a cura di Pierluigi Panunzi

Matematica, matematica ed ancora matematica...

È innegabile che il campo in cui le TI sono più adatte è proprio quello in cui si devono effettuare calcoli....

Anche se gli argomenti trattati in questo numero non sono certo nuovi, li abbiamo considerati in quanto presentano delle novità nella risoluzione.

Il primo programma riguarda il calcolo della frazione che meglio approssima un numero dato, argomento già trattato nel numero 8 di MC, ma risolto questa volta (in maniera anche più veloce) con un diverso algoritmo, quello delle frazioni continue, troncato al valore desiderato.

Il secondo programma invece è dedicato fondamentalmente a chi ha a che fare con le derivate, ad esempio i "vecchi" liceali ed i "giovani" universitari i quali troveranno nella propria TI un valido aiutante, seppur non molto preciso come vedremo in dettaglio, ma non certo per colpa della 58 o 59.

Frazioni continue

di Marco Panareo (Lecce)

Il programma consente la determinazione della frazione che meglio approssima un numero dato.

Spieghiamo con un esempio: dato il numero $\pi = 3.141592654...$ è possibile trovare due interi il cui rapporto è molto vicino a π, ad esempio 22 e 7.

Una curiosità: per garantire una univoca determinazione delle aree degli appezzamenti di terreno, nello stato del Kentucky, USA, il numero π è stato legalmente assunto pari a 22/7).

Data una sequenza di numeri a_0, a_1, a_2,..., a_n,... interi positivi e diversi da zero, l'algoritmo:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \ldots}}}$$

è detto frazione continua. È noto che un qualsiasi numero reale è sviluppabile in frazione continua: lo sviluppo sarà limitato (cioè con un numero finito in termini a_i) se il numero dato è razionale e viceversa sarà illimitato (cioè con un numero infinito di a_i) se il numero è irrazionale. Esistono opportuni algoritmi atti a valutare i coefficienti a_i noto il numero, come anche esistono algoritmi che effettuano il calcolo inverso.

Il programma in questione opera nel modo seguente: dato il numero, calcola i coefficienti a_i sino all'indice "n" prefissato; quindi tramite questi, risale al numeratore ed al denominatore del numero razionale che, sviluppato in frazione continua, darebbe proprio questi n + 1 coefficienti. Impostiamo il programma in memoria e vediamolo in azione: diamo alla calcolatrice il numero "e" = 2.718281828 con "1 INV log", premiamo "A" e quindi inseriamo n = 5 premendo "5 B"; calcoliamo ora il numeratore della frazione approssimante premendo "C". Dopo un certo tempo, dipendente dal valore di "n" impostato, sul visualizzatore appare 87; calcoliamo il denominatore premendo "D" ed otterremo 32.

La frazione approssimante è allora 87/32; la differenza tra questo numero ed il valore di "e" risulta nell'ordine di 10^{-4}.

Se invece poniamo n = 20 otteniamo la frazione

15749589/5793950

con la quale otteniamo un errore NULLO, ovvero per la calcolatrice il valore del rapporto dato ed "e" sono uguali a meno di 10^{-12}, essendo il valore di "e" noto alla TI con 12 cifre decimali (2.718281828459 di cui solo le prime 9 visibili sul display).

Sono utilizzati i registri da R01 ad R08 per i calcoli, mentre i registri da R09 a R_{n+8} per memorizzare i coefficienti a_i"

Dobbiamo confessare che il programma, dato che tra l'altro non è noiosamente lento, ci ha alquanto appassionato...

Abbiamo provato con n = 25 ottenendo la mostruosa frazione:
1061211705/390397969!!

Non contenti di ciò (fidarsi è bene, ma...) abbiamo perfino effettuato la divisione, non nascondendo una certa incredulità: inutile dire che il valore ottenuto era proprio la "e" conosciuta dalla TI, con un errore anche stavolta nullo. Non contenti ci siamo rivolti a π: chi non conosce la bellissima frazione approssimatrice 355/113? Ebbene si trova ponendo n = 3. Ecco che nettamente migliore è la frazione 416327/132521 ottenibile con n = 5, per non parlare poi del valore ottenuto con n = 15, valore che per la precisione della TI è prorio π: 18550869/5904925.

