui della regressione

linSolve() Catalogo >

linSolve(*SistemaDiEqLineari*, *Var1*, *Var2*, ...**)⇒***lista*

linSolve(*EqLineare1* **and** *EqLineare2* **and** ..., *Var1*, *Var2*, ...**)⇒***lista*

linSolve({*EqLineare1*, *EqLineare2*, ...**},** *Var1*, *Var2*, ...**) ⇒***lista*

linSolve(*SistemaDiEqLineari*, **{***Var1*, *Var2*, ...**}) ⇒***lista*

linSolve(*EqLineare1* **and** *EqLineare2* **and** ..., **{***Var1*, *Var2*, ...**})⇒***lista*

linSolve({*EqLineare1*, *EqLineare2*, ...**},** **{***Var1*, *Var2*, ...**}) ⇒***lista*

Restituisce una lista di soluzioni per le variabili *Var1*, *Var2*, ...

Il primo argomento deve calcolare un sistema di equazioni lineari o una singola equazione lineare. Diversamente, si ottiene un argomento errato.

$$\text{linSolve}\left(\begin{cases}2\cdot x+4\cdot y=3\\5\cdot x-3\cdot y=7\end{cases},\{x,y\}\right)\qquad\left\{\frac{37}{26},\frac{1}{26}\right\}$$

$$\text{linSolve}\left(\begin{cases}2\cdot x=3\\5\cdot x-3\cdot y=7\end{cases},\{x,y\}\right)\qquad\left\{\frac{3}{2},\frac{1}{6}\right\}$$

$$\text{linSolve}\left(\begin{cases}apple+4\cdot pear=23\\5\cdot apple-pear=17\end{cases},\{apple,pear\}\right)\qquad\left\{\frac{13}{3},\frac{14}{3}\right\}$$

$$\text{linSolve}\left(\begin{cases}apple\cdot 4+\frac{pear}{3}=14\\-apple+pear=6\end{cases},\{apple,pear\}\right)\qquad\left\{\frac{36}{13},\frac{114}{13}\right\}$$

linSolve() Catalogo >

Ad esempio, calcolando linSolve(x=1 e x=2,x) si ottiene un risultato “Argomento errato”.

Δlist() (Differenza in una lista) Catalogo >

Δlist(*Lista1***)⇒***lista*

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`deltaList(…)`**.

Restituisce una lista contenente le differenze tra elementi consecutivi in *Lista1*. Ogni elemento di *Lista1* viene sottratto dal successivo elemento di *Lista1*. La lista risultante è sempre composta da un elemento in meno della *Lista1* originale.

ΔList({20,30,45,70})	{10,15,25}

list▸mat() (Da lista a matrice) Catalogo >

list▸mat(*Lista* [**,** *elementiPerRiga*]**)⇒***matrice*

Restituisce una matrice completata riga per riga con gli elementi di *Lista*.

elementiPerRiga, se incluso, specifica il numero di elementi per riga. L’impostazione predefinita corrisponde al numero di elementi di *Lista* (una riga).

Se *Lista* non completa la matrice risultante, viene aggiunta una serie di zeri.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`list@>mat(…)`**.

list▸mat({1,2,3})	[1 2 3]
list▸mat({1,2,3,4,5},2)	[1 2; 3 4; 5 0]

ln() (Logaritmo naturale) Tasti ctrl e^x

ln(*Valore1***)⇒***valore*

ln(*Lista1***)⇒***lista*

Restituisce il logaritmo naturale dell’argomento.

ln(2.)	0.693147

Se la modalità del formato complesso è Reale:

ln() (Logaritmo naturale) Tasti ctrl e^x

In una lista, restituisce i logaritmi naturali degli elementi.

```
ln({-3,1.2,5})
                "Error: Non-real calculation"
```

Se la modalità del formato complesso è Rettangolare:

```
ln({-3,1.2,5})
   {1.09861+3.14159·i,0.182322,1.60944}
```

ln(*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce il logaritmo naturale della matrice *matriceQuadrata1*. Ciò non equivale a calcolare il logaritmo naturale di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$$\ln\left(\begin{bmatrix} 1 & 5 & 3 \\ 4 & 2 & 1 \\ 6 & -2 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} 1.83145+1.73485\cdot i & 0.009193-1.49086 \\ 0.448761-0.725533\cdot i & 1.06491+0.623491 \\ -0.266891-2.08316\cdot i & 1.12436+1.79018\cdot \end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

LnReg (Regressione logaritmica) Catalogo >

LnReg *X*, *Y*[, [*Freq*] [, *Categoria*, *Includi*]]

Calcola la regressione logaritmica y = a+b ·ln (x) sulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

LnReg (Regressione logaritmica) Catalogo >

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a+b ·ln(x)
stat.a, stat.b	Coefficienti di regressione
stat.r^2	Coefficiente di determinazione lineare di dati trasformati
stat.r	Coefficiente di correlazione per dati trasformati (ln(x), y)
stat.Resid	Residui associati al modello logaritmico
stat.ResidTrans	Residui associati all’adattamento lineare dei dati trasformati
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

Local (Variabile locale) Catalogo >

Local *Var1*[, *Var2*] [, *Var3*] ...

Definisce le *var* specificate come variabili locali. Tali variabili esistono solamente durante il calcolo di una funzione e vengono cancellate al termine dell’esecuzione di tale funzione.

Nota: le variabili locali permettono di risparmiare memoria in quanto esistono solo temporaneamente. Inoltre, esse non influiscono sui valori delle variabili globali esistenti. Le variabili locali devono essere utilizzate per i cicli **For** e per salvare in maniera provvisoria i valori in una funzione su diverse righe, poiché le modifiche sulle variabili globali non sono ammesse in una funzione.

```
Define rollcount()=Func
                   Local i
                   1→i
                   Loop
                   If randInt(1,6)=randInt(1,6)
                   Goto end
                   i+1→i
                   EndLoop
                   Lbl end
                   Return i
                   EndFunc
                                          Done
rollcount()                                 16
rollcount()                                  3
```

Local (Variabile locale) Catalogo >

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Lock Catalogo >

Lock *Var1* [**,** *Var2*] [**,** *Var3*] ...

Lock *Var*.

Blocca le variabili o il gruppo di variabili specificate. Le variabili bloccate non possono essere modificate o eliminate.

Non è possibile bloccare o sbloccare la variabile di sistema *Ans*, inoltre non è possibile bloccare i gruppi di variabili di sistema *stat*. o *tvm*.

Nota: Il comando **Blocca** (**Lock**) cancella la cronologia di Annulla/Ripeti quando è applicato a variabili sbloccate.

Vedere **unLock**, pagina 176 e **getLockInfo()**, pagina 68.

a:=65	65
Lock a	*Done*
getLockInfo(a)	1
a:=75	"Error: Variable is locked."
DelVar a	"Error: Variable is locked."
Unlock a	*Done*
a:=75	75
DelVar a	*Done*

log() (Logaritmo) Tasti ctrl 10^x

log(*Valore1* [**,** *Valore2*] **)⇒** *valore*

log(*Lista1* [**,** *Valore2*] **)⇒** *lista*

Restituisce il logaritmo in base-*Valore2* dell'argomento.

Nota: vedere anche **Modello di logaritmo**, pagina 6.

In una lista, restituisce il logaritmo in base-*Valore2* degli elementi.

Se *Espr2* viene omesso, come base viene utilizzato 10.

$\log_{10}(2.)$	0.30103
$\log_{4}(2.)$	0.5
$\log_{3}(10)-\log_{3}(5)$	0.63093

Se la modalità del formato complesso è Reale:

$\log_{10}(\{-3,1.2,5\})$	"Error: Non−real calculation"

log() (Logaritmo)

Tasti ctrl 10^x

log(*matriceQuadrata1* [,*Valore2*]**)⇒***matriceQuadrata*

Restituisce il logaritmo in base-*Valore2* della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare il logaritmo in base-*Valore2* di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

Se l'argomento base viene omesso, come base viene utilizzato 10.

Se la modalità del formato complesso è Rettangolare:

$$\log_{10}(\{-3,1.2,5\})$$
$$\{0.477121+1.36438\cdot i, 0.079181, 0.69897\}$$

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$$\log_{10}\left(\begin{bmatrix} 1 & 5 & 3 \\ 4 & 2 & 1 \\ 6 & -2 & 1 \end{bmatrix}\right)$$
$$\begin{bmatrix} 0.795387+0.753438\cdot i & 0.003993-0.6474 \\ 0.194895-0.315095\cdot i & 0.462485+0.2707 \\ -0.115909-0.904706\cdot i & 0.488304+0.7774 \end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

Logistic (Regressione logistica)

Catalogo >

Logistic *X*, *Y*[, [*Freq*] [, *Categoria*, *Includi*]]

Calcola la regressione logisticay = (c/(1+a ·e^{-bx}))sulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Logistic (Regressione logistica) Catalogo >

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: $c/(1+a \cdot e^{-bx})$
stat.a, stat.b, stat.c	Coefficienti di regressione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

LogisticD (Rgressione logistica) Catalogo >

LogisticD *X*, *Y* [, [*Iterazioni*], [*Freq*] [, *Categoria*, *Includi*]]

Calcola la regressione logistica $y = (c/(1+a \cdot e^{-bx})+d)$ sulle liste *X* e *Y* con frequenza *Freq*, utilizzando un numero specificato di *Iterazioni*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Iterazioni è un valore opzionale che specifica quante volte al massimo verrà tentata una soluzione. Se omesso, viene utilizzato 64. Di solito valori più alti danno una maggiore accuratezza ma richiedono tempi di esecuzione più lunghi, e viceversa.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	**Descrizione**
stat.RegEqn	Equazione di regressione: $c/(1+a \cdot e^{-bx})+d)$
stat.a, stat.b, stat.c, stat.d	Coefficienti di regressione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

Loop

Catalogo >

Loop
Blocco
EndLoop

Esegue ciclicamente le istruzioni di *Blocco*. Si noti che un ciclo viene eseguito infinite volte, se non si trovano istruzioni **Goto** o **Exit** all'interno di *Blocco*.

Blocco è una sequenza di istruzioni separate dal carattere.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define rollcount()=Func
                   Local i
                   1→i
                   Loop
                   If randInt(1,6)=randInt(1,6)
                   Goto end
                   i+1→i
                   EndLoop
                   Lbl end
                   Return i
                   EndFunc
                                          Done
rollcount()                                 16
rollcount()                                  3
```

LU (Scomposizione inferiore - superiore)

Catalogo >

LU *Matrice*, *MatriceL*, *MatriceU*, *MatriceP[,Tol]*

Calcola la scomposizione LU (lower-upper, inferiore-superiore) di una matrice reale o complessa. La matrice triangolare inferiore è memorizzata in *MatriceL*, quella superiore in *MatriceU* e la matrice di permutazione (che descrive gli scambi di riga eseguiti durante i calcoli) in *MatriceP*.

MatriceL · *MatriceU* = *MatriceP* · *matrice*

In alternativa, un elemento qualsiasi della matrice viene considerato zero se il suo valore assoluto è minore di *Toll*. Tale tolleranza viene utilizzata solo se la matrice contiene elementi a virgola mobile e non contiene variabili simboliche alle quali non sia stato assegnato un valore. In caso contrario, *Toll* viene ignorato.

- Se si usa ctrl enter oppure se si imposta la modalità **Auto o Approssimato** su Approssimato, i calcoli verranno eseguiti in virgola mobile.
- Se *Toll* viene omesso o non è utilizzato,

$$\begin{bmatrix} 6 & 12 & 18 \\ 5 & 14 & 31 \\ 3 & 8 & 18 \end{bmatrix} \rightarrow m1 \qquad \begin{bmatrix} 6 & 12 & 18 \\ 5 & 14 & 31 \\ 3 & 8 & 18 \end{bmatrix}$$

LU *m1*,*lower*,*upper*,*perm* — *Done*

lower $\begin{bmatrix} 1 & 0 & 0 \\ \frac{5}{6} & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix}$

upper $\begin{bmatrix} 6 & 12 & 18 \\ 0 & 4 & 16 \\ 0 & 0 & 1 \end{bmatrix}$

perm $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

LU (Scomposizione inferiore - superiore) Catalogo >

la tolleranza predefinita viene calcolata come:
5E-14 ·max(dim(*Matrice*)) ·rowNorm (*Matrice*)

L'algoritmo di scomposizione in fattori **LU** usa il pivoting parziale per lo scambio di righe.

M

mat▸list() (Da matrice a lista) Catalogo >

mat▸list(*Matrice***)⇒***lista*

Restituisce una lista completata con gli elementi di *Matrice*. Gli elementi sono copiati da *Matrice* riga per riga.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`mat@>list(…)`**.

$\text{mat}\blacktriangleright\text{list}\left(\begin{bmatrix}1 & 2 & 3\end{bmatrix}\right)$ $\{1,2,3\}$

$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}\rightarrow m1$ $\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}$

$\text{mat}\blacktriangleright\text{list}(m1)$ $\{1,2,3,4,5,6\}$

max() (Massimo) Catalogo >

max(*Valore1***, *Valore2***)⇒***espressione*

max(*Lista1***, *Lista2***)⇒***lista*

max(*Matrice1***, *Matrice2***)⇒***matrice*

Restituisce il massimo di due argomenti; se questi sono due liste o matrici, restituisce una lista o matrice contenente il valore massimo di ciascuna coppia di elementi corrispondenti.

$\max(2.3,1.4)$ 2.3

$\max(\{1,2\},\{-4,3\})$ $\{1,3\}$

max(*Lista***)⇒***espressione*

Restituisce l'elemento massimo di *lista*.

$\max(\{0,1,-7,1.3,0.5\})$ 1.3

max(*Matrice1***)⇒***matrice*

Restituisce un vettore riga contenente l'elemento massimo di ciascuna colonna di *Matrice1*.

$\max\left(\begin{bmatrix}1 & -3 & 7\\-4 & 0 & 0.3\end{bmatrix}\right)$ $\begin{bmatrix}1 & 0 & 7\end{bmatrix}$

max() (Massimo) Catalogo >

Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

Nota: vedere anche **min().**

mean() Media Catalogo >

mean(*Lista***[, *listaFreq*])⇒*espressione***

Restituisce la media degli elementi di *Lista*.

Ogni elemento di *listaFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Lista*.

$\text{mean}(\{0.2,0,1,-0.3,0.4\})$ → 0.26

$\text{mean}(\{1,2,3\},\{3,2,1\})$ → $\frac{5}{3}$

mean(*Matrice1***[, *matriceFreq*]) ⇒*matrice***

Restituisce un vettore riga contenente le medie di tutte le colonne di *Matrice1*.

Ogni elemento di *matriceFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Matrice1*.

Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

In modalità formato vettore rettangolare:

$\text{mean}\left(\begin{bmatrix} 0.2 & 0 \\ -1 & 3 \\ 0.4 & -0.5 \end{bmatrix}\right)$ → $\begin{bmatrix} -0.133333 & 0.833333 \end{bmatrix}$

$\text{mean}\left(\begin{bmatrix} \frac{1}{5} & 0 \\ -1 & 3 \\ \frac{2}{5} & \frac{-1}{2} \end{bmatrix}\right)$ → $\begin{bmatrix} \frac{-2}{15} & \frac{5}{6} \end{bmatrix}$

$\text{mean}\left(\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix},\begin{bmatrix} 5 & 3 \\ 4 & 1 \\ 6 & 2 \end{bmatrix}\right)$ → $\begin{bmatrix} \frac{47}{15} & \frac{11}{3} \end{bmatrix}$

median() (Mediana) Catalogo >

median(*Lista***[, *listaFreq*])⇒*espressione***

Restituisce la mediana degli elementi di *Lista*.

Ogni elemento di *listaFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Lista*.

$\text{median}(\{0.2,0,1,-0.3,0.4\})$ → 0.2

median(*Matrice1***[, *matriceFreq*])⇒*matrice***

Restituisce un vettore riga contenente le mediane degli elementi delle colonne di *Matrice1*.

$\text{median}\left(\begin{bmatrix} 0.2 & 0 \\ 1 & -0.3 \\ 0.4 & -0.5 \end{bmatrix}\right)$ → $\begin{bmatrix} 0.4 & -0.3 \end{bmatrix}$

median() (Mediana) Catalogo >

Ogni elemento di *matriceFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Matrice1*.

Note:

- Tutti gli elementi nella lista o nella matrice devono essere semplificati in numeri.
- Gli elementi vuoti (nulli) di una lista o matrice vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

MedMed (Linea mediana-mediana) Catalogo >

MedMed *X*,*Y* [, *Freq*] [, *Categoria*, *Includi*]]

Calcola la linea mediana-medianay = (m ·x+b)sulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione linea mediana-mediana: m ·x+b
stat.m, stat.b	Coefficienti del modello
stat.Resid	Residui della linea mediana-mediana
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

mid() (In mezzo alla stringa) Catalogo >

mid(*stringaOrigine***,** *Inizio*[**,** *Cont*]**)⇒***stringa*

Restituisce *Cont* caratteri dalla stringa di caratteri *stringaOrigine*, incominciando dal numero di carattere di *Inizio*.

Se *Cont* viene omesso o se è maggiore della dimensione di *stringaOrigine*, restituisce tutti i caratteri di *stringaOrigine*, incominciando dal numero di carattere di *Inizio*.

Cont deve essere ≥ 0. Se *Cont* = 0, viene restituita una stringa vuota.

```
mid("Hello there",2)          "ello there"
mid("Hello there",7,3)        "the"
mid("Hello there",1,5)        "Hello"
mid("Hello there",1,0)        ""
```

mid(*listaOrigine***,** *Inizio* [**,** *Cont*]**)⇒***lista*

Restituisce *Cont* elementi da *listaOrigine*, iniziando dal numero di elemento di *Inizio*.

Se *Cont* viene omesso o è maggiore della dimensione di *listaOrigine*, restituisce tutti gli elementi di *listaOrigine*, incominciando dal numero di elemento di *Inizio*.

Cont deve essere ≥ 0. Se Cont = 0, restituisce una lista vuota.

```
mid({9,8,7,6},3)              {7,6}
mid({9,8,7,6},2,2)            {8,7}
mid({9,8,7,6},1,2)            {9,8}
mid({9,8,7,6},1,0)            {}
```

mid(*listaStringaOrigine***,** *Inizio*[**,** *Cont*]**)⇒***lista*

