a cura di Pierluigi Panunzi

In questo numero ci occupiano di due programmi prettamente matematici entrambi realizzati da Andrea Cantadori di Parma: il primo permette di effettuare moltiplicazioni e divisioni tra polinomi, fornendo anche il resto nel secondo caso.

Il secondo programma invece consente di calcolare il prodotto tra numeri fino a 90 cifre.

Moltiplicazione e divisione di polinomi

di Andrea Cantadori (Parma)

Permette di dividere e moltiplicare tra loro due polinomi tali che la somma dei loro gradi massimi sia al massimo 52, dando anche il resto nel caso si tratti della divisione.

Il listato è adatto alla TI-59. Tuttavia, variando gli indirizzi alle memorie 59, 58, ..., 53 che compaiono nel listato stesso con i corrispondenti 29, 28, ..., 23 (cioè mettendo ad esempio RCL 28 al posto di RCL 58), si dovrebbe avere la compatibilità anche con le TI-58, o almeno credo....

Ovviamente in questo caso la somma dei gradi massimi dei polinomi dovrà essere inferiore a 23. Viceversa, qualora si disponga di una 59, e si abbia bisogno di una maggiore capacità di calcolo, si potrà operare una sostituzione degli indirizzi di memoria 59, 58, ..., 53 con gli indirizzi 89, 88, ..., 83, impostando benintenso la ripartizione di memoria corrispondente a 9 Op 17; in tal caso il limite massimo sale a 83.

Spero di essermi spiegato, perché ora passerò a descrivere il programma in maggiori dettagli. Innanzitutto non credo che l'algoritmo usato abbia bisogno di essere interpretato, perché si tratta proprio dell'algoritmo "classico" che alle scuole medie abbiamo tutti imparato. Con un po' di attenzione la comprensione del listato non dovrebbe risultare difficile, onde non mi dilungherò in noiose analisi dello stesso. Dirò soltanto che il programma può essere distinto in tre blocchi principali:

a) la sezione di input dati (A-B-C)
b) la sezione "moltiplicazione" (E)
c) la sezione "divisione" (D-D').

E veniamo all'uso; supposti dati i due polinomi nella forma seguente:

$$a_o+a_1x+a_2x^2+\ldots+a_nx^n$$
$$b_o+b_1x+b_2x^2+\ldots+b_mx^m$$

dovrà essere $m \leq n$ nel caso in cui intendiamo dividere il primo polinomio per il secondo; se invece eseguiamo la moltiplicazione, ciò non ha importanza.

A questo punto:

a) *impostare n e premere A*
b) *impostare m e premere B*
c) impostare gli "a" e i "b" in ordine di potenza crescente; in altre parole, occorre impostare nell'ordine i coefficienti:

$a_o, a_1, \ldots, a_n, b_o, b_1, \ldots, b_m$

facendo uso del tasto C (vedi oltre, nell'esempio)

d) *per dividere*: si otterrà il coefficiente del termine di grado (n-m) premendo D, e i coefficienti successivi premendo R/S. Il display, lampeggiando una fila di 9, indicherà il termine della divisione.

e) *per il resto della divisione*: premere D' per ottenere il termine noto del polinomio resto, che avrà grado minore di m, ovviamente. Per i coefficienti successivi premere D'.

N.B.: poiché non è stata inserita una routine per indicare il termine dell'output dei coefficienti del resto, occorrerà fermarsi non appena si siano ottenuti i primi m

