



# Parla più FORTH

di Raffaello De Masi

## Seconda parte

### Impariamo qualche parola

Nell'articolo precedente abbiamo visto parlando della RPN alcune semplici word, vale a dire +, -, ., che sono, per definizione, vere e proprie parole. Le altre due parole, che completano le 4 operazioni fondamentali sono la moltiplicazione (\*) e la divisione (/).

Queste quattro operazioni funzionano, rispettando le regole della RPN, come in qualsiasi altro linguaggio, ricordando, comunque, che il risultato finale non viene visualizzato, ma posto nel TOS a meno di non usare il ".". Ad esempio l'operazione:

5 \* 3 - 8 : 4

diviene

5 3 \* 8 4 / - .

con risultato

13 OK

Negli esempi che seguiranno useremo solo ed esclusivamente numeri interi ed i risultati saranno altrettanto numeri interi. Vedremo in un secondo tempo il perché.

Fin qui nulla di speciale. Il Forth possiede, però, una serie di operatori addizionali, sempre riferiti alle quattro operazioni, che risultano estremamente utili per eseguire una serie di operazioni addizionali.

Esiste un gruppo di word che operano direttamente sullo stack e sul suo contenuto. Cominciamo a vederne qualcuna.

#### La parola MOD

Questa parola peraltro presente in altri linguaggi, esegue la normale operazione di divisione ma lascia nel TOS il resto invece del risultato. Ad esempio:

32 5 MOD .

darà

2 OK

Vale a dire il resto (intero) della divisione tra 32 e 5.

#### La parola / MOD (Slash - Mod)

riunisce, come è intuitivo le caratteristiche delle due parole, lasciando nello stack sia il risultato che il resto. Ad esempio:

Stack	/MOD	Stack
32		6
5		2

Notare come il risultato sia posto in TOS ed il resto sia al 2° posto per cui digitando

si avrà

6 2 OK

Un'altra word estremamente utile è la \* (notare l'assenza di spazio tra gli operandi). Necessita della presenza di tre numeri

nello stack; esegue il prodotto del 3° numero per il secondo e divide il risultato per il primo. Come si vede, la sequenza operativa è completamente diversa dalle due operazioni \* e / separate tra di loro e non è esattamente simile (proprio per il fatto di operare su numeri interi) alla sequenza / \*

Esempio 3 6 2 \*/ . Return  
9 OK

Nel caso invece di operazioni con resto si avrebbe

7 7 5 \*/ . Return  
9 OK

mentre

7 7 5 / \* . Return  
7 OK

Word simile alla precedente è \*/MOD (sempre senza spazi separatori); essa esegue le stesse operazioni lasciando in TOS il quoziente ed al 2° posto il resto.

Schematicamente (ed è questa la notazione che useremo d'ora in poi per indicare la sequenza di operazioni e le modifiche che avvengono nello stack) si può rappresentare

a b c --- rem quot

dove a,b,c, ... n indicano un numero, --- l'operatore (nel caso, \*/MOD e ciò che segue lo stato dello stack dopo l'operazione.

L'operazione appena descritta

7 7 5 \*/MOD . . RETURN

darà

9 4 OK

Va posta attenzione alla spaziatura. L'interprete Forth ignora gli spazi superiori ad 1 ma, evidentemente, uno spazio compreso nella parola \*/MOD cambierebbe completamente il risultato. Inoltre, per praticità, sottintenderemo la pressione del tasto RETURN tranne in casi in cui tale mancanza ingeneri confusione.

Consideriamo adesso l'espressione

( a - b ) / c

con a b c già inseriti nello stack in qualsivoglia ordine.

Per quanto ci si sprema le meningi (a meno di non voler reinserire in forma differita i numeri nello stack) non è possibile risolvere l'espressione.

Occorre pertanto avere a disposizione dei comandi che consentono di eseguire spostamenti e modifiche dei valori presenti nello stack. Questi operatori, chiamati appunto manipolatori di stack, sono estremamente utili e meritano di essere descritti

dettagliatamente uno per uno, non avendo corrispondente in altri linguaggi evoluti.

La parola SWAP scambia di posto i due valori sommitali presenti nello stack. Ad esempio:

```
3 5 SWAP . .
```

darà

```
3 5
```

la semplice equazione appena descritta si risolverà

```
c a b - SWAP / .
```

infatti

Stack	-	SWAP	/
b	(a-b)	c	c/(a-b)
a	c	(a-b)	
c			

e numericamente

```
a=10 b=4 c=2
2 10 4 - SWAP / .
3 OK
```

con

Stack iniziale	-	SWAP	/
4	6	2	3 3 OK
10	2	6	
2			

La parola DUP duplica il TOS spostando tutta la catasta in giù di un posto.