Frazioni continue

000	76	LBL	046	97	DSZ	093	02	.02	140	01	1	187	06	06
001	11	A	047	06	06	094	21	.21	141	00	0	188	02	02
002	42	STO	048	00	00	095	69	OP	142	42	STO	189	21	21
003	01	01	049	65	65	096	28	28	143	08	08	190	69	OP
004	91	R/S	051	03	03	097	65	×	144	43	RCL	191	28	28
005	76	LBL	052	42	STO	098	73	RC*	145	06	06	192	65	×
006	12	B	053	02	02	099	08	08	146	29	CP	193	73	RC*
007	42	STO	054	69	OP	100	85	+	147	67	EQ	194	08	08
008	06	06	055	28	28	101	43	RCL	148	02	02	195	85	+
009	42	STO	056	43	RCL	102	02	02	149	22	22	196	43	RCL
010	07	07	057	04	04	103	95	=	150	73	RC*	197	03	03
011	91	R/S	058	42	STO	104	42	STO	151	08	08	198	95	=
012	76	LBL	059	03	03	105	04	04	152	42	STO	199	42	STO
013	13	C	060	43	RCL	106	22	INV	153	02	02	200	05	05
014	01	1	061	05	05	107	97	DSZ	154	22	INV	201	22	INV
015	42	STO	062	61	GTO	108	07	07	155	97	DSZ	202	97	DSZ
016	03	03	063	00	00	109	02	02	156	06	06	203	06	06
017	09	9	064	25	25	110	21	21	157	02	02	204	02	02
018	42	STO	065	43	RCL	111	69	OP	158	21	21	205	21	21
019	08	08	066	07	07	112	28	28	159	65	×	206	42	STO
020	43	RCL	067	29	CP	113	65	×	160	69	OP	207	03	03
021	01	01	068	67	EQ	114	73	RC*	161	28	28	208	43	RCL
022	42	STO	069	02	02	115	08	08	162	73	RC*	209	04	04
023	02	02	070	19	19	116	85	+	163	08	08	210	42	STO
024	59	INT	071	42	STO	117	43	RCL	164	85	+	211	02	02
025	72	ST*	072	06	06	118	03	03	165	01	1	212	43	RCL
026	08	08	073	09	9	119	95	=	166	95	=	213	05	05
027	65	×	074	42	STO	120	42	STO	167	42	STO	214	69	OP
028	43	RCL	075	08	08	121	05	05	168	03	03	215	25	25
029	03	03	076	73	RC*	122	22	INV	169	22	INV	216	61	GTO
030	94	+/-	077	08	08	123	97	DSZ	170	97	DSZ	217	01	01
031	85	+	078	42	STO	124	07	07	171	06	06	218	73	73
032	43	RCL	079	02	02	125	02	02	172	02	02	219	43	RCL
033	02	02	080	65	×	126	21	21	173	21	21	220	09	09
034	95	=	081	69	OP	127	42	STO	174	69	OP	221	91	R/S
035	42	STO	082	28	28	128	03	03	175	28	28	222	01	1
036	04	04	083	73	RC*	129	43	RCL	176	65	×	223	91	R/S
037	55	÷	084	08	08	130	04	04	177	73	RC*	224	00	0
038	43	RCL	085	85	+	131	42	STO	178	08	08	225	00	0
039	03	03	086	01	1	132	02	02	179	85	+	226	00	0
040	95	=	087	95	=	133	43	RCL	180	43	RCL			
041	35	1/X	088	42	STO	134	03	03	181	02	02			
042	59	INT	089	03	03	135	61	GTO	182	95	=			
043	42	STO	090	22	INV	136	00	00	183	42	STO	001	11	A
044	05	05	091	97	DSZ	137	95	95	184	04	04	006	12	B
045	22	INV	092	07	07	138	76	LBL	185	22	INV	013	13	C
						139	14	D	186	97	DSZ	139	14	D

Calcolo di derivate successive

di Andrea Cantadori (Parma)

Questo programma calcola la derivata n-esima di una funzione (nel caso ovviamente che tale derivata esista; la calcolatrice non distingue funzioni derivabili e non) calcolata in un punto x_0 del suo insieme di definizione. È un programma utile e pratico, che potrà servire a quanti hanno a che fare con studi di funzione (ma anche ad altri). Vediamo ora l'algoritmo su cui è basato il programma. Innanzitutto è noto che la derivata di una funzione f (x) è definita come il limite per $h \to 0$ del seguente rapporto incrementale:

$$\lim_{h \to 0} \frac{f(x_0+h) - f(x_0)}{h}$$

Se consideriamo tale formula così come è, non ci potrà essere di grande aiuto, giacché i limiti non hanno senso per la calcolatrice. Però se ci limitiamo a valutare l'espressione del rapporto incrementale con h abbastanza piccolo (e sarà proprio questo "abbastanza piccolo" a porre i limiti più grossi all'uso del programma) si potrà trovare un valore che differisce abbastanza poco dal valore reale della derivata. In effetti si trova che, nel caso della derivata prima, per la maggior parte delle funzioni si arriva ad una precisione fino a circa la quinta o sesta cifra decimale, il che non è poco. Se poi la funzione è sufficientemente regolare come andamento, si potranno trovare valori ancora meglio approssimati.

