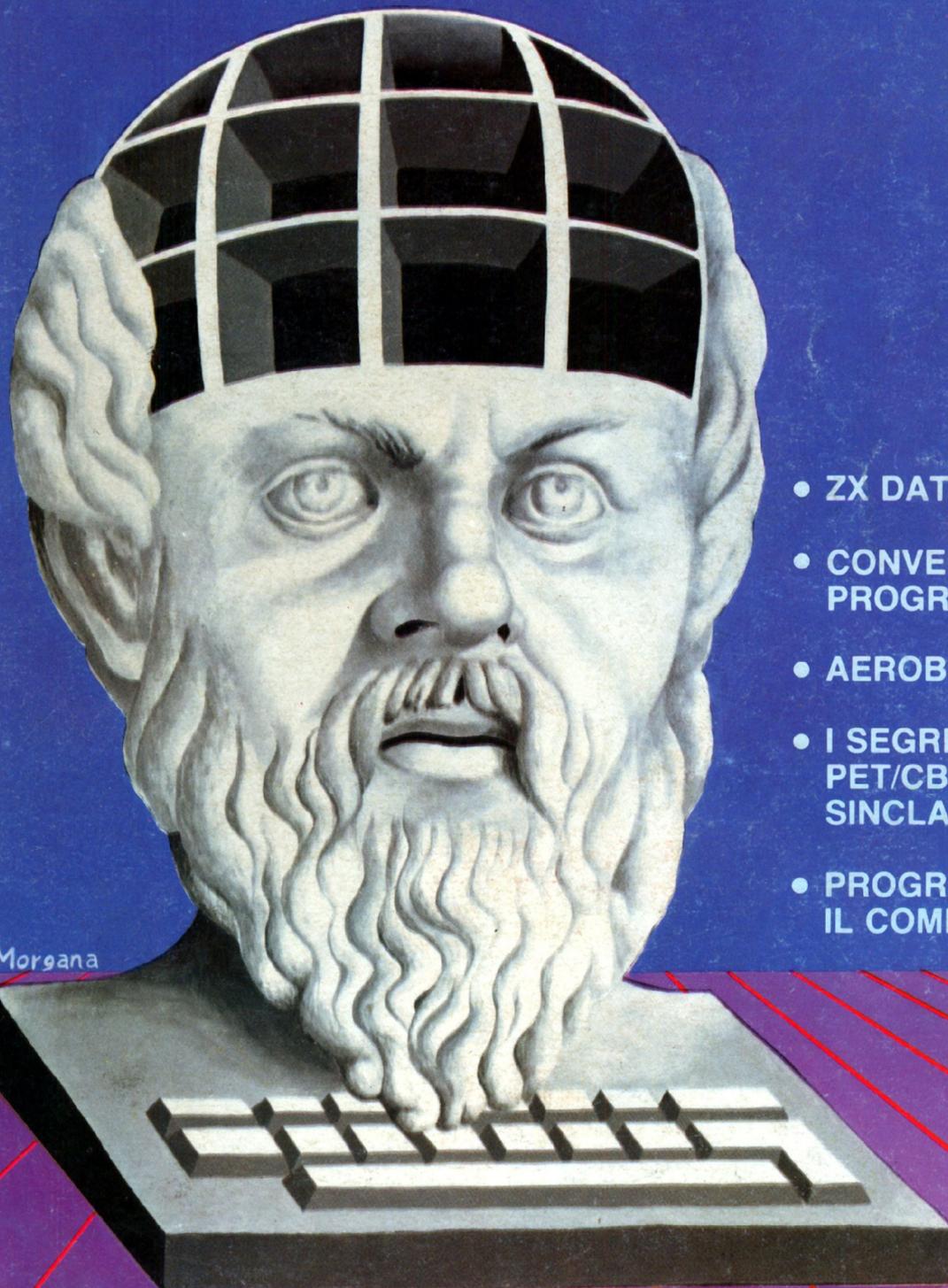


# PERSONAL SOFTWARE

ANNO 2 N. 8-9  
LUGLIO-AGOSTO 1983 L. 3.500

UNA PUBBLICAZIONE DEL GRUPPO EDITORIALE JACKSON



Morgana

- ZX DATA BASE
- CONVERSIONE DI PROGRAMMI PER ZX81
- AEROBICA CON IL TRS-80
- I SEGRETI DEI PERSONAL: PET/CBM, TEXAS 99/4A, SINCLAIR, VIC 20
- PROGRAMMI PER IL COMMODORE 64

# Harden Italia, Il salto di qualità.

*Dal personal computer  
al professional computer.*

Nel quadro di una filosofia aziendale in evoluzione, Harden Italia riconferma la validità della proposta del Sirius 1. Il Sirius 1, con tutta la potenza del suo microprocessore a 16 bit, con 5 MHz, e una memoria centrale che può arrivare a 896 KBytes, è uno dei più avanzati della nuova generazione dei Personal.

Oltre ad una enorme capacità di archiviazione dei dati (dai 1240 KBytes del Sirius 1 agli 11.840 KBytes del Sirius 1b) il Sirius può contare su alcune caratteristiche che un tecnico e un professionista non possono non apprezzare: dall'interfacciamento con due porte seriali e una parallela programmabile da software, ai sistemi operativi (MS-DOS della Microsoft e CP/M86 della Digital Research), fino ai linguaggi di alto livello come il BASIC-86 (interprete e compilatore), l'Assembler, il COBOL, il Fortran, il Pascal.

Oltre che sul software vero e proprio (programmi come il Dbase II, il SuperCalc, il Multiplan o l'Harden-text e l'Harden-data) il Sirius 1 si avvale dei così detti "Tool Kits", una serie cioè di utilities compatibili con qualsiasi linguaggio che permettono una stesura dei programmi più facile e più completa come ad esempio l'AutoSort, il FABS, una gestione sofisticata IS, ecc. In più, il Sirius 1 è distribuito e assistito dalla Harden Italia su tutto il territorio nazionale.

Per saperne di più sul Sirius 1, sui suoi programmi o su dove sono i punti di vendita Harden più vicini, chiamare (0372)-63136 oppure (02)-651645: risponde la Harden Italia.



Harden Italia S.p.A. Direzione generale e uffici commerciali  
20121 Milano - via dei Giardini, 4 - tel. (02) 651645  
Sede operativa e uffici commerciali  
26048 Sospiro (CR) - tel. (0372) 63136 - telex: 3205881

SIRIUS 1 CONFIGURAZIONE BASE  
(128 KBYTES RAM, 1240 KBYTES FLOPPY DISC)  
DA OGGI L.6500000

ANNO 2 N. 8-9 LUGLIO-AGOSTO 1983

# PERSONAL SOFTWARE



In copertina: la rappresentazione fantastica di un data base intelligente nell'interpretazione dello Studio Morgana.