Prodotti e quozienti di polinomi

```
000  76 LBL   054  72 ST*   108  29 CP    162  53  53
001  11  A    055  55  55    109  43 RCL   163  91 R/S
002  85  +    056  01  1     110  56  56    164  25 CLR
003  01  1    057  22 INV    111  42 STO   165  42 STO
004  95  =    058  44 SUM    112  57  57    166  53  53
005  42 STO   059  55  55    113  01  1     167  01  1
006  59  59   060  22 INV    114  44 SUM   168  44 SUM
007  92 RTN   061  44 SUM    115  57  57    169  56  56
008  76 LBL   062  57  57    116  43 RCL   170  43 RCL
009  12  B    063  43 RCL    117  59  59    171  59  59
010  85  +    064  54  54    118  85  +     172  75  -
011  01  1    065  77  GE    119  01  1     173  43 RCL
012  95  =    066  00  00    120  85  +     174  56  56
013  42 STO   067  74  74    121  43 RCL   175  75  -
014  58  58   068  01  1     122  54  54    176  01  1
015  92 RTN   069  44 SUM    123  95  =     177  95  =
016  76 LBL   070  54  54    124  42 STO   178  22 INV
017  13  C    071  61 GTO    125  55  55    179  77  GE
018  69 OP    072  00  00    126  73 RC*   180  01  01
019  20  20   073  45  45    127  57  57    181  85  85
020  72 ST*   074  25 CLR    128  65  ×     182  61 GTO
021  00  00   075  42 STO    129  73 RC*   183  01  01
022  92 RTN   076  54  54    130  55  55    184  09  09
023  76 LBL   077  01  1     131  95  =     185  01  1
024  14  D    078  22 INV    132  44 SUM   186  22 INV
025  43 RCL   079  44 SUM    133  53  53    187  44 SUM
026  58  58   080  59  59    134  01  1     188  56  56
027  32 X:T   081  43 RCL    135  22 INV   189  44 SUM
028  43 RCL   082  58  58    136  44 SUM   190  54  54
029  59  59   083  32 X:T    137  57  57    191  43 RCL
030  42 STO   084  43 RCL    138  44 SUM   192  54  54
031  55  55   085  59  59    139  55  55    193  75  -
032  43 RCL   086  22 INV    140  43 RCL   194  43 RCL
033  00  00   087  77  GE    141  57  57    195  58  58
034  42 STO   088  00  00    142  67  EQ    196  95  =
035  57  57   089  93  93    143  01  01    197  77  GE
036  73 RC*   090  61 GTO    144  61  61    198  00  00
037  59  59   091  00  00    145  43 RCL   199  93  93
038  55  ÷    092  25  25    146  58  58    200  61 GTO
039  73 RC*   093  25 CLR    147  85  +     201  01  01
040  57  57   094  42 STO    148  43 RCL   202  09  09
041  95  =    095  00  00    149  59  59    203  00  0
042  42 STO   096  35 1/X    150  75  -     204  00  0
043  56  56   097  91 R/S    151  43 RCL   205  00  0
044  91 R/S   098  76 LBL    152  55  55    206  00  0
045  73 RC*   099  19 D'     153  95  =     207  00  0
046  55  55   100  69 OP    154  22 INV
047  75  -    101  20  20    155  77  GE
048  73 RC*   102  73 RC*    156  01  01    001  11  A
049  57  57   103  00  00    157  61  61    009  12  B
050  65  ×    104  91 R/S    158  61 GTO   017  13  C
051  43 RCL   105  68 NOP    159  01  01    024  14  D
052  56  56   106  76 LBL    160  26  26    099  19 D'
053  95  =    107  15  E     161  43 RCL   107  15  E
```

termini; in caso contrario, il display mostrerà uno dopo l'altro tutti i contenuti dei vari registri dati.

f) *per moltiplicare*: premere E per ottenere il primo coefficiente, ed R/S per i successivi; essi saranno in ordine di potenza crescente, a partire dal termine noto. Analogamente a prima, il display mostrerà lampeggiando il termine dell'elaborazione.
ESEMPIO: si vuole svolgere la seguente espressione:

$$\frac{(1+X+X^2+X^3+2X^4-3X^5)\,(1+X+0X^2-3X^3)}{4X^3-5X^2+6X-10}$$

Dapprima valutiamo il numeratore; procediamo come segue:

a) impostiamo 5 (il grado del primo polinomio) e premiamo A

b) impostiamo 3 (il grado del secondo polinomio) e premiamo B

c) impostare uno alla volta, premendo C, i seguenti coefficienti:

1, 1, 1, 1, 2, -3, 1, 1, 0, -3

d) premendo dapprima E e poi, successivamente, R/S, si otterranno i seguenti valori:

1, 2, 2, - 1, 0, -4, -6, -6, 9

e pertanto si deve concludere che il numeratore vale:

$1+2X+2X^2-X^3+0X^4-4X^5-6X^6-6X^7+9X^8$

Premiamo poi 2nd CMs ed eseguiamo la divisione, impostando i dati come in precedenza:

a) impostare 8 e premere A

b) impostare 3 e premere B

c) impostare uno alla volta, usando il tasto C, i seguenti coefficienti:

1, 2, 2, -1, 0, -4, -6, -6, 9, -10, 6, -5, 4

d) premiamo D e successivamente R/S. Otterremo i seguenti valori, approssimando alla terza cifra decimale:

2.250, 1.312, -3.234, -1.387, 6.399, 1.743

e pertanto si deve concludere che il quoziente è:

$1.743+6.399X-1.387X^2+$
$-3.234X^3+1.312X^4+2.250X^5$

e) per il resto, premendo ripetutamente D' otteniamo, sempre approssimando alla terza cifra decimale:

18.434, 55.534, -41.547

ed il resto pertanto è:

$18.434+55.534X-41.547X^2$

Da notare che ci siamo fermati correttamente al terzo termine, cioè al termine di numero ordinale pari al grado massimo del polinomio divisore. Questo consente di ot-

L'ANGOLO DELLE TI

Anteprima ... mondiale: "Synthetic Programming" per TI-58, 58C e 59

Prova che ti riprova, tanta tenacia è stata premiata. Stiamo parlando dell'ormai proverbiale tenacia di tal Stefano Laporta di Bologna il quale è oramai giunto ad una svolta decisiva nella ricerca dei segreti delle calcolatrici Texas Instruments: ha trovato il metodo di generare qualsiasi codice di funzione, ben al di là delle possibilità finora note delle TI.

Dimenticavamo di dire che i codici in questione sono esadecimali!!!

Già lo scorso numero avevamo dato un "assaggio" dell'eccezionale scoperta, a quanto ci consta, in "assoluta anteprima mondiale".

Data la novità dell'argomento, quanto è riportato in questa rubrica non può essere liberamente riprodotto o divulgato senza il consenso della redazione e dovrà essere inoltre segnalata la provenienza delle informazioni...

Prima di passare la parola al nostro lettore, desideriamo sottolineare che tutte le notizie riportate sono applicabili indistintamente ai tre modelli di calcolatrici, 58, 58C, 59, con l'unica differenza, salvo diversamente riportato, riguardante la ripartizione iniziale della calcolatrice: 3 Op 17 per le TI-58 e TI-58C, 9 Op 17 per le TI-59.

Ecco quanto ci scrive il nostro Laporta:

"Ho trovato il modo di generare ben 60 NUOVI CODICI ESADECIMALI sulle TI!!!

Nell'ultimo numero di MC ho parlato di come generare due codici, il 12 ed il 32 (ossia 0c e 2c) mediante sequenze lunghissime ed in punti particolari della memoria; ora ho superato questi problemi ed ho realizzato una vera e propria 'Synthetic Programming on TI-58, 58C, 59'.

Passo ora alla descrizione dettagliata di quanto ho scoperto".

1) Caratteristiche principali dei codici

È possibile generare tutti i codici esadecimali aventi la prima cifra compresa tra 0 e 9, ad esempio 1a, 7b, 9f, ma non e2, ab, ecc.

Questi codici possono essere generati in qualunque passo della memoria multiplo di 8, ma a causa di probabili problemi hardware bisogna stare attenti a non spostarli con "Ins" o "Del" altrimenti si trasformano in codici normali.

(N.d.r. Inoltre, almeno secondo le nostre conoscenze attuali, tali codici vengono considerati "normali", quando si registra il programma su scheda, ovviamente per quanto riguarda la TI-59).

2) Visualizzazione

Non essendo disponibili sul visualizzatore le lettere esadecimali "a,...,f" (o no?!) i codici vengono visualizzati come somma fra il valore decimale della cifra hex ed il valore della prima cifra moltiplicata per 10, ovvero "3b" viene visto sul display come $30+11=41$, "7e" come $70+14=84$, ecc.

Se il risultato è maggiore di 100 (decimale), viene trascurato l'"1" più significativo e cioè "9f", che sarebbe $90+15=105$, viene visualizzato come "05".

Ciò naturalmente provoca notevole ambiguità rendendone difficile la corretta interpretazione: ma se il passo di programma può essere trasformato in registro di memoria (oltre il passo 159 nella TI-59) con un'opportuna modifica della ripartizione, allora richiamando tale registro di memoria si otterrà "1E-99" o "9E99", a conferma che il codice è sicuramente esadecimale.