In definitiva ci limiteremo a valutare la seguente espressione:

$$\frac{f(x_0+h) - f(x_0)}{h}$$

Iterando poi il ragionamento fatto per la derivata prima anche sulle derivate seconde, terze, ecc., si perviene facilmente, dopo pochi calcoli alla seguente formula approssimata generale:

$$f^n(x_0) = \frac{1}{h^n} \sum_{k=0}^{n} (-1)^k \binom{n}{k} f\left[x_0 + (n-k)\,h\right]$$

Facendo uso di questa formula, siamo in grado allora di approssimare il valore reale della derivata n-esima nel punto considerato. C'è però da fare una importante considerazione: siccome è noto che la calcolatrice non può lavorare con infinitesimi, è pure ovvio che col crescere di n, cioè dell'ordine di infinitesimo del numeratore del rapporto incrementale approssimato, dovrà crescere h. Ad esempio, consideriamo lo sviluppo della formula precedente per una derivata terza:

$$f'''(x_0) = \frac{1}{h^3}\left[\binom{3}{0} f(x_0+3h) - \binom{3}{1} f(x_0+2h) + \binom{3}{2} f(x_0+h) - \binom{3}{3} f(x_0)\right]$$

Derivate successive

```
250  42 STO      288  50  50      326  01  01      364  61 GTO
251  05  05      289  42 STO      327  95  =       365  02  02
252  01  1       290  02  02      328  16 A'       366  91  91
253  42 STO      291  43 RCL      329  49 PRD      367  43 RCL
254  04  04      292  02  02      330  06  06      368  07  07
255  43 RCL      293  42 STO      331  87 IFF      369  55  ÷
256  05  05      294  06  06      332  01  01      370  43 RCL
257  67  EQ      295  43 RCL      333  03  03      371  09  09
258  02  02      296  03  03      334  45  45      372  45 YX
259  67  67      297  71 SBR      335  86 STF      373  43 RCL
260  49 PRD      298  02  02      336  01  01      374  00  00
261  04  04      299  50  50      337  43 RCL      375  95  =
262  69 OP       300  22 INV      338  06  06      376  91 R/S
263  35  35      301  49 PRD      339  22 INV      377  76 LBL
264  61 GTO      302  06  06      340  44 SUM      378  14  D
265  02  02      303  43 RCL      341  07  07      379  42 STO
266  55  55      304  00  00      342  61 GTO      380  09  09
267  43 RCL      305  75  -       343  03  03      381  92 RTN
268  04  04      306  43 RCL      344  52  52      382  00  0
269  92 RTN      307  03  03      345  22 INV      383  00  0
270  76 LBL      308  95  =       346  86 STF      384  00  0
271  11  A       309  71 SBR      347  01  01
272  42 STO      310  02  02      348  43 RCL
273  00  00      311  50  50      349  06  06
274  92 RTN      312  22 INV      350  44 SUM      LABELS
275  76 LBL      313  49 PRD      351  07  07
276  12  B       314  06  06      352  69 OP       271  11  A
277  42 STO      315  43 RCL      353  23  23       276  12  B
278  01  01      316  00  00      354  43 RCL      281  13  C
279  92 RTN      317  75  -       355  00  00       378  14  D
280  76 LBL      318  43 RCL      356  75  -
281  13  C       319  03  03      357  43 RCL
282  86 STF      320  95  =       358  03  03
283  01  01      321  65  ×       359  95  =
284  43 RCL      322  43 RCL      360  22 INV
285  00  00      323  09  09      361  77  GE
286  71 SBR      324  85  +       362  03  03
287  02  02      325  43 RCL      363  67  67
```

Ora, se prendiamo $h = 10^{-9}$, si può vedere che ogni addendo del numeratore non differisce, nei limiti di precisione della calcolatrice, in modulo dagli altri, onde il numeratore stesso avrà come risultato 0 qualunque sia la f (x) e qualunque sia il punto considerato.