## ARTICOLI

<b>Conversione di programmi per ZX81 e ZX80 nuova ROM</b> .....	Bruno Del Medico.....	... ZX80, ZX81
<b>Master Mind contro il VIC</b> .....	Paolo Pulli.....	... VIC 20
<b>Programmare grafici con il TI 99/4A</b> .....	Roberto Del Giudice.....	... TI 99/4A
<b>Picture</b> .....	Matteo Minischetti.....	... VIC 20
<b>Quattro programmi per il Commodore 64</b> .....	Carlo Sintini.....	... Commodore 64
<b>Gestire file con il PET/CBM</b> .....	Stefano De Monte.....	... PET/CBM
<b>Archivio scacchistico per TI 99/4A</b> .....	Sergio Borsani.....	... TI 99/4A
<b>Procedura di recupero</b> .....	Luciano Gemme.....	... VIC 20
<b>Guida e cacciatore</b> .....	Paolo Ferrari.....	... ZX81
<b>ZX data base</b> .....	Enrico Ferreguti.....	... ZX81
<b>ZX Manager</b> .....	Enrico Ferreguti.....	... ZX81
<b>Paroliamo</b> .....	Fausto Manfredini.....	... ZX81
<b>Il giro del cavallo</b> .....	Giulio Morpurgo.....	... DAI
<b>Dal Basic al Pascal</b> .....	Ronald W. Anderson.....	...
<b>Matematica aerobica con il TRS-80</b> .....	Bruce Douglas.....	... TRS-80

## RUBRICHE

<b>Editoriale</b>		
Come passerò l'estate .....	Mauro Boscarol.....	
<b>Raccolta di routine Basic</b>		
I nidi di FOR .....	Mauro Boscarol.....	... tutti
<b>I segreti dei personal</b>		
Un tasto per una istruzione .....		... PET/CBM
PRINT e INPUT in un contesto grafico .....	Sergio Borsani.....	... TI 99/4A
Scritte giganti, grafici e disegni, routine e messaggi video .....	Enrico Ferreguti.....	... ZX81
Gestione del cursore del VIC 20 .....	Enrico Bossaglia.....	... VIC 20
Come programmare uno schermo scorrevole .....	Marcello Spero.....	... Spectrum
<b>Conversioni</b>		
Julia .....		
<b>Debug</b>		
Come realizzare un listato bidirezionale .....		
<b>Contributi dei lettori</b>		
Il controllo del codice fiscale .....		
Utilizzare le routine della cartridge del VIC .....		

In questa guida sono riportati i personal computer e i microprocessori di cui si parla negli articoli e nelle rubriche.

## Come passerò l'estate

Mauro Boscarol

Sono sul piede di partenza per la Corsica. Il gommone è già caricato sulla macchina, il carrello è pieno di tende, pentole, scatolette, servizi in plastica e melanina. Forzosamente, la destinazione che ho scelto è tra le più economiche: la direzione di questa rivista non mi permette di uscire dagli angusti limiti che mi sono imposto. Non ho però trascurato di infilare nel sacco uno ZX80 nuova ROM, con il quale, nei pomeriggi assolati, calcolare le orbite dei satelliti e la periodicità delle maree.

Mia figlia strilla perché vuole anche il PET/CBM 4032 con doppio floppy e stampante: sto cercando di convincerla che sul posto non avremo la necessaria alimentazione elettrica, e che giocheremo a Invaders quando torneremo in città.

Ho accuratamente progettato queste vacanze con l'aiuto di un Apple II e di Visicalc. Ogni spostamento è stato organizzato nei minimi particolari. Il costo della benzina, dei camping, dei generi alimentari è stato calcolato in anticipo.

Ho pensato anche al problema delle cartoline: con

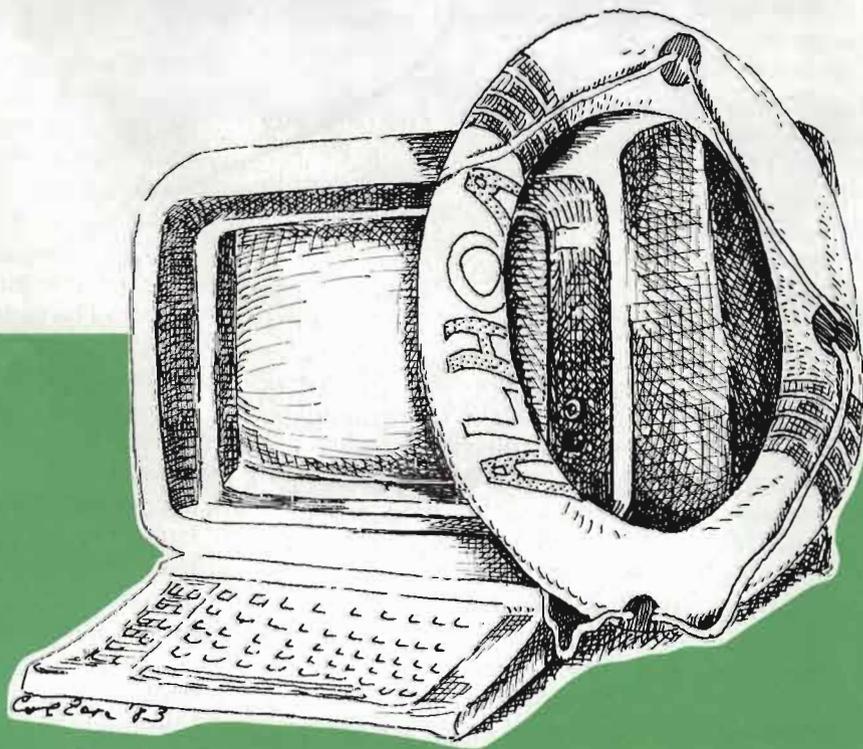
un data base su Atari 400 ho stampato tutti i record del mio indirizzario su etichette: quando sarò ad Ajaccio incollerò le etichette sulle cartoline e, a mano, completerò la parte dei saluti e baci.

Quest'anno ho deciso di fare delle vacanze "intelligenti". Mi sono procurato tutti i numeri dell'*Espresso* e di *Panorama*, ed ho organizzato un itinerario nell'interno della Corsica che mi porterà a contatto con pastori e valligiani. Andrò alla ricerca della genuina anima popolare corsa: registrerò le melodie folk su nastri che farò poi digitalizzare al mio DAI. Oltre a calcolare correlazioni e contingenze, il DAI potrà sintetizzare le melodie, ordinarle, correggerle ed archivarle in un apposito data base musicale. Quest'inverno potrò risentirne in santa pace.