Esempio:

Stack iniziale	DUP
3	3
5	3
8	5
	8

O, il che è lo stesso

```
3 16 27 DUP . . .
27 27 16 3 OK
```

Un esempio pratico della parola DUP può essere dato dall'elevamento a potenza intera di un numero. Ad esempio (ne approfittiamo per ricordare la definizione di nuove word)

```
: CUBO DUP DUP * * . ;
OK
```

e digitando

```
5 CUBO si avrà
125 OK
```

Facciamo un nuovo esempio: si abbia l'espressione

$a \cdot (a + b)$  con  $a$  e  $b$  (TOS) già inseriti nello stack

Abbiamo bisogno di due copie della  $a$  ma questa è sotto la  $b$ . Bisognerebbe avere

una  $a$  in TOS. È possibile farlo con un'altra parola; OVER.

La parola OVER esegue una copia del secondo numero presente nello stack e pone in TOS, spingendo la catasta in giù di un posto.

Avremo così

Stack iniziale	OVER
b	a
a	b
	a

numericamente avremo per  $a=5$  e  $b=3$

```
5 3 OVER + * .
```

Ovvero

Stack iniziale	OVER	+	*	.
3	5	8	40	40 OK
5	3	5		
	5			

Ancora una word, ROT. Essa consente di prelevare il terzo termine dello stack e lo pone in TOS, spingendo in giù i primi due valori. Tanto per intenderci:

Stack iniziale	ROT
5	13
8	5
13	8
21	21

La parola DROP, che rappresenta l'ultimo operatore di stack che presenteremo per adesso, elimina il TOS facendo salire la catasta di un posto. Esempio:

Stack	DROP
3	5
5	8
8	-
-	-

Esistono ancora alcuni comandi cosiddetti doppi, vale a dire che manipolano i valori dello stack a coppia. Per la verità sarebbe più preciso dire che manovrano i byte a coppie (questi comandi furono approntati per lavorare su numeri in doppia precisione), ma per adesso è sufficiente immaginare che operino su valori in tandem.

Gli operatori seguenti funzionano esattamente allo stesso modo di quelli appena descritti, con la differenza di operare su coppie di numeri (o, come vedremo in seguito, su numeri doppi) invece che su numeri singoli. Essi sono

Word	Operaz. eseguite	Commenti
2SWAP	d1 d2--d2 d1	Scambia le due coppie di numeri presenti in sommità dello stack.

Esempio:

Stack	2SWAP	Stack
5	8	
13	6	8 6 5 13
8	5	
6	13	
2DUP	d--d d	duplica la coppia di numeri presenti alla sommità dello stack.

Esempio:

Stack	2DUP	Stack
5		5
8		8
		5
		8
2OVER	(d1 d2 -- d1 d2 d1)	duplica la seconda coppia di numeri e la pone nello stack.

Esempio:

Stack	2OVER	Stack
7		12
5		8
12		7
8		5
		12
		8
2DROP	(d --)	cancella la coppia sommitale di numeri.

Con le parole in possesso è possibile eseguire un grande numero di operazioni coinvolgendo solo lo stack. Ad esempio la espressione:

```
a^2 + ab + c
con in stack
b (TOS)
a
c
```

può essere risolta compilando la parola

```
: RISOLVI OVER + * + ;
```

infatti

Stack iniz.	OVER	+	*	+
b	a	(a+b)	a(a+b)	a(a+b)+c
e	b	a	c	
c	a	c		
	c			

per cui:

3 5 8	RISOLVI	Stack	RISOLVI
8		5 13 65 68 68	OK
5		8 5 3	
3		5 3	
		3	

oppure:

(a + b)/(a-b) con in stack b(TOS)

```

      a
      : SOLUZIONE 2 DUP - ROT ROT + / ;

```

la cui soluzione passo passo lasciamo ai lettori.

Prima di concludere con l'argomento operatori immediati c'è ancora da precisare come è possibile ricevere sullo schermo un messaggio oltre che dei risultati. La cosa è molto semplice: basta includere il messaggio tra virgolette facendolo immediatamente precedere dal punto. Il compilatore comprende che è necessario visualizzare il messaggio presente dopo il punto. Poiché le virgolette iniziali sono a tutti gli effetti una word è necessario che siano seguite da uno spazio così:

```

3 5 2DUP ." il prodotto di " . " e "
      ." è " * .