Se invece prendiamo h = 0.001 (valore ricavato "sperimentalmente") si otterranno buone valutazioni, che inevitabilmente però differiranno dal valore reale più della derivata seconda e ancor più della derivata prima. Questo in sostanza è il limite della nostra calcolatrice in questo genere di calcolo; pertanto al di sopra della derivata terza si dovranno considerare i risultati non come "relativamente esatti", ma come "ben indicativi" dell'andamento della derivata nel punto.

Il programma contiene una subroutine, indicata con SBR 250, che effettua il calcolo del fattoriale, però in una forma un po' particolare per poter essere inserita nel programma; sua caratteristica è inoltre quella di dare come risultato di 0! il numero 1, fatto questo che permette di risparmiare passi di programma nel seguito.

Innanzitutto viene settato il flag 1 il quale, alternativamente, servirà a sottrarre o a sommare, cioè accendendosi e spegnendosi farà da $(-1)^n$. Si parte poi da k = 0, e viene calcolato il coefficiente binomiale (k^n), tramite la SBR 250, appunto. Si procede così fino a k = n, dopodiché si moltiplica il numero così ottenuto per $1/h^n$, che fornisce appunto il valore della derivata. Da notare che il programma è stato inserito a partire dal passo 250 per poter permettere a partire dal passo 000 l'inserzione della funzione con l'etichetta A'. Questo abbrevia i tempi di ricerca della funzione stessa quando ciò è richiesto dal programma. Pertanto andrà registrato il tutto sul lato 2 delle schede magnetiche; per le TI-58, si dovranno operare alcuni cambiamenti, al fine di sistemare il programma come consentito dalle ripartizioni di memoria, tenendo conto che vengono utilizzati i registri da 00 a 09. Si potrebbe lasciare così a patto che si entri nella ripartizione di memoria corrispondente a 1 Op 17. Questa però lascia liberi solo i registri che servono proprio per l'esecuzione del programma, onde non ne restano per l'impostazione di f(x). Si può ovviare all'inconveniente sostituendo un registro, ad esempio 09, con una HIR, poi utilizzare il registro 09 stesso come x.

Vedano comunque i possessori delle 58 come comportarsi.

Veniamo ora alle istruzioni per l'uso del programma:

a) Impostare la funzione LBL A', al passo 000, seguendo le normali istruzioni per caricare una funzione, vale a dire terminando con INV SBR, senza mai usare =, CLR, ecc. Valgono a questo le considerazioni sulle memorie fatte in precedenza.

b) Impostare n (l'ordine della derivata) e premere A.

c) Impostare x_0 (il punto in cui si calcola la derivata) e premere B.
d) Impostare h secondo il seguente criterio che, ribadiamo, è puramente sperimentale, e la sua necessità è dovuta a motivi intrinseci all'uso della calcolatrice:
— per la derivata prima: h = 0.0000001
— per la derivata seconda: h = 0.0001
— per la derivata terza: h = 0.001
— per la derivata quarta: h = 0.01
— per la derivata quinta: h = 0.01
— per la derivata sesta, settima, ecc.: usare h = 0.1;
notare che se l'ordine della derivata è ancora maggiore, è consigliabile utilizzare il programma solo per funzioni il più "regolari" possibile. Per n molto elevato si consiglia di usare h = .5 oppure addirittura h = 1. C'è un criterio semplice per distinguere un h adatto: l'h è adatto quando, oltre ad essere il più piccolo possibile, è anche quello per il quale al variare del punto la calcolatrice non risponde identicamente 0.
e) Premere C per il risultato.

Un esempio servirà a chiarire le cose: sia data la funzione:

$$f(x) = \frac{1 + x - x^2}{\log(x)} \qquad x \in (1, +\infty)$$

Vogliamo studiarne il comportamento in prossimità del punto x = 2. Impostiamo pertanto la f(x) in Lbl A' come già spiegato. Per la derivata prima:
a) premere 1 e il tasto A
b) premere 2 e il tasto B
c) impostare 0.0000001 e premere D
d) premere C: si otterrà:
-3.2874, valore che differisce da quello reale (-3.287400632) per diciannove parti per milione. Una approssimazione tale da permettere anche utilissimi confronti a studenti che abbiano a che fare con questi simpatici calcoli ...