Bene, mi pare che sia tutto pronto: mia moglie ha già acceso il motore della Diesel per farlo scaldare, e mi sta suonando con il clacson. "Arrivo", urlo "finisco di scrivere questo editoriale con il word processor del TRS-80 Color Computer e vengo".

Cari coniglietti, mi spiace lasciarvi per tutto questo tempo. Spero che anche voi passiate delle belle vacanze con il software e l'hardware a portata di mano.

Al ritorno in città troverete un altro numero della vostra rivista, e tutto riprenderà come prima. Non scoraggiatevi, su con la vita e "tokenizzatevi". Ciao.



## Programmi di geologia

Egregio Direttore, sono un ricercatore universitario assiduo lettore della Vostra Rivista che trovo interessante ed utile.

Ho letto con molto interesse il Suo editoriale "Il software oggi e domani" apparso sul n° 5 nel quale Lei fa riferimento a sistemi "esperti" afferenti a diverse discipline.

In particolare sarei interessato a ricevere maggiori informazioni circa i programmi dedicati alla Geologia. Le sarei quindi veramente grato se mi fornisse maggiori dettagli sulla reperibilità di tali programmi e/o, se in Suo possesso, inviarmi del materiale illustrativo per il quale rimborserei le spese.

Certo che la mia richiesta verrà esaudita, colgo l'occasione per complimentarmi con Lei per la Rivista (nella quale spero di trovare in futuro anche una sezione applicativa scientifica) e per porgerLe i miei cordiali saluti.

Gerardo Brancucci

*Un "sistema esperto" è un programma che usa le tecniche dell'intelligenza artificiale per fare deduzioni basate su specifiche conoscenze fornite da uno o più esperti. Sono stati realizzati sistemi esperti in differenti discipline. Per quanto riguarda la geologia, è stato sviluppato un sistema esperto di nome PROSPECTOR, descritto nell'articolo "Model Design in the Prospector Consultant System for Mineral Exploration" di R.O. Duda, J.G. Gaschnig e P.E. Hart, comparso in Expert System in the Microelectronic Age, un volume pubblicato nel 1979 dalla Edinburgh University Press, a cura di D. Michie. Il progetto è stato finanziato dalla US Geological Survey e dalla National Science Foundation. PROSPECTOR può dare pareri su: (1) la possibilità che in una certa zona ci siano giacimenti di un certo minerale; (2) la convenienza di una particolare esplorazione; (3) i migliori punti di trivellazione. La prima previsione di PROSPECTOR è stata quella relativa ad un giacimento di molib-*

*deno, che pare sia stato effettivamente trovato là dove previsto (ma su questo non abbiamo notizie sicure).*

*Può trovare un bell'articolo di rassegna dei sistemi esperti (che contiene anche una spiegazione abbastanza dettagliata di PROSPECTOR e una buona bibliografia) nel numero di settembre 81 di BYTE (R.O. Duda J.G. Gaschnig Knowledge Based Expert Systems Come of Age): gli autori sono gli stessi di PROSPECTOR.*

## Parliamo dell'Atom

Gentilissima Redazione di Personal Software, vorrei sapere il perché dei pochi programmi pubblicati per il sistema ACORN ATOM.

Ho aspettato pazientemente fino al numero cinque della rivista, sperando che dopo il programma RALLY pubblicato sul numero tre, ce ne fossero degli altri.

Supponendo che non possiate accontentarmi, vi chiedo cortesemente di codificare le seguenti istruzioni per il mio sistema: READ; PRINT AT; DATA; INKEYS; CODE; CHR\$; MIDS; POKE; PEEK ed infine l'assegnamento dei tasti, per simulare il pilotaggio di una astronave.

Tutto questo è dovuto alla non validità del manuale del mio sistema.

Aggiungo che la rivista è magnifica, e nel futuro sarà ancora più fornita.

Baffigi Armando  
Roma

*È la legge della domanda e dell'offerta che fa sì che i programmi per l'Atom siano pochi. Il fatto è che, purtroppo, non è una macchina molto diffusa.*

*Ci è oscuro comunque il significato della sua richiesta. Se per "codificare" intende dire il codice con il quale quelle istruzioni vengono memorizzate, dobbiamo dire che l'Atom non memorizza le istruzioni con un codice, ma lettera per lettera. Se invece intende chiederci come si "traducono" per l'Atom, dobbiamo no-*

*tare che READ e DATA possono essere simulate come indicato a pag. 92 del manuale italiano, MIDS a sua volta può essere simulata come spiegato a pag. 90 del manuale italiano, la POKE è spiegata a pag. 28 e così via. Ma lei, il manuale, l'ha mai letto?*

*Teniamo infine a precisare che quello italiano è l'esatta traduzione di quello inglese, quindi può trovare anche tutto lì, se possiede solo quello.*

## Tante critiche e qualche lode

Non ho l'abitudine di scrivere alle riviste perché quelle che seguo assiduamente mi soddisfano in pieno o quanto meno mi premunisco non comprandole dopo averle "assaggiate", ma in questo caso... e qui casca l'asino, Voi siete gli unici (fino ad oggi) a pubblicare una rivista vertente su un argomento, lo ammetto, tanto volubile e tanto mutevole sia come idee di realizzazione, sia come concetti che ne sono alla base. Non voglio dire con questo che se domani trovassi in edicola un'altra rivista di software, pianterei P.S. di volata, non fraintendetemi. Voglio però far presente alcune vostre prese di posizione nella gestione della Rivista. Innanzitutto credo che una rivista che si occupi esclusivamente di software, non dovrebbe fidare solo sull'invio dei programmi da parte di chi gli articoli dovrebbe leggerli piuttosto che redigerli. Sono d'accordo sulla pubblicazione dei programmi "questo l'ho fatto io" ma, a mio parere, dovrete essere voi per primi a generare questi articoli e non già rispondere a domande del tipo "come mi suggerite di operare nei riguardi di questo problema" con: "Ci appelliamo alla clemenza dei nostri lettori: se c'è qualcuno tra voi di buon cuore in grado di sciogliere questo nodo, ci scriva mandandoci il listato, una cassetta e due righe di commento!". Leggendo cose del genere su P.S. accompagnate da un sottofondo di "Metodi di protezione della esclusiva esclusività all'uso dei vostri programmi" a dire il vero, ap-

parate in una ben strana luce di "dispensatori" del sapere.

D'altro canto però, devo ammettere che gli articoli che appaiono su P.S. sono quantomeno interessanti quando non sono illuminanti (a proposito: un plauso per il Dizionario di Basic); però quello che cerco di dirvi è che non dovrete aspettare che le montagne vengano a Voi prima di muovere un dito.