```
mid({"A","B","C","D"},2,2)
                        {"B","C"}
```

mid() (In mezzo alla stringa) Catalogo >

Restituisce *Cont* stringhe della lista delle stringhe *listaStringheOrigine*, incominciano dal numero di elemento di *Inizio*.

min() (Minimo) Catalogo >

min(*Valore1***, *Valore2***)⇒***espressione*

min(*Lista1, Lista2***)⇒***lista*

min(*Matrice1, Matrice2***)⇒***matrice*

Restituisce il minimo di due argomenti. Se gli argomenti sono due liste o matrici, viene restituita una lista o una matrice contenente il valore minimo di ciascuna coppia di elementi corrispondenti.

$\min(2.3,1.4)$	1.4
$\min(\{1,2\},\{-4,3\})$	$\{-4,2\}$

min(*Lista***)⇒***espressione*

Restituisce l'elemento minimo di *Lista*.

$\min(\{0,1,-7,1.3,0.5\})$	-7

min(*Matrice1***)⇒***matrice*

Restituisce un vettore di riga contenente l'elemento minimo di ciascuna colonna di *Matrice1*.

$\min\left(\begin{bmatrix} 1 & -3 & 7 \\ -4 & 0 & 0.3 \end{bmatrix}\right)$	$\begin{bmatrix} -4 & -3 & 0.3 \end{bmatrix}$

Nota: vedere anche **max()**.

mirr() Catalogo >

mirr
(
tassoFinanziamento
,*tassoReinvestimento***,***CF0***,***CFLista*
[**,***CFFreq*]**)**

Funzione finanziaria che restituisce l'indice interno di rendimento modificato di un investimento.

tassoFinanziamento è il tasso di interesse che si paga sugli importi di cash flow.

tassoReinvestimento è il tasso di interesse in corrispondenza del quale i cash flow vengono reinvestiti.

$list1:=\{6000,-8000,2000,-3000\}$	$\{6000,-8000,2000,-3000\}$
$list2:=\{2,2,2,1\}$	$\{2,2,2,1\}$
$\text{mirr}(4.65,12,5000,list1,list2)$	13.41608607

mirr() Catalogo >

CF0 è il cash flow iniziale al tempo 0; deve essere un numero reale.

CFLista è una lista di importi di cash flow dopo il cash flow iniziale CF0.

CFFreq è una lista opzionale in cui ciascun elemento specifica la frequenza di occorrenza di un importo di cash flow raggruppato (consecutivo), che è l'elemento corrispondente di *CFLista*. L'impostazione predefinita è 1; è possibile inserire valori che siano numeri interi positivi < 10000.

Nota: vedere anche **irr()**, pagina 79.

mod() (Modulo) Catalogo >

mod(*Valore1***,** *Valore2***)⇒***espressione*

mod(*Lista1***,** *Lista2***)⇒***lista*

mod(*Matrice1***,** *Matrice2***)⇒***matrice*

Restituisce il primo argomento modulo secondo argomento, come definito dalle identità:

mod(x,0) = x

mod(x,y) = x − y floor(x/y)

Quando il secondo argomento è diverso da zero, il risultato è periodico in tale argomento. Il risultato può essere zero oppure ha lo stesso segno del secondo argomento.

Se gli argomenti sono due liste o due matrici, viene restituita una lista o una matrice contenente il modulo di ciascuna coppia di elementi corrispondenti.

Nota: vedere anche **remain()**, pagina 133

mod(7,0)	7
mod(7,3)	1
mod(⁻7,3)	2
mod(7,⁻3)	⁻2
mod(⁻7,⁻3)	⁻1
mod({12,⁻14,16},{9,7,⁻5})	{3,0,⁻4}

mRow() (Operazione con righe di matrice)

Catalogo >

mRow(*Valore***, *Matrice1*, *Indice***)⇒***matrice***

Restituisce una copia di *Matrice1* dove ciascun elemento della riga *Indice* di *Matrice1* viene moltiplicato per *Valore*.

$$\text{mRow}\left(\frac{-1}{3},\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix},2\right) \qquad \begin{bmatrix}1 & 2\\ -1 & \frac{-4}{3}\end{bmatrix}$$

mRowAdd() (Moltiplicazione e somma di righe di matrice)

Catalogo >

mRowAdd(*Valore***, *Matrice1*, *Indice1*, *Indice2*) ⇒***matrice***

Restituisce una copia di *Matrice1* dove ciascun elemento della riga *Indice2* di *Matrice1* viene sostituito con:

Valore × *riga Indice1* + *riga Indice2*

$$\text{mRowAdd}\left(-3,\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix},1,2\right) \qquad \begin{bmatrix}1 & 2\\ 0 & -2\end{bmatrix}$$

MultReg (Regressione lineare multipla)

Catalogo >

MultReg *Y*, *X1*[,*X2*[,*X3*,...[,*X10*]]]

Calcola la regressione lineare multipla della lista *Y* sulle liste *X1*, *X2*, ..., *X10*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere la stessa dimensione.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: b0+b1 ·x1+b2 ·x2+ ...
stat.b0, stat.b1, ...	Coefficienti di regressione
stat.R^2	Coefficiente di determinazione multipla
stat.ŷList	ŷList = b0+b1 ·x1+ ...
stat.Resid	Residui della regressione

MultRegIntervals (Intervalli di confidenza della previsione di regressione multipla)

Catalogo >

MultRegIntervals *Y*, *X1*[,*X2*[,*X3*,...[,*X10*]]],*listaValX*[,*livelloConfidenza*]

Calcola un valore y previsto, un intervallo di previsione del livello C per una singola osservazione e un intervallo di confidenza del livello C per la risposta media.

Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere la stessa dimensione.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: b0+b1 ·x1+b2 ·x2+ ...
stat.$\hat{y}$	Una stima del punto: $\hat{y}$ = b0 + b1 · xl + ... per *listaValX*
stat.dfError	Gradi di libertà degli errori
stat.CLower, stat.CUpper	Intervallo di confidenza per una risposta media
stat.ME	Margine di errore dell’intervallo di confidenza
stat.SE	Errore standard della risposta media
stat.LowerPred, stat.UpperrPred	Intervallo di previsione per una singola osservazione
stat.MEPred	Margine di errore dell’intervallo di previsione
stat.SEPred	Errore standard per la previsione
stat.bList	Lista dei coefficienti di regressione, {b0,b1,b2,...}
stat.Resid	Residui della regressione

MultRegTests (Verifica sulla regressione lineare multipla)

Catalogo >

MultRegTests *Y*, *X1*[,*X2*[,*X3*,...[,*X10*]]]

MultRegTests (Verifica sulla regressione lineare multipla)

Catalogo >

Il test di regressione lineare multipla calcola una regressione lineare multipla sui dati assegnati e fornisce la statistica F test globale e le statistiche t test per i coefficienti.

Il riepilogo dei risultati è memorizzato nella variabile $stat.results$. (pagina 156).

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

`Output`

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: b0+b1 ·x1+b2 ·x2+ …
stat.F	Statistica della verifica F globale
stat.PVal	Valore P associato alla statistica F globale
stat.R^2	Coefficiente di determinazione multipla
stat.AdjR^2	Coefficiente modificato di determinazione multipla
stat.s	Deviazione standard dell’errore
stat.DW	Statistica d Durbin-Watson; utilizzata per determinare se la correlazione automatica di primo ordine è presente nel modello
stat.dfReg	Gradi di liberà della regressione
stat.SSReg	Somma dei quadrati della regressione
stat.MSReg	Quadrato medio della regressione
stat.dfError	Gradi di libertà degli errori
stat.SSError	Somma dei quadrati degli errori
stat.MSError	Quadrato medio degli errori
stat.bList	{b0,b1,…} Lista dei coefficienti
stat.tList	Lista di statistiche t, una per ogni coefficiente di bList
stat.PList	Lista di valori P per ogni statistica t
stat.SEList	Lista di errori standard per coefficienti di bList

Variabile di output	Descrizione
stat.ŷList	listaŷ = b0+b1 ·x1+ . . .
stat.Resid	Residui della regressione
stat.sResid	Residui standardizzati; valore ottenuto dividendo un residuo per la sua deviazione standard
stat.CookDist	Distanza di Cook; misura dell'influenza di un'osservazione basata sui valori residui e di leverage
stat.Leverage	Misura della distanza dei valori della variabile indipendente dai rispettivi valori medi

N

nand

ctrl = tasti

BooleanExpr1 **nand** *BooleanExpr2*
restuisce *Boolean expression*

BooleanList1 **nand** *BooleanList2* restituisce *Boolean list*

BooleanMatrix1 **nand** *BooleanMatrix2*
restituisce *Boolean matrix*

Restituisce la negazione di un'operazione **and** logica riguardo i due argomenti. Restituisce true, false o una forma semplificata dell'equazione.

Nel caso di liste e matrici, restituisce un confronto elemento per elemento.

Integer1 **nand** *Integer2*⇒*integer*

Confronta due interi reali bit per bit tramite un'operazione **nand**. Internamente, entrambi gli interi vengono convertiti in numeri binari a 64 bit con segno. Quando vengono confrontati i bit corrispondenti, il risultato sarà 1 se entrambi sono uguali a 1; in caso contrario, il risultato sarà 0. Il valore restituito rappresenta il risultato dei bit e viene visualizzato secondo la modalità base.

3 and 4	0
3 nand 4	-1
{1,2,3} and {3,2,1}	{1,2,1}
{1,2,3} nand {3,2,1}	{-2,-3,-2}

nand

ctrl = tasti

È possibile inserire gli interi in qualsiasi base numerica. Nel caso di un numero binario o esadecimale, è necessario utilizzare rispettivamente il prefisso 0b o 0h. Senza prefisso, gli interi vengono considerati decimali (base 10)

nCr() (Combinazioni)

Catalogo >

nCr(*Valore1***,** *Valore2***)⇒***espressione*

Dati i numeri interi *Valore1* e *Valore2* con *Valore1* ≥ *Valore2* ≥ 0, **nCr()** è il numero di combinazioni degli elementi di *Valore1* presi nel numero di *Valore2* per volta (questa procedura è nota anche come coefficiente binomiale).

$\mathrm{nCr}(z,3)\mid z=5$	10
$\mathrm{nCr}(z,3)\mid z=6$	20

nCr(*Valore1***, 0)⇒1**

nCr(*Valore1***,** *interoNeg***)⇒0**

nCr(*Valore1***,** *interoPos***)⇒** *Valore1* · **(***Valore1*−1**)... (***Valore1*−*interoPos***+1)/** *interoPos***!**

nCr(*Valore1***,** *nonIntero***)⇒***espressione***!/ ((***Valore1*−*nonIntero***)! ·***nonIntero***!)**

nCr(*Lista1***,** *Lista2***)⇒***lista*

Restituisce una lista di combinazioni sulla base delle coppie di elementi corrispondenti nelle due liste. Le liste degli argomenti devono avere dimensioni uguali.

$\mathrm{nCr}(\{5,4,3\},\{2,4,2\})$	$\{10,1,3\}$

nCr(*Matrice1***,** *Matrice2***)⇒***matrice*

Restituisce una matrice di combinazioni, sulla base delle coppie di elementi corrispondenti nelle due matrici. Le matrici degli argomenti devono avere le stesse dimensioni.

$\mathrm{nCr}\left(\begin{bmatrix}6&5\\4&3\end{bmatrix},\begin{bmatrix}2&2\\2&2\end{bmatrix}\right)$	$\begin{bmatrix}15&10\\6&3\end{bmatrix}$

nDerivative() Catalogo >

nDerivative(*Espr1***,***Var=Valore* [**,***Ordine*]**)⇒***valore*

nDerivative(*Espr1***,***Var*[**,***Ordine*]**) |** *Var=Valore*⇒*valore*

Restituisce la derivata numerica calcolata con metodi di differenziazione automatica.

Se il *Valore* è specificato, lo stesso sostituisce qualsiasi assegnazione di variabile precedente o qualsiasi sovrapposizione corrente "|" della variabile.

Se la variabile *Var* non contiene un valore numerico, è necessario specificare *Valore*.

Ordine della derivata deve essere **1** o **2**.

$\text{nDerivative}(\lvert x\rvert, x=1)$	1
$\text{nDerivative}(\lvert x\rvert, x)\lvert x=0$	undef
$\text{nDerivative}(\sqrt{x-1}, x)\lvert x=1$	undef

Nota: L'algoritmo nDerivative() ha una limitazione: funziona in modo ricorsivo tramite l'espressione non semplificata, calcolando il valore numerico della derivata prima (e seconda, se applicabile) e di ciascuna sottoespressione, il che può produrre un risultato imprevisto.

Si consideri l'esempio sulla destra. La derivata prima di x·(x^2+x)^(1/3) con x=0 è uguale a 0. Tuttavia, poiché la derivata prima della sottoespressione (x^2+x)^(1/3) è indefinita con x=0, e questo valore è utilizzato per calcolare la derivata dell'espressione totale, **nDerivative()** riporta il risultato come indefinito e visualizza un messaggio di avvertenza.

Se si incontra questa limitazione, verificare la soluzione graficamente. È anche possibile provare a utilizzare **centralDiff()**.

$\text{nDerivative}\left(x\cdot\left(x^2+x\right)^{\frac{1}{3}}, x, 1\right)\lvert x=0$	undef
$\text{centralDiff}\left(x\cdot\left(x^2+x\right)^{\frac{1}{3}}, x\right)\lvert x=0$	0.000033

newList() (Nuova lista) Catalogo >

newList(*numElementi***)⇒***lista*

Restituisce una lista le cui dimensioni sono *numElementi*. Ciascun elemento è zero.

$\text{newList}(4)$	$\{0,0,0,0\}$

newMat() (Nuova matrice) Catalogo >

newMat(*numRighe***,** *numColonne***)⇒***matrice*

Restituisce una matrice di zeri le cui dimensioni sono date da *numRighe* per *numColonne*.

newMat(2,3)	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

nfMax() (Massimo di una funzione calcolato numericamente) Catalogo >

nfMax(*Espr***,** *Var***)⇒***valore*

nfMax(*Espr***,** *Var, estremoInf***)⇒***valore*

nfMax(*Espr***,** *Var, estremoInf***,** *estremoSup***)⇒***valore*

nfMax(*Espr, Var***) | ***estremoInf*≤*Var* ≤*estremoSup*⇒*valore*

Restituisce un possibile valore calcolato numericamente della variabile *Var* che produce il massimo locale di *Espr*.

Se si fornisce *estremoInf* e *estremoSup*, la funzione controlla nell'ambito dell'intervallo chiuso [*estremoInf,estremoSup*] per il massimo locale.

$\text{nfMax}\left(-x^2-2\cdot x-1,x\right)$	-1.
$\text{nfMax}\left(0.5\cdot x^3-x-2,x,-5,5\right)$	5.

nfMin() (Minimo di una funzione calcolato numericamente) Catalogo >

nfMin(*Espr***,** *Var***)⇒***valore*

nfMin(*Espr***,** *Var, estremoInf***)⇒***valore*

nfMin(*Espr***,** *Var, estremoInf***,** *estremoSup***)⇒***valore*

nfMin(*Espr, Var***) | ***estremoInf*≤*Var* ≤*estremoSup*⇒*valore*

Restituisce un possibile valore calcolato numericamente della variabile *Var* che produce il minimo locale di *Espr*.

$\text{nfMin}\left(x^2+2\cdot x+5,x\right)$	-1.
$\text{nfMin}\left(0.5\cdot x^3-x-2,x,-5,5\right)$	-5.

nfMin() (Minimo di una funzione calcolato numericamente) Catalogo >

Se si fornisce *estremoInf* e *estremoSup*, la funzione controlla nell'ambito dell'intervallo chiuso [*estremoInf*,*estremoSup*] per il minimo locale.

nInt() (Integrale numerico) Catalogo >

nInt(*Espr1, Var, Inferiore, Superiore***)⇒***espressione*

$\text{nInt}\left(e^{-x^2},x,-1,1\right)$ 1.49365

Se la funzione da integrare *Espr1* non contiene nessun'altra variabile oltre a *Var* e se *Inferiore* e *Superiore* sono costanti, ∞ positivo o ∞ negativo, allora **nInt()** restituisce un'approssimazione di ∫**(***Espr1*,*Var*, *Inferiore*, *Superiore***)**. Tale approssimazione è una media pesata di alcuni valori esemplificativi della funzione da integrare nell'intervallo *Inferiore*<*Var*<*Superiore*.

$\text{nInt}\left(\cos(x),x,-\pi,\pi+1.\text{E}-12\right)$ $-1.04144\text{E}-12$

L'obiettivo sono sei cifre significative. L'algoritmo adattivo termina quando sembra che l'obiettivo sia stato raggiunto, oppure quando sembra che ulteriori esempi non potrebbero portare alcun miglioramento significativo.

Viene visualizzato un avvertimento ("Accuratezza dubbia") quando sembra che l'obiettivo non sia stato raggiunto.

$\text{nInt}\left(\text{nInt}\left(\frac{e^{-x\cdot y}}{\sqrt{x^2-y^2}},y,-x,x\right),x,0,1\right)$ 3.30423

È possibile nidificare **nInt()** per un'integrazione numerica multipla. I limiti di integrazione possono dipendere da variabili di integrazione esterne ad essi.

nom() Catalogo >

nom(*tassoEffettivo,CpY***)⇒***valore*

$\text{nom}(5.90398,12)$ 5.75

Funzione finanziaria che converte il tasso di interesse effettivo annuo *tassoEffettivo* in un tasso nominale, essendo *CpY* il numero di periodi di capitalizzazione per anno.

nom() Catalogo >

tassoEffettivo deve essere un numero reale e *CpY* deve essere un numero reale > 0.

Nota: vedere anche **eff()**, pagina 50.

nor ctrl = tasti

BooleanExpr1 **nor** *BooleanExpr2* restituisce *Boolean expression*

BooleanList1 **nor** *BooleanList2* restituisce *Boolean list*

BooleanMatrix1 **nor** *BooleanMatrix2* restituisce *Boolean matrix*

Restituisce la negazione di un'operazione **or** logica riguardo i due argomenti. Restituisce true, false o una forma semplificata dell'equazione.

Nel caso di liste e matrici, restituisce un confronto elemento per elemento.

Integer1 **nor** *Integer2*⇒*integer*

Confronta due interi reali bit per bit tramite una operazione **nor**. Internamente, entrambi gli interi vengono convertiti in numeri binari a 64 bit con segno. Quando vengono confrontati i bit corrispondenti, il risultato sarà 1 se entrambi sono uguali a 1; in caso contrario, il risultato sarà 0. Il valore restituito rappresenta il risultato dei bit e viene visualizzato secondo la modalità base.

È possibile inserire gli integer in qualsiasi base numerica. Se si tratta di un numero binario o esadecimale, utilizzare rispettivamente il prefisso 0b o 0h. Senza prefisso, gli integer vengono considerati decimali (base 10)

3 or 4	7
3 nor 4	-8
{1,2,3} or {3,2,1}	{3,2,3}
{1,2,3} nor {3,2,1}	{-4,-3,-4}

norm() (Norma di Froebius) Catalogo >

norm(*Matrice***)⇒***espressione*

norm(*Vettore***)⇒***espressione*

Restituisce la norma di Frobenius.

$\text{norm}\left(\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}\right)$	5.47723
$\text{norm}\left(\begin{bmatrix}1 & 2\end{bmatrix}\right)$	2.23607
$\text{norm}\left(\begin{bmatrix}1\\ 2\end{bmatrix}\right)$	2.23607

normCdf() (Probabilità di distribuzione normale) Catalogo >

normCdf(*valoreInferiore***,***valoreSuperiore* [,μ[,σ]])⇒*numero* se *valoreInferiore* e *valoreSuperiore* sono numeri, *lista* se *valoreInferiore* e *valoreSuperiore* sono liste

Calcola la probabilità di distribuzione normale tra *valoreInferiore* e *valoreSuperiore* per μ (default=0) e σ (default=1) specificati.

Per P(X ≤ *valoreSuperiore*), impostare *valoreInferiore* = −9E999.

normPdf() (Densità di probabilità) Catalogo >

normPdf(*ValX***[,μ[,σ]])⇒*numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola la funzione della densità di probabilità per la distribuzione normale in un valore *valX* specificato per μ σ specificati.

not Catalogo >

not *EsprBooleana1*⇒*espressione booleana*

Restituisce vero o falso o una forma semplificata dell'argomento.

not *Intero1*⇒*intero*

not (2≥3)	true
not 0hB0▸Base16	0hFFFFFFFFFFFFFF4F
not not 2	2

In modalità base Esadecimale:

Importante: è zero, non la lettera O.

not

Catalogo >

Restituisce il complemento a uno di un intero reale. Internamente, *Intero1* viene convertito in numero binario a 64 bit con segno. Il valore di ciascun bit viene scambiato (0 diventa 1 e viceversa) per il complemento a uno. I risultati vengono visualizzati nella modalità Base che è stata impostata.

È possibile inserire l'intero in qualsiasi base numerica. Se si tratta di un numero binario o esadecimale, utilizzare rispettivamente il prefisso 0b o 0h. Senza prefisso, l'intero viene considerato decimale (base 10).

Se viene indicato un intero decimale troppo grande per una forma binaria con segno a 64 bit, verrà utilizzata un'operazione a modulo simmetrico per portare il valore nell'intervallo appropriato. Per ulteriori informazioni, vedere ▶**Base2**, pagina 21.

not 0h7AC36	0hFFFFFFFFFFF853C9

In modalità base Bin:

0b100101▶Base10	37
not 0b100101	
0b1111111111111111111111111111111▶	
not 0b100101▶Base10	-38

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

Nota: un numero binario può contenere fino a 64 cifre (oltre al prefisso 0b). Un numero esadecimale può contenere fino ad 16 cifre.

nPr() (Disposizioni semplici)

Catalogo >

nPr(*Valore1***, *Valore2*)⇒*espressione***

nPr(*Valore1***, 0)⇒1**

nPr(*Valore1***, *interoNeg*) ⇒ 1/ ((*Valore1*+1) ·(*Valore1*+2)... (*Valore1*−*interoNeg*))**

nPr(*Valore1***, *interoPos*) ⇒ *Valore1* · (*Valore1*−1)... (*Valore1*−*interoPos*+1)**

nPr(*Valore1***, *nonIntero*) ⇒*Valore1*! / (*Valore1*−*nonIntero*)!**

nPr(z,3)|z=5	60
nPr(z,3)|z=6	120
nPr({5,4,3},{2,4,2})	{20,24,6}
$\text{nPr}\left(\begin{bmatrix}6&5\\4&3\end{bmatrix},\begin{bmatrix}2&2\\2&2\end{bmatrix}\right)$	$\begin{bmatrix}30&20\\12&6\end{bmatrix}$

nPr(*Lista1***, *Lista2*)⇒*lista***

Restituisce una lista delle disposizioni semplici sulla base delle coppie di elementi corrispondenti nelle due liste. Le liste degli argomenti devono avere dimensioni uguali.

nPr({5,4,3},{2,4,2})	{20,24,6}

nPr(*Matrice1***, *Matrice2*)⇒*matrice***

$\text{nPr}\left(\begin{bmatrix}6&5\\4&3\end{bmatrix},\begin{bmatrix}2&2\\2&2\end{bmatrix}\right)$	$\begin{bmatrix}30&20\\12&6\end{bmatrix}$

nPr() (Disposizioni semplici) Catalogo >

Restituisce una matrice di disposizioni semplici sulla base delle coppie di elementi corrispondenti nelle due matrici. Le matrici degli argomenti devono avere le stesse dimensioni.

npv() Catalogo >

npv(*tassoInteresse***,***CFO***,***CFLista* [,*CFFreq*]**)**

Funzione finanziaria che calcola il valore presente netto (Net Present Value), la somma dei valori presenti per i cash flow in entrata (somme ricevute) e in uscita (somme pagate). Un risultato positivo per npv indica un investimento proficuo.

tassoInteresse è il tasso a cui scontare i cash flow (il costo del denaro) di un dato periodo.

CF0 è il cash flow iniziale al tempo 0; deve essere un numero reale.

CFLista è una lista di importi di cash flow dopo il cash flow iniziale *CF0*.

CFFreq è una lista in cui ciascun elemento specifica la frequenza di occorrenza di un importo di cash flow raggruppato (consecutivo), che è l'elemento corrispondente di *CFLista*. L'impostazione predefinita è 1; è possibile inserire valori che siano numeri interi positivi < 10000.

$list1:=\{6000,^-8000,2000,^-3000\}$	$\{6000,^-8000,2000,^-3000\}$
$list2:=\{2,2,2,1\}$	$\{2,2,2,1\}$
$\text{npv}(10,5000,list1,list2)$	4769.91

nSolve() (Soluzione numerica) Catalogo >

nSolve(*Equazione***,***Var* [=*Campione*]**)**⇒*numero o stringa_errore*

nSolve(*Equazione***,***Var* [=*Campione*]**,***estremoInf***)** ⇒*numero o stringa_errore*

nSolve(*Equazione***,***Var* [=*Campione*]**,***estremoInf***,***estremoSup***)** ⇒*numero o stringa_errore*

$\text{nSolve}(x^2+5\cdot x-25=9,x)$	3.84429
$\text{nSolve}(x^2=4,x=^-1)$	$^-2.$
$\text{nSolve}(x^2=4,x=1)$	2.

Nota: se vi sono soluzioni multiple, è possibile usare un valore campione per trovare una soluzione particolare.

nSolve() (Soluzione numerica) Catalogo >

nSolve(*Equazione*,*Var*[=*Campione*]) | *estremoInf*≤*Var*≤*estremoSup* ⇒*numero o stringa_errore*

Ricerca iterativamente una soluzione numerica reale approssimata dell'*Equazione* per la sua variabile. Specificare la variabile come:

variabile

– o –

variabile = *numero reale*

Ad esempio, x è valido come pure x=3.

nSolve() cerca di determinare un punto in cui il resto sia zero oppure due punti relativamente vicini, nei quali il resto abbia segni opposti e la grandezza del resto non sia eccessiva. Se non è possibile ottenere ciò utilizzando un numero limitato di punti campione, viene restituita la stringa "Nessuna soluzione trovata."

$\text{nSolve}\left(x^2+5\cdot x-25=9,x\right)\|x<0$	-8.84429
$\text{nSolve}\left(\frac{(1+r)^{24}-1}{r}=26,r\right)\|r>0 \text{ and } r<0.25$	0.006886
$\text{nSolve}\left(x^2=-1,x\right)$	"No solution found"

O

OneVar (Statistiche a una variabile) Catalogo >

OneVar [**1,**]*X*[,[*Freq*][,*Categoria*,*Includi*]]

OneVar [*n*,]*X1*,*X2*[*X3*[,...[,*X20*]]]

Calcola le statistiche ad una variabile fino a 20 liste. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

Gli argomenti di X sono liste di dati.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

OneVar (Statistiche a una variabile) Catalogo >

Categoria è una lista di codici numerici di categoria dei dati corrispondenti di *X*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Un elemento vuoto (nullo) in qualsiasi lista *X*, *Freq* o *Categoria* produce un corrispondente elemento vuoto in tutte queste liste. Un elemento vuoto (nullo) in qualsiasi lista da *X1* a *X20* produce un corrispondente elemento vuoto in tutte queste liste. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

Variabile di output	Descrizione
stat.$\bar{x}$	Media dei valori X
Statistiche.Σx	Somma dei valori X
stat.Σx^2	Somma dei valori X^2
stat.sx	Deviazione standard del campione di X
stat. x	Deviazione standard della popolazione di X
stat.n	Numero dei punti di dati
stat.MinX	Minimo dei valori x
stat.Q_1X	1° quartile di x
stat.MedianX	Mediana di x
stat.Q_3X	3° quartile di x
stat.MaxX	Massimo dei valori x
stat.SSX	Somma dei quadrati delle deviazioni dalla media di x

or Catalogo >

BooleanExpr1 **or** *BooleanExpr2* restituisce *Boolean expression*

BooleanList1 **or** *BooleanList2* restituisce *Boolean list*

BooleanMatrix1 **or** *BooleanMatrix2*

restituisce *Boolean matrix*

Restituisce vero o falso o una forma semplificata dell'espressione immessa originariamente.

Restituisce vero se la semplificazione di una o di entrambe le espressioni risulta vera. Restituisce falso solo se il calcolo di entrambe le espressioni risulta falso.

Nota: vedere **xor**.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Define $g(x)$=Func If $x\le 0$ or $x\ge 5$ Goto *end* Return $x\cdot 3$ Lbl *end* EndFunc		*Done*
$g(3)$		9
$g(0)$	*A function did* not return *a value*	

Intero1 **or** *Intero2*⇒*intero*

Confronta due interi reali bit per bit tramite un'operazione or. Internamente, entrambi gli interi vengono convertiti in numeri binari a 64 bit con segno. Quando vengono confrontati bit corrispondenti, il risultato sarà 1 se uno dei due bit è uguale a 1; se entrambi i bit sono uguali a 0, il risultato sarà 0. Il valore restituito rappresenta il risultato dei bit e viene visualizzato nella modalità base che è stata impostata.

È possibile inserire gli interi in qualsiasi base numerica. Se si tratta di un numero binario o esadecimale, utilizzare rispettivamente il prefisso 0b o 0h. Senza prefisso, gli interi vengono considerati decimali (base 10).