Un'eccezione curiosa: il codice 9a viene visualizzato come 00.

3) Generazione

Descrivo la sequenza che genera qualunque codice hex ai passi 48, 72, 80, 96 e 120; è possibile generarli in altri passi ma la sequenza è leggermente diversa, come vedremo.

Dopo aver predisposto la ripartizione come segnalato, supponiamo di voler mettere al passo NNN un certo codice hex. Si calcola allora come verrà visualizzato tale codice dalla TI e lo si introduce al passo NNN; poi si preme CLR Pgm 19 SBR 045 Dms LRN Ins Ins LRN RST = CLR dove "=CLR" può essere trascurato nelle TI-59.

È un po' più corta delle precedenti, vero?

Faccio un esempio chiarificatore: voglio generare il codice "3c" al passo 048.

"3c" verrà visualizzato come $30+12=42$, dunque imposto il codice normale 42 (cioè STO) al passo 048 con GTO 048 LRN STO BST LRN e poi imposto la sequenza di generazione.

Vediamo cosa abbiamo creato: con GTO 048 LRN vedremo un "42" che però non è più il nostro vecchio STO che avevamo introdotto... Fantastico!

Chi non ne fosse convinto (con la TI58) prema LRN (per tornare in modo di calcolo) e poi GTO SST LRN: con stupore (anche se oramai siamo entrati in un mondo che di normale ha ben poco!) vedrà sul display 7992 28 e per di più con SST si può arrivare al passo 7999! Altro che STO!

Invece sulla TI-59 si avrà un "6" sul display ed entrando in LRN si visualizzerà, in maniera alquanto fugace, il passo 002

tenere come resto appunto un polinomio di grado minore del divisore (ovviamente potrebbe capitare che tale grado del resto sia ancora inferiore, oppure che sia addirittura zero, cioè che il resto si annulli, e questo nel caso che dividendo e divisore siano perfettamente divisibili).

Supermoltiplicazione

di Andrea Cantadori (Parma)

Questo programma consente di moltiplicare fra di loro due numeri di al massimo 50 cifre con la TI-59 e di 20 cifre con la TI-58, anche se tale limite è elevabile rispettivamente a 90 e 30 con una semplicissima modifica, che io non ho qui riportato per maggiore semplicità (ma che dirò poi in cosa consiste). Ecco ora una breve descrizione dell'algoritmo usato, più difficile in realtà a spiegarsi che non ad adoperarsi: supponiamo dati due numeri, ad es.:

$$a_n\ a_{n-1}\ a_{n-2} \ldots a_1$$
$$b_m\ b_{m-1}\ b_{m-2} \ldots b_1$$

dove ogni lettera indica una cifra. Come ci si può rendere conto intuitivamente, con una analisi appena riflettuta dell'algoritmo della moltiplicazione aritmetica, moltiplicare tra loro questi due numeri equivale a moltiplicare tra di loro i seguenti due polinomi:

$$(a_nX^{n-1}+a_{n-1}X^{n-2}+\ldots.+a_1)$$
$$(b_mX^{m+1}+b_{m-1}X^{m-2}+\ldots+b_1)$$

e questo è possibile, e con facilità direi anche, con la routine E del programma precedentemente sviluppato. Si otterrà in generale un polinomio del tipo:

$$c_{m+n}X^{n+m-1}+c_{m+n-1}X^{n+m-2}+\ldots.+c_1$$

dove i coefficienti c_i non saranno però in generale di una sola cifra. In altre parole, il prodotto così sotto forma di polinomio è di difficoltosa interpretazione, e pertanto serve a ben poco. Ecco allora che scatta l'aggiunta propria di questo programma, che riguarda essenzialmente la decodifica dei dati. Essa è inserita dal passo di programma Ø71 al 116.

Supponiamo a questo punto che ogni c_i abbia un certo numero di cifre, ed indichiamole con:

$$di_1\ di_2\ di_3 \ldots di_k$$

Ora, prendiamo c_i, che è il primo risultato della routine E, e mostriamo sul display il suo d_{1k}: esso è l'ultima cifra del numero cercato. Togliendo da c_i il suo d_{1k} resta evidentemente:

$$dl_1\ dl_2\ dl_3 \ldots dl_{k-1}\ 0$$

e questa quantità va sommata a C_2, sul

---

contenente il codice 28: è fugace perché basta andare al passo 1 e poi di nuovo al passo 2 per trovarlo vuoto...