Per la derivata seconda:
a) premere 2 e il tasto A
b) premere 2 e il tasto B
c) impostare 0.0001 e premere D
d) premere C: si otterrà 1.3364 anziché 1.336984262, con una approssimazione diremmo ancora molto buona.

L'ANGOLO DELLE TI

Eccoci dunque alla continuazione dell'argomento trattato sul numero scorso e che oramai ci accompagna da parecchie puntate: l'analisi del comportamento delle nuovissime funzioni ottenibili "sinteticamente" sulle TI-58 e 59, funzioni ovviamente non citate su alcun manuale ed aventi la notevole caratteristica (oramai lo sanno anche i sassi...) di essere codificate con un valore esadecimale (anche se però sul display vengono mostrate in "decimale" dopo una traduzione interna).

I sassi di cui sopra sanno anche qual è la "fonte" da cui attingiamo queste notizie: solo per i lettori che ancora non lo sapessero, ricordiamo il nome del "creatore" di queste stranissime funzioni: tal Stefano Laporta di Bologna.

Ecco quanto ancora ha da dirci:

"Molto utili sono i codici 6D ("tradotto" dalla stampante con "*ST"), 8E e 4E (questi ultimi due non "tradotti" dalla PC-100).

Il codice 6D incrementa di uno il registro interno delle TI che contiene il fissaggio dei decimali: ovvero se abbiamo fissato 3 decimali con "Fix 3", il codice 6D esegue un Fix 4. Piuttosto esplicativo è il programmino seguente, che va introdotto al passo 000: 6D Pause RST.

Introdotto il codice 6D al passo 000, tramite la "sequenza di Laporta" (per delucidazioni rivolgersi ai sassi...), si impostano le funzioni "Pause" e RST; tornati in modo esecuzione si introduce un numero (ad esempio π) e si preme RST R/S: si vedrà via via il numero dei decimali andare da 0 a 9. Il codice 8E invece fa una cosa particolare: arrotonda il numero visualizzato eliminando le cifre non visibili (e fin qui è uguale ad EE INV EE) ma, se il numero possiede N decimali, lo divide per 10^{12-N}. Si badi che N è il numero di decimali visualizzati ed in generale non corrisponde al fissaggio dei decimali. Introduciamo il codice 8E al passo 048 con
GTO 048 LRN +/- BST LRN e con
CLR Pgm 19 SBR 045 DMS LRN Ins Ins LRN RST CLR
ed infine inseriamo "Lbl A" ai passi 046 e 047, nonché "INV SBR" al passo 049: attenzione a non cancellare il codice appena nato al passo 048, mentre viceversa potete chiudere un occhio sui passi 050-055 contenenti, (come dicono gli inglesi) "garbage"...

Torniamo perciò al modo esecuzione e premiamo "π A": se tutto è andato per il meglio vedremo sul display

0.0031415927:

dato che π era visualizzato con 9 cifre decimali, la TI ha diviso tale valore per 10^{12-9} e cioè 1000.

Analogamente premendo "3.2 A" otterremo 3.2 E-11. Questo codice può essere utile per determinare rapidamente quante cifre decimali ha un certo numero.

Inseriamo a questo proposito quest'altro piccolo programma a partire dal passo 000, avendo cura di *non* cancellare il precedente (passi 046-049):
Lbl E EE INV EE ÷ A = log +/- + 12 = INV SBR.

Tale programma fornisce il numero dei decimali visualizzati del numero impostato: per esempio "1.234 E" fornisce "3", mentre tre "π E" dà come risultato 9.

Inutile dire che senza l'ausilio del codice 8E il programma sarebbe ben più complicato.

Dulcis in fundo (come dicevano i pasticceri latini), il codice 4E necessita di un discorso a parte.

Prima ho parlato, a proposito del codice 6D, di un registro interno della TI: ebbene il codice 4E permette di leggere tale registro.

Questo è lungo quanto un normale registro di memoria (8 byte) e contiene varie informazioni: lo stato dei 10 flag, il fissaggio dei decimali, il passo di destinazione dell'ultimo GTO eseguito ed un registro di "servizio".

Non è poco vero?!

Il contenuto di questo registro ("di stato", ragionando in termini computeristici) è formato dunque da 16 cifre, raggruppate, a seconda della loro funzione, nel modo seguente:

XXXXX 000 PPPP 0 RR Y.