Poi c'è un altro fatto che mi fa letteralmente impazzire: la rivista è giovane... il costo della carta... il costo della stampa..., ma si può sapere perché una rivista che si cibi di Enervit Protein (leggi snellissima) e a cui collaborano redattori non professionisti (leggi articoli pagati a persone estranee alla Redazione), debba costare, a paragone di altre riviste, una cifra tanto alta?

In collaborazione con il mio Commodore 64 (un po' di pubblicità non guasta, visto l'orientamento di P.S.), ho svolto questa piccola indagine, riscontrando che P.S. è, in assoluto, la più "cara" in tutti i sensi.

P.S.	p. 68
costo L. 3.500 L./pag.	51.47
M. & P. Comp.	p. 100
costo L. 3.000 L./pag.	30.00
MC Comp.	p. 116
costo L. 3.000 L./pag.	25.86
Comm. Comp. C.	p. 68
costo L. 2.000 L./pag.	29.41
BIT	p. 164
costo L. 3.000 L./pag.	18.29

Cercate di rivedere un po' le cose in modo da non dare un'immagine di rivista parrocchiale ad una pubblicazione di per sé tanto importante.

Siete la prima rivista europea di software, ma a noi non basta: vogliamo che siate la MIGLIORE in Europa e, scusate se tiro un po' l'acqua al mio mulino, la più conveniente come rapporto contenuti/prezzo.

Le mie proposte sono queste:

1. Aumentate il numero delle pagine (che tra l'altro questo mese si sono ridotte a 68 dalle già poche 84!).
2. Universalizzate, nei casi in cui

ciò sia possibile, i programmi, di modo che il Basic prenda il posto dello sfortunato Esperanto.

3. Pubblicate articoli sulle conversioni di dialetto da macchina e macchina.
4. Pubblicate confronti tra metodologie diverse tese a risolvere lo stesso problema.
5. Chi più ne ha, più ne metta.

Scusate infine lo sfogo di un Vostro assiduo lettore, ma le cose che ho scritto mi ronzavano in testa già da un po' di tempo e "quando ce vo', ce vo'", come dicono su in Brianza.

Con la speranza di aver dato anche una pur minima collaborazione, colgo l'occasione per salutarVi distintamente.

Vincenzo Iorio  
Salerno

*Tutte le maggiori riviste scientifiche, ed anche la nostra che è di una particolare divulgazione scientifica, si basano sulle collaborazioni esterne. Provi a dare un'occhiata a BYTE, Microcomputer, Creative Computing, per restare nell'ambito del software. I collaboratori interni, per quanto capaci, non possono avere un'idea brillante ogni 15 giorni, né scrivere un programma complesso ogni mese.*

*In quanto alla sua statistica sul costo della rivista, non ha specificato quali fascicoli ha preso in considerazione; tuttavia abbiamo l'impressione che abbia conteggiato anche le pagine di pubblicità, che per correttezza andrebbero escluse dal computo. Ma probabilmente, anche dopo aver fatto questo calcolo, PS resta la più cara. Le ragioni? Su PS non trovate le solite prove di hardware. Su PS non trovate articoli di "filosofia dell'informatica", né pagine e pagine sul computer a 8 anni (e per noi di 30?); né radici dell'informatica, né tabelle degli stipendi dei programmatori. Su PS trovate idee (come anche il nostro amico lettore ammette), e le idee, interne o esterne, costano.*

*Per quanto riguarda le sue richieste, le stiamo discutendo in redazione. Probabilmente qualcosa di nuovo accadrà.*

## Come collaborare e dizionario di Basic

Spett.le Redazione, possiedo un Texas TI99 4/a con il Basic esteso e avrei intenzione di mandarvi alcuni miei programmi di giochi e soprattutto delle conversioni. Però non so se mandarveli già scritti in Basic esteso (per giochi non riesco più a farne a meno), o mandarveli scritti in TI Basic e lasciare il lavoro di modifica (e soprattutto di miglioria) ai fortunati come me.

Poi volevo chiedere qualcosa sul famigerato "Dizionario di Basic", di cui ho sentito parlare e ne ho letto qualche "pezzo". Ma qui sta il punto, dopo qualche fascicolo la succitata rubrica è letteralmente sparita dalle pagine di PS. Poi ho letto una risposta che avete dato a un lettore dicendo che la state ancora pubblicando.

Dov'è finito il Dizionario di Basic? È una domanda che mi sono posto e a cui non ho trovato risposta. Allora mi sono deciso a scrivervi sperando di avere una risposta.

Complimenti per la vostra favolosa rivista, i miei saluti.

Barbati Alberto  
Milano

P.S.: Gradirei ricevere la "Guida per gli Autori" di cui leggo spesso nella vostra rivista.

*Riguardo alla prima questione, deve decidere lei: nel momento in cui sceglie di collaborare con la rivista, deve fare tutte le considerazioni del caso sull'argomento, di cui lei, in quanto collaboratore, è l'esperto. Per il Dizionario di Basic, che ha fatto preoccupare più di un lettore, nessuna paura. Dopo una breve pausa per motivi tecnici, riprenderà e andrà avanti fino alla Z.*

In questa rubrica rispondiamo alle lettere di carattere generale.

Scrivete a

Personal Software  
Via Rosellini 12  
20124 Milano

# SAVING COMPUTER '83



## La sorgente per le necessità del tuo computer

*Nella nostra sala mostra  
potrai ammirare e provare prodotti come:*

- stampanti
- floppy disk
- programmi
- biblioteca specializzata

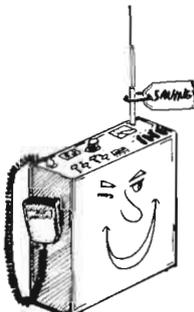
## Le migliori marche di Personal Computer

*Disponiamo infatti pronta consegna  
di APPLE II, APPLE II E, SIRIUS,  
SORD M 23, AVT COMP 2, VIC 20,  
VIC 64, ZX81, SPECTRUM, MPF II*

**Non perdere  
questa occasione!!!**

**DISTRIBUTORI ESCLUSIVI DEL FAVOLOSO  
"THE LAST ONE" PER IL VENETO**

Vendita anche per corrispondenza,  
telefona per le quotazioni, saremo lieti di accontentarti.



# SAVING ELETTRONICA

VIA GRAMSCI 40 - MIRANO (VE) - TEL. (041) 432876

---

# Conversione di programmi per ZX81 e ZX80 nuova ROM

— prima parte —

---

Tutti i segreti per trasformare qualunque programma Basic in un programma per lo ZX81

---

di Bruno Del Medico

**P**ochi possono immaginare la rabbia di un sinclarista il quale, sfogliando *Personal Software*, vede sfilare sotto gli occhi decine di programmi convinto che non potrà mai usarli: programmi per Apple, Tandy, Atari, VIC 20, Atom e chi più ne ha più ne metta. È vero, ci sono anche programmi per ZX81, ma al nostro amico sembrano pochi e inadeguati. Gli altri invece sembrano più belli, hanno sicuramente il fascino del proibito.