```

darà (STARE BENE ATTENTI ALLE SPAZIATURE!)

il prodotto di 3 e 5 è 15

Oppure definendo una nuova word

```

: MOLTIPLICAZIONE 2 DUP . " ; il prodotto
dei due numeri precedenti dà " * . ;

```

per cui

```

7 21 MOLTIPLICAZIONE darà

```

```

7 21 ; il prodotto dei due numeri preceden-
ti dà 147

```

Anche qui valgono le regole di composizione delle word. Ad esempio, utilizzando ancora la word CR (carriage return):

```

: PRIMOVERSO ."Oh, che bel castello " CR ;
: SECONDOVERSO ."Marcondirondera " CR ;
: TERZOVERSO ."Marcondironda' " CR ;
: POESIA PRIMOVERSO SECONDOVERSO
PRIMOVERSO TERZOVERSO ;

```

darà la prima strofa della famosa filastrocca (un po' di pazienza e tra un po' di tempo vi daremo anche la musica), seguita dall'imperturbabile OK.

A questo punto dovrebbe essere abbastanza chiaro tutto ma abbiamo accumulato tanto di quel ciarpame di word che sarebbe opportuno fare un po' di pulizia. In Forth è possibile cancellare le parole già esistenti con la nuova word, FORGET, seguita da un'altra word già definita. Attenti però: FORGET cancella dalla parola che la segue fino all'ultima definita. Ad esempio, nel caso precedente:

```
FORGET SECONDOVERSO
```

cancellerà dal vocabolario le parole SECONDOVERSO e TERZOVERSO ma non PRIMOVERSO.

È pertanto buona norma inserire, prima o durante la definizione delle word, delle

parole chiave senza alcun effetto, che hanno la funzione di pietre miliari nel programma.

In pratica, nei punti cruciali è opportuno inserire dei flag che, all'occorrenza, consentono di cancellare solo spezzoni del vocabolario.

Facciamo un esempio (il calcolatore è stato appena acceso e non esistono parole già definite) con la solita filastrocca:

```

: GIOVANNI ;
: PRIMOVERSO ."Oh, che bel castello " CR ;
: SECONDOVERSO ."marcondirondera " CR ;
: TERZOVERSO ."marcondirondà " CR ;
: PRIMASTROFA PRIMOVERSO SECONDO-
VERSO
PRIMOVERSO TERZOVERSO ;

```

```
: NICOLA ;
```

```

: QUARTOVERSO ."è più bello il nostro" CR ;
: SECONDASTROFA QUARTOVERSO SE-
CONDOVERSO QUARTOVERSO TERZOVER-
SO ;
: FILASTROCCA PRIMASTROFA SECONDA-
STROFA ;

```

tutte le definizioni, a meno di errori di digitazione saranno seguite da imperturbabili OK.

Orbene battiamo FILASTROCCA seguito dal solito RETURN

Sul visore avremo:

```

Oh che bel castello
marcondirondera
Oh che bel castello
marcondirondà
è più bello il nostro
marcondirondero
è più bello il nostro
marcondirondà
OK

```

Proviamo adesso a battere

```
FORGET NICOLA
```

che darà il nuovo OK

Se battiamo FILASTROCCA ci capiterà di vedere

```
FILASTROCCA ?
```

mentre battendo PRIMASTROFA avremo solo i primi quattro versi della poesio-la.

Chiara, vero, l'utilità di questi indicatori? È opportuno che almeno uno venga inserito all'inizio così da ripulire agevolmente il dizionario alla fine di un'operazione (ad onor del vero esiste una word già implementata (EMPTY) che cancella tutto il vocabolario definito dall'utente) ma visto che il FORTH è un linguaggio che lascia la massima libertà al programmatore, non vi pare più originale lavorare di testa pro-

pria? Ad esempio, si può utilizzare come segnalatore una parola facilmente ricordabile (ad esempio il nome del calcolatore su cui si sta lavorando, eventualmente seguito da numeri progressivi); chi scrive usa il nome della moglie e non immaginate il piacere che si prova a battere il FORGET seguito dal nome della consorte.

Ci fermiamo qui con la definizione di nuove word, riservandoci di seguire il discorso più tardi. È opportuno però precisare, prima di chiudere, che una parola già esistente può essere ridefinita.

Ad esempio:

```

: ASSIOMA DUP 2DUP . " per " . " fa " * . ;
3 ASSIOMA

```

3 per 3 fa 9

Se effettuiamo una ridefinizione

```

: ASSIOMA DUP 2DUP 2DUP . " mogli + " .
." suocere accoppiano " * * * . " mariti " ;

```

avremo come messaggio

```
ASSIOMA NOT UNIQUE OK
```

che ci avvisa come esistono più word eguali nel vocabolario. L'OK comunque ci fa intendere che la parola è stata comunque accettata.