Le prime cinque cifre XXXXX contengono lo stato dei flag accoppiati secondo lo schema 94 83 72 61 50. In particolare ognuna delle cifre X può valere 0, 1, 2, o 3 secondo una logica prettamente booleana e ben nota agli informatici: se il flag è spento si avrà uno "0" ed in caso di flag settato si avrà un "1", il tutto poi letto in "decimale" apparirà con un valore variabile tra 0 e 3; un esempio pratico: settando il flag 3, le cifre XXXXX varranno 01000, mentre settando tutti i 10 flag si ottiene 33333.

Le quattro cifre PPPP contengono il passo di destinazione dell'ultima istruzione di salto usato (GTO, SBR, Dsz e simili) in forma pseudo-ottale: un valore 0247, ad esempio, significa 24 × 8 + 7 = 199, indicando che è stato effettuato un salto al passo 199.

Le due cifre RR vengono usate quando si esegue un'istruzione a più byte, mentre infine Y contiene il valore del fissaggio dei decimali incrementato di 2 (ma Y = 0 corrisponde al fissaggio standard).

In definitiva l'istruzione 4E pone nel display questo registro interno come se si trattasse di un qualunque registro di memoria e come tale le prime 13 cifre saranno intese come mantissa, RR come esponente ed Y darà i segni di ambedue. Il tutto secondo le "regole" abituali, regole che si apprendono facilmente andandosi a studiare, la codifica interna della TI. In particolare è lecito aspettarsi, e così è infatti, che alcuni valori di Y causeranno la visualizzazione di un errore (9.999 ecc lampeggiante).

Vediamo dunque un esempio: immettiamo il codice 4E al passo 048 con:

GTO 048 LRN) BST LRN e poi
CLR Pgm 19 SBR 045 DMS LRN Ins Ins LRN RST = CLR

al solito senza preoccuparsi dei vari lampeggiamenti che mano a mano si avranno sul display.

Impostiamo come prima la "Lbl A" e il "RTN" rispettivamente prima e dopo il codice neonato e torniamo al modo esecuzione. A questo punto premiamo DMS (serve per alterare il valore di RR), settiamo da tastiera i flag 0, 3, 5, 7 e premiamo A. Sul display vedremo 12030000.06.

Controlliamo: XXXXX vale 01203 a conferma dell'accensione dei flag 3, 7, 5, 0 (con gli ultimi due "accoppiati", come visto in precedenza); PPPP vale 0060, corrispondente al passo 6 × 8 = 48, e cioè l'indirizzo di salto di "A"; RR ed Y invece non sono direttamente visibili: ma dalla posizione del punto decimale si può arguire che RR è 08 (forse perché la funzione DMS utilizza la HIR 08) e che Y vale 0, corrispondente a nessun fissaggio predisposto.

Potete ora sbizzarrirvi a settare e resettare flag, a cambiare fissaggio dei decimali o usare RR (per esempio la sequenza STO 0 seguita da un tasto qualunque mette il codice del tasto in RR).

Non sono riuscito purtroppo a trovare un'istruzione (tipo STO), che permetta di modificare questo registro interno: una volta trovatala si potrebbe modificare lo stato della TI in un solo colpo, così come succede con l'HP 41.

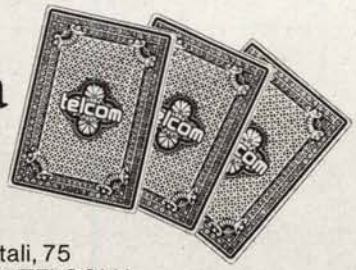
Risulta chiara una cosa: l'hardware delle TI è fortemente legato a strutture a 8 byte (come i registri di memoria) ognuna delle quali contiene anche più di una informazione: questo spiega la proliferazione di "8", ottali e pseudo-ottali, che ho notato in tutte le mie scoperte."

L'IMBATTIBILE

VIDEO TERMINALE VT-4100

Il meglio della tecnologia racchiuso nel terminale video TATUNG VT 4100.
Un microprocessore ne rappresenta il cuore e gestisce il buffer di 2 K di memoria per l'editing locale e la gestione della trasmissione a caratteri, a blocchi o totale.
Comandi remoti permettono un completo controllo dello schermo e la gestione ottimale dei records.
Un'uscita ausiliaria consente il collegamento ad una stampante.
TATUNG VT-4100 IMBATTIBILE NEL RAPPORTO PRESTAZIONE-PREZZO

gioca la carta telcom



Telcom s.r.l.- 20148 Milano-Via M. Civitali, 75
Tel. 4047648 (6 linee ric. aut.)-Telex 335654 TELCOM I