Per di più il visualizzatore è ora posto in "fix 5".

Peggio ancora si ha con la stampante: portiamoci al passo 048, poniamo la printer in TRACE, premiamo GTO (che la stampante riporterà fedelmente) e poi SST che ora la nostra fida PC-100C tradurrà con un misterioso "V2N". Vi ricordate l'istruzione TRL dello scorso numero? E questo non è che l'inizio!

Si può vedere che la decina di passi successiva a 048 è piena di passi piovuti da chissà dove: sono dei codici "residui" del tutto normali che possono essere eliminati. Faccio notare che quando si batte la sequenza "meravigliosa" il contatore di programma si deve trovare al passo in cui vogliamo il codice: dico questo per chi non capisse l'utilità del BST usato nell'esempio.

4) Effetti dei codici

Dato che per la descrizione dei 100 codici "normali" il manuale della Texas impiega circa 300 pagine, per questi 60 codici nuovi dovrei impiegarne 180...

Scherzi a parte, inserirò qui solo alcune considerazioni generali.

Tutti i codici interagiscono con la chiamata delle etichette dando luogo a strani malfunzionamenti; in generale ogni codice ha due funzioni diverse: una è quella del codice stesso, l'altra è data da GTO "codice" che talvolta richiama alcuni programmi del modulo.

Osservando il registro di memoria corrispondente (nel caso del passo 048 ciò vale solo per la TI-58, n.d.r.), la cifra esadecimale corrisponde all'esponente del numero: i conoscitori della TI sapranno che una cifra 8 o 9 in questa posizione dà sul display "9.99 E 99" lampeggiante, qualsiasi siano le altre cifre del numero.

Ora si osserva che in questa posizione:

le cifre	a b	danno -9.99 E 99 lampeggiante
le cifre	c d	danno 1 E 99 lampeggiante
le cifre	e f	danno -1 E 99 lampeggiante

È chiaro che i codici esadecimali sono usati correntemente dalla TI per segnalazione di Overflow e di Underflow.

Ho notato che alcuni codici hanno una funzione simile a quella dei tasti più vicini, per es. "2a" esegue un CLR. Fra i 60 codici vale la pena ricordare i codici "0a, 0b,..., 0f" (visti come 10, 11,..., 15), che impostano il corrispondente valore nell'esponente.

I codici 1d, 2d, 3d (23, 33, 43) cancellano vari pezzi di memoria. Il codice 8e moltiplica per potenze di 10 e arrotonda al numero visualizzato.

Con GTO "3c" e GTO "7c" il program counter si porta oltre il passo 7900 (TI-58) oppure vengono creati dei passi "fugaci" come visto prima (TI-59).

Con CLR GTO "7f" vengono disabilitati i tasti numerici (ma non gli altri!).

Il codice "3b", tanto per farne una nuova, risulta veramente misterioso: esegue la funzione "seno", tenendo conto della notazione angolare, ma INV "3b" calcola il valore di una funzione misteriosa periodica di forma trapezoidale cui non corrisponde alcuna funzione trigonometrica o logaritmica a me nota: probabilmente INV "3b" chiama una routine utilizzata durante il calcolo delle funzioni trigonometriche.

Vi sono inoltre codici a due byte, che operano per la maggior parte come RCL XY e sono ad esempio 2e, 4c, 5f, 6f, 7e, 8c, 9d; altri sono codici a tre o più byte, quali 4f, 5d, 6e, 7b, 8b, 8d, 8f.

Come si vede, ce n'è per tutti i gusti!

5) Meccanismo di generazione

Per i patiti descrivo come funziona la mia sequenza generatrice.

Tutto avviene quando si preme il tasto "Ins": ogni volta che viene premuto, viene sommato un certo valore esadecimale al codice presente in quel momento nella RAM (che non è quello visualizzato, in quanto sappiamo oramai che si tratta della ROM); tale valore esadecimale risulta dipendente dal passo in questione.

Riprendendo l'esempio, al passo 048 l'"Ins" aggiunge 4d; premuto due volte, come nel caso della sequenza, aggiunge 4d + 4d = 9a al codice immesso al passo 048 e cioè 42: perciò 9a + 42 = 13c, che, trascurando l'"1", diventa proprio "3c", visualizzato come "42".