In questo articolo vengono illustrate alcune tecniche per convertire programmi scritti per altri microcomputer in un Basic comprensibile allo ZX81 o ZX80 con nuova ROM: quando parleremo del primo intenderemo sempre anche il secondo.

Diciamo subito che la pratica è essenziale in questo genere di esercizi, quindi l'articolo è diviso in due parti: nella prima parte sono elencati dei principi di carattere generale ed alcuni semplici esempi, nella seconda parte affronteremo praticamente due programmi scritti per il PET Commodore e per l'Atari 800, trasformandoli in due programmi per ZX81.

Successivamente toccherà a voi fare dei tentativi. Se riuscite a convertire anche solo un breve programma, facendolo girare sullo ZX81, vedrete poi quanto sarà facile ottenere sempre di più.

## Principi di carattere generale

Le maggiori difficoltà nella conversione di programmi sono determinate dalla diversità dei seguenti elementi:

- gli indirizzi di memoria,
- la grafica.

Tutte le altre differenze, come quelle dovute alla sintassi del Basic o alla mancanza di particolari istruzioni, possono essere superate abbastanza agevolmente.

Non esiste una grammatica che insegni le regole per convertire i programmi. La pratica aiuta moltissimo, quindi per le prime esperienze è consigliabile scegliere programmi abbastanza semplici, evitando in modo particolare quelli che contengono molte istruzioni PEEK e POKE o molte istruzioni READ, DATA e RESTORE. Inoltre è decisamente sconsigliabile scegliere subito dei giochi grafici complicati.

Un programma va riscritto anche più volte in fase di conversione. Possiamo chiamare "passata" ogni successiva riscrittura del programma. Quindi avremo una prima passata, nella quale cominceremo a cambiare alcuni elementi, poi una seconda passata, e se necessario anche passate successive. Nella prima passata consiglio di attenermi ai principi generali illustrati qui di seguito.

## Il LET

Inserite dappertutto il LET. Per esempio la linea di programma:

```
10 Y=22
```

va riscritta così:

```
10 LET Y=22
```

## I nomi delle variabili

Cambiate dappertutto dove è necessario:

- i nomi delle variabili stringa,
- i nomi delle variabili di controllo dei cicli,
- i nomi delle matrici.

Questo perché nel Sinclair tutte queste variabili devono avere un nome composto da una sola lettera. Quindi:

```
10 FOR INC=1 TO 10
20 DIM BUZ(4)
30 LET GIO$="CIAO"
40 NEXT INC
```

deve diventare:

```
10 FOR I=1 TO 10
20 DIM B(4)
30 LET G$="CIAO"
40 NEXT I
```

Non è necessario cambiare i nomi delle variabili numeriche, perché in quel caso sono consentiti nomi con più caratteri. Attenzione a non usare per i nuovi nomi delle lettere già usate nel programma per definire altre variabili. Prima di sostituire FOR INC con FOR I assicuratevi che nel programma non sia già presente una variabile I.

## I numeri di linea

Il Sinclair accetta numeri di linea fino a 9999, mentre altri microcomputer consentono l'uso di numeri di linea fino a 65535. Se nel programma da convertire le linee hanno numeri superiori al 9999, dovette cambiarli. Fate molta attenzione ai GOTO ed ai GOSUB che vanno ugualmente modificati.

## Intero e decimale

Eliminate dove sono presenti i segni % e ! nei nomi delle variabili numeriche; questi segni servono in molti microcomputer per distinguere rispettivamente le variabili intere e quelle decimali. Una tipica linea di programma:

```
10 A%=25/B
```

può essere sostituita con:

```
10 LET A=INT (25/B)
```

Il segno ! può essere ommesso senza conseguenze. Quando non sono associati ai nomi delle variabili numeriche, i segni % e ! possono avere altri significati. In particolare nell'Atom i segni ? e ! sostituiscono i comandi POKE e PEEK.

## Completezza delle istruzioni

Tenete presente che non sono mai consentite abbreviazioni nelle istruzioni. Quindi una linea di programma come questa:

```
10 IF A=25 THEN 200
```

va riscritta così:

```
10 IF A=25 THEN GOTO 200
```

Anche i NEXT vanno definiti con completezza. In molti micro-

computer è possibile usare la forma:

```
10 FOR W=1 TO 10
20 NEXT
```

Nello ZX81 dovrete specificare: NEXT W.

Un'altra abbreviazione non consentita è l'uso del punto interrogativo al posto dell'istruzione PRINT. Su programmi per PET, VIC 20, DAI potrete trovare:

```
10 ? A$
```

In questo caso dovrete riscrivere:

```
10 PRINT A$
```

## I cicli FOR...NEXT

Per quanto riguarda i cicli FOR...NEXT, alcuni microcomputer eseguono il seguente programma

```
10 FOR HAZ=1 TO 0
15 PRINT "OK"
30 NEXT
```

scrivendo OK quando viene dato il run.

Per ottenere il medesimo risultato con il Sinclair occorre usare lo STEP. Quindi:

```
10 FOR H=1 TO 0 STEP -1
15 PRINT "OK"
30 NEXT H
```

CICLO	OUTPUT
10 FOR K=1 TO 2 20 PRINT "OK" 30 NEXT K	2 volte OK
10 FOR K=1 TO 1 20 PRINT "OK" 30 NEXT K	1 volta OK
10 FOR K=1 TO 0 20 PRINT "OK" 30 NEXT K	nessun output
10 FOR K=1 TO 2 STEP -1 20 PRINT "OK" 30 NEXT K	nessun output
10 FOR K=1 TO 1 STEP -1 20 PRINT "OK" 30 NEXT K	1 volta OK

Fig. 1. Contrariamente ad altri microcomputer, nei quali il ciclo FOR NEXT viene sempre comunque eseguito almeno una volta, vi sono dei casi nello ZX81 in cui il ciclo non viene eseguito. Nella figura sono illustrati i più interessanti. Di questa particolarità occorre tenere conto nella conversione dei programmi.

nello ZX81 devono avere un numero fisso di caratteri e ciò comporta principalmente due inconvenienti:

```
10 DIM A$(3,8)
20 INPUT A$(1)
```

Se inserite BUONGIORNO lo ZX81 conserva solo 8 caratteri, BUONGIOR.

```
30 LET A$(2)="CHI"
40 LET A$(3)="SEI"
50 PRINT A$(2);" ";A$(3);
  " ?"
```

Dando il run otteniamo:

```
CHI SEI ?
```

anziché: CHI SEI?, perché i caratteri sono comunque otto per ogni elemento. Quando non se ne può fare a meno, per non guastare l'estetica dell'output occorre inserire una subroutine come la seguente che elimini gli spazi a destra:

```
500 LET B$=A$(x)
510 FOR W=1 TO LEN B$
520 IF B$(LEN B$)< >" "
  THEN RETURN
530 LET B$=B$(TO LEN
  B$-1)
540 NEXT W
550 RETURN
```

x indica l'elemento della matrice. In fase di output naturalmente dovremo stampare B\$ e non A\$(x). La linea 550 assicura il RETURN anche quando la stringa è composta di soli spazi.