Se viene indicato un intero decimale troppo grande per una forma binaria con segno a 64 bit, verrà utilizzata un'operazione a modulo simmetrico per portare il valore nell'intervallo appropriato. Per ulteriori informazioni, vedere ▶**Base2**, pagina 21.

Nota: vedere **xor.**

In modalità base Esadecimale:

0h7AC36 or 0h3D5F	0h7BD7F

Importante: è zero, non la lettera O.

In modalità base Bin:

0b100101 or 0b100	0b100101

Nota: un numero binario può contenere fino a 64 cifre (oltre al prefisso 0b). Un numero esadecimale può contenere fino ad 16 cifre.

ord() Codice numerico di carattere — Catalogo >

ord(*Stringa***)⇒***intero*

ord(*Lista1***)⇒***lista*

Restituisce il codice numerico del primo carattere nella stringa di caratteri *Stringa*, oppure una lista dei primi caratteri di ciascun elemento della lista.

ord("hello")	104
char(104)	"h"
ord(char(24))	24
ord({"alpha","beta"})	{97,98}

P

P▸Rx() (Coordinata x rettangolare) — Catalogo >

P▸Rx(*rEspr***, θ***Espr***)⇒***espressione*

P▸Rx(*rLista***, θ***Lista***)⇒***lista*

P▸Rx(*rMatrice***, θ***Matrice***)⇒***matrice*

Restituisce la coordinata x equivalente della coppia (r, θ).

Nota: l'argomento θ viene interpretato come angolo in gradi, gradianti o radianti, conformemente alla modalità di misurazione degli angoli impostata. Se l'argomento è un'espressione, si può utilizzare il simbolo °, G o r per escludere tale impostazione provvisoriamente.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **P@>Rx(...)**.

In modalità angolo in radianti:

P▸Rx(4,60°)	2.
P▸Rx({-3,10,1.3},{π/3,-π/4,0})	{-1.5,7.07107,1.3}

P▸Ry() (Coordinata y rettangolare) — Catalogo >

P▸Ry(*rValore***, θ***Valore***)⇒***valore*

P▸Ry(*rLista***, θ***Lista***)⇒***lista*

P▸Ry(*rMatrice***, θ***Matrice***)⇒***matrice*

Restituisce la coordinata y equivalente della coppia (r, θ).

Nota: l'argomento θ viene interpretato come angolo in gradi, gradianti o radianti, conformemente alla modalità di misurazione degli angoli impostata.

In modalità angolo in radianti:

P▸Ry(4,60°)	3.4641
P▸Ry({-3,10,1.3},{π/3,-π/4,0})	{-2.59808,-7.07107,0}

P▸Ry() (Coordinata y rettangolare) Catalogo >

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **P@>Ry(...)**.

PassErr Catalogo >

PassErr

Passa un errore al livello successivo.

Se la variabile di sistema *errCode* è zero, **PassErr** non esegue alcuna azione.

L'istruzione **Else** del blocco **Try...Else...EndTry** dovrebbe utilizzare **ClrErr** o **PassErr**. Se l'errore deve essere elaborato o ignorato, utilizzare **ClrErr**. Se non si sa quale azione applicare all'errore, utilizzare **PassErr** per inviarlo al successivo blocco di gestione degli errori. Se non ci sono ulteriori blocchi di gestione degli errori **Try...Else...EndTry** in attesa di applicazione, la finestra di dialogo dell'errore viene visualizzata come normale.

Nota: Controlla anche **ClrErr**, pagina 28, e **Try**, pagina 168.

Nota per l'inserzione dell'esempio: Nell'applicazione Calcolatrice sul palmare, è possibile inserire definizioni su più linee premendo ↵ invece di enter alla fine di ogni riga. Sulla tastiera del computer, mantenere premuto **Alt** e premere **Invio**.

Per un esempio di **PassErr**, vedere l'esempio 2 del comando **Try**, pagina 169.

piecewise() (Funzione definita a tratti) Catalogo >

piecewise(*Espr1* **[,** *Condizione1* **[,** *Espr2* **[,** *Condizione2* **[,** ... **]]]])**

Restituisce definizioni di una funzione piecewise (definita a tratti) sotto forma di elenco. È inoltre possibile creare definizioni piecewise utilizzando un modello.

Nota: vedere anche **Modello di funzione piecewise** a pagina 7.

Define $p(x)=\begin{cases} x, & x>0 \\ \text{undef}, & x\le 0 \end{cases}$	*Done*
$p(1)$	1
$p(-1)$	undef

poissCdf() (Probabilità cumulata per la distribuzione discreta di Poisson) Catalogo >

poissCdf
(
λ
,
valoreInferiore,*valoreSuperiore*)⇒*numero* se *valoreInferiore* e *valoreSuperiore* sono numeri, *lista* se *valoreInferiore* e *valoreSuperiore* sono liste

poissCdf(λ,*valoreSuperiore*)(per P(0≤X ≤*valoreSuperiore*)⇒*numero* se *valoreSuperiore* è un numero, lista se *valoreSuperiore* è una lista

Calcola una probabilità cumulata per la distribuzione discreta di Poisson con la media λ specificata.

Per P(X ≤ *valoreSuperiore*), impostare *valoreInferiore*=0

poissPdf() (Probabilità per la distribuzione discreta di Poisson) Catalogo >

poissPdf(λ,*ValX*)⇒*numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola una probabilità per la distribuzione discreta di Poisson con la media λ specificata.

▸Polar (Visualizza come vettore polare) Catalogo >

Vettore **▸Polare**

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **@>Polar**.

Visualizza *vettore* in forma polare [r∠θ]. Il vettore deve essere bidimensionale e può essere sia una riga che una colonna.

Nota: ▸Polar è un'istruzione in formato di visualizzazione, non una funzione di conversione. Tale istruzione può essere impiegata solo alla fine di una riga di introduzione e non aggiorna *ans*.

$[1 \quad 3.]$▸Polar $[3.16228 \quad \angle 71.5651]$

▸Polar (Visualizza come vettore polare) Catalogo >

Nota: vedere anche **▸Rect**, pagina 130.

valoreComplesso **▸Polar**

Visualizza *vettoreComplesso* in forma polare.

- In modalità angolo in gradi, restituisce (r∠θ).
- In modalità angolo in radianti, restituisce $re^{i\theta}$.

valoreComplesso può avere una forma complessa qualsiasi. Tuttavia una voce $re^{i\theta}$ causa un errore nella modalità di misurazione degli angoli in gradi.

Nota: è necessario usare le parentesi per un inserimento polare (r∠θ).

In modalità angolo in radianti:

$(3+4\cdot i)$▸Polar	$e^{.927295\cdot i}\cdot 5$
$\left(\left(4\angle\frac{\pi}{3}\right)\right)$▸Polar	$e^{1.0472\cdot i}\cdot 4.$

In modalità angolo in gradianti (gradi centesimali):

$(4\cdot i)$▸Polar	$(4\angle 100)$

In modalità angolo in gradi:

$(3+4\cdot i)$▸Polar	$(5\angle 53.1301)$

polyEval() (Calcola polinomio) Catalogo >

polyEval(*Lista1***,** *Espr1***)⇒***espressione*

polyEval(*Lista1***,** *Lista2***)⇒***espressione*

Interpreta il primo argomento come coefficiente di un polinomio di grado decrescente e restituisce il polinomio calcolato per il valore del secondo argomento.

polyEval({1,2,3,4},2)	26
polyEval({1,2,3,4},{2,-7})	{26,-262}

polyRoots() Catalogo >

polyRoots(*Poli***,***Var***)** ⇒*lista*

polyRoots(*ListaDiCoeff***)** ⇒*lista*

La prima sintassi, **polyRoots(***Poli***,***Var***)**, restituisce una lista di radici reali del polinomio *Poli* in funzione della variabile *Var*. Se non esistono radici reali, restituisce una lista vuota: { }.

Poli deve essere un polinomio in forma normale in una variabile. Non utilizzare polinomi non ridotti a forma normale, quali $y^2\cdot y+1$ o $x\cdot x+2\cdot x+1$

polyRoots(y^3+1,y)	{-1}
cPolyRoots(y^3+1,y)	{-1,0.5−0.866025·i,0.5+0.866025·i}
polyRoots($x^2+2\cdot x+1,x$)	{-1,-1}
polyRoots({1,2,1})	{-1,-1}

polyRoots() Catalogo >

La seconda sintassi, **PolyRoots (***ListaDiCoeff***)**, restituisce una lista di radici reali per i coefficienti di *ListaDiCoeff*.

Nota: vedere anche **cPolyRoots()**, pagina 37.

PowerReg (Regressione su potenza) Catalogo >

PowerReg *X*,*Y* [, *Freq*] [, *Categoria*, *Includi*]]

Calcola la regressione su potenzay = (a · (x)^b)sulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*..

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a ·(x)^b
stat.a, stat.b	Coefficienti di regressione

Variabile di output	Descrizione
stat.r^2	Coefficiente di determinazione lineare di dati trasformati
stat.r	Coefficiente di correlazione per dati trasformati (ln(x), ln(y))
stat.Resid	Residui associati al modello di potenza
stat.ResidTrans	Residui associati all'adattamento lineare dei dati trasformati
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

Prgm — Catalogo >

Prgm
Blocco
EndPrgm

Modello per la creazione di un programma definito dall'utente. Deve essere utilizzato con il comando **Define**, **Define LibPub** o **Define LibPriv**.

Blocco può essere una singola istruzione, una serie di istruzioni separate dal carattere ":" o una serie di istruzioni su righe separate.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Calcolare il massimo comune divisore e visualizzare i risultati intermedi.

```
Define proggcd(a,b)=Prgm
                    Local d
                    While b≠0
                    d:=mod(a,b)
                    a:=b
                    b:=d
                    Disp a," ",b
                    EndWhile
                    Disp "GCD=",a
                    EndPrgm
                                    Done
```

```
proggcd(4560,450)
                                450 60
                                60 30
                                30 0
                                GCD=30
                                    Done
```

prodSeq() — Vedere P(), pagina 200.

Product (PI) (Prodotto) Vedere P(), pagina 200.

product() (Prodotto) Catalogo >

product(*Lista***[, *Inizio*[, *Fine*]])⇒*espressione***

Restituisce il prodotto degli elementi contenuti in *Lista*. *Inizio* e *Fine* sono opzionali e specificano un intervallo di elementi.

$\text{product}(\{1,2,3,4\})$ 24

$\text{product}(\{4,5,8,9\},2,3)$ 40

product(*Matrice1***[, *Inizio*[, *Fine*]])⇒*matrice***

Restituisce un vettore di riga contenente i prodotti degli elementi nelle colonne di *Matrice1*. *Inizio* e *Fine* sono opzionali e specificano un intervallo di elementi.

Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$\text{product}\left(\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}\right)$ $\begin{bmatrix}28&80&162\end{bmatrix}$

$\text{product}\left(\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix},1,2\right)$ $\begin{bmatrix}4&10&18\end{bmatrix}$

propFrac() (Frazione propria) Catalogo >

propFrac(*Valore1***[, *Var*])⇒*valore***

propFrac(*numero_razionale***)** restituisce *numero_razionale* sotto forma di somma di un numero intero e di una frazione, aventi lo stesso segno e denominatore di grandezza maggiore del numeratore.

propFrac(*espressione_razionale*,*Var***)** restituisce la somma delle frazioni proprie ed un polinomio rispetto a *Var*. Il grado di *Var* nel denominatore supera il grado di *Var* nel numeratore di ciascuna frazione propria. Le potenze simili di *Var* sono ridotte. I termini ed i rispettivi fattori sono ordinati considerando *Var* la variabile principale.

Se *Var* è omesso, viene eseguita l'espansione della frazione propria rispetto alla variabile più significativa. I coefficienti della parte polinomiale vengono trasformati in propri rispetto alla prima variabile più significativa, e così di seguito.

$\text{propFrac}\left(\frac{4}{3}\right)$ $1+\frac{1}{3}$

$\text{propFrac}\left(\frac{-4}{3}\right)$ $-1-\frac{1}{3}$

QR (Scomposizione QR) Catalogo >

QR *Matrice*, *MatriceQ*, *MatriceR*[, *Tol*]

Calcola la scomposizione QR di Householder di una matrice reale o complessa. Le matrici Q ed R che si ottengono vengono memorizzate nei *Matrice* specificati. La matrice Q è unitaria. La matrice R è triangolare superiore.

In alternativa, un elemento qualsiasi della matrice viene considerato zero se il suo valore assoluto è minore di *Tol*. Tale tolleranza viene utilizzata solo se la matrice contiene elementi a virgola mobile e non contiene variabili simboliche alle quali non sia stato assegnato un valore. In caso contrario, *Tol* viene ignorato.

- Se si usa ctrl enter oppure se si imposta la modalità **Auto o Approssimato** su Approssimato, i calcoli verranno eseguiti in virgola mobile.
- Se *Tol* viene omesso o non è utilizzato, la tolleranza predefinita viene calcolata come:
  5E⁻14 ·max(dim(*Matrice*)) ·rowNorm (*Matrice*)

La scomposizione QR viene calcolata in modo numerico tramite trasformazioni di Householder; la soluzione simbolica tramite Gram-Schmidt. Le colonne in *nomeMatriceQ* sono i vettori della base ortonormale con estensione pari allo spazio definito da *matrice*.

Il numero a virgola mobile (9.) in m1 fa sì che i risultati vengano calcolati nella forma a virgola mobile.

$\begin{bmatrix}1&2&3\\4&5&6\\7&8&9.\end{bmatrix}\rightarrow m1$	$\begin{bmatrix}1&2&3\\4&5&6\\7&8&9.\end{bmatrix}$
QR *m1*,*qm*,*rm*	*Done*
qm	$\begin{bmatrix}0.123091&0.904534&0.408248\\0.492366&0.301511&-0.816497\\0.86164&-0.301511&0.408248\end{bmatrix}$
rm	$\begin{bmatrix}8.12404&9.60114&11.0782\\0.&0.904534&1.80907\\0.&0.&0.\end{bmatrix}$

QuadReg (Regressione quadratica) Catalogo >

QuadReg *X*,*Y* [, *Freq*] [, *Categoria*, *Includi*]]

Calcola la regressione polinomiale quadraticay = a ·x²+b ·x+csulle liste *X* e *Y* con frequenza *Freq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

QuadReg (Regressione quadratica) Catalogo >

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a ·x^2+b ·x+c
stat.a, stat.b, stat.c	Coefficienti di regressione
stat.R^2	coefficiente di determinazione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

QuartReg (Regressione quartica) Catalogo >

QuartReg *X*,*Y* [, *Freq*] [, *Categoria*, *Includi*]]

QuartReg (Regressione quartica) Catalogo >

Calcola la regressione polinomiale quarticay = a ·x^4+b ·x^3+c · x^2+d ·x+esulle liste X e Y con frequenza $Freq$. Il riepilogo dei risultati è memorizzato nella variabile $stat.results$. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di $Includi$.

X e Y sono liste di variabili indipendenti e dipendenti.

$Freq$ è una lista opzionale di valori di frequenza. Ciascun elemento di $Freq$ specifica la frequenza di occorrenza di ogni dato corrispondente di X e Y. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

$Categoria$ è una lista di codici numerici o di stringa di categoria dei dati corrispondenti X e Y.

$Includi$ è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	**Descrizione**
stat.RegEqn	Equazione di regressione: a ·x^4+b ·x^3+c · x^2+d ·x+e
stat.a, stat.b, stat.c, stat.d, stat.e	Coefficienti di regressione
stat.R^2	coefficiente di determinazione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della $Lista\ X$ modificata attualmente usata nella regressione secondo le restrizioni di $Freq$, $Lista\ Categoria$ e $Includi\ Categorie$
stat.YReg	Lista di punti dati della $Lista\ Y$ modificata attualmente usata nella regressione secondo le restrizioni di $Freq$, $Lista\ Categoria$ e $Includi\ Categorie$
stat.FreqReg	Lista di frequenze corrispondenti a $stat.XReg$ e $stat.YReg$

R▸Pθ() (Coordinata polare) Catalogo >

R▸Pθ (*xValore***, *yValore*)⇒*valore***

R▸Pθ (*xLista***, *yLista*)⇒*lista***

R▸Pθ (*xMatrice***, *yMatrice*)⇒*matrice***

Restituisce la coordinata θ equivalente alla coppia di argomenti (*x*,*y*).

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`R@>Ptheta(...)`**.

In modalità angolo in gradi:

R▸Pθ(2,2)	45.

In modalità angolo in gradianti (gradi centesimali):

R▸Pθ(2,2)	50.

In modalità angolo in radianti:

R▸Pθ(3,2)	0.588003
R▸Pθ([3 -4 2],[0 π/4 1.5])	[0. 2.94771 0.643501]

R▸Pr() Coordinata polare Catalogo >

R▸Pr (*xValore***, *yValore*)⇒*valore***

R▸Pr (*xLista***, *yLista*)⇒*lista***

R▸Pr (*xMatrice***, *yMatrice*)⇒*matrice***

Restituisce la coordinata r equivalente alla coppia di argomenti (*x*,*y*).

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`R@>Pr(...)`**.

In modalità angolo in radianti:

R▸Pr(3,2)	3.60555
R▸Pr([3 -4 2],[0 π/4 1.5])	[3 4.07638 5/2]

▸Rad in angolo radiante Catalogo >

Valore1▸*Rad*⇒*valore*

Converte l'argomento in una misura d'angolo in radianti.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **`@>Rad`**.

In modalità angolo in gradi:

(1.5)▸Rad	$(0.02618)^r$

In modalità angolo in gradianti (gradi centesimali):

▸Rad in angolo radiante — Catalogo >

(1.5)▸Rad	$(0.023562)^r$

rand() (Numero casuale) — Catalogo >

rand()⇒*espressione*

rand(*numTentativi***)**⇒*lista*

rand() restituisce un numero casuale compreso tra 0 e 1.

rand(*numTentativi***)** restituisce una lista contenente *numTentativi* valori casuali compresi tra 0 e 1.

Imposta il seme dei numeri casuali.

RandSeed 1147	*Done*
rand(2)	{0.158206,0.717917}

randBin() (Numero casuale da distribuzione binomiale) — Catalogo >

randBin(*n***, *** p***)**⇒*espressione*

randBin(*n***, *** p***, *** numTentativi***)**⇒*lista*

randBin(*n***, *** p***)** restituisce un numero reale casuale da una distribuzione binomiale specificata.

randBin(*n***, *** p***, *** numTentativi***)** restituisce una lista contenente *numTentativi* numeri reali casuali da una distribuzione binomiale specificata.

randBin(80,0.5)	46.
randBin(80,0.5,3)	{43.,39.,41.}

randInt() (Intero casuale) — Catalogo >

randInt (*EstremoInf***,***EstremoSup***)**⇒*espressione*

randInt (*EstremoInf* ,*EstremoSup***,***numTentativi***)**⇒*lista*

randInt(*EstremoInf***,***EstremoSup***)** restituisce un intero casuale all'interno dell'intervallo specificato dagli numeri interi *EstremoInf* e *EstremoSup*.

randInt(3,10)	3.
randInt(3,10,4)	{9.,3.,4.,7.}

randInt() (Intero casuale) Catalogo >

randInt(*EstremoInf*,*EstremoSup*,*numTentativi***)** restituisce un elenco contenente *NumTentativi* interi casuali all'interno dell'intervallo specificato.

randMat() (Matrice casuale) Catalogo >

randMat(*numRighe*, *numColonne***)⇒***matrice*

Restituisce una matrice di numeri interi compresi tra -9 e 9 della dimensione specificata.

Entrambi gli argomenti devono potere essere semplificati in numeri interi.

RandSeed 1147	*Done*
randMat(3,3)	$\begin{bmatrix} 8 & -3 & 6 \\ -2 & 3 & -6 \\ 0 & 4 & -6 \end{bmatrix}$

Nota: i valori di questa matrice cambiano ogni volta che si preme enter.

randNorm() (Normale casuale) Catalogo >

randNorm(μ, σ**)⇒***espressione*

randNorm(μ, σ, *numTentativi***)⇒***lista*

Restituisce un numero decimale dalla distribuzione normale specifica. Può essere qualsiasi numero reale, anche se con maggiore probabilità sarà compreso nell'intervallo [μ−3 ·σ, μ+3 ·σ].

randNorm(μ, σ, *numTentativi***)** restituisce una lista contenente *numTentativi* valori decimali dalla distribuzione normale specificata.

RandSeed 1147	*Done*
randNorm(0,1)	0.492541
randNorm(3,4.5)	-3.54356

randPoly() (Polinomio casuale) Catalogo >

randPoly(*Var*, *Ordine***)⇒***espressione*

Restituisce un polinomio in *Var* dell'*Ordine* specificato. I coefficienti sono numeri interi casuali compresi nell'intervallo tra -9 e 9. Il primo coefficiente non può essere zero.

Ordine deve essere tra 0 e 99.

RandSeed 1147	*Done*
randPoly(x,5)	$-2\cdot x^5+3\cdot x^4-6\cdot x^3+4\cdot x-6$

randSamp() (Campione casuale) Catalogo >

randSamp(*Lista*,*NumTentativi* [,*noSostituzione*]**)⇒***lista*

Restituisce una lista contenente un campione casuale di *NumTentativi* da *Lista* con l'opzione di sostituire (*noSostituzione*=0) o meno (*noSostituzione*=1) il campione. L'impostazione predefinita prevede la sostituzione del campione.

Define *list3*={1,2,3,4,5}	*Done*
Define *list4*=randSamp(*list3*,6)	*Done*
list4	{1.,3.,3.,1.,3.,1.}

RandSeed (Seme numero casuale) Catalogo >

RandSeed *Numero*

Se *Numero* = 0, imposta i semi ai valori predefiniti per il generatore di numeri casuali. Se Numero ≠ 0, viene utilizzato per generare due semi, memorizzati nelle variabili di sistema seed1 e seed2.

RandSeed 1147	*Done*
rand()	0.158206

real() (Reale) Catalogo >

real(*Valore1***)⇒***valore*

Restituisce la parte reale dell'argomento.

Nota: tutte le variabili non definite vengono considerate come variabili reali. vedere anche **imag()**, pagina 75.

real(2+3·*i*)	2

real(*Lista1***)⇒***lista*

Restituisce le parti reali di tutti gli elementi.

real({1+3·*i*,3,*i*})	{1,3,0}

real(*Matrice1***)⇒***matrice*

Restituisce le parti reali di tutti gli elementi.

real$\left(\begin{bmatrix} 1+3\cdot i & 3 \\ 2 & i \end{bmatrix}\right)$	$\begin{bmatrix} 1 & 3 \\ 2 & 0 \end{bmatrix}$

▸Rect (Visualizza come vettore rettangolare) Catalogo >

Vettore **▸Rect**

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@>Rect`.

$\left(\begin{bmatrix} 3 & \angle\frac{\pi}{4} & \angle\frac{\pi}{6} \end{bmatrix}\right)$▸Rect
[1.06066 1.06066 2.59808]

▶Rect (Visualizza come vettore rettangolare) Catalogo >

Visualizza *Vettore* nella forma rettangolare [x, y, z]. Le dimensioni del vettore devono essere 2 o 3 e il vettore può essere una riga o una colonna.

Nota: **▶Rect** è un'istruzione del formato di visualizzazione, non una funzione di conversione. Tale istruzione può essere impiegata solo alla fine di una riga di introduzione e non aggiorna *ans*.

Nota: vedere anche **▶Polar**, pagina 119.

valoreComplesso **▶Rect**

Visualizza *valoreComplesso* nella forma rettangolare a+bi. *valoreComplesso* può avere una forma complessa qualsiasi. Tuttavia un inserimento nella forma $re^{i\theta}$ causa un errore nella modalità di misurazione degli angoli in gradi.

Nota: è necessario usare le parentesi per un inserimento polare ($r\angle\theta$).

In modalità angolo in radianti:

$$\left(4\cdot e^{\frac{\pi}{3}}\right)\blacktriangleright\text{Rect} \qquad 11.3986$$

$$\left(\left(4\angle\frac{\pi}{3}\right)\right)\blacktriangleright\text{Rect} \qquad 2.+3.4641\cdot i$$

In modalità angolo in gradianti (gradi centesimali):

$$\left(\left(1\angle 100\right)\right)\blacktriangleright\text{Rect} \qquad i$$

In modalità angolo in gradi:

$$\left(\left(4\angle 60\right)\right)\blacktriangleright\text{Rect} \qquad 2.+3.4641\cdot i$$

Nota: per inserire ∠ dalla tastiera, selezionarlo nell'elenco dei simboli del Catalogo.

ref() (Forma a scalini per righe) Catalogo >

ref(*Matrice1*[**,** *Tol*]**)⇒***matrice*

Restituisce la forma a scalini per righe di *Matrice1*.

$$\text{ref}\left(\begin{bmatrix}-2 & -2 & 0 & -6\\ 1 & -1 & 9 & -9\\ -5 & 2 & 4 & -4\end{bmatrix}\right) \qquad \begin{bmatrix}1 & \frac{-2}{5} & \frac{-4}{5} & \frac{4}{5}\\ 0 & 1 & \frac{4}{7} & \frac{11}{7}\\ 0 & 0 & 1 & \frac{-62}{71}\end{bmatrix}$$

In alternativa, un elemento qualsiasi della matrice viene considerato zero se il suo valore assoluto è minore di *Tol*. Tale tolleranza viene utilizzata solo se la matrice contiene elementi a virgola mobile e non contiene variabili simboliche alle quali non sia stato assegnato un valore. In caso contrario, *Tol* viene ignorato.

- Se si usa [ctrl] [enter] oppure se si imposta la modalità **Auto o Approssimato** su Approssimato, i calcoli verranno eseguiti in virgola mobile.
- Se *Tol* viene omesso o non è utilizzato, la tolleranza predefinita viene calcolata come:
  5E⁻14 ·max(dim(*Matrice1*)) ·rowNorm (*Matrice1*)

Evitare elementi indefiniti in *Matrice1*. Possono produrre risultati imprevisti.

Ad esempio, se a è indefinito nella seguente espressione, viene visualizzato un messaggio di avvertenza e il risultato viene mostrato come:

$$\text{ref}\left(\begin{bmatrix} a & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right) \qquad \begin{bmatrix} 1 & \frac{1}{a} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Il messaggio di avvertenza appare perché l'elemento generalizzato $1/a$ non è valido per a=0.

È possibile evitare questa situazione memorizzando in anticipo un valore in a oppure utilizzando l'operatore di vincolo (“|”), come mostrato nell'esempio che segue.

$$\text{ref}\left(\begin{bmatrix} a & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right) | a=0 \qquad \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Nota: vedere anche **rref()**, pagina 141.

remain() (Resto) Catalogo >

remain(*Valore1***,** *Valore2***)⇒***valore*

remain(*Lista1***,** *Lista2***)⇒***lista*

remain(*Matrice1***,** *Matrice2***)⇒***matrice*

Restituisce il resto del primo argomento rispetto al secondo argomento, come definito dalle identità:

remain(x,0) x

remain(x,y) x−y ·iPart(x/y)

Si noti che, di conseguenza **remain(**−x,y**)** −**remain(**x,y**)**. Il risultato può essere zero oppure può avere lo stesso segno del primo argomento.

Nota: vedere anche **mod()**, pagina 101.

remain(7,0)	7
remain(7,3)	1
remain(⁻7,3)	⁻1
remain(7,⁻3)	1
remain(⁻7,⁻3)	⁻1
remain({12,⁻14,16},{9,7,⁻5})	{3,0,1}

$$\text{remain}\left(\begin{bmatrix} 9 & -7 \\ 6 & 4 \end{bmatrix}, \begin{bmatrix} 4 & 3 \\ 4 & -3 \end{bmatrix}\right) \qquad \begin{bmatrix} 1 & -1 \\ 2 & 1 \end{bmatrix}$$

Request Catalogo >

Request*stringaPrompt***,** *var***[,** *FlagDispl* **[,** *varStato***]]**

Request*stringaPrompt***,** *funz***(***arg1***,** *...argn***)** **[,** *FlagDispl* **[,** *varStato***]]**

Istruzione di programmazione: Sospende il programma e visualizza una finestra di dialogo contenente il messaggio *stringaPrompt* e un riquadro di testo per la risposta dell'utente.

Dopo che l'utente ha digitato una risposta e ha fatto clic su **OK**, il contenuto del riquadro di testo viene assegnato alla variabile *var*.

Se l'utente fa clic su **Cancel (Annulla)**, il programma procede senza accettare input. Il programma utilizza il precedente valore di *var* se *var* è già stata definita.

L'argomento opzionale *FlagDispl* può essere qualsiasi espressione.

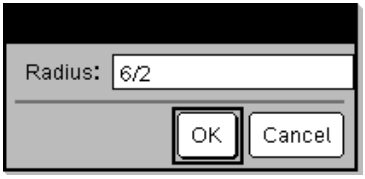
- Se *FlagDispl* viene omesso o dà come risultato **1**, la richiesta e la risposta da parte dell'utente vengono visualizzate nella cronologia della calcolatrice.

Definire un programma:

```
Define request_demo()=Prgm
  Request "Radius: ",r
  Disp "Area = ",pi*r²
EndPrgm
```

Eseguire il programma e digitare una risposta:

```
request_demo()
```



Risultato dopo aver selezionato **OK**:

```
Radius: 6/2
Area= 28.2743
```

- Se *FlagDispl* dà come risultato **0**, la richiesta e la risposta non vengono visualizzate nella cronologia.

L'argomento *varStato* opzionale offre al programma la possibilità di determinare la modalità di uscita dell'utente dalla finestra di dialogo. Notare che *varStato* richiede l'argomento *FlagDispl*.

- Se l'utente ha fatto clic su **OK** o ha premuto **Invio** oppure **Ctrl+Invio**, la variabile *varStato* viene impostata sul valore **1**.
- Altrimenti, la variabile *varStato* viene impostata sul valore **0**.

L'argomento *funz*() consente a un programma di memorizzare la risposta dell'utente come una definizione di funzione. Questa sintassi opera come se l'utente avesse eseguito il comando:

Define *funz*(*arg1*, ...*argn*) = *risposta utente*

Il programma può quindi usare la funzione definita *funz*(). La *stringaPrompt* dovrebbe portare l'utente ad inserire una *risposta utente* appropriata che completi la definizione di funzione.

Nota: è possibile utilizzare il comando **Request** all'interno di un programma definito dall'utente, ma non di una funzione.

Per arrestare un programma che contiene un comando **Request in un loop infinito:**

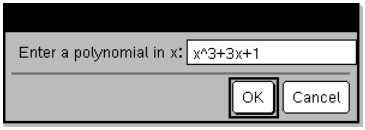
- **Palmare:** Tenere premuto il tasto [on] e premere [enter] più volte.
- **Windows®:** Tenere premuto il tasto **F12** e premere **Invio** più volte.
- **Macintosh®:** Tenere premuto il tasto **F5** e premere **Invio** più volte.
- **iPad®:** L'app mostra un prompt. È possibile continuare ad attendere o annullare.

Definire un programma:

```
Define polynomial()=Prgm
  Request "Enter a polynomial in x:",p(x)
  Disp "Real roots are:",polyRoots(p(x),x)
EndPrgm
```

Eseguire il programma e digitare una risposta:

```
polynomial()
```



Risultato dopo aver selezionato **OK**:

```
Enter a polynomial in x: x^3+3x+1
Le radici reali sono: {-0.322185}
```

Request Catalogo >

Nota: vedere anche **RequestStr, pagina 135.**

RequestStr Catalogo >

RequestStr*stringaPrompt*, *var*[, *FlagDispl*]

Istruzione di programmazione: Opera in modo identico alla prima sintassi del comando **Request**, eccetto che la risposta dell'utente viene sempre interpretata come stringa. A differenza di questo, il comando **Request** interpreta la risposta come espressione a meno che l'utente non la racchiuda tra virgolette (“”).

Nota: è possibile utilizzare il comando **RequestStr** all'interno di un programma definito dall'utente, ma non di una funzione.

Per arrestare un programma che contiene un comando **RequestStr** in un loop infinito:

- **Palmare:** Tenere premuto il tasto [on] e premere [enter] più volte.
- **Windows®:** Tenere premuto il tasto **F12** e premere **Invio** più volte.
- **Macintosh®:** Tenere premuto il tasto **F5** e premere **Invio** più volte.
- **iPad®:** L'app mostra un prompt. È possibile continuare ad attendere o annullare.

Nota: vedere anche **Request, pagina 133.**

Definire un programma:

```
Define requestStr_demo()=Prgm
  RequestStr “Your name:”,name,0
  Disp “Response has “,dim(name),”
characters.”
EndPrgm
```

Eseguire il programma e digitare una risposta:

```
requestStr_demo()
```

Risultato dopo aver selezionato **OK** (Si noti che l'argomento *FlagDispl* **0** omette la richiesta e la risposta dalla cronologia):

```
requestStr_demo()
La risposta ha 5 caratteri.
```

Return (Restituisci) Catalogo >

Return [*Espr*]

Restituisce *Espr* quale risultato della funzione. Questo comando viene utilizzato all’interno di un blocco **Func**...**EndFunc**.

Nota: utilizzare **Return** senza alcun argomento all’interno di un blocco **Prgm**...**EndPrgm** per uscire da un programma.

```
Define factorial(nn)=
Func
Local answer,counter
1→answer
For counter,1,nn
answer·counter→answer
EndFor
Return answer
EndFunc
```

factorial(3) 6

Return (Restituisci) Catalogo >

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

right() (Destra) Catalogo >

right(*Lista1***[, *Num*])⇒*lista***

Restituisce i *Num* elementi più a destra contenuti in *Lista1*.

Se si omette *Num*, restituisce l'intera *Lista1*.

```
right({1,3,-2,4},3)                    {3,-2,4}
```

right(*stringaOrigine***[, *Num*])⇒*stringa***

Restituisce i caratteri *Num* più a destra contenuti nella stringa di caratteri *stringaOrigine*.

Se si omette *Num*, restituisce l'intera *stringaOrigine*.

```
right("Hello",2)                       "lo"
```

right(*Confronto***)⇒*espressione***

Restituisce il secondo membro di un'equazione o di una disequazione.

rk23 () Catalogo >

rk23(*Espr***, *Var*, *varDipendente*, {*Var0*, *VarMax*}, *varDipendente0*, *incrVar* [, *tollErrore*])⇒*matrice***

rk23(*SistemaDiEspr***, *Var*, *ListaDiVarDipendenti*, {*Var0*, *VarMax*}, *ListaDiVarDipendenti0*, *incrVar* [, *tollErrore*]) ⇒*matrice***

rk23(*ListaDiEspr***, *Var*, *ListaDiVarDipendenti*, {*Var0*, *VarMax*}, *ListaDiVarDipendenti0*, *incrVar* [, *tollErrore*]) ⇒*matrice***

Utilizza il metodo di Runge-Kutta per risolvere il sistema

Equazione differenziale:

y'=0.001*y*(100-y) e y(0)=10

```
rk23(0.001·y·(100−y),t,y,{0,100},10,1)
[ 0.    1.        2.        3.        4.
 10.   10.9367   11.9493   13.042    14.2
```

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

Stessa equazione con *tollErrore* impostata su 1.E−6

$$\frac{d\,depVar}{d\,Var} = Expr(Var, depVar)$$

con *varDipendente* (*Var0*)=*varDipendente0* nell'intervallo [*Var0*,*VarMax*]. Restituisce una matrice la cui prima riga definisce i valori ottenuti di *Var* come definito da *incrVar*. La seconda riga definisce il valore del primo componente della soluzione per i valori di *Var* corrispondenti, e così via.

Espr è il secondo membro che definisce l'equazione differenziale ordinaria (ODE).

SistemaDiEspr è un sistema secondi membri che definiscono il sistema di ODE (corrisponde all'ordine di variabili dipendenti in *ListaDivarDipendenti*).

ListaDiEspr è una lista di secondi membri che definisce il sistema di ODE (corrisponde all'ordine di variabili dipendenti in *ListaDiVarDipendenti*).

Var è la variabile indipendente.

ListaDiVarDipendenti è una lista di variabili dipendenti.

{*Var0*, *VarMax*} è una lista a due elementi che indica alla funzione di integrare da *Var0* a *VarMax*.

ListaDiVarDipendenti0 è una lista di valori iniziali di variabili dipendenti.

Se *incrVar* dà come risultato un numero diverso da zero: sign(*incrVar*) = sign (*VarMax*-*Var0*) e sono restituite soluzioni a *Var0*+i**incrVar* per tutti i valori di i=0,1,2,... tali che *Var0*+i**incrVar* sia in [*var0*,*VarMax*] (potrebbe non esserci un valore di soluzione a *VarMax*).

Se *incrVar* dà come risultato zero, le soluzioni sono restituite ai valori *Var* di "Runge-Kutta".

$$\text{rk23}\left(0.001\cdot y\cdot(100-y), t, y, \{0,100\}, 10, 1, 1.\text{E}{-6}\right)$$

$$\begin{bmatrix} 0. & 1. & 2. & 3. & 4. \\ 10. & 10.9367 & 11.9495 & 13.0423 & 14.2189 \end{bmatrix}$$

Sistema di equazioni

$$\begin{cases} y1' = -y1 + 0.1\cdot y1\cdot y2 \\ y2' = 3\cdot y2 - y1\cdot y2 \end{cases}$$

con $y1(0)=2$ e $y2(0)=5$

$$\text{rk23}\left(\begin{cases} -y1+0.1\cdot y1\cdot y2 \\ 3\cdot y2-y1\cdot y2 \end{cases}, t, \{y1, y2\}, \{0,5\}, \{2,5\}, 1\right)$$

$$\begin{bmatrix} 0. & 1. & 2. & 3. & 4. \\ 2. & 1.94103 & 4.78694 & 3.25253 & 1.82848 \\ 5. & 16.8311 & 12.3133 & 3.51112 & 6.27245 \end{bmatrix}$$

rk23 ()

Catalogo >

tollErrore è la tolleranza d'errore (predefinita a 0.001).

root()

Catalogo >

root(*Valore***)⇒** *radice*

root(*Valore1***, *Valore2***)⇒** *radice*

root(*Valore***)** restituisce la radice quadrata di *Valore***.**

root(*Valore1***, *Valore2***)** restituisce la radice *Valore2* di *Valore1*. *Valore1* può essere una costante reale o complessa a virgola mobile, una costante razionale intera o complessa.

Nota: vedere anche **Modello di radice ennesima**, pagina 5.

$\sqrt[3]{8}$	2
$\sqrt[3]{3}$	1.44225

rotate() (Ruota)

Catalogo >

rotate(*Intero1***[,***numRotazioni***])⇒***intero***

Ruota i bit di un numero intero binario. È possibile inserire *Intero1* in qualsiasi base numerica; esso viene convertito automaticamente in forma binaria a 64 bit con segno. Se *Intero1* è troppo grande per questa forma, un'operazione a modulo simmetrico lo porta all'interno dell'intervallo. Per ulteriori informazioni, vedere ▶**Base2**, pagina 21.

Se *numRotazioni* è positivo, la rotazione avviene verso sinistra. Se *numRotazioni* è negativo, la rotazione avviene verso destra. L'impostazione predefinita è ⁻1 (ruota a destra di un bit).

Ad esempio, in una rotazione a destra:

Ogni bit ruota verso destra.

0b00000000000000111101011000011010 1

L'ultimo bit a destra diventa l'ultimo a sinistra.

In modalità base Bin:

rotate(0b11111111111111111111111111111111)	
0b10000000000000000000000000000001▸	
rotate(256,1)	0b1000000000

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

In modalità base Esadecimale:

rotate(0h78E)	0h3C7
rotate(0h78E,⁻2)	0h80000000000001E3
rotate(0h78E,2)	0h1E38

Importante: per inserire un numero binario o esadecimale, utilizzare sempre il prefisso 0b o 0h (è uno zero, non la lettera O).

rotate() (Ruota) Catalogo >

Dà:

0b10000000000000011110101100001101 0

I risultati vengono visualizzati nella modalità Base che è stata impostata.

rotate(*Lista1*[,*numRotazioni*]**)⇒***lista*

Restituisce una copia di *Lista1* ruotata a destra o a sinistra di *numRotazioni* elementi. Non modifica *Lista1*.

Se *numRotazioni* è positivo, la rotazione avviene verso sinistra. Se *numRotazioni* è negativo, la rotazione avviene verso destra. L'impostazione predefinita è ⁻1 (ruota a destra di un bit).

In modalità base Dec:

rotate({1,2,3,4})	{4,1,2,3}
rotate({1,2,3,4},⁻2)	{3,4,1,2}
rotate({1,2,3,4},1)	{2,3,4,1}

rotate(*Stringa1*[,*numRotazioni*]**)⇒***stringa*

Restituisce una copia di *Stringa1* ruotata a destra o a sinistra di *numRotazioni* caratteri. Non modifica *Stringa1*.

Se *numRotazioni* è positivo, la rotazione avviene verso sinistra. Se *numRotazioni* è negativo, la rotazione avviene verso destra. L'impostazione predefinita è ⁻1 (ruota a destra di un carattere).

rotate("abcd")	"dabc"
rotate("abcd",⁻2)	"cdab"
rotate("abcd",1)	"bcda"

round() (Arrotondamento) Catalogo >

round(*Valore1*[, *cifre*]**)⇒***valore*

Restituisce l'argomento arrotondato ad un numero specifico di cifre dopo la virgola decimale.

cifre deve essere un numero intero compreso tra 0 e 12. Se *cifre* non viene incluso, la funzione restituisce l'argomento arrotondato a 12 cifre significative.

Nota: la visualizzazione dipende dalla modalità selezionata.

round(1.234567,3)	1.235

round(*Lista1*[, *cifre*]**)⇒***lista*

Restituisce una lista degli elementi arrotondati al numero specifico di cifre.

round({π,√2,ln(2)},4)	{3.1416,1.4142,0.6931}

round() (Arrotondamento) Catalogo >

round(*Matrice1*[, *cifre*]**)⇒***matrice*

Restituisce una matrice degli elementi arrotondati al numero specifico di cifre.

$$\text{round}\left(\begin{bmatrix}\ln(5) & \ln(3)\\ \pi & e^1\end{bmatrix},1\right) \qquad \begin{bmatrix}1.6 & 1.1\\ 3.1 & 2.7\end{bmatrix}$$

rowAdd() (Somma di righe di matrice) Catalogo >

rowAdd(*Matrice1*, *rIndice1*, *rIndice2*)⇒***matrice***

Restituisce una copia di *Matrice1* nella quale la riga *rIndice2* è sostituita dalla somma delle righe *rIndice1* e *rIndice2*.

$$\text{rowAdd}\left(\begin{bmatrix}3 & 4\\ -3 & -2\end{bmatrix},1,2\right) \qquad \begin{bmatrix}3 & 4\\ 0 & 2\end{bmatrix}$$

rowDim() (Dimensione righe matrice) Catalogo >

rowDim(*Matrice***)⇒***espressione***

Restituisce il numero di righe di *Matrice*.

Nota: vedere anche **colDim()**, pagina 29.

$$\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}\rightarrow m1 \qquad \begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}$$

$$\text{rowDim}(m1) \qquad 3$$

rowNorm() (Norma righe matrice) Catalogo >

rowNorm(*Matrice***)⇒***espressione***

Restituisce il massimo delle somme dei valori assoluti degli elementi nelle righe di *Matrice*.

Nota: tutti gli elementi della matrice devono potere essere semplificati in numeri. Vedere anche **colNorm()**, pagina 29.

$$\text{rowNorm}\left(\begin{bmatrix}-5 & 6 & -7\\ 3 & 4 & 9\\ 9 & -9 & -7\end{bmatrix}\right) \qquad 25$$

rowSwap() (Inverti righe matrice) Catalogo >

rowSwap(*Matrice1*, *rIndice1*, *rIndice2*)⇒***matrice***

Restituisce *Matrice1* con le righe *rIndice1* e *rIndice2* scambiate.

$$\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}\rightarrow mat \qquad \begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}$$

$$\text{rowSwap}(mat,1,3) \qquad \begin{bmatrix}5 & 6\\ 3 & 4\\ 1 & 2\end{bmatrix}$$

rref() (Forma a scalini ridotta per righe)

Catalogo >

rref(*Matrice1*[, *Tol*]**)⇒***matrice*

Restituisce la forma a scalini ridotta per righe di *Matrice1*.

$$\text{rref}\left(\begin{bmatrix} -2 & -2 & 0 & -6 \\ 1 & -1 & 9 & -9 \\ -5 & 2 & 4 & -4 \end{bmatrix}\right) \qquad \begin{bmatrix} 1 & 0 & 0 & \frac{66}{71} \\ 0 & 1 & 0 & \frac{147}{71} \\ 0 & 0 & 1 & \frac{-62}{71} \end{bmatrix}$$

In alternativa, un elemento qualsiasi della matrice viene considerato zero se il suo valore assoluto è minore di *Tol*. Tale tolleranza viene utilizzata solo se la matrice contiene elementi a virgola mobile e non contiene variabili simboliche alle quali non sia stato assegnato un valore. In caso contrario, *Tol* viene ignorato.

- Se si usa ctrl enter oppure se si imposta la modalità **Auto o Approssimato** su Approssimato, i calcoli verranno eseguiti in virgola mobile.
- Se *Tol* viene omesso o non è utilizzato, la tolleranza predefinita viene calcolata come:
  5E⁻14 ·max(dim(*Matrice1*)) ·rowNorm (*Matrice1*)

Nota: vedere anche **ref()**, pagina 131.

S

sec() (Secante)

Tasto trig

sec(*Valore1***)** ⇒ *valore*

sec(*Lista1***)** ⇒ *lista*

Restituisce la secante di *Valore1* oppure restituisce una lista contenente le secanti di tutti gli elementi in *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, l'argomento viene interpretato come angolo in gradi, gradianti o radianti. È possibile utilizzare °, G o r per escludere provvisoriamente la modalità d'angolo selezionata.

In modalità angolo in gradi:

sec(45)	1.41421
sec({1,2.3,4})	{1.00015,1.00081,1.00244}

$\sec^{-1}()$ (Secante inversa) — Tasto trig

$\sec^{-1}$(*Valore1*) ⇒ *valore*

$\sec^{-1}$(*Lista1*) ⇒ *lista*

Restituisce l'angolo la cui secante è *Valore1* oppure restituisce una lista contenente le secanti inverse di ciascun elemento di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arcsec(...)`.

In modalità angolo in gradi:

$\sec^{-1}(1)$	0.

In modalità angolo in gradianti (gradi centesimali):

$\sec^{-1}(\sqrt{2})$	50.

In modalità angolo in radianti:

$\sec^{-1}(\{1,2,5\})$	$\{0,1.0472,1.36944\}$

sech() (Secante iperbolica) — Catalogo >

sech(*Valore1*) ⇒ *valore*

sech(*Lista1*) ⇒ *lista*

Restituisce la secante iperbolica di *Valore1* oppure restituisce una lista contenente le secanti iperboliche di tutti gli elementi di *Lista1*.

$\text{sech}(3)$	0.099328
$\text{sech}(\{1,2.3,4\})$	$\{0.648054,0.198522,0.036619\}$

$\text{sech}^{-1}()$ (Secante iperbolica inversa) — Catalogo >

sech^{-1}(*Valore1*) ⇒ *valore*

sech^{-1} (*Lista1*) ⇒ *lista*

Restituisce la secante iperbolica inversa di *Valore1* oppure restituisce una lista contenente le secanti iperboliche inverse di ciascun elemento di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arcsech(...)`.

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$\text{sech}^{-1}(1)$	0
$\text{sech}^{-1}(\{1,-2,2.1\})$	$\{0,2.0944\cdot i,8.\text{E}^{-}15+1.07448\cdot i\}$

Send — Menu Hub

Send *exprOrString1*[, *exprOrString2*] ...

Esempio: attivare l'elemento blu del LED RGB integrato per 0,5 secondi.

Send — Menu Hub

Istruzione di programmazione: invia uno o più TI-Innovator™ Hub comandi a un hub collegato.

exprOrString deve essere un TI-Innovator™ Hub comando valido. Normalmente, *exprOrString* contiene un comando **"SET ..."** per controllare un dispositivo o un comando **"READ ..."** per richiedere i dati.

Gli argomenti vengono inviati in successione all'hub.

Nota: è possibile utilizzare il comando **Send** all'interno di un programma definito dall'utente ma non di una funzione.

Nota: vedere anche **Get** (pagina 66), **GetStr** (pagina 70) e **eval()** (pagina 53).

Send "SET COLOR.BLUE ON TIME .5"	*Done*

Esempio: richiedere il valore attuale del sensore di livello luminosità integrato dell'hub. Un comando **Get** recupera il valore e lo assegna alla variabile *lightval*.

Send "READ BRIGHTNESS"	*Done*
Get *lightval*	*Done*
lightval	0.347922

Esempio: inviare una frequenza calcolata al diffusore integrato dell'hub. Utilizzare la variabile speciale *iostr.SendAns* per mostrare il comando hub con l'espressione valutata.

n:=50	50
m:=4	4
Send "SET SOUND eval(m· n)"	*Done*
iostr.SendAns	"SET SOUND 200"

seq() (Sequenza) — Catalogo >

seq(*Espr***, *Var*, *Basso*, *Alto*[, *Incr*])⇒*lista***

Aumenta *Var* da *Basso* a *Alto* con incrementi di *Incr*, calcola *Espr* e restituisce i risultati in forma di lista. Il contenuto originale di *Var* è intatto dopo l'esecuzione di **seq()**.

Il valore predefinito di *Incr* è 1.

$\text{seq}(n^2,n,1,6)$	$\{1,4,9,16,25,36\}$
$\text{seq}\left(\frac{1}{n},n,1,10,2\right)$	$\left\{1,\frac{1}{3},\frac{1}{5},\frac{1}{7},\frac{1}{9}\right\}$
$\text{sum}\left(\text{seq}\left(\frac{1}{n^2},n,1,10,1\right)\right)$	$\frac{1968329}{1270080}$

Nota: Per forzare un risultato approssimativo,

Palmare: Premere [ctrl] [enter].
Windows®: Premere **Ctrl+Invio**.
Macintosh®: Premere ⌘+**Invio**.
iPad®: Tenere premuto **Invio** e selezionare [≈].

$\text{sum}\left(\text{seq}\left(\frac{1}{n^2},n,1,10,1\right)\right)$	1.54977

seqGen() Catalogo >

seqGen(*Espr*, *Var*, *varDipendente*, {*Var0*, *VarMax*}[, *ListaDiTermIniziali* [, *incrVar* [, *ValArrotPerEccesso*]]]) ⇒*lista*

Genera una lista di termini per la successione *varDipendente*(*Var*)=*Espr* come segue: Incrementa la variabile indipendente *Var* da *Var0* fino a *VarMax* di *incrVar*, calcola *varDipendente*(*Var*) per i valori corrispondenti di *Var* utilizzando la formula *Espr* e *ListaDiTermIniziali* e restituisce i risultati in forma di lista.

seqGen(*ListaOSistemaDiEspr*, *Var*, *ListaDiVarDipendenti*, {*Var0*, *VarMax*} [, *MatriceDiTermIniziali* [, *incrVar* [, *ValArrotPerEccesso*]]]) ⇒*matrice*

Genera una matrice di termini per un sistema (o lista) di successioni *ListaDiVarDipendenti*(*Var*)=*ListaOSistemaDiEspr* come segue: Incrementa la variabile indipendente *Var* da *Var0* fino a *VarMax* di *incrVar*, calcola *ListaDiVarDipendenti*(*Var*) per i valori corrispondenti di *Var* utilizzando la formula *ListaOSistemaDiEspr* e *MatriceDiTermIniziali* e restituisce i risultati in forma di matrice.

Il contenuto originale di *Var* è intatto dopo l'esecuzione di **seqGen()**.

Il valore predefinito di *incrVar* è **1**.

Generare i primi 5 termini della sequenza $u(n) = u(n-1)^2/2$, con $u(1)=$**2** e *incrVar*=**1**.

$$\text{seqGen}\left(\frac{(u(n-1))^2}{n}, n, u, \{1,5\}, \{2\}\right) \qquad \left\{2, 2, \frac{4}{3}, \frac{4}{9}, \frac{16}{405}\right\}$$

Esempio in cui Var0=2:

$$\text{seqGen}\left(\frac{u(n-1)+1}{n}, n, u, \{2,5\}, \{3\}\right) \qquad \left\{3, \frac{4}{3}, \frac{7}{12}, \frac{19}{60}\right\}$$

Sistema di due successioni:

$$\text{seqGen}\left(\left\{\frac{1}{n}, \frac{u2(n-1)}{2}+u1(n-1)\right\}, n, \{u1, u2\}, \{1,5\}, \left\{\frac{_}{2}\right\}\right) \qquad \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ 2 & 2 & \frac{3}{2} & \frac{13}{12} & \frac{19}{24} \end{bmatrix}$$

Nota: il simbolo di nullo (_) nella matrice di termini iniziali sopra viene utilizzato per indicare che il termine iniziale per u1(n) è calcolato utilizzando la formula di sequenza esplicita u1(n)=1/n.

seqn() Catalogo >

seqn(*Espr*(*u*, *n* [, *ListaDiTermIniziali*[, *nMax* [, *ValArrotPerEccesso*]]])⇒*lista*

Genera una lista di termini per una successione *u*(*n*)=*Espr*(*u*, *n*) come segue: incrementa *n* da 1 a *nMax* di 1, calcola *u*(*n*) per i valori di *n* corrispondenti utilizzando la formula *Espr*(*u*, *n*) e *ListaDiTermIniziali* e restituisce i risultati in forma di lista.

seqn(*Espr*(*n* [, *nMax* [, *ValArrotPerEccesso*]])⇒*lista*

Genera i primi 6 termini della successione $u(n) = u(n-1)/2$, con $u(1)=$**2**.

$$\text{seqn}\left(\frac{u(n-1)}{n}, \{2\}, 6\right) \qquad \left\{2, 1, \frac{1}{3}, \frac{1}{12}, \frac{1}{60}, \frac{1}{360}\right\}$$

$$\text{seqn}\left(\frac{1}{n^2}, 6\right) \qquad \left\{1, \frac{1}{4}, \frac{1}{9}, \frac{1}{16}, \frac{1}{25}, \frac{1}{36}\right\}$$

seqn() Catalogo >

Genera una lista di termini per una successione non ricorsiva $u(n)=Espr(n)$ come segue: Incrementa n da 1 a $nMax$ di 1, calcola $u(n)$ per i valori di n corrispondenti utilizzando la formula $Espr(n)$ e restituisce i risultati in forma di lista.

Se manca $nMax$, $nMax$ viene impostata su 2500

Se $nMax$=0, $nMax$ viene impostato su 2500

Nota: seqn() chiama **seqGen( )** con $n0$=**1** e $nincr$ =**1**

setMode() Catalogo >

setMode(*interoNomeModo***,** *interoImpostazione***)** ⇒*intero*

setMode(*lista***)** ⇒*lista interi*

Valido solo all'interno di una funzione o di un programma.

setMode(*interoNomeModo***,** *interoImpostazione***)** imposta temporaneamente il modo *interoNomeModo* sulla nuova impostazione *interoImpostazione* e restituisce un intero corrispondente all'impostazione originale di quel modo. La modifica è limitata alla durata dell'esecuzione del programma o della funzione.

interoNomeModo specifica il modo da impostare. Deve essere uno degli interi dei modi della tabella seguente.

interoImpostazione specifica la nuova impostazione per il modo. Deve essere uno dei numeri interi per le impostazioni elencati di seguito per lo specifico modo che si sta impostando.

Visualizzare il valore approssimato di π utilizzando l'impostazione predefinita di Mostra cifre (Display Digits), quindi visualizzare π con l'impostazione Fissa2. Verificare che l'impostazione predefinita venga ripristinata dopo l'esecuzione del programma.