Degli altri quattro passi da me citati, alcuni sommano 4d, altri 9d, ma il risultato (9a) è sempre lo stesso.

Ovviamente i codici hex possono essere generati in un qualunque passo (multiplo di 8!), ma bisogna tener conto del diverso valore che verrà sommato.

Tale valore potrà essere ricavato ponendo nel passo in questione il codice 00 ed eseguendo la sequenza "meravigliosa" con un solo "Ins": quello che si ottiene nel passo è appunto l'addendo esadecimale.

Ad esempio il passo 008 aggiunge 9c, il passo 016 aggiunge 4b ed il passo 152 aggiunge 9a".

Per ora, soprattutto per motivi di spazio, non aggiungiamo altro...

Diciamo solo che ora tocca ai lettori scoprire la "semantica" di ognuna di queste funzioni. A voi scoprire perché la stampante tira fuori nomi quali "OSS", "TLR", "V2N", "XIN" ed altri ancora.

Ovviamente molte caratteristiche le abbiamo già trovate, ma non vogliamo togliervi il gusto di trovarle da soli....

quale andrà poi eseguito lo stesso ragionamento che su c_1, e così via finché non si è giunti al termine della moltiplicazione.

Alla fine otterremo un output di cifre da 0 a 9 le quali, raggruppate e lette alla rovescia, daranno il numero cercato con tutte le cifre significative...

Prima di ulteriori descrizioni e di un esempio per chiarire un po' la cosa, siccome sono stato certamente un po' oscuro, voglio specificare che se magari il programma vi sembra un po' lento a girare (anche se in effetti non lo è), e questo soprattutto rispetto ad altri programmi di mia conoscenza la cui capacità massima è però di 20 cifre, pensate un po' se doveste fare una moltiplicazione del genere a mano!

N.B.: precisiamo che il programma è inutile, qualora si voglia calcolare ad es. 125×36... Ma mettete la soddisfazione!

Diamo ora un esempio, e poi ritorneremo sull'argomento; supponiamo di voler calcolare:
9.3174659627542 × 1003545.675438867;
il primo numero ha 14 cifre, il secondo 16.

Bisogna allora:

a) impostare 14 e premere A

b) impostare 16 e premere B

c) impostare una per una, premendo C, le cifre nel seguente ordine
2, 4, 5, 7, 2, 6, 9, 5, 6, 4, 7, 1, 3, 9, 7, 6, 8, 8, 3, 4, 5, 7, 6, 5, 4, 5, 3, 0, 0, 1

d) premendo D e successivamente R/S finché il display non indicherà il termine dell'elaborazione, si ottiene:
4, 1, 9, 4, 7, 4, 0, 1, 6, 7, 2, 3, 8, 6, 1, 8, 0, 7, 9, 2, 7, 6, 2, 0, 5, 0, 5, 3, 9

Pertanto il risultato cercato è, leggendo le cifre alla rovescia e mettendo al posto giusto la virgola:
9350502.6729708168327610474914,
e vi assicuro che questo risultato è corretto, avendo eseguito la moltiplicazione anche a mano (con un po' più di tempo...) ed avendo ottenuto lo stesso risultato.

Passiamo ora a considerazioni ulteriori sul programma. Esso è, come ho già avuto occasione di ripetere, identico nella sostanza con la parte etichettata con E nel programma precedente. Le uniche differenze sono che:

1) nella sezione input i valori di n e di m non vengono incrementati di 1; questo perché coi polinomi è più pratico operare col grado massimo, che è in difetto di 1 rispetto al numero di addendi del polinomio stesso, mentre coi numeri è più facile usare il numero di cifre direttamente.

2) Invece di essere subito mostrato, il contenuto del registro 53 viene ulteriormente elaborato con un semplice procedimento (passi da 071 a 116) allo scopo di mostrare una per una le cifre, eseguendo nel contempo l'operazione di "riporto" cui avevo già accennato all'inizio.