### MID\$, RIGHT\$ e LEFT\$

Le funzioni di stringa del Basic standard LEFT\$(A\$,x), MID\$(A\$,x,y) e RIGHT\$(A\$,x) non sono disponibili nel Sinclair. Indicano rispettivamente:

- gli x caratteri più a sinistra di A\$
- gli y caratteri della stringa A\$ partendo dal carattere x
- gli x caratteri più a destra di A\$

Se A\$ = "BUONGIORNO", allora:

```
LEFT$(A$,3) ="BUO"
RIGHT$(A$,4)="ORNO"
```

```
MID$(A$,4,2) ="NG"
MID$(A$,3,1) ="O"
```

Nello ZX81 potete avere rispettivamente:

```
A$(TO 3) ="BUO"
A$(7 TO) ="ORNO"
A$(4 TO 5)="NG"
A$(3) ="O"
```

### CODE, CHR\$

La funzione di stringa ASC di molti microcomputer equivale nella forma alla funzione CODE dello ZX81. ASC(A\$) indica il numero ottenuto premendo il tasto equivalente al primo carattere di A\$, secondo il codice ASCII standard. Se A\$ = "BUONGIORNO" allora:

- in vari microcomputer: ASC(A\$)=66,
- nello ZX81: CODE(A\$)=39,

perché lo ZX81 non usa il codice ASCII. Nella tabella 1 potete leggere, nella colonna al centro, i numeri di codice usati nel Sinclair per le cifre da 0 a 9 e per le lettere da A a Z; nella colonna a destra i corrispondenti numeri di codice ASCII.

Le medesime osservazioni valgono per la funzione CHR\$, per cui

```
PRINT CHR$(66)
```

va tradotto così:

```
PRINT CHR$ 39
```

e provoca la stampa della lettera B.

### DEF FN

L'istruzione DEF FN richiede una breve spiegazione. Sono possibili diverse DEF FN, da DEF FNA a DEF FNZ. Ciascuna può rappresentare una diversa funzione. Vediamo un esempio:

```
10 DEF FNA(Y)=Y*Y+Y
20 K=FNA(5)
30 W=10+FNA(2)
```

Nella linea 20, la variabile numerica K assume il valore di FNA (cioè Y\*Y+Y) dove Y vale 5. Quindi K = 30.

Nella linea 30 la variabile W assume il valore di 10 + FNA (cioè 10+Y\*Y+Y) dove Y vale 2. Quindi W = 16.

Nel Sinclair potremo scrivere:

```
10 LET A$="Y*Y+Y"
20 LET Y=5
22 LET K=VAL A$
30 LET Y=2
32 LET W=10+VAL A$
```

### La funzione RND

La funzione RND può assumere diversi aspetti nei vari microcomputer.

```
10 A=RND(1)
```

0	28	48
1	29	49
2	30	50
3	31	51
4	32	52
5	33	53
6	34	54
7	35	55
8	36	56
9	37	57
A	38	65
B	39	66
C	40	67
D	41	68
E	42	69
F	43	70
G	44	71
H	45	72
I	46	73
J	47	74
K	48	75
L	49	76
M	50	77
N	51	78
O	52	79
P	53	80
Q	54	81
R	55	82
S	56	83
T	57	84
U	58	85
V	59	86
W	60	87
X	61	88
Y	62	89
Z	63	90

Tabella 1. Corrispondenze tra il codice dei caratteri usato dallo ZX81 ed il codice standard ASCII. La colonna a sinistra indica il carattere; la colonna al centro indica il numero di codice corrispondente usato sul Sinclair, e la colonna di destra indica il codice ASCII.

Equivalentente ZX81	PET, VIC 20	DAI	ATOM	APPLE
CLS	PRINT CHR\$(147)	PRINT CHR\$(12)	PRINT\$ 12	
PRINT AT 0,0 (pagina a capo)	PRINT CHR\$(19)	CURSOR 23,0	PRINT\$ 30	HOME
PRINT AT x,y	Disponibile solo la funzione TAB(x). Nel VIC quando x supera 21 si passa alla linea successiva			HTAB x VTAB y
NEWLINE (ENTER)	RETURN	RETURN	RETURN	
A\$=INKEY\$	GET A\$	GETC=X\$		
CODE	ASC	ASC	CH	ASC
LN x	LOG (x)	LOG (x)	LOG (x)	LOG (x)
ARCTAN x	ATN (x)	ATN (x)	ATN (x)	ATN (x)
USR indirizzo	SYS indirizzo	UT	LINK label	CALL indirizzo
STOP	END-STOP	END-STOP	END	END-STOP
PAUSE	WAIT	WAIT TIME		WAIT

Tabella 2. Equivalenza di alcune istruzioni dello ZX81 con quelle di altri microcomputer.

genera di solito un numero casuale da 0 a 1 che può essere anche negativo. Può essere sostituito da:

```
10 LET A=RND
```

oppure:

```
10 LET A=-RND
```

Invece la linea:

```
20 A=RND (150)
```

genera di solito un numero casuale da 0 a 150. Nello ZX81 possiamo scrivere:

```
20 LET A=INT(RND*150)+1
```

ed otteniamo un numero casuale da 1 a 150. Invece con la linea:

```
20 LET A=INT (RND*151)
```

otteniamo un numero casuale da 0 a 150.

### READ, DATA, RESTORE

Le istruzioni READ, DATA e RESTORE servono ad immagazzinare dei dati ed a recuperarli con un certo ordine, e non sono disponibili nello ZX81. Vi sono molti metodi alternativi all'uso di queste istruzioni. Per esempio, i dati possono essere immagazzinati in singole variabili, in linee di programma REM, in matrici numeriche o meglio in matrici stringa. L'argo-

mento è abbastanza vasto e potrà essere oggetto di un prossimo articolo. Il listato 5 fornisce un esempio di conversione delle linee READ e DATA.