```
Define prog1()=Prgm                    Done
               Disp π
               setMode(1,16)
               Disp π
               EndPrgm
prog1()
                                 3.14159
                                 3.14
                                    Done
```

setMode(*lista***)** consente di modificare impostazioni multiple. *lista* contiene coppie di interi dei modi e di interi delle impostazioni. **setMode(***lista***)** restituisce una lista simile le cui coppie di interi rappresentano i modi e le impostazioni originali.

Se si salvano tutte le impostazioni di modo con **getMode(0)** → *var*, è possibile utilizzare **setMode(***var***)** per ripristinare tali impostazioni fintantoché la funzione o il programma esistono. Vedere **getMode()**, pagina 69.

Nota: le impostazioni di modo correnti vengono passate alle sottoroutine chiamate. Se una sottoroutine cambia un'impostazione di modo, tale modifica viene perduta quando il controllo torna alla routine di chiamata.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Nome modo	Intero modo	Interi impostazioni
Mostra cifre (Display digits)	1	**1**=Mobile, **2**=Mobile1, **3**=Mobile2, **4**=Mobile3, **5**=Mobile4, **6**=Mobile5, **7**=Mobile6, **8**=Mobile7, **9**=Mobile8, **10**=Mobile9, **11**=Mobile10, **12**=Mobile11, **13**=Mobile12, **14**=Fissa0, **15**=Fissa1, **16**=Fissa2, **17**=Fissa3, **18**=Fissa4, **19**=Fissa5, **20**=Fissa6, **21**=Fissa7, **22**=Fissa8, **23**=Fissa9, **24**=Fissa10, **25**=Fissa11, **26**=Fissa12
Angolo (Angle)	2	**1**=Radianti, **2**=Gradi, **3**=Gradianti
Formato esponenziale (Exponential Format)	3	**1**=Normale, **2**=Scientifico, **3**=Tecnico

Nome modo	Intero modo	Interi impostazioni
Reale o Complesso (Real or Complex)	4	**1**=Reale, **2**=Rettangolare, **3**=Polare
Auto o Approssimato (Auto or Approx.)	5	**1**=Auto, **2**=Approssimato
Formato vettoriale (Vector Format)	6	**1**=Rettangolare, **2**=Cilindrico, **3**=Sferico
Base	7	**1**=Decimale, **2**=Esa, **3**=Binario

shift() (Sposta) Catalogo >

shift(*Intero1*[,*numSpostamenti*]**)**⇒*intero*

Sposta i bit di un numero intero binario. È possibile inserire *Intero1* in qualsiasi base numerica; esso viene convertito automaticamente in forma binaria a 64 bit con segno. Se *Intero1* è troppo grande per questa forma, un'operazione a modulo simmetrico lo porta all'interno dell'intervallo. Per ulteriori informazioni, vedere ▶**Base2**, pagina 21.

Se *numSpostamenti* è positivo, lo spostamento avviene verso sinistra. Se *numSpostamenti* è negativo, lo spostamento avviene verso destra. L'impostazione predefinita è ⁻1 (sposta a destra di un bit).

In uno spostamento a destra, il bit più a destra viene eliminato e al suo posto viene inserito 0 o 1, in modo che corrisponda al bit più a sinistra. In uno spostamento a sinistra, il bit più a sinistra viene eliminato e viene inserito 0 come bit più a destra.

Ad esempio, in uno spostamento a destra:

ogni bit viene spostato verso destra.

0b0000000000000111101011000011010

In modalità base Bin:

shift(0b1111010110000110101)	0b111101011000011010
shift(256,1)	0b1000000000

In modalità base Esadecimale:

shift(0h78E)	0h3C7
shift(0h78E,⁻2)	0h1E3
shift(0h78E,2)	0h1E38

Importante: per inserire un numero binario o esadecimale, utilizzare sempre il prefisso 0b o 0h (è uno zero, non la lettera O).

shift() (Sposta) Catalogo >

Inserisce 0 se il bit più a sinistra è 0,

oppure 1 se il bit più a sinistra è 1.

Dà:

0b00000000000000011110101100001101 0

I risultati vengono visualizzati nella modalità Base che è stata impostata. Gli zeri iniziali non vengono visualizzati.

shift(*Lista1* [**,***numSpostamenti*]**)⇒***lista*

Restituisce una copia di *Lista1* spostata a destra o a sinistra di *numSpostamenti* elementi. Non modifica *Lista1*.

Se *numSpostamenti* è positivo, lo spostamento avviene verso sinistra. Se *numSpostamenti* è negativo, lo spostamento avviene verso destra. L'impostazione predefinita è ⁻1 (sposta a destra di un elemento).

Gli elementi introdotti all'inizio o alla fine di *lista* a seguito dello spostamento sono contrassegnati con il simbolo "undef".

In modalità base Dec:

shift({1,2,3,4})	{undef,1,2,3}
shift({1,2,3,4},⁻2)	{undef,undef,1,2}
shift({1,2,3,4},2)	{3,4,undef,undef}

shift(*Stringa1* [**,***numSpostamenti*]**)⇒***stringa*

Restituisce una copia di *Stringa1* spostata a sinistra o a destra di *numSpostamenti* caratteri. Non modifica *Stringa1*.

Se *numSpostamenti* è positivo, lo spostamento avviene verso sinistra. Se *numSpostamenti* è negativo, lo spostamento avviene verso destra. L'impostazione predefinita è ⁻1 (sposta a destra di un carattere).

I caratteri introdotti all'inizio o alla fine di *stringa* a seguito dello spostamento sono costituiti da uno spazio.

shift("abcd")	" abc"
shift("abcd",⁻2)	" ab"
shift("abcd",1)	"bcd "

sign() (Segno) Catalogo >

sign(*Valore1***)⇒***valore*

sign(*Lista1***)⇒***lista*

sign(⁻3.2)	⁻1
sign({2,3,4,⁻5})	{1,1,1,⁻1}

sign() (Segno) Catalogo >

sign(*Matrice1***)⇒***matrice*

Quando *Valore1* è reale o complesso, restituisce *Valore1* / **abs(***Valore1***)** se *Valore1* ≠ 0.

Restituisce 1 se *Valore1* è positivo.

Restituisce ⁻1 se *Valore1* è negativo.

sign(0) restituisce ±1 se la modalità Formato complesso è Reale; altrimenti restituisce se stesso.

sign(0) rappresenta la circonferenza unitaria del dominio complesso.

Con liste o matrici, restituisce i segni di tutti gli elementi.

Se la modalità del formato complesso è Reale:

$\text{sign}\left(\begin{bmatrix} -3 & 0 & 3 \end{bmatrix}\right)$ $\begin{bmatrix} -1 & \text{undef} & 1 \end{bmatrix}$

simult() Sistema di equazioni simultanee Catalogo >

simult(*matriceCoeff, vettoreCost [,Tol]***)⇒***matrice*

Restituisce un vettore colonna che contiene le soluzioni di un sistema di equazioni lineari.

Nota: vedere anche **linSolve()**, pagina 87.

matriceCoeff deve essere una matrice quadrata contenente i coefficienti delle equazioni.

vettoreCost deve avere lo stesso numero di righe (stesse dimensioni) di *matriceCoeff* e deve contenere i termini noti.

In alternativa, un elemento qualsiasi della matrice viene considerato zero se il suo valore assoluto è minore di *Tol*. Tale tolleranza viene utilizzata solo se la matrice contiene elementi a virgola mobile e non contiene variabili simboliche alle quali non sia stato assegnato un valore. In caso contrario, *Tol* viene ignorato.

- Se si imposta la modalità **Auto o Approssimato** su Approssimato, i calcoli

Risolvere rispetto a x e y:

x + 2y = 1

3x + 4y = ⁻1

$\text{simult}\left(\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right)$ $\begin{bmatrix} -3 \\ 2 \end{bmatrix}$

La soluzione è x=⁻3 e y=2.

Risolvere:

ax + by = 1

cx + dy = 2

$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \rightarrow matx1$ $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$

$\text{simult}\left(matx1, \begin{bmatrix} 1 \\ 2 \end{bmatrix}\right)$ $\begin{bmatrix} 0 \\ \frac{1}{2} \end{bmatrix}$

verranno eseguiti in virgola mobile.

- Se *Tol* viene omesso o non è utilizzato, la tolleranza predefinita viene calcolata come: 5E⁻14 ·max(dim(*matriceCoeff*)) ·rowNorm(*matriceCoeff*)

simult(*matriceCoeff***,** *matriceCost*[**,** *Tol*]**)⇒***matrice*

Risolve sistemi multipli di equazioni lineari in cui ogni sistema ha coefficienti delle equazioni uguali ma termini noti diversi.

Ogni colonna di *matriceCost* deve contenere i termini noti per un sistema di equazioni. Ogni colonna della matrice risultante contiene la soluzione per il sistema corrispondente.

Risolvere:

x + 2y = 1

3x + 4y = ⁻1

x + 2y = 2

3x + 4y = ⁻3

$$\text{simult}\left(\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix},\begin{bmatrix}1 & 2\\ -1 & -3\end{bmatrix}\right) \qquad \begin{bmatrix}-3 & -7\\ 2 & \frac{9}{2}\end{bmatrix}$$

Per il primo sistema, x=⁻3 e y=2. Per il secondo sistema, x=⁻7 e y=9/2.

sin() (Seno)

Tasto trig

sin(*Valore1***)⇒***valore*

sin(*Lista1***)⇒***lista*

sin(*Valore1***)** restituisce il seno dell'argomento.

sin(*Lista1***)** restituisce una lista di seni di tutti gli elementi di *Lista1*.

Nota: l'argomento è interpretato come angolo in gradi, gradianti o radianti, a seconda della modalità angolo correntemente impostata. Si può usare °,G o ʳ per escludere tale impostazione provvisoriamente.

In modalità angolo in gradi:

$\sin\left(\left(\frac{\pi}{4}\right)r\right)$	0.707107
$\sin(45)$	0.707107
$\sin(\{0,60,90\})$	$\{0.,0.866025,1.\}$

In modalità angolo in gradianti (gradi centesimali):

$\sin(50)$	0.707107

In modalità angolo in radianti:

$\sin\left(\frac{\pi}{4}\right)$	0.707107
$\sin(45°)$	0.707107

sin() (Seno)

Tasto trig

sin(*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce il seno della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare il seno di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti:

$$\sin\left(\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}\right)$$

$$\begin{bmatrix}0.9424 & -0.04542 & -0.031999\\ -0.045492 & 0.949254 & -0.020274\\ -0.048739 & -0.00523 & 0.961051\end{bmatrix}$$

$\sin^{-1}$() (Arcoseno)

Tasto trig

$\sin^{-1}$(*Valore1***)⇒***valore*

$\sin^{-1}$(*Lista1***)⇒***lista*

$\sin^{-1}$(*Valore1***)** restituisce l'angolo il cui seno è *Valore1***.**

$\sin^{-1}$(Lista1) restituisce una lista contenente l'inversa del seno di ciascun elemento di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`arcsin(…)`**.

In modalità angolo in gradi:

$\sin^{-1}(1)$ 90.

In modalità angolo in gradianti (gradi centesimali):

$\sin^{-1}(1)$ 100.

In modalità angolo in radianti:

$\sin^{-1}(\{0,0.2,0.5\})$ $\{0.,0.201358,0.523599\}$

$\sin^{-1}$(*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce l'inversa del seno della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare l'inversa del seno di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$$\sin^{-1}\left(\begin{bmatrix}1 & 5\\ 4 & 2\end{bmatrix}\right)$$

$$\begin{bmatrix}-0.174533-0.12198\cdot i & 1.74533-2.35591\cdot i\\ 1.39626-1.88473\cdot i & 0.174533-0.593162\cdot i\end{bmatrix}$$

sinh() (Seno iperbolico) Catalogo >

sinh(*Valore1***)⇒***valore*

sinh(*Lista1***)⇒***lista*

sinh (*Valore1***)** restituisce il seno iperbolico dell'argomento.

sinh (*Lista1***)** restituisce una lista del seno iperbolico di ciascun elemento di *Lista1*.

$\sinh(1.2)$ → 1.50946

$\sinh(\{0,1.2,3.\})$ → $\{0,1.50946,10.0179\}$

sinh (*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce il seno iperbolico della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare il seno iperbolico di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti:

$$\sinh\left(\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}\right) \rightarrow \begin{bmatrix}360.954 & 305.708 & 239.604\\ 352.912 & 233.495 & 193.564\\ 298.632 & 154.599 & 140.251\end{bmatrix}$$

$\sinh^{-1}()$ (Arcoseno iperbolico) Catalogo >

sinh⁻¹(*Valore1***)⇒***valore*

sinh⁻¹(*Lista1***)⇒***lista*

sinh⁻¹(*Valore1***)** restituisce l'inversa del seno iperbolico dell'argomento.

sinh⁻¹(*List1***)** restituisce una lista contenente l'inversa del seno iperbolico di ciascun elemento di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`arcsinh(…)`**.

$\sinh^{-1}(0)$ → 0

$\sinh^{-1}(\{0,2.1,3\})$ → $\{0,1.48748,1.81845\}$

sinh⁻¹ (*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce l'inversa del seno iperbolico della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare l'inversa del seno iperbolico di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

In modalità angolo in radianti:

$$\sinh^{-1}\left(\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}\right) \rightarrow \begin{bmatrix}0.041751 & 2.15557 & 1.1582\\ 1.46382 & 0.926568 & 0.112557\\ 2.75079 & -1.5283 & 0.57268\end{bmatrix}$$

$\sinh^{-1}()$ (Arcoseno iperbolico) Catalogo >

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

SinReg (Regressione sinusoidale) Catalogo >

SinReg *X*, *Y* [, [*Iterazioni*], [*Periodo*] [, *Categoria*, *Includi*]]

Calcola la regressione sinusoidale sulle liste *X* e *Y*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Iterazioni è un valore che specifica quante volte al massimo (da 1 a 6) verrà tentata una soluzione. Se omesso, viene utilizzato 8. Di solito valori più alti danno una maggiore accuratezza ma richiedono tempi di esecuzione più lunghi, e viceversa.

Periodo indica il periodo stimato. Se viene omesso, la differenza tra i valori di *X* deve essere uguale e in ordine sequenziale. Se invece *Periodo* viene specificato, le differenze tra valori x possono non essere uguali.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

L'output di **SinReg** è sempre espresso in radianti, indipendentemente dall'impostazione corrente della modalità di misurazione degli angoli.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.RegEqn	Equazione di regressione: a ·sin(bx+c)+d
stat.a, stat.b, stat.c, stat.d	Coefficienti di regressione
stat.Resid	Residui della regressione
stat.XReg	Lista di punti dati della *Lista X* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.YReg	Lista di punti dati della *Lista Y* modificata attualmente usata nella regressione secondo le restrizioni di *Freq*, *Lista Categoria* e *Includi Categorie*
stat.FreqReg	Lista di frequenze corrispondenti a *stat.XReg* e *stat.YReg*

SortA (Ordinamento ascendente) Catalogo >

SortA *Lista1*[, *Lista2*] [, *Lista3*] ...

SortA *Vettore1*[, *Vettore2*] [, *Vettore3*] ...

Questo comando permette di ordinare in modo ascendente gli elementi del primo argomento.

Se vengono inclusi altri argomenti, il comando permette di ordinare gli elementi di ciascuno di essi in modo che le loro nuove posizioni coincidano con le nuove posizioni degli elementi del primo argomento.

Tutti gli argomenti devono essere nomi di liste o di vettori e devono avere le stesse dimensioni.

Gli elementi vuoti (nulli) nel primo argomento si spostano in fondo. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$\{2,1,4,3\} \rightarrow list1$	$\{2,1,4,3\}$
SortA *list1*	*Done*
list1	$\{1,2,3,4\}$
$\{4,3,2,1\} \rightarrow list2$	$\{4,3,2,1\}$
SortA *list2*,*list1*	*Done*
list2	$\{1,2,3,4\}$
list1	$\{4,3,2,1\}$

SortD (Ordinamento discendente) Catalogo >

SortD *Lista1*[, *Lista2*] [, *Lista3*] ...

SortD *Vettore1*[,*Vettore2*] [,*Vettore3*] ...

Questo comando è identico a **SortA**, con la differenza che **SortD** ordina gli elementi in modo discendente.

Gli elementi vuoti (nulli) nel primo argomento si spostano in fondo. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$\{2,1,4,3\} \rightarrow list1$	$\{2,1,4,3\}$
$\{1,2,3,4\} \rightarrow list2$	$\{1,2,3,4\}$
SortD *list1*,*list2*	*Done*
list1	$\{4,3,2,1\}$
list2	$\{3,4,1,2\}$

▶Sphere (Visualizza come vettore sferico) Catalogo >

Vettore **▶Sphere**

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@>Sphere`.

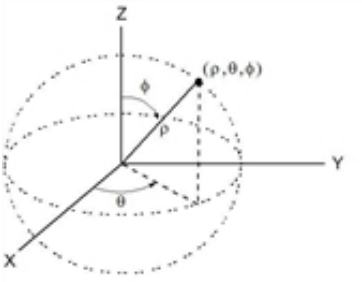
Visualizza il vettore di riga o colonna nel formato sferico [ρ ∠θ ∠ϕ].

Vettore deve avere dimensione 3 e può essere un vettore di riga o colonna.

Nota: ▶Sphere è un'istruzione nel formato di visualizzazione, ma non è una funzione di conversione. Tale istruzione può essere impiegata solo alla fine di una riga di introduzione.

$$\begin{bmatrix}1 & 2 & 3\end{bmatrix}\blacktriangleright\text{Sphere}$$
$$\begin{bmatrix}3.74166 & \angle 1.10715 & \angle 0.640522\end{bmatrix}$$

$$\left(\begin{bmatrix}2 & \angle\frac{\pi}{4} & 3\end{bmatrix}\right)\blacktriangleright\text{Sphere}$$
$$\begin{bmatrix}3.60555 & \angle 0.785398 & \angle 0.588003\end{bmatrix}$$



sqrt() (Radice quadrata) Catalogo >

sqrt(*Valore1***)⇒***valore*

sqrt(*Lista1***)⇒***lista*

Restituisce la radice quadrata dell'argomento.

$\sqrt{4}$	2
$\sqrt{\{9,2,4\}}$	$\{3,1.41421,2\}$

sqrt() (Radice quadrata) Catalogo >

In una lista, restituisce le radici quadrate di tutti gli elementi di *Lista1*.

Nota: vedere anche **Modello di radice quadrata**, pagina 5.

stat.results (Risultati dell'analisi statistica) Catalogo >

stat.results

Visualizza risultati di un calcolo statistico.

I risultati sono visualizzati sotto forma di coppie nome-valore. I nomi specifici mostrati dipendono dall'ultima funzione o comando statistico applicato.

È possibile copiare un nome o un valore e incollarlo in altre posizioni.

xlist:={1,2,3,4,5}	{1,2,3,4,5}
ylist:={4,8,11,14,17}	{4,8,11,14,17}
LinRegMx *xlist*,*ylist*,1: *stat.results*	
"Title"	"Linear Regression (mx+b)"
"RegEqn"	"m*x+b"
"m"	3.2
"b"	1.2
"r²"	0.996109
"r"	0.998053
"Resid"	"{...}"
stat.values	"Linear Regression (mx+b)"
	"m*x+b"
	3.2
	1.2
	0.996109
	0.998053
	"{⁻0.4,0.4,0.2,0.,⁻0.2}"

Nota: evitare di definire variabili con gli stessi nomi delle variabili utilizzate per l'analisi statistica. In alcuni casi, potrebbe prodursi un errore. I nomi di variabile utilizzati per l'analisi statistica sono elencati nella seguente tabella.

stat.a	stat.dfDenom	stat.MedianY	stat.Q3X	stat.SSBlock
stat.AdjR²	stat.dfBlock	stat.MEPred	stat.Q3Y	stat.SSCol
stat.b	stat.dfCol	stat.MinX	stat.r	stat.SSX
stat.b0	stat.dfError	stat.MinY	stat.r²	stat.SSY
stat.b1	stat.dfInteract	stat.MS	stat.RegEqn	stat.SSError
stat.b2	stat.dfReg	stat.MSBlock	stat.Resid	stat.SSInteract
stat.b3	stat.dfNumer	stat.MSCol	stat.ResidTrans	stat.SSReg
stat.b4	stat.dfRow	stat.MSError	stat.σx	stat.SSRow
stat.b5	stat.DW	stat.MSInteract	stat.σy	stat.tList
stat.b6	stat.e	stat.MSReg	stat.σx1	stat.UpperPred
stat.b7	stat.ExpMatrix	stat.MSRow	stat.σx2	stat.UpperVal
stat.b8	stat.F	stat.n	stat.Σx	stat.x̄

stat.b9	stat.FBlock	stat.$\hat{p}$	stat.Σx^2	stat.$\bar{x}1$
stat.b10	stat.Fcol	stat.$\hat{p}1$	stat.Σxy	stat.$\bar{x}2$
stat.bList	stat.FInteract	stat.$\hat{p}2$	stat.Σy	stat.$\bar{x}$Diff
stat.χ^2	stat.FreqReg	stat.$\hat{p}$Diff	stat.Σy^2	stat.$\bar{x}$List
stat.c	stat.Frow	stat.PList	stat.s	stat.XReg
stat.CLower	stat.Leverage	stat.PVal	stat.SE	stat.XVal
stat.CLowerList	stat.LowerPred	stat.PValBlock	stat.SEList	stat.XValList
stat.CompList	stat.LowerVal	stat.PValCol	stat.SEPred	stat.$\bar{y}$
stat.CompMatrix	stat.m	stat.PValInteract	stat.sResid	stat.$\hat{y}$
stat.CookDist	stat.MaxX	stat.PValRow	stat.SEslope	stat.$\hat{y}$List
stat.CUpper	stat.MaxY	stat.Q1X	stat.sp	stat.YReg
stat.CUpperList	stat.ME	stat.Q1Y	stat.SS	
stat.d	stat.MedianX			

Nota: ogni volta che Foglio elettronico calcola risultati statistici, l'applicazione copia le variabili di gruppo "stat." in un gruppo "stat#.", dove # è un numero che viene incrementato automaticamente. Questa funzione consente di mantenere i risultati precedenti mentre si eseguono calcoli multipli.

stat.values (Valori dei risultati) Catalogo >

stat.values

Visualizza una matrice dei valori calcolati per l'ultima funzione o comando statistico calcolato.

Diversamente da **stat.results**, **stat.values** omette i nomi associati ai valori.

È possibile copiare un valore e incollarlo in altre posizioni.

Per un esempio vedere **stat.results**.

stDevPop() (Deviazione standard della popolazione) Catalogo >

stDevPop(*Lista***[, ***listaFreq***])⇒***espressione***

Restituisce la deviazione standard degli elementi di *Lista*.

In modalità angolo in radianti:

stDevPop({1,2,5,-6,3,-2})	3.59398
stDevPop({1.3,2.5,-6.4},{3,2,5})	4.11107

stDevPop() (Deviazione standard della popolazione)

Catalogo >

Ogni elemento di *listaFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Lista*.

Nota: *Lista* deve avere almeno due elementi. Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

stDevPop(*Matrice1***[,** *matriceFreq***])⇒***matrice*

Restituisce un vettore di riga delle deviazioni standard della popolazione delle colonne di *Matrice1*.

Ogni elemento di *matriceFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Matrice1*.

Nota: *Matrice1* deve avere almeno due righe. Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$$\text{stDevPop}\left(\begin{bmatrix} 1 & 2 & 5 \\ -3 & 0 & 1 \\ 5 & 7 & 3 \end{bmatrix}\right)$$

$$\begin{bmatrix} 3.26599 & 2.94392 & 1.63299 \end{bmatrix}$$

$$\text{stDevPop}\left(\begin{bmatrix} -1.2 & 5.3 \\ 2.5 & 7.3 \\ 6 & -4 \end{bmatrix}, \begin{bmatrix} 4 & 2 \\ 3 & 3 \\ 1 & 7 \end{bmatrix}\right)$$

$$\begin{bmatrix} 2.52608 & 5.21506 \end{bmatrix}$$

stDevSamp() (Deviazione standard del campione)

Catalogo >

stDevSamp(*Lista***[,** *listaFreq***])⇒***espressione*

Restituisce la deviazione standard del campione di *Lista*.

Ogni elemento di *listaFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Lista*.

Nota: *Lista* deve avere almeno due elementi. Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$$\text{stDevSamp}(\{1,2,5,-6,3,-2\}) \qquad 3.937$$

$$\text{stDevSamp}(\{1.3,2.5,-6.4\},\{3,2,5\}) \qquad 4.33345$$

stDevSamp() (Deviazione standard del campione)

Catalogo >

stDevSamp(*Matrice1***[,** *matriceFreq***])⇒***matrice*

Restituisce un vettore di riga delle deviazioni standard del campione delle colonne di *Matrice1*.

Ogni elemento di *matriceFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Matrice1*.

Nota: *Matrice1* deve avere almeno due righe. Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$$\text{stDevSamp}\left(\begin{bmatrix}1 & 2 & 5\\ -3 & 0 & 1\\ 5 & 7 & 3\end{bmatrix}\right)$$

$$\begin{bmatrix}4. & 3.60555 & 2.\end{bmatrix}$$

$$\text{stDevSamp}\left(\begin{bmatrix}-1.2 & 5.3\\ 2.5 & 7.3\\ 6 & -4\end{bmatrix},\begin{bmatrix}4 & 2\\ 3 & 3\\ 1 & 7\end{bmatrix}\right)$$

$$\begin{bmatrix}2.7005 & 5.44695\end{bmatrix}$$

Stop

Catalogo >

Stop

Istruzione di programmazione: termina il programma.il programma.

Stop non è ammesso nelle funzioni.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
i:=0                                   0
Define prog1()=Prgm                    Done
              For i,1,10,1
              If i=5
              Stop
              EndFor
              EndPrgm
prog1()                                Done
i                                      5
```

Store (Memorizza)

Vedere → (memorizza), pagina 209.

string() (Da espressione a stringa)

Catalogo >

string(*Espr***)⇒***stringa*

Semplifica *Espr* e restituisce il risultato come una stringa di caratteri.

```
string(1.2345)                         "1.2345"
string(1+2)                            "3"
```

subMat() (Sottomatrice) Catalogo >

subMat(*Matrice1*[, *rigaInizio*] [, *colInizio*] [, *rigaFine*] [, *collFine*]**)** ⇒*matrice*

Restituisce la sottomatrice specificata di *Matrice1*.

Impostazioni predefinite: *rigaInizio*=1, *colInizio*=1, *rigaFine*=ultima riga, *colFine*=ultima colonna.

$$\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}\rightarrow m1 \qquad \begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$$

$$\text{subMat}(m1,2,1,3,2) \qquad \begin{bmatrix}4&5\\7&8\end{bmatrix}$$

$$\text{subMat}(m1,2,2) \qquad \begin{bmatrix}5&6\\8&9\end{bmatrix}$$

Sum (Sigma) Vedere Σ(), pagina 201.

sum() (Somma) Catalogo >

sum(*Lista*[, *Inizio*[, *Fine*]])⇒*espressione*

Restituisce la somma degli elementi di *Lista*.

Inizio e *Fine* sono opzionali e specificano un intervallo di elementi.

Un argomento vuoto produce un risultato vuoto. Gli elementi vuoti (nulli) di *Lista* vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$$\text{sum}(\{1,2,3,4,5\}) \qquad 15$$

$$\text{sum}(\{a,2\cdot a,3\cdot a\})$$

"Error: Variable is not defined"

$$\text{sum}(\text{seq}(n,n,1,10)) \qquad 55$$

$$\text{sum}(\{1,3,5,7,9\},3) \qquad 21$$

sum(*Matrice1*[, *Inizio*[, *Fine*]])⇒*matrice*

Restituisce un vettore di riga contenente le somme degli elementi nelle colonne della *Matrice1*.

Inizio e *Fine* sono opzionali e specificano un intervallo di elementi.

Un argomento vuoto produce un risultato vuoto. Gli elementi vuoti (nulli) di *Matrice1* vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$$\text{sum}\left(\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}\right) \qquad \begin{bmatrix}5&7&9\end{bmatrix}$$

$$\text{sum}\left(\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}\right) \qquad \begin{bmatrix}12&15&18\end{bmatrix}$$

$$\text{sum}\left(\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix},2,3\right) \qquad \begin{bmatrix}11&13&15\end{bmatrix}$$

sumIf() Catalogo >

sumIf(*Lista***,***Criteri*[, *listaSomma*]**)**⇒*valore*

Restituisce la somma cumulata di tutti gli elementi di *Lista* che soddisfano i *Criteri* specificati. Opzionalmente è possibile specificare una lista alternativa, *ListaSomma*, per fornire gli elementi da accumulare.

Lista può essere un'espressione, una lista o una matrice. *listaSomma*, se specificata, deve avere le stesse dimensioni di *Lista*.

Criteri può essere:

- Un valore, un'espressione o una stringa. Ad esempio, **34** accumula solo quegli elementi di *Lista* che sono semplificati nel valore 34.
- Un'espressione booleana contenente il simbolo **?** come segnaposto di ciascun elemento. Ad esempio, **?<10** accumula solo quegli elementi di *Lista* che sono minori di 10.

Quando un elemento di *Lista* soddisfa i *Criteri*, esso viene aggiunto alla somma accumulata. Se si include *listaSomma*, l'elemento corrispondente di *listaSomma* viene aggiunto invece alla somma.

In Foglio elettronico, è possibile utilizzare un intervallo di celle al posto di *Lista* e di *listaSomma*.

Gli elementi vuoti (nulli) vengono ignorati. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

Nota: vedere anche **countIf()**, pagina 36.

```
sumIf({1,2,e,3,π,4,5,6},2.5<?<4.5)
                          12.859874482
sumIf({1,2,3,4},2<?<5,{10,20,30,40})
                                   70
```

sumSeq() Vedere Σ(), pagina 201

system() (Sistema) Catalogo >

system(*Valore1* [, *Valore2* [, *Valore3* [,

system() (Sistema) Catalogo >

...]]])

Restituisce un sistema di equazioni, formattato come una lista. È inoltre possibile creare un sistema utilizzando un modello.

T

T (Trasposizione) Catalogo >

*Matrice1*T⇒*matrice*

Restituisce la trasposta dei complessi coniugati di *Matrice1*.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@t`.

$$\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}^T \qquad \begin{bmatrix}1&4&7\\2&5&8\\3&6&9\end{bmatrix}$$

tan() (Tangente) Tasto trig

tan(*Valore1***)⇒***valore*

tan(*Lista1***)⇒***lista*

tan(*Valore1***)** restituisce la tangente dell'argomento.

tan(*Lista1***)** restituisce una lista delle tangenti di tutti gli elementi di *Lista1*.

Nota: l'argomento è interpretato come angolo in gradi, gradianti o radianti, a seconda della modalità angolo correntemente impostata. Si può usare °, G o r per escludere tale impostazione provvisoriamente.

In modalità angolo in gradi:

$$\tan\left(\left(\frac{\pi}{4}\right)^r\right) \qquad 1.$$

$$\tan(45) \qquad 1.$$

$$\tan(\{0,60,90\}) \qquad \{0.,1.73205,\text{undef}\}$$

In modalità angolo in gradianti (gradi centesimali):

$$\tan\left(\left(\frac{\pi}{4}\right)^r\right) \qquad 1.$$

$$\tan(50) \qquad 1.$$

$$\tan(\{0,50,100\}) \qquad \{0.,1.,\text{undef}\}$$

In modalità angolo in radianti:

tan() (Tangente) Tasto trig

$\tan\left(\frac{\pi}{4}\right)$ 1.

$\tan(45°)$ 1.

$\tan\left(\left\{\pi,\frac{\pi}{3},-\pi,\frac{\pi}{4}\right\}\right)$ $\{0.,1.73205,0.,1.\}$

tan (*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce la tangente della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare la tangente di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti:

$$\tan\left(\begin{bmatrix}1&5&3\\4&2&1\\6&-2&1\end{bmatrix}\right)$$
$$\begin{bmatrix}-28.2912&26.0887&11.1142\\12.1171&-7.83536&-5.48138\\36.8181&-32.8063&-10.4594\end{bmatrix}$$

$\tan^{-1}$() (Arcotangente) Tasto trig

$\tan^{-1}$(*Valore1***)⇒***valore*

$\tan^{-1}$(*Lista1***)⇒***lista*

$\tan^{-1}$(*Valore1***)** restituisce l'angolo la cui tangente è *Valore1***.**

$\tan^{-1}$(*Lista1***)** restituisce una lista dell'inversa della tangente di ciascun elemento di *Lista1*.

Nota: conformemente alla modalità di misurazione degli angoli impostata, il risultato è in gradi, gradianti o radianti.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arctan(…)`.

$\tan^{-1}$ (*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce la tangente inversa della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare la tangente inversa di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

In modalità angolo in gradi:

$\tan^{-1}(1)$ 45

In modalità angolo in gradianti (gradi centesimali):

$\tan^{-1}(1)$ 50

In modalità angolo in radianti:

$\tan^{-1}(\{0,0.2,0.5\})$ $\{0,0.197396,0.463648\}$

In modalità angolo in radianti:

$$\tan^{-1}\left(\begin{bmatrix}1&5&3\\4&2&1\\6&-2&1\end{bmatrix}\right)$$
$$\begin{bmatrix}-0.083658&1.26629&0.62263\\0.748539&0.630015&-0.070012\\1.68608&-1.18244&0.455126\end{bmatrix}$$

tan⁻¹() (Arcotangente) Tasto trig

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

tanh() (Tangente iperbolica) Catalogo >

tanh(*Valore1***)⇒***valore*

tanh(*Lista1***)⇒***lista*

tanh(*Valore1***)** restituisce la tangente iperbolica dell'argomento.

tanh(*Lista1***)** restituisce una lista delle tangenti iperboliche di ciascun elemento di *Lista1*.

$\tanh(1.2)$ → 0.833655

$\tanh(\{0,1\})$ → $\{0.,0.761594\}$

tanh (*matriceQuadrata1***)⇒***matriceQuadrata*

Restituisce la tangente iperbolica della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare la tangente iperbolica di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

In modalità angolo in radianti:

$$\tanh\left(\begin{bmatrix} 1 & 5 & 3 \\ 4 & 2 & 1 \\ 6 & -2 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} -0.097966 & 0.933436 & 0.425972 \\ 0.488147 & 0.538881 & -0.129382 \\ 1.28295 & -1.03425 & 0.428817 \end{bmatrix}$$

tanh⁻¹() (Arcotangente iperbolica) Catalogo >

tanh⁻¹(*Valore1***)⇒***valore*

tanh⁻¹(*Lista1***)⇒***lista*

tanh⁻¹(*Valore1***)** restituisce l'inversa della tangente iperbolica dell'argomento.

tanh⁻¹(*Lista1***)** restituisce una lista dell'inversa della tangente iperbolica di ciascun elemento di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `arctanh(...)`.