A questo punto specifico che potevo anche inserire una routine per mostrare sul display le cifre del risultato a gruppi di 10, aumentando la praticità d'uso per l'operatore. Ho preferito tuttavia non farlo (anche se i lettori lo potranno fare da soli, con un po' di pazienza e "buttando via" almeno un paio di registri ancora) per questo motivo: aumentare la possibilità di calcolo. Noterete infatti che il programma occupa 158 passi di programma; questo permette di entrare nella ripartizione di memoria 159.99 (corrispondente a 10 Op 17) per la TI-59, e in 159.39 (corrispondente a 4 Op 17) per la TI-58. E allora, chi avesse bisogno per un motivo o per l'altro, di una potenza di calcolo ancora superiore, può operare una sostituzione sugli indirizzi alle **memorie 59, 58, ..., 51 con i seguenti:**

— **per le TI-59: 99, 98, ..., 91**

— **per le TI-58: 39, 38, ..., 31**

con conseguente elevamento a 90 e a 30 dei limiti delle cifre impostabili. Ad esempio nel caso di una TI-59, dopo essere entrati nella ripartizione di memoria suddetta, si mette al posto di un RCL 59 un RCL 99, e così via.

Vi ho spiegato in pratica come funziona tutto; ed ora si potrebbe, chissà, estendere l'algoritmo usato anche al caso della divisione, facendo uso del programma precedente; oppure si potrebbe, sempre facendo riferimento alla moltiplicazione, usare il programma DATA PACKING del modulo Math/Utilities, tenendo presente che le cifre sono sempre intere; se ci si riuscisse, compattando i dati penso che si potrebbe arrivare a capacità dell'ordine delle 200 e oltre cifre, facendola in barba a calcolatori molto più potenti e costosi.... MC

Supermoltiplicazione

000	76	LBL	043	53	53	086	42	STO	129	01	1
001	11	A	044	01	1	087	51	51	130	95	=
002	42	STO	045	22	INV	088	55	÷	131	22	INV
003	59	59	046	44	SUM	089	01	1	132	77	GE
004	92	RTN	047	57	57	090	00	0	133	01	01
005	76	LBL	048	44	SUM	091	95	=	134	38	38
006	12	B	049	55	55	092	22	INV	135	61	GTO
007	42	STO	050	43	RCL	093	59	INT	136	00	00
008	58	58	051	57	57	094	65	×	137	19	19
009	92	RTN	052	67	EQ	095	01	1	138	01	1
010	76	LBL	053	00	00	096	00	0	139	22	INV
011	13	C	054	71	71	097	95	=	140	44	SUM
012	69	OP	055	43	RCL	098	99	PRT	141	56	56
013	20	20	056	58	58	099	43	RCL	142	44	SUM
014	72	ST*	057	85	+	100	53	53	143	54	54
015	00	00	058	43	RCL	101	55	÷	144	43	RCL
016	92	RTN	059	59	59	102	01	1	145	54	54
017	76	LBL	060	75	-	103	00	0	146	75	-
018	14	D	061	43	RCL	104	85	+	147	43	RCL
019	43	RCL	062	55	55	105	53	(	148	58	58
020	56	56	063	95	=	106	43	RCL	149	95	=
021	42	STO	064	22	INV	107	51	51	150	77	GE
022	57	57	065	77	GE	108	55	÷	151	01	01
023	01	1	066	00	00	109	01	1	152	56	56
024	44	SUM	067	71	71	110	00	0	153	61	GTO
025	57	57	068	61	GTO	111	54	)	154	00	00
026	43	RCL	069	00	00	112	59	INT	155	19	19
027	59	59	070	36	36	113	95	=	156	25	CLR
028	85	+	071	43	RCL	114	59	INT	157	35	1/X
029	01	1	072	53	53	115	42	STO	158	91	R/S
030	85	+	073	55	÷	116	52	52	159	00	0
031	43	RCL	074	01	1	117	25	CLR	160	00	0
032	54	54	075	00	0	118	42	STO	161	00	0
033	95	=	076	95	=	119	53	53	162	00	0
034	42	STO	077	22	INV	120	01	1			
035	55	55	078	59	INT	121	44	SUM			
036	73	RC*	079	65	×	122	56	56			
037	57	57	080	01	1	123	43	RCL	001	11	A
038	65	×	081	00	0	124	59	59	006	12	B
039	73	RC*	082	85	+	125	75	-	011	13	C
040	55	55	083	43	RCL	126	43	RCL	018	14	D
041	95	=	084	52	52	127	56	56			
042	44	SUM	085	95	=	128	75	-			