Infatti:

```
3 ASSIOMA
```

3 mogli + 3 suocere accoppiano 81 mariti OK

a segno che viene utilizzata solo l'ultima definizione.

Battendo FORGET ASSIOMA viene cancellata solo l'ultima definizione e quella precedente ridiventa operativa; vale a dire:

```
5 ASSIOMA
```

darà:

```
5 per 5 fa 25 OK
```

Un'ultima considerazione brevissima: è possibile ridefinire anche parole già esistenti nel vocabolario di base.

Ad esempio:

```
: - + ;
```

darà:

```
- NOT UNIQUE OK
```

e

```
13 5 -
```

darà:

```
18 OK
```

È evidente la pericolosità dell'operazione per cui se ne sconsiglia l'uso a meno di una improbabilissima necessità.

Bene, digerite con calma, ci rivediamo presto!



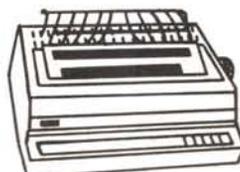
**Verbatim**



Minidisks



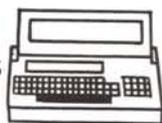
**Diablo**



Stampanti

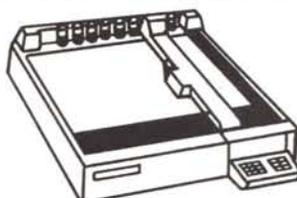
**EPSON**

Computers



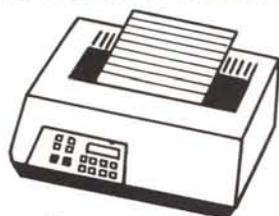
Stampanti

**GRAPHTEC**



Plotters / Digitizers

**datasouth**



Stampanti

*Esprit*



Terminali Video

**Olympia**



Stampanti a margherita

**SCELTE OBBLIGATE  
PER COMBINAZIONI  
VINCENTI.**

**TREPIU'**

INFORMATICA VIDEO TELEMATICA

TRE PIU' s.r.l. 00199 Roma, Via Asmara 58  
Tel. 06-8395766 - 8394458 - Telex 616130

Per ulteriori informazioni e per conoscere l'indirizzo del  
RIVENDITORE più vicino, rivolgersi a TREPIU'

# PROGRAMMA 2000!

**IL BIMESTRALE DI 20 PROGRAMMI SU CASSETTA!!**

**NASTRO + MANUALE IN ITALIANO A LIT. 20.000,  
SOLO 1.000 LIRE A PROGRAMMA!!**

THE BEST FOR YOU!

## COMMODORE 64

In questa raccolta 20 fantastici programmi!  
Vi ricordiamo la WORKIT: 14 comandi aggiunti al basic residente per il debug e la programmazione grafica, USER-CALC: Una versione Programma 2000 del famoso foglio elettronico, SIMULA: un completo simulatore di volo, STREET: una nuova ed inedita avventura sul computer, SPEED-CAR: una appassionante gara automobilistica ad alta velocità, PAINT: il simulatore di tavoletta grafica e tanti altri favolosi programmi.

## SPECTRUM 16-48

20 programmi di giochi, didattica, grafica e utilità. In questo numero BONUS: calcolate il costo del vostro premio assicurativo, IVA: calcola i valori IVA con lo scorporo, ALIEN: un extraterrestre vi insegue in un difficile percorso, TAB: un programma didattico per insegnare le tabelline ai bambini, MORSE: i segreti del famoso alfabeto insegnati dallo spectrum e tanti altri programmi.

## TEXAS TI 99/4A

La grande sorpresa programma 2000! La raccolta bimestrale di 20 programmi per utenti TI99/4A. Nel primo numero TOTO: pronostici e sviluppo colonne per il totocalcio, GRAF: gestione delle figure su tre dimensioni, FILETTO: il famoso gioco in versione computer, DECIDERE: un aiuto dal vostro TI99/4A per prendere decisioni, BILANCIO: tenete sotto controllo la vostra contabilità personale e tanti altri fantastici programmi.

ECCEZIONALE!! I PROGRAMMI SONO LISTABILI E DI QUALITÀ. UN APPOSITO SERVIZIO DI ASSISTENZA RISPONDERÀ ALLE VOSTRE DOMANDE PER QUALSIASI PROBLEMA INERENTE ALL'USO DEI PROGRAMMI.

Spedire in busta chiusa a:  
PROGRAMMA 2000  
VIA G. FELICI, 20  
00144 ROMA

Nome e Cognome: .....  
Indirizzo: .....  
Città: ..... CAP: .....  
Computer: ..... N. copie: .....

allego pagamento     pagherò alla consegna