In modalità formato rettangolare complesso:

$\tanh^{-1}(0)$ → 0.

$\tanh^{-1}(\{1,2.1,3\})$
$\{\text{undef},0.518046-1.5708\cdot i,0.346574-1.570$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

tanh⁻¹ (*matriceQuadrata1***)⇒***matriceQuadrata*

In modalità angolo in radianti e in modalità formato rettangolare complesso:

$\tanh^{-1}()$ (Arcotangente iperbolica) Catalogo >

Restituisce la tangente iperbolica inversa della matrice di *matriceQuadrata1*. Ciò non equivale a calcolare la tangente iperbolica inversa di ogni elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

$$\tanh^{-1}\left(\begin{bmatrix}1 & 5 & 3\\ 4 & 2 & 1\\ 6 & -2 & 1\end{bmatrix}\right)$$

$$\begin{bmatrix}-0.099353+0.164058\cdot i & 0.267834-1.4908 \\ -0.087596-0.725533\cdot i & 0.479679-0.9473 \\ 0.511463-2.08316\cdot i & -0.878563+1.7901\end{bmatrix}$$

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

tCdf() (Probabilità di distribuzione t di Student) Catalogo >

tCdf(*estremoInf***,***estremoSup***,***gl***)⇒***numero*
se *estremoInf* e *estremoSup* sono numeri, *lista* se *estremoInf* e *estremoSup* sono liste

Calcola la probabilità della distribuzione *t* di Student tra il *estremoInf* e il *estremoSup* per i gradi di libertà *gl* specificati.

Per P(X ≤ *valoreSuperiore*), impostare *valoreInferiore* = **−9E**999.

Text Catalogo >

Text*stringaPrompt*[**,** *FlagDispl*]

Istruzione di programmazione: Sospende il programma e visualizza la stringa di caratteri *stringaPrompt* in una finestra di dialogo.

Selezionando **OK**, l'esecuzione del programma continua.

L'argomento opzionale *flag* può essere un'espressione.

- Se *FlagDispl* viene omesso o dà come risultato **1**, il messaggio testuale viene aggiunto alla cronologia di Calcolatrice.
- Se *FlagDispl* dà come risultato **0**, il messaggio testuale non viene aggiunto alla cronologia.

Definire un programma che si arresta momentaneamente per visualizzare ciascuno dei cinque numeri casuali in una finestra di dialogo.

All'interno del modello Prgm...EndPrgm, completare ciascuna riga premendo ↵ invece di enter. Sulla tastiera del computer, mantenere premuto **Alt** e premere **Invio**.

```
Define text_demo()=Prgm
  For i,1,5
    strinfo:=”Random number “ &
string(rand(i))
    Text strinfo
  EndFor
```

Text

Catalogo >

Se il programma richiede di digitare una risposta, vedere **Request, pagina 133 oppure RequestStr, pagina 135.**

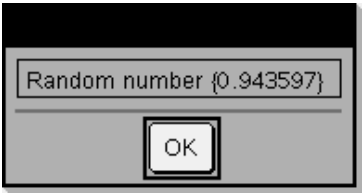
Nota: è possibile utilizzare questo comando all'interno di un programma definito dall'utente, ma non di una funzione.

```
EndPrgm
```

Eseguire il programma:

```
text_demo()
```

Esempio di finestra di dialogo:



Then

Vedere If, pagina 73.

tInterval (Intervallo di confidenza t)

Catalogo >

tInterval *Lista*[**,***Freq*[**,***livelloConfidenza*]]

(Input lista dati)

tInterval $\bar{x}$**,***sx***,***n***[,***livelloConfidenza***]**

(Input statistiche riepilogo)

Calcola un intervallo di confidenza *t*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.CLower, stat.CUpper	Intervallo di confidenza per una media non nota di una popolazione
stat.$\bar{x}$	Media del campione della successione di dati dalla distribuzione casuale normale
stat.ME	Margine di errore

Variabile di output	Descrizione
stat.df	Gradi di libertà
stat.σx	Deviazione standard del campione
stat.n	Lunghezza della successione di dati con media del campione

tInterval_2Samp (Intervallo di confidenza t su due campioni)

Catalogo >

tInterval_2Samp *Lista1*,*Lista2*[,*Freq1*[,*Freq2*[,*livelloConfidenza*[,*Aggregata*]]]]

(Input lista dati)

tInterval_2Samp $\bar{x}1$,*sx1*,*n1*,$\bar{x}2$,*sx2*,*n2*[,*livelloConfidenza*[,*Aggregata*]]

(Input statistiche riepilogo)

Calcola un intervallo di confidenza *t* su due campioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Aggregata=**1** aggrega le varianze; *Aggregata*=**0** non aggrega le varianze.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.CLower, stat.CUpper	Intervallo di confidenza contenente la probabilità di distribuzione del livello di confidenza
stat.$\bar{x}$1-$\bar{x}$2	Medie dei campioni delle successioni di dati dalla distribuzione casuale normale
stat.ME	Margine di errore
stat.df	Gradi di libertà
stat.$\bar{x}$1, stat.$\bar{x}$2	Medie dei campioni delle successioni di dati dalla distribuzione casuale normale
stat.σx1, stat.σx2	Deviazioni standard dei campioni di *Lista 1* e *Lista 2*
stat.n1, stat.n2	Numero di campioni nelle successioni di dati
stat.sp	Deviazione standard aggregata. Calcola quando *Aggregata* = sì.

tPdf() (Densità di probabilità t di Student)

Catalogo >

tPdf(*ValX***,***gl***)⇒***numero* se *ValX* è un numero, *lista* se *ValX* è una lista

Calcola la funzione della densità di probabilità (pdf) per la distribuzione *t* di Student in corrispondenza di un valore *x* specificato con i gradi di libertà *gl* specificati.

trace() (Traccia)

Catalogo >

trace(*matriceQuadrata***)⇒***valore*

Restituisce la traccia (somma di tutti gli elementi sulla diagonale principale) di *matriceQuadrata*.

$$\text{trace}\left(\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}\right) \qquad 15$$

$$a:=12 \qquad 12$$

$$\text{trace}\left(\begin{bmatrix}a & 0\\ 1 & a\end{bmatrix}\right) \qquad 24$$

Try (Tentativo)

Catalogo >

Try

blocco1

Else

blocco2

EndTry

Esegue *blocco1* a meno che non si verifichi un errore. In questo caso, l'esecuzione del programma viene trasferita a *blocco2*. La variabile di sistema *errCode* contiene il codice di errore che consente al programma di eseguire il ripristino dell'errore Per un elenco dei codici di errore, vedere "*Codici di errore e messaggi*", pagina 218.

blocco1 e *blocco2* possono essere o una singola istruzione o una serie di istruzioni separate dal carattere ":".

```
Define prog1()=Prgm
                Try
                z:=z+1
                Disp "z incremented."
                Else
                Disp "Sorry, z undefined."
                EndTry
                EndPrgm
                                        Done
z:=1:prog1()
                                z incremented.
                                        Done
DelVar z:prog1()
                                Sorry, z undefined.
                                        Done
```

Try (Tentativo) Catalogo >

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Esempio 2

Per vedere i comandi **Try**, **ClrErr** e **PassErr** in funzione, il programma eigenvals() riportato sulla destra. Avviare il programma eseguendo ciascuna delle seguenti espressioni.

$$eigenvals\left(\begin{bmatrix} -3 \\ -41 \\ 5 \end{bmatrix}, \begin{bmatrix} -1 & 2 & -3.1 \end{bmatrix}\right)$$

Nota: vedere anche **ClrErr**, pagina 28 e **PassErr**, pagina 118.

```
Define eigenvals(a,b)=Prgm
© Questo programma visualizza gli autovalori di A·B
Try
  Disp "A= ",a
  Disp "B= ",b
  Disp " "
  Disp "Eigenvalues of A·B are:",eigVl(a*b)
Else
  If errCode=230 Then
    Disp "Error: Product of A·B must be a square matrix"
    ClrErr
  Else
    PassErr
  EndIf
EndTry
EndPrgm
```

tTest (Verifica t) Catalogo >

tTest $\mu 0, Lista[, Freq[, Ipotesi]]$

(Input lista dati)

tTest $\mu 0, \bar{x}, sx, n, [Ipotesi]$

(Input statistiche riepilogo)

tTest (Verifica t) Catalogo >

Esegue una verifica dell'ipotesi su un'unica media μ non nota di una popolazione quando la deviazione standard σ della popolazione non è nota. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Viene verificata l'ipotesi nulla H_0: μ = μ0 in contrapposizione a una delle alternative seguenti:

Per H_a: μ < μ0, impostare *Ipotesi*<0

Per H_a: μ ≠ μ0 (default), impostare *Ipotesi*=0

Per H_a: μ > μ0, impostare *Ipotesi*>0

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.t	$(\bar{x} - \mu 0) / (stdev / sqrt(n))$
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.df	Gradi di libertà
stat.$\bar{x}$	Media del campione della sequenza di dati in *Lista*
stat.sx	Deviazione standard del campione della sequenza di dati
stat.n	Dimensione dei campioni

tTest_2Samp (Verifica t su due campioni) Catalogo >

tTest_2Samp *Lista1*,*Lista2*[,*Freq1*[,*Freq2*[,*Ipotesi*[,*Aggregata*]]]]

(Input lista dati)

tTest_2Samp $\bar{x}$*1*,*sx1*,*n1*,$\bar{x}$*2*,*sx2*,*n2*[,*Ipotesi*[,*Aggregata*]]

(Input statistiche riepilogo)

tTest_2Samp (Verifica t su due campioni)

Catalogo >

Esegue una verifica *t* su due campioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Viene verificata l'ipotesi nulla H_0: $\mu 1 = \mu 2$ in contrapposizione a una delle alternative seguenti:

Per H_a: $\mu 1 < \mu 2$, impostare *Ipotesi*<0

Per H_a: $\mu 1 \neq \mu 2$ (default), impostare *Ipotesi*=0

Per H_a: $\mu 1 > \mu 2$, impostare *Ipotesi*>0

Aggregata=**1** aggrega le varianze

Aggregata=**0** non aggrega le varianze

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	**Descrizione**
stat.t	Valore normale standard calcolato per la differenze delle medie
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.df	Gradi di libertà della statistica t
stat.$\bar{x}$1, stat.$\bar{x}$2	Medie dei campioni delle sequenze di dati di *Lista1* e *Lista2*
stat.sx1, stat.sx2	Deviazioni standard dei campioni delle sequenze di dati di *Lista1* e *Lista2*
stat.n1, stat.n2	Dimensione dei campioni
stat.sp	Deviazione standard aggregata. Calcolato quando *Aggregata*=1.

tvmFV()

Catalogo >

tvmFV(*N*,*I*,*PV*,*Pmt*,[*PpY*],[*CpY*],[*PmtAt*]**)**⇒*valore*

Funzione finanziaria che calcola il valore futuro del denaro.

```
tvmFV(120,5,0,-500,12,12)        77641.1
```

tvmFV() Catalogo >

Nota: gli argomenti utilizzati nelle funzioni TVM sono descritti nella tabella degli argomenti TVM, pagina 173. Vedere anche **amortTbl()**, pagina 11.

tvmI() Catalogo >

tvmI(*N***,***PV***,***Pmt***,***FV***,**[*PpY*]**,**[*CpY*]**,** [*PmtAt*]**)⇒***valore*

Funzione finanziaria che calcola il tasso di interesse annuo.

Nota: gli argomenti utilizzati nelle funzioni TVM sono descritti nella tabella degli argomenti TVM, pagina 173. Vedere anche **amortTbl()**, pagina 11.

```
tvmI(240,100000,-1000,0,12,12)        10.5241
```

tvmN() Catalogo >

tvmN(*I***,***PV***,***Pmt***,***FV***,**[*PpY*]**,**[*CpY*]**,** [*PmtAt*]**)⇒***valore*

Funzione finanziaria che calcola il numero di periodi di pagamento.

Nota: gli argomenti utilizzati nelle funzioni TVM sono descritti nella tabella degli argomenti TVM, pagina 173. Vedere anche **amortTbl()**, pagina 11.

```
tvmN(5,0,-500,77641,12,12)            120.
```

tvmPmt() Catalogo >

tvmPmt(*N***,***I***,***PV***,***FV***,**[*PpY*]**,**[*CpY*]**,** [*PmtAt*]**)⇒***valore*

Funzione finanziaria che calcola l'importo di ciascuna rata.

Nota: gli argomenti utilizzati nelle funzioni TVM sono descritti nella tabella degli argomenti TVM, pagina 173. Vedere anche **amortTbl()**, pagina 11.

```
tvmPmt(60,4,30000,0,12,12)            -552.496
```

tvmPV() Catalogo >

tvmPV(*N*,*I*,*Pmt*,*FV*,[*PpY*],[*CpY*], [*PmtAt*]**)**⇒*valore*

tvmPV(48,4,⁻500,30000,12,12) ⁻3426.7

Funzione finanziaria che calcola il valore presente.

Nota: gli argomenti utilizzati nelle funzioni TVM sono descritti nella tabella degli argomenti TVM, pagina 173. Vedere anche **amortTbl()**, pagina 11.

Argomento TVM *	Descrizione	Tipo di dati
N	Numero di periodi di pagamento	Numero reale
I	Tasso di interesse annuo	Numero reale
PV	Valore presente	Numero reale
Pmt	Importo della rata	Numero reale
FV	Valore futuro	Numero reale
PpY	Rate all'anno, impostazione predefinita=1	Numero intero > 0
CpY	Periodi di capitalizzazione all'anno, impostazione predefinita=1	Numero intero > 0
PmtAt	Pagamento versato alla fino o all'inizio di ogni periodo di pagamento, impostazione predefinita=fine	Numero intero (0=fine, 1=inizio)

***** Questi nomi di argomenti TVM sono simili ai nomi delle variabili TVM (come ad esempio **tvm.pv** e **tvm.pmt**) che sono utilizzate dal Risolutore finanziario dell'applicazione *Calculator*. Le funzioni finanziarie, tuttavia, non memorizzano automaticamente i valori degli argomenti o i risultati in variabili TVM.

TwoVar (Risultati a due variabili) Catalogo >

TwoVar *X*, *Y*[, [*Freq*] [, *Categoria*, *Includi*]]

Calcola le statistiche TwoVar. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Tutte le liste devono avere le stesse dimensioni, ad eccezione di *Includi*.

X e *Y* sono liste di variabili indipendenti e dipendenti.

Freq è una lista opzionale di valori di frequenza. Ciascun elemento di *Freq* specifica la frequenza di occorrenza di ogni dato corrispondente di *X* e *Y*. Il valore predefinito è 1. Tutti gli elementi devono essere numeri interi ≥ 0.

Categoria è una lista di codici numerici o di stringa di categoria dei dati corrispondenti *X* e *Y*.

Includi è una lista di uno o più codici di categoria. Solo quei dati il cui codice di categoria è inserito in questa lista vengono inclusi nel calcolo.

Un elemento vuoto (nullo) in qualsiasi lista *X*, *Freq* o *Categoria* produce un corrispondente elemento vuoto in tutte queste liste. Un elemento vuoto (nullo) in qualsiasi lista da *X1* a *X20* produce un corrispondente elemento vuoto in tutte queste liste. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

Variabile di output	**Descrizione**
stat.$\bar{x}$	Media dei valori X
stat. x	Somma dei valori X
stat. x2	Somma dei valori x2
stat.sx	Deviazione standard del campione di X
stat. x	Deviazione standard della popolazione di X
stat.n	Numero dei punti di dati
stat.$\bar{y}$	Media dei valori y
stat. y	Somma dei valori y
stat. y^2	Somma dei valori y2
stat.sy	Deviazione standard dei campione di y
stat. y	Deviazione standard della popolazione di y
stat. xy	Somma dei valori x ·y
stat.r	Coefficiente di correlazione

Variabile di output	Descrizione
stat.MinX	Minimo dei valori x
stat.Q_1X	1° quartile di x
stat.MedianX	Mediana di x
stat.Q_3X	3° quartile di x
stat.MaxX	Massimo dei valori x
stat.MinY	Minimo dei valori y
stat.Q_1Y	1° quartile di x
stat.MedY	Mediana di y
stat.Q_3Y	3° quartile di y
stat.MaxY	Massimo dei valori Y
stat. $(x-)^2$	Somma dei quadrati delle deviazioni dalla media di x
stat. $(y-)^2$	Somma dei quadrati delle deviazioni dalla media di y

U

unitV() (Vettore unità) Catalogo >

unitV(*Vettore1***)⇒***vettore*

Restituisce un vettore unità riga o colonna, a seconda della forma di *Vettore1*.

Vettore1 deve essere una matrice composta da una sola riga o da una sola colonna.

$$\text{unitV}\left(\begin{bmatrix}1 & 2 & 1\end{bmatrix}\right)$$
$$\begin{bmatrix}0.408248 & 0.816497 & 0.408248\end{bmatrix}$$

$$\text{unitV}\left(\begin{bmatrix}1\\2\\3\end{bmatrix}\right)$$
$$\begin{bmatrix}0.267261\\0.534522\\0.801784\end{bmatrix}$$

unLock

Catalogo >

unLock *Var1* [, *Var2*] [, *Var3*] ...

unLock *Var*.

Sblocca le variabili o il gruppo di variabili specificate. Le variabili bloccate non possono essere modificate o eliminate.

Vedere **Lock**, pagina 91 e **getLockInfo()**, pagina 68.

a:=65	65
Lock a	*Done*
getLockInfo(a)	1
a:=75	"Error: Variable is locked."
DelVar a	"Error: Variable is locked."
Unlock a	*Done*
a:=75	75
DelVar a	*Done*

V

varPop() (Varianza della popolazione)

Catalogo >

varPop(*Lista*[, *listaFreq*]**)⇒***espressione*

Restituisce la varianza della popolazione di *Lista*.

Ogni elemento di *listaFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Lista*.

Nota: *Lista* deve contenere almeno due elementi.

Se un elemento di una lista è vuoto (nullo), l'elemento viene ignorato, come pure l'elemento corrispondente nell'altra lista. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

varPop({5,10,15,20,25,30})	72.9167

varSamp() (Varianza campione)

Catalogo >

varSamp(*Lista*[, *listaFreq*]**)⇒***espressione*

Restituisce la varianza campione di *Lista*.

Ogni elemento di *listaFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Lista*.

Nota: *Lista* deve contenere almeno due elementi.

varSamp({1,2,5,⁻6,3,⁻2})	$\frac{31}{2}$
varSamp({1,3,5},{4,6,2})	$\frac{68}{33}$

varSamp() (Varianza campione) Catalogo >

Se un elemento di una lista è vuoto (nullo), l'elemento viene ignorato, come pure l'elemento corrispondente nell'altra lista. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

varSamp(*Matrice1***[,** *matriceFreq***])⇒***matrice*

Restituisce un vettore riga contenente la varianza campione di ciascuna colonna di *Matrice1*.

Ogni elemento di *matriceFreq* conta il numero di occorrenze consecutive dell'elemento corrispondente di *Matrice1*.

Nota: *Matrice1* deve contenere almeno due righe.

Se un elemento di una matrice è vuoto (nullo), l'elemento viene ignorato, come pure l'elemento corrispondente nell'altra matrice. Per ulteriori informazioni sugli elementi vuoti, vedere a pagina 212.

$$\text{varSamp}\left(\begin{bmatrix} 1 & 2 & 5 \\ -3 & 0 & 1 \\ .5 & .7 & 3 \end{bmatrix}\right) \qquad \begin{bmatrix} 4.75 & 1.03 & 4 \end{bmatrix}$$

$$\text{varSamp}\left(\begin{bmatrix} -1.1 & 2.2 \\ 3.4 & 5.1 \\ -2.3 & 4.3 \end{bmatrix}, \begin{bmatrix} 6 & 3 \\ 2 & 4 \\ 5 & 1 \end{bmatrix}\right) \qquad \begin{bmatrix} 3.91731 & 2.08411 \end{bmatrix}$$

W

Wait Catalogo >

Wait *tempoInSecondi*

Sospende l'esecuzione per un periodo di *tempoInSecondi* secondi.

Wait è particolarmente utile in un programma che necessita di un breve ritardo per consentire che i dati richiesti diventino disponibili.

L'argomento *tempoInSecondi* deve essere un'espressione che viene semplificata a un valore decimale da 0 a 100. Il comando arrotonda questo valore al decimo di secondo più vicino.

Per annullare un **Wait** in corso,

- **Palmare:** Tenere premuto il tasto [on] e premere [enter] più volte.

Per attendere 4 secondi:

Wait 4

Per attendere 1/2 secondo:

Wait 0.5

Per attendere 1,3 secondi utilizzando la variabile *seccount*:

seccount:=1.3
Wait seccount

Questo esempio fa accendere un LED verde per 0,5 secondi e quindi lo spegne.

Send "SET GREEN 1 ON"
Wait 0.5
Send "SET GREEN 1 OFF"

Wait

Catalogo >

- **Windows®:** Tenere premuto il tasto **F12** e premere **Invio** più volte.
- **Macintosh®:** Tenere premuto il tasto **F5** e premere **Invio** più volte.
- **iPad®:** L'app mostra un prompt. È possibile continuare ad attendere o annullare.

Nota: È possibile utilizzare il comando **Wait** all'interno di un programma definito dall'utente, ma non di una funzione.

warnCodes ()

Catalogo >

warnCodes(*Espr1***,** *varStato***)⇒***espressione*

Calcola l'espressione *Espr1*, restituisce il risultato e memorizza i codici di tutte le avvertenze generate nella variabile di lista *varStato*. Se non sono generate avvertenze, questa funzione assegna a *varStato* una lista vuota.

Espr1 può essere qualsiasi espressione matematica valida di TI-Nspire™ o TI-Nspire™ CAS. Non è possibile utilizzare un comando o una assegnazione come *Espr1*.

varStato deve essere un nome di variabile valido.

Per un elenco dei codici di avvertenza e dei messaggi associati, vedere pagina 227.

warnCodes(det([1.23456E-999]),*warn*)

1.23456E-999

warn {10029}

when() (Quando)

Catalogo >

when(*Condizione***,** *risultatoVero* [**,** *risultatoFalso*][**,** *risultatoSconosciuto*]**)** **⇒***espressione*

Restituisce *risultatoVero*, *risultatoFalso* o *risultatoSconosciuto*, a seconda che *Condizione* sia vera, falsa o sconosciuta. Restituisce la voce inserita se gli argomenti immessi sono insufficienti per specificare un risultato appropriato.

warnCodes () Catalogo >

Omettere sia *risultatoFalso* che *risultatoSconosciuto* se si desidera che l'espressione venga definita solamente nella regione in cui *Condizione* è vera.

Utilizzare **undef** come *risultatoFalso* per definire un'espressione che esegua una rappresentazione grafica solo in un intervallo.

$\text{when}(x<0,x+3)\|x=5$	undef

when() è utile per la definizione di funzioni ricorsive.

$\text{when}(n>0,n\cdot factoral(n-1),1)\rightarrow factoral(n)$	*Done*
$factoral(3)$	6
$3!$	6

While Catalogo >

While *Condizione*
Blocco
EndWhile

Esegue le istruzioni di *Blocco* fino a quando *Condizione* è vera.

Blocco può essere una singola espressione o una serie di espressioni separate dal carattere ":".