### Altre istruzioni

Vi è poi tutta una serie di istruzioni che possono essere convertite semplicemente cambiando il nome: molte vengono illustrate nella tabella 2.

Il punto e virgola associato all'istruzione PRINT conserva sempre la funzione tipica dello ZX81, cioè evita di andare a capo. (Solo

nell'Atom il punto e virgola sostituisce i due punti nel separare più istruzioni sulla stessa linea). Anche la virgola svolge una funzione simile, cioè sposta il cursore di un campo. Però se il campo dello ZX81 è di otto spazi, non è sempre così in altri microcomputer. Il PET spazia 10 colonne, il VIC 20 dodici colonne, il DAI quattordici colonne. I caratteri grafici usati per tracciare linee o segni possono essere sostituiti con caratteri analoghi disponibili sullo ZX81. I caratteri che indicano i semi delle carte (cuori, quadri, fiori e picche) possono essere sostituiti rispettivamente con C, Q, F e P in campo inverso.

```

1 INPUT N
2 INPUT X
3 GOSUB 1090
4 GOTO 1
1090 LET N$=STR$(N)
1100 LET L=LEN N$-2
1110 LET X$=" "
1120 IF L>2 THEN GOTO 1150
1130 LET I=L
1140 GOTO 1180
1150 FOR I=L TO 3 STEP -3
1160 LET X$="." +N$(I TO I+2)+X$
1170 NEXT I
1180 LET X$=N$ (TO I+2)+X$
1190 PRINT TAB X-LEN X$;X$
1200 RETURN

```

Listato 1. Ripropone la routine "PUNTEGGIATURA" apparsa su PS1. La conversione è stata effettuata in una sola passata ed il programma gira, ma presenta il difetto di non punteggiare i numeri composti da 4 cifre.

---

---

# Master mind contro il VIC

---

---

## Un esempio "casalingo" di intelligenza artificiale

---

---

di Paolo Pulli

**Q**uesto programma è una versione del famoso gioco di logica ed intelligenza. È un gioco indubbiamente sfruttato, ma questo programma si differenzia da quelli che ho visto realizzare finora per il fatto che la partita viene giocata anche dal calcolatore che tenterà, con frequente successo, di indovinare il numero segreto del giocatore. È un programma interessante nonostante il suo carattere ludico, in quanto il calcolatore non si riduce a controllare le mosse del giocatore ma gioca attivamente la sua partita.

È un programma apparentemente "intelligente" anche se, è bene precisare, di intelligente non c'è niente perché il calcolatore arriva ad indovinare il numero segreto del suo avversario solo seguendo un algoritmo che spiegherò più avanti.

### Spiegazione del gioco

Ciascuno dei 2 giocatori (voi ed il calcolatore) deve pensare un numero segreto di 5 cifre tutte diverse tra loro, con lo zero che può occupare anche la prima posizione (es. 12345, 02497, ma non 14618). L'avversario dovrà tentare di indovinarlo basandosi sulle risposte ricevute ai propri tentativi. Ad ogni tentativo del proprio avversario si dovrà rispondere con due numeri: il primo è il numero di "+", ossia

di cifre indovinate al posto giusto, ed il secondo è il numero di "-", vale a dire di cifre indovinate ma al posto sbagliato. Il programma lascia il primo tentativo al giocatore e se questi ottiene per primo l'atteso 5+, il calcolatore ha un altro tentativo a disposizione che se è azzeccato porta la partita a concludersi in parità.

Il numero dei numeri pensabili, all'inizio del gioco, è molto elevato e cioè pari a 30240, però il numero di possibilità si riduce già dopo il primo tentativo. Con semplici calcoli combinatori si ha che, a seconda della somma totale dei segni + e - ottenuti al primo tentativo il numero di possibilità  $P$  che restano è:

con 0 o 5  $P < 120$   
con 4 o 1  $P < 3000$   
con 3 o 2  $P < 12000$

Nei casi più sfortunati sono necessari 7 tentativi per essere battuti o per indovinare (a meno di errori logici).

Il programma è stato scritto per il VIC 20 versione standard con 3.5 Kbyte di RAM. Con poche modifiche è immediatamente trasportabile su ogni tipo di PET (è lo stesso interprete Basic 4.00) e anche su altre macchine, in quanto si sono utilizzate solo istruzioni Basic.

Dunque munitevi di carta e matita, rispondete sempre alle domande del calcolatore e soprattutto

chiamano combinazioni di classe  $K$  di  $N$  oggetti distinti tutti i possibili insiemi di  $K$  oggetti distinti scelti tra gli  $N$  oggetti dati (con  $K \leq N$ ) riguardati però come insiemi non ordinati, cioè in ogni insieme non si tiene conto dell'ordine secondo cui si susseguono gli oggetti dell'insieme stesso (es. le combinazioni di classe 2 dei 3 oggetti 1 2 3 sono: 12,23,31).

Si dicono invece permutazioni di  $N$  oggetti distinti tutti i possibili insiemi costruibili disponendo gli  $N$  oggetti in ordine diverso. Ogni permutazione differisce da un'altra soltanto per l'ordine con cui si susseguono gli stessi oggetti. Il numero delle permutazioni di  $N$  oggetti è dato dal fattoriale di  $N$  cioè:  $N!$ .

Si può dire numero pensabile un numero solo ipotizzabile in base al risultato ottenuto da un generico tentativo del calcolatore. Un numero pensabile può non essere logicamente compatibile con i tentativi precedenti. Chiarirò con un esempio come vengono calcolati tutti i numeri pensabili.

Supponiamo che al primo tentativo del calcolatore il giocatore risponda con  $N$  segni complessivi (con  $N \neq 0$  ed  $N \neq 5$ , i casi ovvi  $N=0$  ed  $N=5$  vengono trattati diversamente) non preoccupandoci per il momento di quanti siano + e quanti -. Per ricercare un numero pensabile il calcolatore dovrà mantenere  $N$  cifre tra quelle giocate e sostituire le rimanenti ( $5-N$ ) con cifre scelte tra quelle 5 cifre non giocate (le cifre arabe sono 10).

Ma scegliere  $N$  cifre tra le 5 giocate in tutti i modi possibili significa formare tutte le combinazioni di classe  $N$  di 5 oggetti; e sostituire ( $5-N$ ) cifre, sempre in tutti i modi possibili, equivale a costruire le combinazioni di classe ( $5-N$ ) di altri 5 oggetti che sono le cifre escluse. Per ognuna delle combinazioni del primo tipo dovranno essere tentate tutte quelle del secondo tipo. Seguendo questa strada è possibile costruire un certo numero  $I$  (dipendente da  $N$  e facilmente calcolabile con il calcolo combinatorio) di insiemi di 5 cifre, che differiscono tra loro in almeno una ci-

fra, ancora pensabili dopo il primo tentativo del calcolatore.