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define sum_of_recip(n)=Func
                       Local i,tempsum
                       1→i
                       0→tempsum
                       While i≤n
                       tempsum+1/i→tempsum
                       i+1→i
                       EndWhile
                       Return tempsum
                       EndFunc
                                          Done
sum_of_recip(3)                           11/6
```

X

xor Catalogo >

*BooleanExpr1***xor***BooleanExpr2* restituisce *Boolean expression*

*BooleanList1***xor***BooleanList2* restituisce *Boolean list*

*BooleanMatrix1***xor***BooleanMatrix2* restituisce *Boolean matrix*

true xor true	false
5>3 xor 3>5	true

xor Catalogo >

Restituisce true se *Espressione booleana1* è vera e *Espressione booleana2* è falsa, o viceversa.

Restituisce false se gli argomenti sono entrambi veri o falsi. Restituisce un'espressione booleana semplificata se nessuno degli argomenti può essere risolto in vero o falso.

Nota: vedere **or**, pagina 115.

Intero1 **xor** *Intero2* ⇒ *intero*

Confronta due interi reali bit per bit tramite un'operazione **xor**. Internamente, entrambi gli interi vengono convertiti in numeri binari a 64 bit con segno. Quando vengono confrontati bit corrispondenti, il risultato sarà 1 se uno dei due bit (ma non entrambi) è uguale a 1; se entrambi i bit sono uguali a 0 o a 1, il risultato sarà 0. Il valore restituito rappresenta il risultato dei bit e viene visualizzato nel modo Base che è stato impostato.

È possibile inserire gli interi in qualsiasi base numerica. Se si tratta di un numero binario o esadecimale, utilizzare rispettivamente il prefisso 0b o 0h. Senza prefisso, gli interi vengono considerati decimali (base 10).

Se viene indicato un intero decimale troppo grande per una forma binaria con segno a 64 bit, verrà utilizzata un'operazione a modulo simmetrico per portare il valore nell'intervallo appropriato. Per ulteriori informazioni, vedere ▶**Base2**, pagina 21.

Nota: vedere **or**, pagina 115.

In modalità base Esadecimale:

Importante: è zero, non la lettera O.

0h7AC36 xor 0h3D5F	0h79169

In modalità base Bin:

0b100101 xor 0b100	0b100001

Nota: un numero binario può contenere fino a 64 cifre (oltre al prefisso 0b). Un numero esadecimale può contenere fino ad 16 cifre.

Z

zInterval (Intervallo di confidenza Z) Catalogo >

zInterval σ,*Lista*[,*Freq*[,*livelloConfidenza*]]

(Input lista dati)

zInterval σ,x̄,*n* [,*livelloConfidenza*]

zInterval (Intervallo di confidenza Z) Catalogo >

(Input statistiche riepilogo)

Calcola un intervallo di confidenza *z*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156)

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.CLower, stat.CUpper	Intervallo di confidenza per una media non nota di una popolazione
stat.$\bar{x}$	Media del campione della successione di dati dalla distribuzione casuale normale
stat.ME	Margine di errore
stat.sx	Deviazione standard del campione
stat.n	Lunghezza della successione di dati con media del campione
stat.σ	Deviazione standard nota della popolazione per la sequenza di dati *Lista*

zInterval_1Prop (Intervallo di confidenza z per una proporzione) Catalogo >

zInterval_1Prop *x*,*n* [,*livelloConfidenza*]

Calcola un intervallo di confidenza *z* per una proporzione. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

x è un intero non negativo.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.CLower, stat.CUpper	Intervallo di confidenza contenente la probabilità di distribuzione del livello di confidenza
stat.$\hat{p}$	Proporzione calcolata di casi favorevoli
stat.ME	Margine di errore
stat.n	Numero di campioni nelle successioni di dati

zInterval_2Prop (Intervallo di confidenza z per due proporzioni)

Catalogo >

zInterval_2Prop *x1*,*n1*,*x2*,*n2* [,*livelloConfidenza*]

Calcola l'intervallo di confidenza *z* per due proporzioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

x1 e *x2* sono interi non negativi.

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.CLower, stat.CUpper	Intervallo di confidenza contenente la probabilità di distribuzione del livello di confidenza
stat.$\hat{p}$Diff	Differenza calcolata tra le proporzioni
stat.ME	Margine di errore
stat.$\hat{p}$1	Stima della proporzione del primo campione
stat.$\hat{p}$2	Stima della proporzione del secondo campione
stat.n1	Dimensione del campione nella successione di dati uno
stat.n2	Dimensione del campione nella successione di dati due

zInterval_2Samp (Intervallo di confidenza z su due campioni)

Catalogo >

zInterval_2Samp σ_1,σ_2,*List1*,*Lista2*[,*Freq1* [,*Freq2*, [*livelloConfidenza*]]]

(Input lista dati)

zInterval_2Samp σ_1,σ_2,$\bar{x}$*1*,*n1*,$\bar{x}$*2*,*n2* [,*livelloConfidenza*]

(Input statistiche riepilogo)

Calcola l'intervallo di confidenza *z* su due campioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

zInterval_2Samp (Intervallo di confidenza z su due campioni) Catalogo >

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.CLower, stat.CUpper	Intervallo di confidenza contenente la probabilità di distribuzione del livello di confidenza
stat.$\bar{x}$1-$\bar{x}$2	Medie dei campioni delle successioni di dati dalla distribuzione casuale normale
stat.ME	Margine di errore
stat.$\bar{x}$1, stat.$\bar{x}$2	Medie dei campioni delle successioni di dati dalla distribuzione casuale normale
stat.σx1, stat.σx2	Deviazioni standard dei campioni di *Lista 1* e *Lista 2*
stat.n1, stat.n2	Numero di campioni nelle successioni di dati
stat.r1, stat.r2	Deviazioni standard note della popolazione per la successione di dati *Lista1* e *Lista2*

zTest (Verifica z) Catalogo >

zTest μ*0*,σ,*Lista*,[*Freq*[,*Ipotesi*]]

(Input lista dati)

zTest μ*0*,σ,$\bar{x}$,*n*[,*Ipotesi*]

(Input statistiche riepilogo)

Esegue una verifica *z* con frequenza *listaFreq*. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Viene verificata l’ipotesi nulla H_0: μ = μ0 in contrapposizione a una delle alternative seguenti:

Per H_a: μ < μ0, impostare *Ipotesi*<0

Per H_a: μ ≠ μ0 (default), impostare *Ipotesi*=0

Per H_a: μ > μ0, impostare *Ipotesi*>0

zTest (Verifica z) Catalogo >

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.z	$(\bar{x} - \mu 0) / (\sigma / \text{sqrt}(n))$
stat.P Valore	Probabilità minima in cui l’ipotesi nulla può essere rifiutata
stat.$\bar{x}$	Media del campione della sequenza di dati in *Lista*
stat.sx	Deviazione standard del campione della sequenza di dati. Restituita solo per input di *Dati*.
stat.n	Dimensione dei campioni

zTest_1Prop (Verifica z per una proporzione) Catalogo >

zTest_1Prop *p0*,*x*,*n*[,*Ipotesi*]

Esegue una verifica *z* per una proporzione. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

x è un intero non negativo.

Viene verificata l’ipotesi nulla H_0: $p = p0$ in contrapposizione a una delle alternative seguenti:

Per H_a: $p > p0$, impostare *Ipotesi*>0

Per H_a: $p \neq p0$ *(default)*, impostare *Ipotesi*=0

Per H_a: $p < p0$, impostare *Ipotesi*<0

Per informazioni sull'effetto di elementi vuoti in una lista, vedere “Elementi vuoti (nulli)”, pagina 212.

Variabile di output	Descrizione
stat.p0	Proporzione della popolazione ipotizzata
stat.z	Valore normale standard calcolato per la proporzione

Variabile di output	Descrizione
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.$\hat{p}$	Proporzione attesa del campione
stat.n	Dimensione dei campioni

zTest_2Prop (Verifica z per due proporzioni)

Catalogo >

zTest_2Prop *x1*,*n1*,*x2*,*n2*[,*Ipotesi*]

Esegue una verifica *z* per due proporzioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

x1 e *x2* sono interi non negativi.

Viene verificata l'ipotesi nulla H_0: $p1 = p2$ in contrapposizione a una delle alternative seguenti:

Per H_a: $p1 > p2$, impostare *Ipotesi*>0

Per H_a: $p1 \neq p2$ *(default)*, impostare *Ipotesi*=0

Per H_a: $p < p0$, impostare *Ipotesi*<0

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.z	Valore normale standard calcolato per la differenza delle proporzioni
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.$\hat{p}$1	Stima della proporzione del primo campione
stat.$\hat{p}$2	Stima della proporzione del secondo campione
stat.$\hat{p}$	Stima delle proporzioni del campione aggregate
stat.n1, stat.n2	Numero di campioni presi nelle prove 1 e 2

zTest_2Samp (Verifica su due campioni)

Catalogo >

zTest_2Samp σ_1,σ_2,*Lista1*,*Lista2*[,*Freq1*[,*Freq2*[,*Ipotesi*]]]

(Input lista dati)

zTest_2Samp σ_1,σ_2,$\bar{x}1$,*n1*,$\bar{x}2$,*n2*[,*Ipotesi*]

(Input statistiche riepilogo)

Esegue una verifica *z* su due campioni. Il riepilogo dei risultati è memorizzato nella variabile *stat.results*. (pagina 156).

Viene verificata l'ipotesi nulla H_0: $\mu1 = \mu2$ in contrapposizione a una delle alternative seguenti:

Per H_a: $\mu1 < \mu2$, impostare *Ipotesi*<0

Per H_a: $\mu1 \neq \mu2$ (default), impostare *Ipotesi*=0

Per H_a: $\mu1 > \mu2$, *Ipotesi*>0

Per informazioni sull'effetto di elementi vuoti in una lista, vedere "Elementi vuoti (nulli)", pagina 212.

Variabile di output	Descrizione
stat.z	Valore normale standard calcolato per la differenze delle medie
stat.PVal	Livello minimo di significatività in corrispondenza del quale l'ipotesi nulla può essere rifiutata
stat.$\bar{x}$1, stat.$\bar{x}$2	Medie dei campioni delle sequenze di dati di *Lista1* e *Lista2*
stat.sx1, stat.sx2	Deviazioni standard dei campioni delle sequenze di dati di *Lista1* e *Lista2*
stat.n1, stat.n2	Dimensione dei campioni

Simboli

+ (addizione) — Tasto +

Valore1 + *Valore2*⇒*valore*

Restituisce la somma dei due argomenti.

56	56
56+4	60
60+4	64
64+4	68
68+4	72

Lista1 + *Lista2*⇒*lista*

Matrice1 + *Matrice2*⇒*matrice*

Restituisce una lista (o matrice) contenente le somme degli elementi corrispondenti di *Lista1* e *Lista2* (oppure di *Matrice1* e *Matrice2*).

Gli argomenti devono avere uguali dimensioni.

$\left\{22,\pi,\frac{\pi}{2}\right\}\rightarrow l1$	$\{22,3.14159,1.5708\}$
$\left\{10,5,\frac{\pi}{2}\right\}\rightarrow l2$	$\{10,5,1.5708\}$
$l1+l2$	$\{32,8.14159,3.14159\}$

Valore + *Lista1*⇒*lista*

Lista1 + *Valore*⇒*lista*

Restituisce una lista contenente le somme di *Valore* e di ciascun elemento di *Lista1*.

$15+\{10,15,20\}$	$\{25,30,35\}$
$\{10,15,20\}+15$	$\{25,30,35\}$

Valore + *Matrice1*⇒*matrice*

Matrice1 + *Valore*⇒*matrice*

Restituisce una matrice nella quale *Valore* viene sommato a ciascun elemento sulla diagonale di *Matrice1*. *Matrice1* deve essere quadrata.

Nota: utilizzare **.+** (punto più) per aggiungere un'espressione a ciascun elemento.

$20+\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$	$\begin{bmatrix}21 & 2\\3 & 24\end{bmatrix}$

−(sottrazione) — Tasto −

Valore1 − *Valore2*⇒*valore*

Restituisce *Valore1* meno *Valore2*.

6−2	4
$\pi-\frac{\pi}{6}$	2.61799

−(sottrazione) Tasto −

Lista1 − *Lista2*⇒*lista*

Matrice1 − *Matrice2*⇒*matrice*

Sottrae ciascun elemento di *Lista2* (o di *Matrice2*) dal corrispondente elemento di *Lista1* (o di *Matrice1*) e restituisce il risultato.

Gli argomenti devono avere uguali dimensioni.

$\left\{22,\pi,\frac{\pi}{2}\right\}-\left\{10,5,\frac{\pi}{2}\right\}$ $\{12,-1.85841,0.\}$

$[3\ \ 4]-[1\ \ 2]$ $[2\ \ 2]$

Valore − *Lista1*⇒*lista*

Lista1 − *Valore*⇒*lista*

Sottrae ciascun elemento di *Lista1* da *Valore* oppure sottrae *Valore* da ciascun elemento di *Lista1*, quindi restituisce una lista contenente i risultati.

$15-\{10,15,20\}$ $\{5,0,-5\}$

$\{10,15,20\}-15$ $\{-5,0,5\}$

Valore − *Matrice1*⇒*matrice*

Matrice1 − *Valore*⇒*matrice*

Valore − *Matrice1* restituisce una matrice di numero volte *Valore* la matrice di identità, meno *Matrice1*. *Matrice1* deve essere quadrata.

Matrice1 − *Valore* restituisce una matrice di *Valore* volte la matrice d'identità, sottratta da *Matrice1*. *Matrice1* deve essere quadrata.

Nota: Utilizzare **.**− (punto meno) per sottrarre un'espressione da ciascun elemento.

$20-\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$ $\begin{bmatrix}19 & -2\\-3 & 16\end{bmatrix}$

·(moltiplicazione) Tasto ×

Valore1 · *Valore2*⇒*valore*

Restituisce il prodotto di due argomenti.

$2\cdot 3.45$ 6.9

Lista1 · *Lista2*⇒*lista*

Restituisce una lista contenente i prodotti degli elementi corrispondenti di *Lista1* e *Lista2*.

Le liste devono avere uguali dimensioni.

$\{1.,2,3\}\cdot\{4,5,6\}$ $\{4,10,18\}$

·(moltiplicazione) Tasto ×

Matrice1 · *Matrice2*⇒*matrice*

Restituisce il prodotto di matrice di *Matrice1* per *Matrice2*.

Il numero di colonne di *Matrice1* deve essere uguale al numero di righe di *Matrice2*.

$$\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}.\begin{bmatrix}7&8\\7&8\\7&8\end{bmatrix} \qquad \begin{bmatrix}42&48\\105&120\end{bmatrix}$$

Valore · *Lista1*⇒*lista*

Lista1 · *Valore*⇒*lista*

Restituisce una lista contenente i prodotti di *Valore* per ciascun elemento di *Lista1*.

$$\pi\cdot\{4,5,6\} \qquad \{12.5664,15.708,18.8496\}$$

Valore · *Matrice1*⇒*matrice*

Matrice1 · *Valore*⇒*matrice*

Restituisce una matrice contenente i prodotti di *Valore* per ciascun elemento di *Matrice1*.

Nota: Utilizzare **.** ·(punto moltiplicazione) per moltiplicare un'espressione per ciascun elemento.

$$\begin{bmatrix}1&2\\3&4\end{bmatrix}\cdot 0.01 \qquad \begin{bmatrix}0.01&0.02\\0.03&0.04\end{bmatrix}$$

$$6\cdot\text{identity}(3) \qquad \begin{bmatrix}6&0&0\\0&6&0\\0&0&6\end{bmatrix}$$

/ (divisione) Tasto ÷

Valore1 / *Valore2*⇒*valore*

Restituisce il quoziente di *Valore1* diviso per *Valore2*.

Nota: vedere anche **Modello di frazione**, pagina 5.

$$\frac{2}{3.45} \qquad .57971$$

Lista1 / *Lista2*⇒*lista*

Restituisce una lista contenente i quozienti di *Lista1* diviso per *Lista2*.

Le liste devono avere uguali dimensioni.

$$\frac{\{1.,2,3\}}{\{4,5,6\}} \qquad \left\{0.25,\frac{2}{5},\frac{1}{2}\right\}$$

Valore / *Lista1* ⇒ *lista*

Lista1 / *Valore* ⇒ *lista*

Restituisce una lista contenente i quozienti di *Valore* diviso per *Lista1* o quelli di *Lista1* diviso per *Valore*.

$$\frac{6}{\{3,6,\sqrt{6}\}} \qquad \{2,1,2.44949\}$$

$$\frac{\{7,9,2\}}{7\cdot 9\cdot 2} \qquad \left\{\frac{1}{18},\frac{1}{14},\frac{1}{63}\right\}$$

/ (divisione) — Tasto ÷

Valore / Matrice1⇒ matrice

Matrice1 / Valore ⇒ matrice

Restituisce una matrice contenente i quozienti di Matrice1÷*Valore*.

Nota: utilizzare **. /** (punto divisione) per dividere un'espressione per ciascun elemento.

$$\frac{[7\ \ 9\ \ 2]}{7\cdot 9\cdot 2} \qquad \begin{bmatrix} \frac{1}{18} & \frac{1}{14} & \frac{1}{63} \end{bmatrix}$$

^ (elevamento a potenza) — Tasto ^

Valore1 **^** *Valore2 ⇒ valore*

Lista1 **^** *Lista2 ⇒ lista*

Restituisce il primo argomento elevato alla potenza del secondo argomento.

Nota: vedere anche **Modello di esponente**, pagina 5.

In una lista, restituisce gli elementi di *Lista1* elevati alla potenza dei corrispondenti elementi di *Lista2*.

Nel dominio reale, le potenze frazionarie con esponenti ridotti e denominatori dispari utilizzano l'ambito reale rispetto a quello principale, impiegata nel modo complesso.

$$4^2 \qquad 16$$

$$\{2,4,6\}^{\{1,2,3\}} \qquad \{2,16,216\}$$

Valore ^ *Lista1 ⇒ lista*

Restituisce *Valore* elevato alla potenza degli elementi di *Lista1*.

$$\pi^{\{1,2,-3\}} \qquad \{3.14159,9.8696,0.032252\}$$

Lista1 ^ *Valore ⇒ lista*

Restituisce gli elementi di *Lista1* elevati alla potenza di *Valore*.

$$\{1,2,3,4\}^{-2} \qquad \left\{1,\frac{1}{4},\frac{1}{9},\frac{1}{16}\right\}$$

^ (elevamento a potenza)

Tasto [^]

matriceQuadrata1 **^** *intero* ⇒ *matrice*

Restituisce *matriceQuadrata1* elevata alla potenza di *intero*.

matriceQuadrata1 deve essere una matrice quadrata.

Se *intero* = ⁻1, viene calcolata l'inversa della matrice.

Se *intero* < ⁻1, viene calcolata l'inversa della matrice a una potenza positiva appropriata.

$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}^2$	$\begin{bmatrix}7 & 10\\15 & 22\end{bmatrix}$
$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}^{-1}$	$\begin{bmatrix}-2 & 1\\\frac{3}{2} & \frac{-1}{2}\end{bmatrix}$
$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}^{-2}$	$\begin{bmatrix}\frac{11}{2} & \frac{-5}{2}\\\frac{-15}{4} & \frac{7}{4}\end{bmatrix}$

x² (quadrato)

Tasto [x²]

Valore1 ^{2} ⇒ *valore*

Restituisce il quadrato dell'argomento.

Lista1 **²** ⇒ *lista*

Restituisce una lista contenente i quadrati degli elementi di *Lista1*.

matriceQuadrata1 ² ⇒ *matrice*

Restituisce la matrice *matriceQuadrata1* elevata al quadrato. Ciò non equivale a calcolare il quadrato di ogni elemento. Utilizzare .^2 per calcolare il quadrato di ogni elemento.

4^2	16
$\{2,4,6\}^2$	$\{4,16,36\}$
$\begin{bmatrix}2 & 4 & 6\\3 & 5 & 7\\4 & 6 & 8\end{bmatrix}^2$	$\begin{bmatrix}40 & 64 & 88\\49 & 79 & 109\\58 & 94 & 130\end{bmatrix}$
$\begin{bmatrix}2 & 4 & 6\\3 & 5 & 7\\4 & 6 & 8\end{bmatrix}.^\wedge 2$	$\begin{bmatrix}4 & 16 & 36\\9 & 25 & 49\\16 & 36 & 64\end{bmatrix}$

.+ (punto addizione)

Tasti [.][+]

Matrice1 **.+** *Matrice2* ⇒ *matrice*

Valore .+ *Matrice1* ⇒ *matrice*

Matrice1 **.+** *Matrice2* restituisce una matrice che è la somma di ciascuna coppia di elementi corrispondenti di *Matrice1* e *Matrice2*.

Valore **.+** *Matrice1* restituisce una matrice che è la somma di *Valore* e di ciascun elemento di *Matrice1*.

$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}.+\begin{bmatrix}10 & 30\\20 & 40\end{bmatrix}$	$\begin{bmatrix}11 & 32\\23 & 44\end{bmatrix}$
$5.+\begin{bmatrix}10 & 30\\20 & 40\end{bmatrix}$	$\begin{bmatrix}15 & 35\\25 & 45\end{bmatrix}$

.− (punto sottrazione). Tasti . −

Matrice1 **.−** *Matrice2* ⇒ *matrice*

Valore .−*Matrice1* ⇒ *matrice*

Matrice1 **.−***Matrice2* restituisce una matrice che è la differenza tra ciascuna coppia di elementi corrispondenti di *Matrice1* e *Matrice2*.

Valore **.−***Matrice1* restituisce una matrice che è la differenza di *Valore* e di ciascun elemento di *Matrice1*.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} .- \begin{bmatrix}10 & 20\\30 & 40\end{bmatrix} \qquad \begin{bmatrix}-9 & -18\\-27 & -36\end{bmatrix}$$

$$5 .- \begin{bmatrix}10 & 20\\30 & 40\end{bmatrix} \qquad \begin{bmatrix}-5 & -15\\-25 & -35\end{bmatrix}$$

.· (punto moltiplicazione) Tasti . ×

Matrice1 **.·** *Matrice2* ⇒ *matrice*

Valore **.·** *Matrice1* ⇒ *matrice*

Matrice1 **.·** *Matrice2* restituisce una matrice che è il prodotto di ciascuna coppia di elementi corrispondenti di *Matrice1* e *Matrice2*.

Valore **.·** *Matrice1* restituisce una matrice contenente i prodotti di *Valore* per ciascun elemento di *Matrice1*.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} .\cdot \begin{bmatrix}10 & 20\\30 & 40\end{bmatrix} \qquad \begin{bmatrix}10 & 40\\90 & 160\end{bmatrix}$$

$$5 .\cdot \begin{bmatrix}10 & 20\\30 & 40\end{bmatrix} \qquad \begin{bmatrix}50 & 100\\150 & 200\end{bmatrix}$$

. / (punto divisione) Tasti . ÷

Matrice1 **. /** *Matrice2* ⇒ *matrice*

Valore . / *Matrice1* ⇒ *matrice*

Matrice1 **./** *Matrice2* restituisce una matrice che è il quoziente di ciascuna coppia di elementi corrispondenti di *Matrice1* e di *Matrice2*.

Valore **./** *Matrice1* restituisce una matrice che è il quoziente di *Valore* diviso per ciascun elemento di *Matrice1*.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} ./ \left(\begin{bmatrix}10 & 20\\30 & 40\end{bmatrix}\right) \qquad \begin{bmatrix}\frac{1}{10} & \frac{1}{10}\\\frac{1}{10} & \frac{1}{10}\end{bmatrix}$$

$$5 ./ \left(\begin{bmatrix}10 & 20\\30 & 40\end{bmatrix}\right) \qquad \begin{bmatrix}\frac{1}{2} & \frac{1}{4}\\\frac{1}{6} & \frac{1}{8}\end{bmatrix}$$

.^ (punto elevato a potenza) — Tasto [.] [^]

Matrice1 **.^** *Matrice2* ⇒ *matrice*

Valore **. ^** *Matrice1* ⇒ *matrice*

Matrice1 **.^** *Matrice2* restituisce una matrice nella quale ciascun elemento di *Matrice2* è l'esponente del corrispondente elemento di *Matrice1*.

Valore **.^** *Matrice1* restituisce una matrice nella quale ciascun elemento di *Matrice1* è l'esponente di *Valore*.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}.^{\wedge}\begin{bmatrix}0 & 2\\3 & -1\end{bmatrix} \qquad \begin{bmatrix}1 & 4\\27 & \frac{1}{4}\end{bmatrix}$$

$$5.^{\wedge}\begin{bmatrix}0 & 2\\3 & -1\end{bmatrix} \qquad \begin{bmatrix}1 & 25\\125 & \frac{1}{5}\end{bmatrix}$$

⁻(negazione) — Tasto [(−)]

⁻*Valore1* ⇒ *valore*

⁻*Lista1* ⇒ *lista*

⁻*Matrice1* ⇒ *matrice*

Restituisce l'argomento con il segno cambiato.

In una lista o in una matrice, restituisce tutti gli elementi con il segno cambiato.

Se l'argomento è un intero binario o esadecimale, la negazione dà il complemento a due.

```
⁻2.43                                  ⁻2.43
⁻{⁻1,0.4,1.2E19}        {1.,⁻0.4,⁻1.2E19}
```

`In modalità base Bin:`

Importante: è zero, non la lettera O.

```
⁻0b100101
 0b11111111111111111111111111111111▸
```

Per vedere l'intero risultato, premere ▲, quindi utilizzare ◀ e ▶ per spostare il cursore.

% (percentuale) — Tasti [ctrl] [📖]

Valore1 **%** ⇒ *valore*

Lista1 **%** ⇒ *lista*

Matrice1 **%** ⇒ *matrice*

Restituisce $\frac{argument}{100}$

In una lista o in una matrice, restituisce rispettivamente una lista o una matrice, nelle quali ciascun elemento è diviso per 100.

Nota: Per forzare un risultato approssimativo,

Palmare: Premere [ctrl] [enter].
Windows®: Premere **Ctrl+Invio**.
Macintosh®: Premere ⌘+**Invio**.
iPad®: Tenere premuto **Invio** e selezionare [≈].

```
13%                                     0.13
```

% (percentuale)

Tasti ctrl 📖

```
({1,10,100})%                    {0.01,0.1,1.}
```

= (uguale)

Tasto =

Espr1 = *Espr2*⇒*espressione booleana*

Lista1 = *Lista2*⇒ *lista booleana*

Matrice1 = *Matrice2*⇒ *matrice booleana*

Restituisce true se *Espr1* è determinata in modo da essere uguale a *Espr2*.

Restituisce false se *Espr1* è determinata in modo da non essere uguale ad *Espr2*.

Negli altri casi, restituisce una forma semplificata dell'equazione.

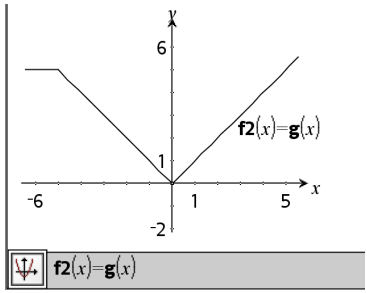
Nelle liste e nelle matrici, restituisce un confronto elemento per elemento.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

Esempio di funzione che usa gli operatori relazionali: =, ≠, <, ≤, >, ≥

```
Define g(x)=Func
        If x≤-5 Then
        Return 5
        ElseIf x>-5 and x<0 Then
        Return -x
        ElseIf x≥0 and x≠10 Then
        Return x
        ElseIf x=10 Then
        Return 3
        EndIf
        EndFunc
                                    Done
```

Risultato della rappresentazione grafica g(x)



≠ (diverso)

Tasti ctrl =

Espr1 ≠ *Espr2* ⇒ *espressione booleana*

Lista1 ≠ *Lista2* ⇒ *lista booleana*

Matrice1 ≠ *Matrice2* ⇒ *Matrice booleana*

Restituisce true se *Espr1* è determinata in modo da non essere uguale a *Espr2*.

Vedere l'esempio di "=" (uguale).

≠ (diverso)

Tasti ctrl =

Restituisce false se *Espr1* è determinata in modo da essere uguale a *Espr2*.

Negli altri casi, restituisce una forma semplificata dell'equazione.

Nelle liste e nelle matrici, restituisce un confronto elemento per elemento.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **/=**

< (minore di)

Tasti ctrl =

Espr1 **<** *Espr2* ⇒ *espressione booleana*

Lista1 **<** *Lista2* ⇒ *lista booleana*

Matrice1 **<** *Matrice2* ⇒ *matrice booleana*

Restituisce true se *Espr1* è determinata in modo da essere minore di *Espr2*.

Restituisce false se *Espr1* è determinata in modo da non essere maggiore o uguale a *Espr2*.

Negli altri casi, restituisce una forma semplificata dell'equazione.

Nelle liste e nelle matrici, restituisce un confronto elemento per elemento.

Vedere l'esempio di "=" (uguale).

≤ (minore di o uguale a)

Tasti ctrl =

Espr1 ≤ *Espr2* ⇒ *espressione booleana*

Lista1 ≤ *Lista2* ⇒ *lista booleana*

Matrice1 ≤ *Matrice2* ⇒ *matrice booleana*

Restituisce true se *Espr1* è determinata in modo da essere minore o uguale a *Espr2*.

Restituisce false se *Espr1* è determinata in modo da non essere maggiore di *Espr2*.

Negli altri casi, restituisce una forma semplificata dell'equazione.

Vedere l'esempio di "=" (uguale).

≤ (minore di o uguale a)

Tasti ctrl =

Nelle liste e nelle matrici, restituisce un confronto elemento per elemento.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **<=**

> (maggiore di)

Tasti ctrl =

Espr1 **>** *Espr2* ⇒ *espressione booleana*

Lista1 **>** *Lista2* ⇒ *lista booleana*

Matrice1 **>** *Matrice2* ⇒ *matrice booleana*

Restituisce true se *Espr1* è determinata in modo da essere maggiore di *Espr2*.

Restituisce false se *Espr1* è determinata in modo da non essere minore o uguale a *Espr2*.

Negli altri casi, restituisce una forma semplificata dell'equazione.

Nelle liste e nelle matrici, restituisce un confronto elemento per elemento.

Vedere l'esempio di "=" (uguale).

≥ (maggiore di o uguale a)

Tasti ctrl =

Espr1 ≥ *Espr2* ⇒ *espressione booleana*

Lista1 ≥ *Lista2* ⇒ *lista booleana*

Matrice1 ≥ *Matrice2* ⇒ *matrice booleana*

Restituisce true se *Espr1* è determinata in modo da essere maggiore o uguale a *Espr2*.

Restituisce false se *Espr1* è determinata in modo da non essere minore o uguale a *Espr2*.

Negli altri casi, restituisce una forma semplificata dell'equazione.

Nelle liste e nelle matrici, restituisce un confronto elemento per elemento.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **>=**

Vedere l'esempio di "=" (uguale).

⇒ (implicazione logica)

ctrl = tasti

BooleanExpr1 ⇒ *BooleanExpr2* restituisce *Boolean expression*

BooleanList1 ⇒ *BooleanList2* restituisce *Boolean list*

BooleanMatrix1 ⇒ *BooleanMatrix2* restituisce *Boolean matrix*

Integer1 ⇒ *Integer2* restituisce *Integer*

5>3 or 3>5	true
5>3 ⇒ 3>5	false
3 or 4	7
3 ⇒ 4	-4
{1,2,3} or {3,2,1}	{3,2,3}
{1,2,3} ⇒ {3,2,1}	{-1,-1,-3}

Valuta l'espressione **not** <argumento1> **or** <argumento2> e restituisce true, false oppure una forma semplificata dell'equazione.

Nel caso di liste e matrici, restituisce un confronto elemento per elemento.

Nota: È possibile inserire questo operatore dalla tastiera digitando **=>**

⇔ (doppia implicazione logica, XNOR)

ctrl = tasti

BooleanExpr1 ⇔ *BooleanExpr2* restituisce *Boolean expression*

BooleanList1 ⇔ *BooleanList2* restituisce *Boolean list*

BooleanMatrix1 ⇔ *BooleanMatrix2* restituisce *Boolean matrix*

Integer1 ⇔ *Integer2* restituisce *Integer*

5>3 xor 3>5	true
5>3 ⇔ 3>5	false
3 xor 4	7
3 ⇔ 4	-8
{1,2,3} xor {3,2,1}	{2,0,2}
{1,2,3} ⇔ {3,2,1}	{-3,-1,-3}

Restituisce la negazione di un'operazione **XOR** logica riguardo i due argomenti. Restituisce true, false oppure una forma semplificata dell'equazione.

Nel caso di liste e matrici, restituisce un confronto elemento per elemento.

Nota: È possibile inserire questo operatore dalla tastiera digitando **<=>**

! (fattoriale)

Tasto [?!▸]

Valore1! ⇒ *valore*

*Lista1***!** ⇒ *lista*

*Matrice1***!** ⇒ *matrice*

Restituisce il fattoriale dell'argomento.

In una lista o in una matrice, restituisce rispettivamente una lista o una matrice dei fattoriali di ciascun elemento.

5!	120
$\{\{5,4,3\}\}!$	$\{120,24,6\}$
$\left(\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\right)!$	$\begin{bmatrix}1 & 2\\6 & 24\end{bmatrix}$

& (aggiunge)

Tasti [ctrl] [📖]

Stringa1 **&** *Stringa2* ⇒ *stringa*

Restituisce una stringa di testo, aggiungendo *Stringa2* a *Stringa1*.

"Hello "&"Nick"	"Hello Nick"

d() (derivata)

Catalogo > [📖]

d(*Espr1***,** *Var***[,** *Ordine***])** | *Var=Valore*⇒*valore*

d(*Espr1***,** *Var***[,** *Ordine***])**⇒*valore*

d(*Lista1***,** *Var***[,** *Ordine***])**⇒*lista*

d(*Matrice1***,** *Var***[,** *Ordine***])**⇒*matrice*

Tranne quando si utilizza la prima sintassi, è necessario memorizzare un valore numerico nella variabile *Var* prima di calcolare d**()**. Vedere gli esempi.

d**()** può essere utilizzato per calcolare numericamente la derivata di primo e di secondo ordine in un punto con metodi di differenziazione automatica.

Ordine, se incluso, deve essere = **1** o **2**. L'impostazione predefinita è **1**.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`derivative(...)`**.

Nota: Vedere anche **Derivata prima**, **pagina 9** o **Derivata seconda**, pagina 10.

$\frac{d}{dx}(\lvert x\rvert)\lvert x=0$	undef
$x:=0:\frac{d}{dx}(\lvert x\rvert)$	undef
$x:=3:\frac{d}{dx}(\{x^2,x^3,x^4\})$	$\{6,27,108\}$

d() (derivata) Catalogo >

Nota: l'algoritmo **d()** ha una limitazione: funziona in modo ricorsivo tramite l'espressione non semplificata, calcolando il valore numerico della derivata prima (e seconda, se applicabile) e di ciascuna sottoespressione, il che può produrre un risultato imprevisto.

Si consideri l'esempio sulla destra. La derivata prima di x·(x^2+x)^(1/3) con x=0 è uguale a 0. Tuttavia, poiché la derivata prima della sottoespressione (x^2+x)^(1/3) è indefinita con x=0, e questo valore è utilizzato per calcolare la derivata dell'espressione totale, **d()** riporta il risultato come indefinito e visualizza un messaggio di avvertenza.

Se si incontra questa limitazione, verificare la soluzione graficamente. È anche possibile provare a utilizzare **centralDiff()**.

$$\frac{d}{dx}\left(x\cdot\left(x^2+x\right)^{\frac{1}{3}}\right)\Big|_{x=0} \qquad \text{undef}$$

$$\text{centralDiff}\left(x\cdot\left(x^2+x\right)^{\frac{1}{3}},x\right)\Big|_{x=0} \qquad 0.000033$$

∫() (integrale) Catalogo >

∫(*Espr1*, *Var*, *Inferiore*, *Superiore*) ⇒ *valore*

Restituisce l'integrale di *Espr1* rispetto alla variabile *Var* da *Inferiore* a *Superiore*. Può essere utilizzato per calcolare numericamente l'integrale definito, con lo stesso metodo di nInt().

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando `integral(...)`.

Nota: vedere anche **nInt()**, pagina 109 e **Modello di integrale definito**, pagina 10.

$$\int_0^1 x^2\,dx \qquad 0.333333$$

√() (radice quadrata) Tasti ctrl x²

√ (*Valore1*)⇒*valore*

√ (*List1*)⇒*lista*

$$\sqrt{4} \qquad 2$$

$$\sqrt{\{9,2,4\}} \qquad \{3,1.41421,2\}$$

√() (radice quadrata) **Tasti** ctrl x^2

Restituisce la radice quadrata dell'argomento.

In una lista, restituisce le radici quadrate di tutti gli elementi di *Lista1*.

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`sqrt(…)`**

Nota: vedere anche **Modello di radice quadrata**, pagina 5.

Π() (prodSeq) **Catalogo >**

Π(*Espr1*, *Var*, *Basso*, *Alto*)⇒*espressione*

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **`prodSeq(…)`**.

Calcola *Espr1* per ciascun valore di *Var* da *Basso* ad *Alto* e restituisce il prodotto dei risultati.

Nota: vedere anche **Modello di prodotto** (Π), pagina 9.

Π(*Espr1*, *Var*, *Basso*, *Basso*−1)⇒1

Π(*Espr1*, *Var*, *Basso*, *Alto*) ⇒**1/**Π(*Espr1*, *Var*, *Alto*+*1*, *Basso*−*1*) if *Alto* < *Basso*−*1*

Le formule di prodotto utilizzate sono derivate dal seguente riferimento:

Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Reading, Massachusetts: Addison-Wesley, 1994.

$$\prod_{n=1}^{5}\left(\frac{1}{n}\right) \qquad \frac{1}{120}$$

$$\prod_{n=1}^{5}\left(\left\{\frac{1}{n},n,2\right\}\right) \qquad \left\{\frac{1}{120},120,32\right\}$$

$$\prod_{k=4}^{3}(k) \qquad 1$$

$$\prod_{k=4}^{1}\left(\frac{1}{k}\right) \qquad 6$$

$$\prod_{k=4}^{1}\left(\frac{1}{k}\right)\cdot\prod_{k=2}^{4}\left(\frac{1}{k}\right) \qquad \frac{1}{4}$$

Σ() (sumSeq) Catalogo >

Σ(*Espr1***,** *Var***,** *Basso***,** *Alto***)⇒***espressione*

Nota: è possibile inserire questa funzione dalla tastiera del computer digitando **sumSeq(...)**.

Calcola *Espr1* per ciascun valore di *Var* da *Basso* ad *Alto* e restituisce la somma dei risultati.

Nota: vedere anche **Modello di somma**, pagina 9.

$$\sum_{n=1}^{5}\left(\frac{1}{n}\right) \qquad \frac{137}{60}$$

Σ(*Espr1***,** *Var***,** *Basso***,** *Basso−1***)⇒**0

Σ(*Espr1***,** *Var***,** *Basso***,** *Alto***) ⇒-Σ(***Espr1***,** *Var***,** *Alto+1***,** *Basso***−1)** if *Alto* < *Basso−1*

$$\sum_{k=4}^{3}(k) \qquad 0$$

Le formule di sommatoria utilizzate sono derivate dal seguente riferimento:

Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Reading, Massachusetts: Addison-Wesley, 1994.

$$\sum_{k=4}^{1}(k) \qquad -5$$

$$\sum_{k=4}^{1}(k)+\sum_{k=2}^{4}(k) \qquad 4$$

ΣInt() Catalogo >

ΣInt(*NPmt1***,** *NPmt2***,** *N***,** *I***,** *PV***,[***Pmt***],** [*FV*], [*PpY*], [*CpY*], [*PmtAt*], [*valoreArrotondato*]**)⇒***valore*

ΣInt
(
NPmt1
,*NPmt2***,***tabellaAmmortamento***)⇒***valore*

Funzione di ammortamento che calcola la somma dell'interesse pagato durante un intervallo specifico di pagamenti.

NPmt1 e *NPmt2* definiscono l'inizio e la fine dell'intervallo dei pagamenti.

N, *I*, *PV*, *Pmt*, *FV*, *PpY*, *CpY* e *PmtAt* sono descritti nella tabella degli argomenti TVM, pagina 173.

ΣInt(1,3,12,4.75,20000,,12,12) -213.48

ΣInt() Catalogo >

- Se si omette *Pmt*, viene utilizzata l'impostazione predefinita *Pmt*=**tvmPmt(***N,I,PV,FV,PpY,CpY,PmtAt***)**.
- Se si omette *FV*, viene utilizzata l'impostazione predefinita *FV*=0.
- Le impostazioni predefinite di *PpY*, *CpY* e *PmtAt* sono le stesse delle funzioni TVM.

valoreArrotondato specifica il numero di cifre decimali di arrotondamento. Impostazione predefinita=2.

ΣInt(*NPmt1,NPmt2,tabellaAmmortamento***)** calcola la somma degli interessi sulla base della tabella di ammortamento *tabellaAmmortamento*. L'argomento *tabellaAmmortamento* deve essere una matrice avente la forma descritta in **amortTbl()**, pagina 11.

Nota: vedere anche ΣPrn(), sotto, e **Bal()**, pagina 20.

tbl:=amortTbl(12,12,4.75,20000,,12,12)

0	0.	0.	20000.
1	−77.49	−1632.43	18367.6
2	−71.17	−1638.75	16728.8
3	−64.82	−1645.1	15083.7
4	−58.44	−1651.48	13432.2
5	−52.05	−1657.87	11774.4
6	−45.62	−1664.3	10110.1
7	−39.17	−1670.75	8439.32
8	−32.7	−1677.22	6762.1
9	−26.2	−1683.72	5078.38
10	−19.68	−1690.24	3388.14
11	−13.13	−1696.79	1691.35
12	−6.55	−1703.37	−12.02

ΣInt(1,3,*tbl*) −213.48

ΣPrn() Catalogo >

ΣPrn(*NPmt1***,** *NPmt2***,** *N***,** *I***,** *PV***,** [*Pmt*]**,** [*FV*]**,** [*PpY*]**,** [*CpY*]**,** [*PmtAt*]**,** [*valoreArrotondato*]**)⇒***valore*

ΣPrn(*NPmt1* **,***NPmt2***,***tabellaAmmortamento***)⇒***valore*

Funzione di ammortamento che calcola la somma del capitale versato durante un intervallo specifico di pagamenti.

NPmt1 e *NPmt2* definiscono l'inizio e la fine dell'intervallo dei pagamenti.

*N***,** *I***,** *PV*, *Pmt*, *FV*, *PpY*, *CpY* e *PmtAt* sono descritti nella tabella degli argomenti TVM, pagina 173.

- Se si omette *Pmt*, viene utilizzata l'impostazione predefinita *Pmt*=**tvmPmt(***N,I,PV,FV,PpY,CpY,PmtAt***)**.

ΣPrn(1,3,12,4.75,20000,,12,12) −4916.28

tbl:=amortTbl(12,12,4.75,20000,,12,12)

0	0.	0.	20000.
1	−77.49	−1632.43	18367.57
2	−71.17	−1638.75	16728.82
3	−64.82	−1645.1	15083.72
4	−58.44	−1651.48	13432.24
5	−52.05	−1657.87	11774.37
6	−45.62	−1664.3	10110.07
7	−39.17	−1670.75	8439.32
8	−32.7	−1677.22	6762.1
9	−26.2	−1683.72	5078.38
10	−19.68	−1690.24	3388.14
11	−13.13	−1696.79	1691.35
12	−6.55	−1703.37	−12.02

ΣPrn(1,3,*tbl*) −4916.28

ΣPrn() Catalogo >

- Se si omette *FV*, viene utilizzata l'impostazione predefinita *FV*=0.
- Le impostazioni predefinite di *PpY*, *CpY* e *PmtAt* sono le stesse delle funzioni TVM.

valoreArrotondato specifica il numero di cifre decimali di arrotondamento. Impostazione predefinita=2.

ΣPrn (*NPmt1,NPmt2,tabellaAmmortamento***)** calcola la somma del capitale versato sulla base della tabella di ammortamento *tabellaAmmortamento*. L'argomento *tabellaAmmortamento* deve essere una matrice avente la forma descritta in **amortTbl()**, pagina 11.

Nota: vedere anche ΣInt(), sopra, e **Bal()**, pagina 20.

(conversione indiretta) Tasti ctrl

*stringaNomeVariabile*

Questo operatore, che si riferisce alla variabile chiamata *stringaNomeVariabile*, permette di creare e di modificare le variabili dall'interno di una funzione mediante le stringhe.

xyz:=12	12
#("x"&"y"&"z")	12

Crea o fa riferimento alla variabile xyz.

10→*r*	10
"r"→*s1*	"r"
#*s1*	10

Restituisce il valore della variabile (r) il cui nome è memorizzato nella variabile s1.

E (notazione scientifica) Tasto EE

*mantissa***E***esponente*

Permette di inserire un numero in notazione scientifica. Il numero viene interpretato come *mantissa* × $10^{esponente}$.

23000.	23000.
2300000000.+4.1E15	4.1E15
$3\cdot10^4$	30000

E (notazione scientifica) — Tasto EE

Suggerimento: se si desidera inserire una potenza di 10 senza ottenere un valore decimale per risultato, utilizzare 10^*intero*.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando `@E`. Per esempio, digitare `2.3@E4` per inserire 2.3E4.

g (gradianti) — Tasto π▸

*Espr1*g ⇒*espressione*

*Espr1*g ⇒*espressione*

*Lista1*g ⇒*lista*

*Matrice1*g ⇒*matrice*

Questa funzione consente di utilizzare un angolo espresso in gradianti pure essendo attiva la modalità di misurazione degli angoli in gradi o in radianti.

In modalità angolo in radianti, moltiplica *Espr1* per π/200.

In modalità angolo in gradi, moltiplica *Espr1* per g/100.

In modalità gradianti, restituisce *Espr1* senza modifiche.

Nota: è possibile inserire questo simbolo dalla tastiera del computer digitando `@g`.

In modalità di misurazione degli angoli in gradi, gradianti o radianti:

$\cos(50^g)$	0.707107
$\cos(\{0,100^g,200^g\})$	$\{1,0.,-1.\}$

r (radianti) — Tasto π▸

*Valore1*r ⇒*valore*

*Lista1*r ⇒*lista*

*Matrice1*r ⇒*matrice*

In modalità angolo in gradi, gradianti o radianti:

$\cos\left(\frac{\pi}{4^r}\right)$	0.707107
$\cos\left(\left\{0^r,\left(\frac{\pi}{12}\right)^r,-(\pi)^r\right\}\right)$	$\{1,0.965926,-1.\}$

ʳ (radianti) Tasto π▸

Questa funzione consente di utilizzare un angolo espresso in radianti pur essendo attiva la modalità di misurazione degli angoli in gradi o in gradianti.

In modalità angolo in gradi, moltiplica l'argomento per 180/π.

In modalità angolo in radianti, restituisce l'argomento immutato.

In modalità angolo in gradianti, moltiplica l'argomento per 200/π.

Suggerimento: ricorrere al simbolo ʳ se si desidera forzare l'uso dei radianti in una definizione, indipendentemente dalla modalità prevalente nella funzione.

Nota: è possibile inserire questo simbolo dalla tastiera del computer digitando **@r**.

° (gradi) Tasto π▸

Valore1°⇒valore

Lista1°⇒lista

Matrice1°⇒matrice

Questa funzione consente di utilizzare un angolo espresso in gradi pur essendo attiva la modalità di misurazione degli angoli in gradianti o in radianti.

In modalità angolo in radianti, moltiplica l'argomento per π/180.

In modalità angolo in gradi, restituisce l'argomento immutato.

In modalità angolo in gradianti, moltiplica l'argomento per 10/9.

Nota: è possibile inserire questo simbolo dalla tastiera del computer digitando **@d**.

In modalità angolo in gradi, gradianti o radianti:

$\cos(45°)$ 0.707107

In modalità angolo in radianti:

$\cos\left(\left\{0,\frac{\pi}{4},90°,30.12°\right\}\right)$

$\{1.,0.707107,0.,0.864976\}$

°, ’, ” (gradi/primi/secondi)

Tasti ctrl

gg°*pp*’*ss.ss*”⇒*espressione*

*gg*un numero positivo o negativo

*pp*un numero non negativo

*ss.ss*un numero non negativo

Restituisce *gg*+(*pp*/60)+(*ss.ss*/3600).

Questo formato di introduzione in base -60 consente di:

- Inserire un angolo in gradi/primi/secondi indipendentemente dalla modalità di misurazione degli angoli corrente.
- Inserire un orario nella forma ore/minuti/secondi.

Nota: far seguire ss.ss da due apostrofi (’’) e non dal simbolo di virgolette ("). In modalità angolo in gradi:

25°13'17.5"	25.2215
25°30'	$\frac{51}{2}$

∠ (angolo)

Tasti ctrl

[*Raggio*,∠θ_*Angolo*]⇒*vettore*

(formato polare)

[*Raggio*,∠θ_*Angolo*,*Z_Coordinate*]⇒*vettore*

(formato cilindrico)

[*Raggio*,∠θ_*Angolo*,∠θ_*Angolo*]⇒*vettore*

(formato sferico)

Restituisce le coordinate nella forma di un vettore a seconda dell’impostazione della modalità formato vettore: rettangolare, cilindrico o sferico.

Nota: è possibile inserire questo simbolo dalla tastiera del computer digitando @<.

In modalità angolo in radianti e con il formato vettore impostata su:

rettangolare

[5 ∠60° ∠45°]	
	[1.76777 3.06186 3.53553]

cilindrico

[5 ∠60° ∠45°]	
	[3.53553 ∠1.0472 3.53553]

sferico

[5 ∠60° ∠45°]	
	[5. ∠1.0472 ∠0.785398]

(*Grandezza* ∠ *Angolo*)⇒*valoreComplesso*

(formato polare)

In modalità angolo in radianti e in modalità formato rettangolare complesso:

∠ (angolo)

Tasti ctrl 📖

Inserisce un valore complesso in formato polare (r∠θ). *Angolo* viene interpretato conformemente alla modalità di misurazione degli angoli impostata.

$5+3\cdot i-\left(10\angle\frac{\pi}{4}\right)$	$-2.07107-4.07107\cdot i$

_ (trattino basso come elemento vuoto)

Vedere "Elementi vuoti (nulli)", pagina 212.

_ (trattino basso come designatore di unità)

Tasti ctrl ␣

*Variabile*_

Se *Variabile* non ha un valore, essa verrà trattata come se indicasse un numero complesso. Per impostazione predefinita, senza _, la variabile verrà considerata un numero reale.

Se a *Variabile* è associato un valore, il carattere _ viene ignorato e *Variabile* conserverà il tipo di dati originale.

Assumendo che z non sia definito:

real(z)	z
real(z_)	real(z_)
imag(z)	0
imag(z_)	imag(z_)

10^()

Catalogo > 🔡

10^ (*Valore1***)⇒***valore*

10^ (*Lista1***)⇒***lista*

Restituisce 10 elevato alla potenza dell'argomento.

In una lista, restituisce 10 elevato alla potenza degli elementi di *Lista1*.

$10^{1.5}$	31.6228

10^(matriceQuadrata1)⇒*matriceQuadrata*

Restituisce 10 elevato alla potenza di *matriceQuadrata1*. Ciò non equivale a calcolare 10 elevato alla potenza di ciascun elemento. Per informazioni sul metodo di calcolo, vedere **cos()**.

matriceQuadrata1 deve essere diagonalizzabile. Il risultato contiene sempre numeri a virgola mobile.

$$10^{\begin{bmatrix}1&5&3\\4&2&1\\6&-2&1\end{bmatrix}}$$

$$\begin{bmatrix}1.14336\text{E}7&8.17155\text{E}6&6.67589\text{E}6\\9.95651\text{E}6&7.11587\text{E}6&5.81342\text{E}6\\7.65298\text{E}6&5.46952\text{E}6&4.46845\text{E}6\end{bmatrix}$$

^⁻¹(reciproco) Catalogo >

Valore1 ^⁻¹⇒*valore*

Lista1 ^⁻¹⇒*lista*

Restituisce il reciproco dell'argomento.

In una lista, restituisce i reciproci degli elementi di *Lista1*.

$(3.1)^{-1}$	0.322581

matriceQuadrata1 ^⁻¹⇒*matriceQuadrata*

Restituisce l'inversa di *matriceQuadrata1*.

matriceQuadrata1 deve essere una matrice quadrata non singolare.

$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^{-1}$	$\begin{bmatrix} -2 & 1 \\ \frac{3}{2} & \frac{-1}{2} \end{bmatrix}$

| (operatore di vincolo) ctrl tasti

Expr | *BooleanExpr1* [**and***BooleanExpr2*]...

Expr | *BooleanExpr1* [**or***BooleanExpr2*]...

Il simbolo di vincolo ("|") serve da operatore binario. L'operando a sinistra di | è un'espressione. L'operando a destra di | definisce una o più relazioni destinate ad avere un effetto sulla semplificazione dell'espressione. Le relazioni multiple dopo | devono essere collegate da operatori logici "**and" o "or".**

$x+1\|x=3$	4
$x+55\|x=\sin(55)$	54.0002

L'operatore di vincolo fornisce tre tipi primari di funzionalità:

- Sostituzioni
- Vincoli d'intervallo
- Esclusioni

Le sostituzioni si presentano nella forma di equazione, come per esempio x=3 o y=sin (x). I migliori risultati si ottengono quando la parte sinistra è una variabile semplice. *Expr* | *Variabile* = *valore* sostituirà *valore* per ogni ricorrenza di *Variabile* in *Expr*.

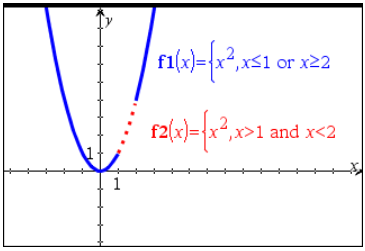
$x^3-2\cdot x+7\rightarrow f(x)$	*Done*
$f(x)\|x=\sqrt{3}$	8.73205

| (operatore di vincolo)

ctrl 📖 tasti

I vincoli d'intervallo si presentano nella forma di una o più disequazioni, collegate da operatori logici “**and**” o “**or**”. I vincoli d'intervallo permettono inoltre la semplificazione, la quale sarebbe altrimenti non valida o non calcolabile.

$\text{nSolve}(x^3+2\cdot x^2-15\cdot x=0,x)$	0.
$\text{nSolve}(x^3+2\cdot x^2-15\cdot x=0,x)\mid x>0 \text{ and } x<5$	3.



Le esclusioni utilizzano l'operatore relazionale “diverso da” (/= or ≠) per escludere dalla valutazione un valore specifico.

→ (memorizza)

Tasti ctrl var

Valore → *Var*

List →

Matrix → *Var*

Espr → *Funzione*(*Param1*,...)

Lista → *Funzione*(*Param1*,...)

Matrice → *Funzione*(*Param1*,...)

Se la variabile *Var* non esiste, *Var* viene creata e inizializzata in *Valore*, *Lista* o *Matrice*.

Se la variabile *Var* esiste già e non è bloccata o protetta, l’operatore ne sostituisce il contenuto con *Valore*, *Lista* o *Matrice*.

Nota: è possibile inserire questo operatore dalla tastiera del computer digitando **=:** come scelta rapida. Per esempio, digitare **`pi/4 =: Miavar`**.

$\frac{\pi}{4}\rightarrow myvar$	0.785398
$2\cdot\cos(x)\rightarrow y1(x)$	*Done*
$\{1,2,3,4\}\rightarrow lst5$	$\{1,2,3,4\}$
$\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}\rightarrow matg$	$\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}$
"Hello" → *str1*	"Hello"

:= (assegna)

Tasti ctrl

Var **:=** *Valore*

Var **:=** *Lista*

Var **:=** *Matrice*

Funzione(*Param1*,...) **:=** *Espr*

Funzione(*Param1*,...) **:=** *Lista*

Funzione(*Param1*,...) **:=** *Matrice*

Se la variabile *Var* non esiste, *Var* viene creata e inizializzata in *Valore*, *Lista* o *Matrice*.

Se la variabile *Var* esiste già e non è bloccata o protetta, l'operatore ne sostituisce il contenuto con *Valore*, *Lista* o *Matrice*.

$myvar:=\frac{\pi}{4}$	.785398
$y1(x):=2\cdot\cos(x)$	*Done*
$lst5:=\{1,2,3,4\}$	$\{1,2,3,4\}$
$matg:=\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}$	$\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}$
$str1:=$"Hello"	"Hello"

© (commento)

Tasti ctrl

© [*testo*]

© considera *testo* come una riga di commento che consente di annotare le funzioni e i programmi creati.

© può trovarsi all'inizio o in qualsiasi altro punto della riga. Tutto quanto si trova a destra del segno © fino alla fine della riga viene considerato come commento.

Nota per l'inserimento dell'esempio: per istruzioni sull'inserimento di definizioni di programmi e funzioni costituite da più righe, consultare la sezione Calcolatrice del manuale del prodotto.

```
Define g(n)=Func
    © Declare variables
    Local i,result
    result:=0
    For i,1,n,1 ©Loop n times
    result:=result+i^2
    EndFor
    Return result
    EndFunc
                                  Done
g(3)                                14
```

0b, 0h

Tasti 0 B, **tasti** 0 H

0b *numeroBinario*

0h *numeroEsadecimale*

In modalità base Dec:

0b10+0hF+10	27

In modalità base Bin:

0b, 0h

Tasti 0 B, tasti 0 H

Indica, rispettivamente, un numero binario o un numero esadecimale. Per inserire un numero binario o esadecimale, inserire il prefisso 0b o 0h indipendentemente dalla modalità Base che è stata impostata. Senza prefisso, un numero viene considerato decimale (base 10).

I risultati vengono visualizzati nella modalità Base che è stata impostata.

0b10+0hF+10	0b11011

In modalità base Esadecimale:

0b10+0hF+10	0h1B

Elementi vuoti (nulli)

Quando si analizzano i dati del mondo reale, può accadere di non disporre sempre di una serie di dati completa. TI-Nspire™ consente l'uso di elementi vuoti o nulli. Ciò permette di proseguire con i dati a disposizione, anziché dover ricominciare da capo o scartare i casi incompleti.

Per un esempio di dati con elementi vuoti, vedere "*Rappresentazione grafica dei dati di un foglio di calcolo*" nel capitolo Foglio elettronico.

La funzione **delVoid()** consente di rimuovere elementi vuoti da una lista. Consente inoltre di testare un elemento vuoto. Per ulteriori informazioni, vedere **delVoid()**, pagina 46 e **isVoid()**, pagina 80.

Nota: per inserire manualmente un elemento vuoto in un'espressione matematica, digitare "_" o la parola chiave **`void`**. La parola chiave **`void`** viene convertita automaticamente in un carattere "_" quando l'espressione viene calcolata. Per digitare il carattere "_" sul palmare, premere [ctrl] [␣].

Calcoli con elementi vuoti

La maggior parte dei calcoli con un inserimento vuoto producono un risultato vuoto. Vedere i casi speciali sotto.

$\lvert _ \rvert$	_
gcd(100,_)	_
3+_	_
{5,_,10}−{3,6,9}	{2,_,1}

Argomenti di lista contenti elementi vuoti

Le funzioni e i comandi seguenti ignorano (saltano) gli elementi vuoti che trovano negli argomenti di lista.

count, **countIf**, **cumulativeSum**, **freqTable▸list**, **frequency**, **max**, **mean**, **median**, **product**, **stDevPop**, **stDevSamp**, **sum**, **sumIf**, **varPop** e **varSamp**, nonché i calcoli di regressione, le statistiche **OneVar**, **TwoVar** e **FiveNumSummary**, gli intervalli di confidenza e i test statistici.

sum({2,_,3,5,6.6})	16.6
median({1,2,_,_,_,3})	2
cumulativeSum({1,2,_,4,5})	{1,3,_,7,12}
cumulativeSum$\left(\begin{bmatrix} 1 & 2 \\ 3 & _ \\ 5 & 6 \end{bmatrix}\right)$	$\begin{bmatrix} 1 & 2 \\ 4 & _ \\ 9 & 8 \end{bmatrix}$

SortA e **SortD** spostano in fondo tutti gli elementi vuoti del primo argomento.

{5,4,3,_,1}→*list1*	{5,4,3,_,1}
{5,4,3,2,1}→*list2*	{5,4,3,2,1}
SortA *list1*,*list2*	*Done*
list1	{1,3,4,5,_}
list2	{1,3,4,5,2}

Argomenti di lista contenti elementi vuoti

$\{1,2,3,_,5\} \rightarrow list1$	$\{1,2,3,_,5\}$
$\{1,2,3,4,5\} \rightarrow list2$	$\{1,2,3,4,5\}$
SortD *list1*,*list2*	*Done*
list1	$\{5,3,2,1,_\}$
list2	$\{5,3,2,1,4\}$

Nelle regressioni, un elemento vuoto in una lista X o Y introduce un elemento vuoto corrispondente nel residuo.

$l1:=\{1,2,3,4,5\}$: $l2:=\{2,_,3,5,6.6\}$	$\{2,_,3,5,6.6\}$
LinRegMx *l1*,*l2*	*Done*
stat.Resid	$\{0.434286,_,-0.862857,-0.011429,0.44\}$
stat.XReg	$\{1.,_,3.,4.,5.\}$
stat.YReg	$\{2.,_,3.,5.,6.6\}$
stat.FreqReg	$\{1.,_,1.,1.,1.\}$

Una categoria omessa nelle regressioni introduce un elemento vuoto corrispondente nel residuo.

$l1:=\{1,3,4,5\}$: $l2:=\{2,3,5,6.6\}$	$\{2,3,5,6.6\}$
$cat:=\{\text{"M"},\text{"M"},\text{"F"},\text{"F"}\}$: $incl:=\{\text{"F"}\}$	$\{\text{"F"}\}$
LinRegMx *l1*,*l2*,1,*cat*,*incl*	*Done*
stat.Resid	$\{_,_,0.,0.\}$
stat.XReg	$\{_,_,4.,5.\}$
stat.YReg	$\{_,_,5.,6.6\}$
stat.FreqReg	$\{_,_,1.,1.\}$

Una frequenza 0 nelle regressioni introduce un elemento vuoto corrispondente nel residuo.

$l1:=\{1,3,4,5\}$: $l2:=\{2,3,5,6.6\}$	$\{2,3,5,6.6\}$
LinRegMx *l1*,*l2*,$\{1,0,1,1\}$	*Done*
stat.Resid	$\{0.069231,_,-0.276923,0.207692\}$
stat.XReg	$\{1.,_,4.,5.\}$
stat.YReg	$\{2.,_,5.,6.6\}$
stat.FreqReg	$\{1.,_,1.,1.\}$

Scelte rapide per l'inserimento di espressioni matematiche

Le scelte rapide permettono di inserire elementi di espressioni matematiche per digitazione invece di usare il Catalogo o la tavolozza Simboli. Per esempio, per inserire l'espressione $\sqrt{6}$, è possibile digitare **sqrt(6)** nella riga di introduzione. Quando si preme enter, l'espressione **sqrt(6)** viene modificata in $\sqrt{6}$. Alcune scelte rapide sono utili sia per il palmare sia per la tastiera del computer. Altre sono utili principalmente dalla tastiera del computer.

Dalla tastiera del palmare o del computer

Per inserire questo:	digitare questa scelta rapida:
π	**pi**
θ	**theta**
∞	**infinity**
$\leq$	**<=**
$\geq$	**>=**
$\neq$	**/=**
$\Rightarrow$ (implicazione logica)	**=>**
$\Leftrightarrow$ (doppia implicazione logica, XNOR)	**<=>**
→ (memorizza operatore)	**=:**
| | (valore assoluto)	**abs(...) (Valore assoluto)**
$\sqrt{}$**()**	**sqrt(...) (Radice quadrata)**
Σ() (Modello di somma)	**sumSeq(...)**
Π**()** (Modello di prodotto)	**prodSeq(...)**
sin^{-1}(), cos^{-1}(), ...	**arcsin(...), arccos(...)**, ...
Δ**Lista()**	**deltaList(...)**

Dalla tastiera del computer

Per inserire questo:	digitare questa scelta rapida:
i (unità immaginaria)	**@i**
e (logaritmo naturale in base e)	**@e**
E (notazione scientifica)	**@E**
^{T} (trasposizione)	**@t**

Per inserire questo:	digitare questa scelta rapida:
ʳ (radianti)	`@r`
° (Gradi)	`@d`
G gradianti	`@g`
∠ (angolo)	`@<`
▸ (conversione)	`@>`
▸Decimal, **▸approxFraction()**, ecc.	`@>Decimal`, `@>approxFraction()`, ecc.

EOS™ (Equation Operating System) gerarchia

Questa sezione descrive il sistema EOS™ (Equation Operating System) utilizzato da TI-Nspire™ Sistema integrato per la matematica e le scienze sperimentali. Numeri, variabili e funzioni vengono introdotte in successione. Il software EOS™ calcola le espressioni e le equazioni utilizzando raggruppamenti racchiusi tra parentesi e in base alle priorità descritte sotto.

Ordine di Valutazione

Livello	Operatore
1	Parentesi tonde ( ), parentesi quadre [ ], parentesi graffe { }
2	Conversione indiretta (#)
3	Chiamate di funzione
4	Post operatori: gradi-prmi-secondi(°,',"), fattoriale (!), percentuale (%), radiante (ʳ), pedice ([ ]), trasposizione (ᵀ)
5	Elevamento a potenza, operatore di potenza (^)
6	Segno negativo (⁻)
7	Concatenazione di stringhe (&)
8	Moltiplicazione (•), divisione (/)
9	Addizione (+), sottrazione (-)
10	Rapporti di uguaglianza: uguale (=), non uguale (≠ o /=), minore di (<), minore di o uguale a (≤ o <=), maggiore di (>), maggiore di o uguale a (≥ o >=)
11	not logico
12	and logico
13	Logico **or**
14	**xor**, **nor**, **nand**
15	Implicazione logica (⇒)
16	Doppia implicazione logica, XNOR (⇔)
17	Operatore di vincolo (“|”)
18	Memorizzazione (→)

Parentesi tonde, quadre e graffe

Tutti i calcoli racchiusi tra parentesi tonde, quadre o graffe vengono eseguiti per primi. Ad esempio, nell'espressione 4(1+2), EOS™ calcola prima la parte di espressione racchiusa tra parentesi tonde, 1+2, quindi moltiplica il risultato, 3, per 4.

In un'espressione o in un'equazione, tutte le parentesi tonde, quadre e graffe aperte devono essere chiuse. Diversamente, viene visualizzato un messaggio di errore a indicare l'elemento mancante. Ad esempio, (1+2)/(3+4 produrrà un messaggio di errore “Manca).”

Nota: poiché il software TI-Nspire™ consente di definire funzioni personalizzate, un nome di variabile seguito da un'espressione racchiusa tra parentesi viene considerato una “chiamata di funzione” e non una moltiplicazione implicita. Ad esempio a(b+c) è la funzione a calcolata per b+c. Per moltiplicare l'espressione b+c per la variabile a, utilizzare la moltiplicazione esplicita: a∗(b+c).

Conversione indiretta

L'operatore di conversione indiretta (#) converte una stringa in un nome di variabile o di funzione. Ad esempio, #(“x”&”y”&”z”) crea il nome di variabile xyz. La conversione indiretta consente inoltre di creare e modificare variabili dall'interno di un programma. Ad esempio, se 10→r e “r”→s1, allora #s1=10.

Post operatori

I post operatori sono operatori che vengono inseriti immediatamente dopo un argomento, come ad esempio 5!, 25% o 60°15' 45". Gli argomenti seguiti da un post operatore vengono calcolati al quarto livello di priorità. Ad esempio, nell'espressione 4^3!, 3! viene calcolato per primo. Il risultato, 6, diventa quindi l'esponente di 4 che darà come risultato 4096.

Elevazione a potenza

L'elevamento a potenza (^) e l'elevazione a potenza elemento per elemento (.^) vengono calcolati da destra a sinistra. Ad esempio, l'espressione 2^3^2 viene calcolata allo stesso modo di 2^(3^2) dando come risultato 512. Ciò è diverso da (2^3)^2, che dà come risultato 64.

Segno negativo

Per introdurre un numero negativo, premere [(−)] quindi il numero. Le post operazioni e l'elevamento a potenza vengono eseguiti prima dell'operazione di cambiamento di segno. Ad esempio, il risultato di ⁻x2 è un numero negativo e $^{-}9^2 = {}^{-}81$. Utilizzare le parentesi per elevare al quadrato un numero negativo, come ad esempio $(^{-}9)^2$ che dà come risultato 81.

Vincolo (“|”)

L'argomento che segue l'operatore di vincolo (“|”) fornisce una serie di restrizioni che influiscono sul calcolo dell'argomento che precede l'operatore.

Codici di errore e messaggi

Quando si produce un errore, il relativo codice viene assegnato alla variabile *errCode*. Programmi e funzioni definite dall'utente possono esaminare *errCode* per determinare la causa dell'errore. Per un esempio dell'uso di *errCode*, vedere l'Esempio 2 del comando **Try** (pagina 169).

Nota: alcune condizioni di errore si riferiscono solo ai prodotti TI-Nspire™ CAS, mentre altre solo ai prodotti TI-Nspire™.

Codice errore	Descrizione
10	Una funzione non ha restituito un valore
20	Una prova non ha saputo stabilire se VERO o FALSO. In generale, non è possibile confrontare variabili non definite. Ad esempio, la prova If a<b causerà questo errore se o a o b sono indefiniti al momento dell'esecuzione dell'istruzione If.
30	Argomento non può essere un nome di cartella.
40	Argomento errato
50	Argomento di tipo errato Due o più argomenti devono essere dello stesso tipo.
60	Argomento deve essere un'espressione booleana o un numero intero
70	Argomento deve essere un numero decimale
90	Argomento deve essere una lista
100	Argomento deve essere una matrice
130	Argomento deve essere una stringa
140	Argomento deve essere un nome di variabile. Accertarsi che il nome: • non inizi con un numero • non contenga spazi o caratteri speciali • non utilizzi trattini bassi o virgole in modo non valido • non superi i limiti di lunghezza Per ulteriori informazioni, vedere il capitolo Calcolatrice nella documentazione.
160	Argomento deve essere un'espressione
165	Carica delle batterie insufficiente per inviare/ricevere Inserire nuove batteria prima di inviare o ricevere.
170	Estremo

Codice errore	Descrizione
	L'estremo inferiore deve essere minore dell'estremo superiore per definire l'intervallo di ricerca.
180	Interruzione È stato premuto il tasto esc o on durante un calcolo lungo o durante l'esecuzione del programma.
190	Definizione circolare Questo messaggio viene visualizzato per evitare l'esaurimento della memoria durante la sostituzione infinta di valori di variabile nel corso di una semplificazione. Ad esempio, a+1->a, dove a è una variabile indefinita, causerà questo errore.
200	Condizione non valida Ad esempio, solve(3x^2-4=0,x) | x<0 or x>5 produrrebbero questo messaggio di errore perché il vincolo è separato da "or" invece che da "and."
210	Tipo di dati non valido Un argomento è un tipo di dati sbagliato.
220	Limite dipendente
230	Dimensione Un indice di lista o di matrice non è valido. Ad esempio, se la lista {1,2,3,4} è memorizzata in L1, allora L1[5] è un errore di dimensione perché L1 contiene solo quattro elementi.
235	Errore di dimensione. Elementi insufficienti nelle liste.
240	Dimensioni non corrispondenti Due o più argomenti devono essere della stessa dimensione. Ad esempio, [1,2]+[1,2,3] è un errore di dimensioni non corrispondenti perché le matrici contengono un numero diverso di elementi.
250	Divisione per zero
260	Argomento errato Deve essere presente un argomento in un dominio specificato. Ad esempio rand(0) non è valido.
270	Nome di variabile duplicato
280	Else ed ElseIf non validi fuori dal blocco If..EndIf
290	EndTry non trova la corrispondente istruzione Else
295	Numero eccessivo di iterazioni

Codice errore	Descrizione
300	Attesa lista o matrice a 2 o 3 elementi
310	Il primo argomento di nSolvedeve essere un'equazione a una sola variabile. Non può contenere una variabile non calcolata che non sia la variabile cercata.
320	Il primo argomento di solve o cSolve deve essere un'equazione o una disequazione. Ad esempio, solve(3x^2-4,x) non è valido perché il primo argomento non è un'equazione.
345	Unità di misura non coerenti
350	Indice non valido
360	La stringa da convertire non è un nome di variabile valido
380	Risultato indefinito Il precedente calcolo non ha prodotto un risultato oppure non è stato inserito alcun calcolo precedente.
390	Assegnazione non valida
400	Valore di assegnazione non valido
410	Comando non valido
430	Non valido per le impostazioni di modo correnti
435	Ipotesi non valida
440	Moltiplicazione sottintesa non valida Ad esempio, x(x+1) non è valido, mentre x*(x+1) è la sintassi corretta. Ciò per evitare confusione tra chiamate di funzione e moltiplicazioni implicite.
450	Non valido in una funzione o nella attuale espressione In una funzione definita dall'utente sono validi solo certi comandi.
490	Non valido nel blocco Try..EndTry
510	Lista o matrice non valida
550	Non valido fuori da una funzione o un programma Alcuni comandi non sono validi fuori da una funzione o un programma. Ad esempio, Local non può essere utilizzato in una funzione o in un programma.
560	Non valido fuori dai blocchi Loop..EndLoop, For..EndFor o While..EndWhile Ad esempio, il comando Exit è valido solo in questi blocchi loop.
565	Non valido fuori da un programma
570	Nome di percorso non valido

Codice errore	Descrizione
	Ad esempio, \var non è valido.
575	Numero complesso in forma polare non valido
580	Chiamata di programma non valida Non è possibile chiamare un programma all'interno di funzioni o espressioni quali 1+p(x), dove p è un programma.
600	Tabella non valida
605	Uso di unità non valido
610	Nome di variabile non valido in una istruzione Local
620	Nomi di variabile o di funzione non valido
630	Chiamata di variabile non valida
640	Sintassi del vettore non valida
650	Errore di collegamento Trasmissione non completata tra due unità. Verificare che il cavo di collegamento sia inserito correttamente ad entrambe le estremità.
665	Matrice non diagonizzabile
670	Memoria quasi esaurita 1. Eliminare alcuni dati in questo documento 2. Salvare e chiudere questo documento Se le istruzioni 1 e 2 non producono l'esito sperato, togliere e rimettere le batterie
672	Risorsa esaurita
673	Risorsa esaurita
680	Manca (
690	Manca)
700	Manca "
710	Manca]
720	Manca }
730	Manca inizio o fine del blocco
740	Manca Then nel blocco If..EndIf
750	Il nome non è una funzione o un programma

Codice errore	Descrizione
765	Nessuna funzione selezionata
780	Nessuna soluzione trovata
800	Risultato non reale Ad esempio, se il software è impostato su Real, √(-1) non è valido. Per consentire risultati complessi, modificare l'impostazione di modalità "Real or Complex" in RETTANGOLARE (RECTANGULAR) o POLARE (POLAR).
830	Superamento della memoria
850	Programma non trovato Impossibile trovare una chiamata di programma all'interno di un altro programma nel percorso specificato durante l'esecuzione.
855	Funzione di tipo Rand non consentita nel grafico
860	Ricorsione troppo profonda
870	Nome riservato o variabile di sistema
900	Argomento errato Impossibile applicare al set di dati il modello mediana-mediana.
910	Errore di sintassi
920	Testo non trovato
930	Argomenti mancanti Nella funzione o nel comando mancano uno o più argomenti.
940	Troppi argomenti L'espressione o equazione contiene un numero eccessivo di argomenti e non può essere calcolata.
950	Troppi indici
955	Troppe variabili non definite
960	La variabile non è definita Nessun valore assegnato alla variabile. Utilizzare uno dei seguenti comandi: • sto → • := • **Define** per assegnare valori alle variabili.

Codice errore	Descrizione
965	SO privo di licenza
970	Variabile in uso, di conseguenza non sono consentite chiamate e modifiche
980	La variabile è protetta
990	Nome variabile non valido Accertarsi che il nome non superi i limiti di lunghezza
1000	Valori dei parametri di Window
1010	Zoom
1020	Errore interno
1030	Violazione della memoria protetta
1040	Funzione non supportata. Questa funzione richiede il sistema CAS (Computer Algebra System). Provare TI-Nspire™ CAS.
1045	Operatore non supportato. Questo operatore richiede il sistema CAS (Computer Algebra System). Provare TI-Nspire™ CAS.
1050	Funzione non supportata. Questo operatore richiede il sistema CAS (Computer Algebra System). Provare TI-Nspire™ CAS.
1060	L'argomento dell'inserimento deve essere numerico. Sono consentiti solo inserimenti contenenti valori numerici.
1070	L'argomento della funzione trigonometrica è troppo grande per una riduzione precisa.
1080	Uso non supportato di Ans. Questa applicazione non supporta Ans.
1090	La funzione non è definita. Utilizzare uno dei seguenti comandi: • **Define** • **:=** • sto → per definire una funzione.
1100	Calcolo non reale Ad esempio, se il software è impostato su "Real" √(-1) non è valido. Per consentire risultati complessi, modificare l'impostazione di modalità "Real or Complex" in RETTANGOLARE (RECTANGULAR) o POLARE (POLAR).
1110	Estremi non validi
1120	Nessun cambio di segno

Codice errore	Descrizione
1130	Argomento non può essere una lista o una matrice
1140	Argomento errato Il primo argomento deve essere un'espressione polinomiale nel secondo argomento. Se il secondo argomento viene omesso, il software tenta di selezionare un'impostazione predefinita.
1150	Argomento errato I primi due argomenti devono essere espressioni polinomiali nel terzo argomento. Se il terzo argomento viene omesso, il software tenta di selezionare un'impostazione predefinita.
1160	Nome di percorso libreria non valido Un nome di percorso deve avere la forma *xxx**yyy*, dove: • La parte *xxx* può avere da 1 a 16 caratteri. • La parte *yyy* può avere da 1 a 15 caratteri. Per ulteriori informazioni, vedere il capitolo Libreria nella documentazione.
1170	Uso di nome percorso libreria non valido • Impossibile assegnare un valore a un nome percorso utilizzando **Define**, **:=** o sto **→**. • Impossibile dichiarare un nome percorso come variabile Local oppure utilizzarlo come parametro in una definizione di funzione o programma.
1180	Nome variabile libreria non valido. Accertarsi che il nome: • non contenga un punto • non inizi con un trattino basso • non contenga più di 15 caratteri Per ulteriori informazioni, vedere il capitolo Libreria nella documentazione.
1190	Documento Libreria non trovato: • Verificare che la libreria sia nella cartella MieLibrerie. • Aggiornare le librerie. Per ulteriori informazioni, vedere il capitolo Libreria nella documentazione.
1200	Variabile libreria non trovata: • Verificare che la variabile libreria esista nella prima attività nella libreria. • Accertarsi che la variabile libreria sia stata definita come LibPub o LibPriv.

Codice errore	Descrizione
	• Aggiornare le librerie. Per ulteriori informazioni, vedere il capitolo Libreria nella documentazione.
1210	Nome collegamento libreria non valido. Accertarsi che il nome: • non contenga un punto • non inizi con un trattino basso • non contenga più di 16 caratteri • non sia un nome riservato Per ulteriori informazioni, vedere il capitolo Libreria nella documentazione.
1220	Errore di dominio. Le funzioni Retta tangente e Retta normale supportano solo funzioni con valori reali.
1230	Errore di dominio. Gli operatori di conversione trigonometrica non sono supportati nelle modalità in gradi o gradianti.
1250	Argomento errato Utilizza un sistema di equazioni lineari. Esempio di sistema di due equazioni lineari con variabili x e y: 3x+7y=5 2y-5x=-1
1260	Argomento errato: Il primo argomento di nfMin o nfMax deve essere un'espressione a una sola variabile. Non può contenere una variabile non calcolata che non sia la variabile cercata.
1270	Argomento errato L'ordine della derivata deve essere uguale a 1 o 2.
1280	Argomento errato Utilizzare un polinomio in forma normale in una sola variabile.
1290	Argomento errato Utilizzare un polinomio in una sola variabile.
1300	Argomento errato I coefficienti del polinomio devono dare come risultato valori numerici.

Codice errore	Descrizione
1310	Argomento errato: Non è stato possibile calcolare una funzione per uno o più dei suoi argomenti.
1380	Argomento errato: Non sono autorizzate le chiamate nidificate alla funzione domain().

Codici di avvertenza e messaggi

È possibile utilizzare la funzione **warnCodes()** per memorizzare i codici delle avvertenze generate dal calcolo di un'espressione. Questa tabella riporta ciascun codice numerico e il relativo messaggio associato.

Per un esempio dei codici di avvertenza, vedere **warnCodes()**, pagina 178.

Codice di avvertenza	**Messaggio**
10000	L'operazione può introdurre soluzioni false
10001	Derivare un'equazione può produrre un'equazione falsa
10002	Soluzione dubbia
10003	Accuratezza dubbia
10004	L'operazione può far perdere soluzioni
10005	cSolve può specificare più zeri
10006	Solve può specificare più zeri
10007	È possibile che ci siano altre soluzioni. Provare a definire i valori inferiori e superiori adeguati e/o una ipotesi. Esempi che usano solve(): • solve(Equation, Var=Approssimativo)|lowBound<Var<upBound • solve(Equation, Var)|lowBound<Var<upBound • solve(Equation,Var=Approssimativo)
10008	Il dominio del risultato può essere minore del dominio dell'introduzione
10009	Il dominio del risultato può essere maggiore del dominio dell'introduzione
10012	Calcolo non reale
10013	∞^0 o undef^0 sostituito da 1
10014	undef^0 sostituito da 1
10015	1^∞ o 1^undef sostituito da 1
10016	1^undef sostituito da 1
10017	Overflow sostituito da ∞ o ⁻∞
10018	L'operazione richiede e restituisce un valore a 64 bit
10019	Risorsa esaurita, semplificazione forse incompleta
10020	L'argomento della funzione trigonometrica è troppo grande per una riduzione precisa

Codice di avvertenza	Messaggio
10021	L'introduzione contiene un parametro non definito. Il risultato potrebbe non essere valido per tutti i valori di parametro possibili.
10022	La specifica di estremi inferiore e superiore appropriati può produrre una soluzione.
10023	Lo scalare è stato moltiplicato per la matrice identica.
10024	Risultato ottenuto con approssimazione aritmetica
10025	L'equivalenza non può essere verificata in modalità EXACT (Esatta).
10026	Il vincolo potrebbe essere ignorato. Specificare il vincolo nella forma "\" 'Variable MathTestSymbol Constant' o da un insieme di queste due forme, per esempio 'x<3 and x>-12'

Informazioni generali

Informazioni sul servizio di manutenzione e riparazione del prodotto TI e sulla garanzia

Informazioni sul prodotto e sui servizi TI	Per ulteriori informazioni sui prodotti e servizi TI, potete contattare TI via e-mail o visiti l'indirizzo Internet di TI. Indirizzo e-mail: ti-cares@ti.com Indirizzo internet: education.ti.com
Informazioni sul servizio di manutenzione e riparazione e sulla garanzia	Per informazioni sulla durata e le condizioni della garanzia o sul servizio di manutenzione e riparazione del prodotto, fate riferimento alla dichiarazione di garanzia allegata al presente prodotto oppure contattate il vostro rivenditore/distributore Texas Instruments locale.

Indice

–

–, sottrazione[*] 187

!

!, fattoriale 198

"

", notazione in secondi 206

#

#, conversione indiretta 203
#, operatore conversione indiretta 217

%

%, percentuale 193

&

&, aggiunge 198

*****

*, moltiplicazione 188

,

, notazione in primi 206

.

.-, punto sottrazione 192
.*, punto moltiplicazione 192
./, punto divisione 192
.^, punto elevato a potenza 193
.+, punto addizione 191

:

:=, assegna 210

^

$^{-1}$, reciproco 208

^, elevamento a potenza 190

_

_, designazione di unità 207

|

|, operatore di vincolo 208

+

+, addizione 187

/

/, divisione[*] 189

=

≠, diverso[*] 194
=, uguale 194

>

>, maggiore di 196

∏

∏, prodotto[*] 200

∑

∑( ), somma[*] 201
∑Int( ) 201
∑Prn( ) 202

√

√, radice quadrata[*] 199

∫

∫, integrale[*] 199

≤

≤, minore di o uguale a 195

≥

≥, maggiore di o uguale a 196

▸

▸, converti in angolo in gradianti[Grad] 72
▸approxFraction( ) 17
▸Base10, visualizza come numero decimale[Base10] 22
▸Base16, visualizza come esadecimale[Base16] 23
▸Base2, visualizza come binario[Base2] 21
▸Cylind, visualizza come vettore in forma cilindrica[Cylind] 41
▸DD, visualizza angolo decimale[DD] 42
▸Decimal, visualizza il risultato nella forma decimale[Decimal] 42
▸DMS, visualizza come gradi/primi/secondi[DMS] 48
▸Polar, visualizza come vettore polare[Polare] 119
▸Rad, converti in radianti[Rad] 127
▸Rect, visualizza come vettore rettangolare[Rect] 130
▸Sphere, visualizza come vettore sferico[Sphere] 155

⇒

⇒ , implicazione logica[*] 197, 214

→

→, memorizza 209

⇔

⇔ , doppia implicazione logica[*] 197

©

©, commento 210

°

°, gradi/primi/secondi[*] 206
°, notazione in gradi[*] 205

0

0b, indicatore binario 210
0h, indicatore esadecimale 210

1

10^( ), potenza di dieci 207

A

abs( ), valore assoluto 11
addizione, + ... 187
affianca/concatena, augment( ) 19
aggiunge, & 198
allinterno della stringa, inString( ) 76
altro, Else 73
amortTbl( ), tabella di ammortamento 11, 20
and, operatore boolean 12
angle( ), angolo 13
angolo, angle( ) 13
ANOVA, analisi della varianza a una variabile 13
ANOVA2way, analisi della varianza a due dimensioni 14
Ans, ultimo risultato 16
approssima, approx( ) 17
approx( ), approssima 17
approxRational( ) 17
arccos() 18
arccosh() 18
arccot() 18
arccoth() 18
arccsc() 18
arccsch() 18
arcocoseno, $\cos^{-1}$( ) 32
arcoseno, $\sin^{-1}$( ) 151
arcotangente, $\tan^{-1}$( ) 163
arcsec() 18
arcsech() 18
arcsin() 18
arcsinh() 18
arctan() 19
arctanh() 19
argomenti nelle funzioni TVM 173
argomenti TVM 173
arrotondamento, round( ) 139
arrotondato per difetto, floor( ) 59
arrotondato per eccesso, ceiling( ) 24-25, 37
augment( ), affianca/concatena 19
autovalore, eigVl( ) 50

autovettore, eigVc( ) 50
avgRC( ), tasso di variazione media 19

B

binario
indicatore, 0b 210
visualizza, ▸Base2 21
binomCdf( ) (Funzione della probabilità cumulativa per la distribuzione binomiale) 24
binomPdf( ) 24
blocco di variabile e gruppi di variabili 91

C

calcola polinomio, polyEval( ) 120
calcolo, ordine di 216
campione casuale 130
cancella
errore, ClrErr 28
caratteri
codice di carattere, ord( ) 117
stringa, char( ) 25
casuale
matrice, randMat( ) 129
normale, randNorm( ) 129
polinomio, randPoly( ) 129
seme numero, RandSeed 130
Cdf( ) 58
ceiling( ), arrotondato per eccesso 24
centralDiff( ) 25
char( ), stringa di caratteri 25
$\chi^2$2way 26
ClearAZ 28
ClrErr, cancella errore 28
codici di errore e messaggi 227
colAugment 29
colDim( ), dimensione colonna di matrice 29
colNorm( ), norma colonna di matrice 29
comando Stop 159
Comando Wait 177
combinazioni, nCr( ) 106
commento, © 210
complesso
coniugato, conj( ) 29
con, | 208

conj( ), complesso coniugato 29
constructMat( ), Costruisci matrice 30
conteggio condizionato elementi in una lista, countif( ) 36
conteggio elementi in una lista, count( ) 36
conteggio giorni tra le date, dbd( ) 41
conversione indiretta, # 203
converti
    4Grad 72
    4Rad 127
coordinata x rettangolare, P▸Rx( ) 117
coordinata y rettangolare, P▸Ry( ) 117
copia variabile o funzione, CopyVar 30
corrMat( ), matrice di correlazione 31
$\cos^{-1}$, arcocoseno 32
cos( ), coseno 31
coseno, cos( ) 31
$\cosh^{-1}$( ), arcocoseno iperbolico 34
cosh( ), coseno iperbolico 33
Costruisci matrice, constructMat( ) 30
$\cot^{-1}$( ), arcocotangente 35
cot( ), cotangente 34
cotangente, cot( ) 34
$\coth^{-1}$( ), arcocotangente iperbolica 35
coth( ), cotangente iperbolica 35
count( ), conteggio elementi in una lista 36
countif( ), conteggio condizionato elementi in una lista 36
cPolyRoots() 37
crossP( ), prodotto vettoriale 37
$\csc^{-1}$( ), cosecante inversa 38
csc( ), cosecante 38
csch^{-1}( ), cosecante iperbolica inversa 39
csch( ), cosecante iperbolica 38
CubicReg, regressione cubica 39
cumulativeSum( ), somma cumulativa 40
Cycle, ripeti 40

D

d ( ), prima derivata 198
da lista a matrice, list▸mat( ) 88
da matrice a lista, mat▸list( ) 96
dbd( ), giorni tra le date 41
decimale
    visualizza angolo, ▸DD 42

visualizza intero, ▸Base10 22
Define 43
Define LibPriv 44
Define LibPub 44
Define, definisci 43
definisci, Define 43
definizione
funzione o programma privato 44
programma o funzione pubblica 44
deltaList() 45
DelVar, elimina variabile 45
delVoid( ), rimuovi elementi vuoti 46
densità di probabilità t di Student, tPdf( ) 168
densità di probabilità, normPdf( ) 111
derivata prima
modello di 9
derivata seconda
modello di 10
derivate
derivata numerica, nDeriv( ) 108
derivata numerica, nDerivative( ) 107
prima derivata, d ( ) 198
destra, right( ) 136
det( ), determinante matrice 46
deviazione standard, stdDev( ) 157-158, 176
diag( ), diagonale matrice 47
dim( ), dimensione 47
dimensione, dim( ) 47
Disp, visualizza dati 48
disposizioni semplici, nPr( ) 112
distribuzione normale cumulativa inversa (invNorm( ) 78
diverso, ≠ 194
divisione intera, intDiv( ) 77
divisione, P 189
doppia implicazione logica, ⇔ 197
dot
prodotto, dotP( ) 48
dotP( ), prodotto scalare 48

E

e alla potenza, e^( ) 55
E, esponente 203
e^( ), funzione esponenziale 49

eff), converti da tasso nominale a effettivo 50
eigVc( ), autovettore 50
eigVl( ), autovalore 50
elementi vuoti 212
elementi vuoti (nulli) 212
elementi vuoti, rimuovi 46
elevamento a potenza, ^ 190
elimina
elementi vuoti dalla lista 46
eliminazione
variabile, DelVar 45
else if, ElseIf 51
ElseIf, else if 51
end
loop, EndLoop 95
per, EndFor 60
se, EndIf 73
EndTry, fine tentativo 168
EndWhile, fine mentre 179
EOS (Equation Operating System) 216
Equation Operating System (EOS) 216
equazioni simultanee, simult( ) 149
errori e soluzione dei problemi
cancella errore, ClrErr 28
passa errore, PassErr 118
esadecimale
indicatore, 0h 210
visualizza, ▸Base16 23
esci, Exit 54
esclusione con loperatore "|" 208
esponente, E 203
esponenti
modello di 5
espressioni
da stringa a espressione, expr( ) 56
etichetta, Lbl 80
euler( ), Euler function 52
Exit, esci 54
exp( ), e alla potenza 55
expr( ), da stringa a espressione 56
ExpReg, regressione esponenziale 56

F

F Test su due campioni 63
factor( ), fattorizza 57
fattoriale, ! 198
fattorizza, factor( ) 57
Fill, riempi matrice 58
fine
    funzione, EndFunc 64
    mentre, EndWhile 179
    tentativo, EndTry 168
fine funzione, EndFunc 64
fine loop, EndLoop 95
fine mentre, EndWhile 179
fine se, EndIf 73
FiveNumSummary 59
floor( ), arrotondato per difetto 59
For 60
For, per 60
forma a scalini per righe, ref( ) 131
forma a scalini ridotta per righe, rref( ) 141
format( ), stringa formato 60
fpart( ), funzione parte frazionaria 61
frazione propria, propFrac 123
frazioni
    frzProp 123
    modello di 5
freqTable( ) 62
frequency( ) 62
Func, funzione 64
Func, funzione programma 64
funzione esponenziale e
    modello di 6
funzione esponenziale, e^( ) 49
funzione parte intera, int( ) 76
funzione piecewise a 2 tratti
    modello di 6
funzione piecewise a N tratti
    modello di 7
funzioni
    definite dallutente 43
    funzione programma, Func 64
    parte frazionaria, fpart( ) 61
funzioni definite dallutente 43

funzioni di distribuzione
binomCdf( ) (Funzione della probabilità cumulativa per la distribuzione binomiale) 24
binomPdf( ) 24
invNorm( ) 78
invt( ) 78
Invχ^2( ) (Funzione di probabilità chi2 cumulativa inversa) 78
normCdf( ) 111
normPdf( ) 111
poissCdf( ) 119
poissPdf( ) 119
tCdf( ) 165
tPdf( ) 168
$\chi^2$2way( ) 26
χ^2Cdf( ) 26
χ^2GOF( ) 27
χ^2Pdf( ) 28
funzioni e programmi definiti dallutente 44
funzioni e variabili
copia 30
funzioni finanziarie, tvmFV( ) 171
funzioni finanziarie, tvmI( ) 172
funzioni finanziarie, tvmN( ) 172
funzioni finanziarie, tvmPmt( ) 172
funzioni finanziarie, tvmPV( ) 173

G

G, gradianti 204
gcd( ), massimo comun divisore 65
geomCdf( ) 65
geomPdf( ) 66
Get 66
getDenom( ), ottieni/restituisci denominatore 67
getLangInfo( ), ottieni/restituisci informazioni sulla lingua 67
getLockInfo( ), testa lo stato bloccato di una variabile o di un gruppo di variabili 68
getMode( ), impostazioni di modo 69
getNum( ), ottieni/restituisci numeratore 70
GetStr 70
getType( ), get type of variable 71
getVarInfo( ), ottieni/restituisci informazioni variabile 71
giorni tra le date, dbd( ) 41
Goto, vai a 72
gruppi, blocco e sblocco 91, 176
gruppi, test stato di blocco 68

I

identity( ), matrice identità 73
If, se 73
ifFn( ) 74
imag( ), parte immaginaria 75
implicazione logica, ⇒ 197, 214
imposta
    modo, setMode( ) 145
impostazioni di modo, getMode( ) 69
impostazioni, visualizza correnti 69
in mezzo alla stringa, mid( ) 99
indice interno di rendimento modificato, mirr( ) 100
inString( ), allinterno della stringa 76
int( ), funzione parte intera 76
intDiv( ), divisione intera 77
integrale definito
    modello di 10
integrale, ∫ 199
interpolate( ), interpolate 77
inversa, ^⁻¹ 208
invF( ) 78
invNorm( ), distribuzione normale cumulativa inversa 78
invt( ) 78
Invχ²( ) 78
iPart( ), parte intera 78
iperbolico
    arcocoseno, $\cosh^{-1}$( ) 34
    arcoseno, $\sinh^{-1}$( ) 152
    arcotangente, $\tanh^{-1}$( ) 164
    coseno, cosh( ) 33
    seno, sinh( ) 152
    tangente, tanh( ) 164
irr( ), indice di rendimento interno
    indice di rendimento interno, irr( ) 79
isPrime( ), verifica numero primo 79
isVoid( ), test per nullità 80

L

Lbl, etichetta 80
lcm, minimo comune multiplo 81
left( ), sinistra 81
LibPriv 44

LibPub 44
libreria
crea collegamenti a oggetti 81
libShortcut( ), crea collegamenti a oggetti libreria 81
linea di regressione mediana-mediana, MedMed 98
lingua
ottieni informazioni sulla lingua 67
LinRegBx, regressione lineare 82
LinRegMx, regressione lineare 83
LinRegtIntervals, regressione lineare 84
LinRegtTest (t Test regressione lineare) 86
linSolve() 87
list▸mat( ), da lista a matrice 88
lista, conteggio condizionato 36
lista, conteggio elementi 36
liste
affianca/concatena, augment( ) 19
da lista a matrice, list▸mat( ) 88
da matrice a lista, mat▸list( ) 96
differenza, @list( ) 88
differenze in una lista, @list( ) 88
elementi vuoti in 212
in mezzo alla stringa, mid( ) 99
massimo, max( ) 96
minimo, min( ) 100
nuova, newList( ) 107
ordinamento ascendente, SortA 154
ordinamento discendente, SortD 155
prodotto scalare, dotP( ) 48
prodotto vettoriale, crossP( ) 37
prodotto, product( ) 123
somma cumulativa, cumulativeSum( ) 40
somma, sum( ) 160-161
ln( ), logaritmo naturale 88
LnReg, regressione logaritmica 89
Local, variabile locale 90
locale, Local 90
Lock, blocca variabile o gruppo di variabili 91
Log
modello di 6
logaritmi 88
logaritmo naturale, ln( ) 88
Logistic, regressione logistica 92

LogisticD, regressione logistica 93
loop, Loop 95
Loop, loop 95
LU, scomposizione matrice inferiore-superiore 95
lunghezza della stringa 47

M

maggiore di o uguale a, | 196
maggiore di, > 196
massimo comune divisore, gcd( ) 65
massimo, max( ) 96
mat▸list( ), da matrice a lista 96
matrice (1 × 2)
    modello di 8
matrice (2 × 1)
    modello di 8
matrice (2 × 2)
    modello di 8
matrice (m × n)
    modello di 8
matrice di correlazione, corrMat( ) 31
matrice identità, identity( ) 73
matrici
    affianca/concatena, augment( ) 19
    autovalore, eigVl( ) 50
    autovettore, eigVc( ) 50
    casuale, randMat( ) 129
    da lista a matrice, list▸mat( ) 88
    da matrice a lista, mat▸list( ) 96
    determinante, det( ) 46
    diagonale, diag( ) 47
    dimensione colonna, colDim( ) 29
    dimensione righe, rowDim( ) 140
    dimensione, dim( ) 47
    forma a scalini per righe, ref( ) 131
    forma a scalini ridotta per righe, rref( ) 141
    identità, identity( ) 73
    inverti righe, rowSwap( ) 140
    massimo, max( ) 96
    minimo, min( ) 100
    moltiplicazione e somma di righe, mRowAdd( ) 102
    norma colonna, colNorm( ) 29
    norma riga, rowNorm( ) 140

nuova, newMat( ) 108
operazione con righe, mRow( ) 102
prodotto, product( ) 123
punto addizione, .+ 191
punto divisione, .P 192
punto elevato a potenza, .^ 193
punto moltiplicazione, .* 192
punto sottrazione, .N 192
riempimento, Fill 58
scomposizione inferiore-superiore, LU 95
scomposizione QR, QR 124
somma cumulativa, cumulativeSum( ) 40
somma di righe, rowAdd( ) 140
somma, sum( ) 160-161
sottomatrice, subMat( ) 160-161
trasposizione, T 162
max( ), massimo 96
mean( ), media 97
media, mean( ) 97
median( ), mediana 97
mediana, median( ) 97
MedMed, linea di regressione mediana-mediana 98
memorizzazione
simbolo, & 209-210
mentre, While 179
mid( ), in mezzo alla stringa 99
min( ), minimo 100
minimo comune multiplo, lcm 81
minimo, min( ) 100
minore di o uguale a, { 195
mirr( ), indice interno di rendimento modificato 100
mod( ), modulo 101
modelli
derivata prima 9
derivata seconda 10
esponente 5
frazione 5
funzione esponenziale e 6
funzione piecewise a 2 tratti 6
funzione piecewise a N tratti 7
integrale definito 10
Log 6
matrice (1 × 2) 8

matrice (2 × 1) 8
matrice (2 × 2) 8
matrice (m × n) 8
prodotto (P) 9
radice ennesima 5
radice quadrata 5
sistema di equazioni (2 equazioni) 7
sistema di equazioni (N equazioni) 7
sommatoria (G) 9
valore assoluto 8
modi
impostazione, setMode( ) 145
modulo, mod( ) 101
moltiplicazione, * 188
mRow( ), operazione con righe di matrice 102
mRowAdd( ), moltiplicazione e somma di righe di matrice 102
MultReg (Regressione lineare multipla) 102
MultRegIntervals ( ) (Intervalli di confidenza della previsione di regressione multipla) 103
MultRegTests( ) (Verifica sulla regressione lineare multipla) 103

N

nand, operatore booleano 105
nCr( ), combinazioni 106
nDerivative( ), derivata numerica 107
newList( ), nuova lista 107
newMat( ), nuova matrice 108
nfMax( ), massimo di una funzione calcolato numericamente 108
nfMin( ), minimo di una funzione calcolato numericamente 108
nInt( ), integrale numerico 109
nom), converti tasso effettivo in nominale 109
nor, operatore boolean 110
norm( ), norma di Frobenius 111
norma di Frobenius, norm( ) 111
normCdf( ) 111
normPdf( ) 111
not, operatore booleano 111
notazione in gradi, - 205
notazione in gradi/primi/secondi 206
notazione in gradianti, G 204
notazione in primi, 206
notazione in secondi, " 206
nPr( ), disposizioni semplici 112

npv( ), valore presente netto 113
nSolve( ), soluzione numerica 113
nullo, test 80
numerica
derivata, nDeriv( ) 108
derivata, nDerivative( ) 107
integrale, nInt( ) 109
soluzione, nSolve( ) 113
nuova
lista, newList( ) 107
matrice, newMat( ) 108

O

oggetti
crea collegamenti a libreria 81
OneVar, statistiche a una variabile 114
operatore conversione indiretta (#) 217
operatore di vincolo "|" 208
operatore di vincolo. ordine di valutazione 216
operatori
ordine di calcolo 216
operatori boolean
and 12
nor 110
operatori booleani
⇒ 197
not 111
or 115
xor 179
Operatori booleani
⇒ 214
⇔ 197
nand 105
or (Boolean), oppure 115
or, operatore booleano 115
ord( ), codice numerico di carattere 117
ordinamento
ascendente, SortA 154
discendente, SortD 155
ottieni/restituisci
denominatore, getDenom( ) 67
informazioni variabile, getVarInfo( ) 67, 71
numeratore, getNum( ) 70

P

P▸Rx( ), coordinata x rettangolare 117
P▸Ry( ), coordinata y rettangolare 117
parte immaginaria, imag( ) 75
parte intera, iPart( ) 78
passa errore, PassErr 118
PassErr, passa errore 118
Pdf( ) 62
per, For 60
percentuale, % 193
piecewise( ) 118
poissCdf( ) 119
poissPdf( ) 119
polare
    coordinata, R▸Pr( ) 127
    coordinata, R▸Pθ( ) 127
    visualizza vettore, ▸Polar 119
polinomi
    calcola, polyEval( ) 120
    casuale, randPoly( ) 129
polyEval( ), calcola polinomio 120
PolyRoots() 120
potenza di dieci, 10^( ) 207
PowerReg, regressione su potenza 121
Prgm, definisci programma 122
probabilità di distribuzione normale, normCdf( ) 111
probabilità di distribuzione t di Student, tCdf( ) 165
prodotto (P)
    modello di 9
prodotto vettoriale, crossP( ) 37
prodotto, product( ) 123
prodotto, Π( ) 200
prodSeq() 122
product( ), prodotto 123
programmazione
    definisci programma, Prgm 122
    display data, Disp 48
    passa errore, PassErr 118
    visualizza dati, Vis 142
programmi
    definizione di libreria privata 44
    definizione di libreria pubblica 44

programmi e programmazione
cancella errore, ClrErr 28
fine tentativo, EndTry 168
tentativo, Try 168
visualizza schermata I/O, Vis 142
visualizza schermo I/O, Disp 48
propFrac, frazione propria 123
punto
addizione, .+ 191
divisione, .P 192
moltiplicazione, .* 192
sottrazione, .N 192
su potenza, .^ 193

Q

QR, scomposizione QR 124
QuadReg, regressione quadratica 124
quando, when( ) 178
QuartReg, regressione quartica 125

R

R, radianti 204
R▸Pr( ), coordinata polare 127
R▸Pθ( ), coordinata polare 127
radianti, R 204
radice ennesima
modello di 5
radice quadrata
modello di 5
radice quadrata, √( ) 155, 199
rand( ), numero casuale 128
randBin, numero casuale 128
randInt( ), intero casuale 128
randMat( ), matrice casuale 129
randNorm( ), normale casuale 129
randPoly( ), polinomio casuale 129
randSamp( ) 130
RandSeed, seme numero casuale 130
real( ), reale 130
reale, real( ) 130
reciproco, ^⁻¹ 208
ref( ), forma a scalini per righe 131

regressione cubica, CubicReg 39
regressione esponenziale, ExpReg 56
regressione lineare, LinRegAx 83
regressione lineare, LinRegBx 82, 84
regressione logaritmica, LnReg 89
regressione logistica, Logistic 92
regressione logistica, LogisticD 93
regressione quadratica, QuadReg 124
Regressione quartica (QuartReg) 125
regressione sinusoidale, SinReg 153
regressione su potenza, PowerReg 120-121, 133, 135, 165
regressioni
cubica, CubicReg 39
esponenziale, ExpReg 56
linea mediana-mediana, MedMed 98
logaritmica, LnReg 89
Logistic (Regressione logistica) 92
logistica, Logistic 93
MultReg (Regressione lineare multipla) 102
quadratica, QuadReg 124
quartica, QuartReg 125
regressione lineare, LinRegAx 83
regressione lineare, LinRegBx 82, 84
regressione su potenza, PowerReg 121
sinusoidale, SinReg 153
regressions, regressioni
regressione su potenza, PowerReg 120, 133, 135, 165
remain( ), resto 133
Request 133
RequestStr 135
restituisci, Return 135
resto, remain( ) 133
Return, restituisci 135
right( ), destra 136
right, right( ) 52, 77, 136, 178
rimuovi
elementi vuoti dalla lista 46
ripeti, Cycle 40
risultati a due variabili, TwoVar 173
risultati, statistica 156
risultato (ultimo), Ans 16
rk23( ), Runge Kutta function 136
rotate( ), ruota 138

round( ), arrotondamento 139
rowAdd( ), somma di righe di matrice 140
rowDim( ), dimensione righe matrice 140
rowNorm( ), norma righe matrice 140
rowSwap( ), inverti righe matrice 140
rref( ), Forma a scalini ridotta per righe 141
ruota, rotate( ) 138

S

sblocco di variabili e gruppi di variabili 176
scelte rapide della tastiera 214
scelte rapide, tastiera 214
scomposizione QR, QR 124
se, If 73
sec^{-1}( ), secante inversa 142
sec( ), secante 141
$sech^{-1}$( ), secante iperbolica inversa 142
sech( ), secante iperbolica 142
segno negativo, introduzione di numeri negativi 217
segno, sign( ) 148
seno, sin( ) 150
seq( ), sequenza 143
seqGen( ) 144
seqn( ) 144
sequence, seq( ) 144
sequenza, seq( ) 143
setMode( ), imposta modo 145
shift( ), sposta 147
sign( ), segno 148
simult( ), equazioni simultanee 149
sin^{-1}( ), arcoseno 151
sin( ), seno 150
$sinh^{-1}$( ), arcoseno iperbolico 152
sinh( ), seno iperbolico 152
sinistra, left( ) 81
SinReg, regressione sinusoidale 153
sistema di equazioni (2 equazioni)
    modello di 7
sistema di equazioni (N equazioni)
    modello di 7
somma cumulativa, cumulativeSum( ) 40
somma dellinteresse pagato 201
somma di capitale versato 202

somma, sum( ) 160
somma, Σ( ) 201
sommatoria (G)
modello di 9
SortA, ordinamento ascendente 154
SortD, ordinamento discendente 155
sostituzione con loperatore "|" 208
sottomatrice, subMat( ) 160-161
sottrazione, - 187
sposta, shift( ) 147
sqrt( ), radice quadrata 155
stat.results 156
stat.values 157
statistica
combinazioni, nCr( ) 106
deviazione standard, stdDev( ) 157-158, 176
disposizioni semplici, nPr( ) 112
fattoriale, ! 198
media, mean( ) 97
mediana, median( ) 97
normale casuale, randNorm( ) 129
risultati a due variabili, TwoVar 173
seme numero casuale, RandSeed 130
statistiche a una variabile, OneVar 114
varianza, variance( ) 176
statistiche a una variabile, OneVar 114
stdDevPop( ), deviazione standard della popolazione 157
stdDevSamp( ), deviazione standard del campione 158
string( ), da espressione a stringa 159
stringa
dimensione, dim( ) 47
lunghezza 47
stringa di caratteri, char( ) 25
stringa formato, format( ) 60
stringhe
aggiunge, & 198
allinterno, InString 76
codice di carattere, ord( ) 117
conversione indiretta, # 203
da espressione a stringa, string( ) 159
da stringa a espressione, expr( ) 56
destra, right( ) 136
dimensione, dim( ) 47

formato, format( ) 60
formattazione 60
in mezzo alla stringa, mid( ) 99
ruota, rotate( ) 138
sinistra, left( ) 81
sposta, shift( ) 147
stringa di caratteri, char( ) 25
utilizzo per creare nomi di variabile 217
strings
right, right( ) 52, 77, 136, 178
subMat( ), sottomatrice 160-161
sum( ), somma 160
sumIf( ) 161
sumSeq() 161

T

T, trasposizione 162
tabella di ammortamento, amortTbl( ) 11, 20
$\tan^{-1}$( ), arcotangente 163
tan( ), tangente 162
tangente, tan( ) 162
$\tanh^{-1}$( ), arcotangente iperbolica 164
tanh( ), tangente iperbolica 164
tasso di variazione media, avgRC( ) 19
tasso effettivo, eff( ) 50
tasso nominale, nom( ) 109
tCdf( ), probabilità di distribuzione t di Student 165
test per nullità, isVoid( ) 80
Test_2S, verifica F su due campioni 63
Text, comando 165
tInterval, intervallo di confidenza t 166
tInterval_2Samp, intervallo di confidenza t su due campioni 167
tPdf( ), densità di probabilità t di Student 168
trace( ) (Traccia) 168
trasposizione, T 162
trattino basso, _ 207
Try, comando gestione errore 168
tTest, verifica t 169
tTest_2Samp, verifica t su due campioni 170
tvmFV( ) 171
tvmI( ) 172
tvmN( ) 172

tvmPmt( ) 172
tvmPV( ) 173
TwoVar, risultati a due variabili 173

U

uguale, = 194
unitV( ), vettore unità 175
unLock, sblocca variabile o gruppo di variabili 176

V

vai a, Goto 72
valore assoluto
    modello di 8
valore del denaro rapportato al tempo, importo della rata 172
valore del denaro rapportato al tempo, interesse 172
valore del denaro rapportato al tempo, numero di rate 172
valore del denaro rapportato al tempo, valore futuro 171
valore del denaro rapportato al tempo, valore presente 173
valore presente netto, npv( ) 113
valori dei risultati, statistica 157
variabile
    creazione di un nome da una stringa di caratteri 217
variabile locale, Local 90
variabili
    cancella tutte con il nome di un solo carattere 28
    elimina, DelVar 45
    locale, Local 90
variabili e funzioni
    copia 30
variabili, blocco e sblocco 68, 91, 176
varianza, variance( ) 176
varPop( ) 176
varSamp( ), varianza campione 176
verifica numero primo, isPrime( ) 79
Verifica t sulla regressione lineare multipla 103
verifica t, tTest 169
vettore unità, unitV( ) 175
vettori
    prodotto scalare, dotP( ) 48
    prodotto vettoriale, crossP( ) 37
    unità, unitV( ) 175
    visualizza vettore in forma cilindrica, ▸Cylind 41
Vis, visualizza dati 142

visualizza come
angolo decimale, ▸DD 42
binario, ▸Base2 21
esadecimale, ▸Base16 23
gradi/primi/secondi, ▸DMS 48
numero decimale, ▸Base10 22
vettore in forma cilindrica, 4Cylind 41
vettore polare, ▸Polar 119
vettore rettangolare, ▸Rect 130
vettore sferico, ▸Sphere 155
visualizza come vettore rettangolare, ▸Rect 130
visualizza dati, Disp 48
visualizza dati, Vis 142
visualizza gradi/primi/secondi, ▸DMS 48
visualizza vettore in forma cilindrica, ▸Cylind 41
visualizza vettore sferico, ▸Sphere 155

W

warnCodes( ), Warning codes 178
when( ), quando 178
While, mentre 179

X

x^2, quadrato 191
XNOR 197
xor, or esclusivo booleano 179

Z

zInterval, intervallo di confidenza z 180
zInterval_1Prop, intervallo di confidenza z per una proporzione 181
zInterval_2Prop, intervallo di confidenza z per due proporzioni 182
zInterval_2Samp, intervallo di confidenza z su due campioni 182
zTest 183
zTest_1Prop, verifica z per una proporzione 184
zTest_2Prop, verifica z per due proporzioni 185
zTest_2Samp, verifica z su due campioni 186

Δ

Δlist( ), differenza in una lista 88

X

χ^2Cdf( ) 26
χ^2GOF 27
χ^2Pdf( ) 28