Questi  $I$  insiemi non sono però ancora i rimanenti numeri pensabili a cui si vuole arrivare perché per costruzione sono insiemi disordinati. La posizione delle cifre è invece fondamentale nel gioco. Sfruttando però la nozione di permutazione è manifesto che ciascuno di questi insiemi dà origine ad un numero di numeri pensabili pari alle permutazioni di 5 oggetti che sono:

$$5! = 120$$

e il numero totale di numeri pensabili dopo il primo tentativo del calcolatore è dato dal prodotto:  $120 \cdot I$ .

Questo numero (i suoi valori sono quelli riportati a inizio articolo e dipendenti da  $N$ ) è certamente minore dei 30240 numeri pensabili originariamente. Tornando al diagramma di flusso, tutti i possibili insiemi di numeri ancora pensabili vengono costruiti nel blocco segnato con A, mentre le permutazioni di ognuno di questi insiemi vengono formate nel blocco B. I due blocchi sono in realtà nidificati nel programma e non separati come schematizzato sul diagramma di flusso (blocco B interno al blocco A).

Gli insiemi di 5 cifre, le permutazioni, i tentativi ecc. vengono manipolati nel programma come vettori integer a 5 elementi di cui ogni elemento contiene una cifra.

Per costruire gli  $I$  insiemi pensabili e le loro permutazioni si deve quindi partire da un certo tentativo già giocato (nell'esempio era il primo) che si può chiamare tentativo base. Il programma riduce sistematicamente e progressivamente il numero dei numeri pensabili seguendo 3 accorgimenti.

Il primo è quello di tenere come tentativo base quello in cui si è ottenuto il massimo numero come somma dei + e dei -. Ad esempio: se nel primo e nel secondo tentativo il calcolatore ha ottenuto 3 segni complessivi e nel terzo tentativo ottiene 4 segni, quest'ultimo diventa il nuovo tentativo base per costruire i successivi in modo che i

numeri ancora da costruire siano 3000 anche se quelli logicamente possibili saranno molti di meno.

Come secondo accorgimento non vengono più costruiti i tentativi già verificati e scartati. Rifacendomi all'esempio precedente: se al quarto tentativo il calcolatore arriva dopo averne scartati 500 dei 3000 ancora pensabili (il primo numero logicamente possibile è il 501.esimo) e a questo tentativo ottiene nuovamente 4 segni (o un numero minore, cioè non migliora la situazione), riprenderà a formare i suoi numeri pensabili dal 502.esimo, perché è evidente che se i primi 500 erano logicamente incompatibili coi primi 3 giocati, lo saranno sicuramente anche coi primi 4.

Il terzo accorgimento è quello che fa diminuire maggiormente i tempi di risposta. Ho detto come la costruzione del nuovo tentativo avviene in due fasi: prima viene formato in un vettore un insieme di 5 cifre ancora pensabile e poi di queste 5 cifre vengono calcolate in un altro vettore tutte le possibili permutazioni che sono:  $5! = 120$ . Si può capire però che un insieme di 5 cifre sarà candidato a generare una permutazione logicamente possibile solo se, confrontando questo insieme con ciascuno dei tentativi precedenti del calcolatore, si riesce ad ottenere la stessa somma totale di segni. Ciò significa che in questo insieme sono contenute delle cifre che, a prescindere dalla loro posizione, combaciano coi tentativi precedenti. L'insieme è allora valido per essere permutato. In caso contrario è inutile calcolare le 120 permutazioni e per ognuna di queste fare i confronti ma l'insieme viene scartato passando al successivo: il programma risulta molto più veloce.

Tutto questo procedimento viene seguito nel caso generale. Nei casi favorevoli in cui (sempre  $N$  numero totale di segni ottenuti) si ottiene  $N=0$  o  $N=5$  si passa direttamente al calcolo delle permutazioni, nel primo caso cambiando e nel secondo mantenendo tutte le cifre.

vettore L% tutte le possibili permutazioni delle cifre del vettore Y%. Non si sono usati cicli FOR-NEXT per non superare il massimo dei cicli nidificabili. La linea 730 verifica se la permutazione calcolata è logicamente compatibile con i precedenti C-1 tentativi giocati dal calcolatore. In caso affermativo questa diventa il C.esimo tentativo e si rimanda per ricominciare alla linea 250.

900 - 940 Viene interrotta la partita. A questo punto il programma arriva solo dopo avere provato tutti i tentativi ipotizzabili senza averne trovato uno logicamente possibile coi precedenti. Siccome l'algoritmo seguito è un algoritmo esatto, ciò significa che nelle risposte date al calcolatore c'è qualche incompatibilità e cioè o il giocatore si è sbagliato nel darle o ha barato. Le 920-930 stampano tentativi e relative risposte per permettere il controllo al giocatore.

1000 - 1130 Subroutine che ri-

### Segue Listato 1.

```

1000 PRINT PRINT PRINT CHE NUMERO GIOCHI INPUT# A=VAL(Y#)I1 G=0+
1020 FOR D=0T04 P=X/D=INT(A/10*(4-D)) P=A-P*X/D#10*(4-D)NEXT D
1040 P=0 M=0 FOR A=0T04 FOR B=0T04
1050 IF(XND(X)OPW(X)THEN1090
1060 IFD=0THENP=P+1 GOTO1050
1070 M=X/I
1080 NEXT B NEXT A
1100 PRINT#1TAB(10)P;"P="M;" " IFD=0THENRETURN
1120 PRINT#1"HAJ INDOVINATO IN"O;" TENTATIVI HO ANCORA UN TIRO PER CUI LA "
1130 PRINT#1"PARTITA POTREBBE FINIRE PA-RI." F1=1 RETURN
1200 M1=0 P1=0 A=0 D=0
1210 IF(XND(X)OR(X)D)THEN1240
1220 IF A=0 THEN P1=F1+1 GOTO1210
1230 M1=M1+1
1240 D=D+1 IFD=0THEN1210
1250 B=0 A=A+1 IF A=0 THEN1210
1260 RETURN
1300 IF(XND(X)=5)AND(F1=1)THENPRINT#1"LA PARTITA E'PARI" GOTO1050
1310 IF F1=1 THENPRINT#1"HAJ VINTO COMPLIMENTI" GOTO1050
1320 PRINT#1"HO VINTO IN"O;"I TENTATIVI HO STAMMATI 1240
1330 PRINT#1"O CON UN PUNTO " COSTURA."
1340 Y#="" FOR D=0T04 Y#=#FRONT(CSTR(XND(X)I1)NEXT D
1345 PRINT PRINT#1"IL MIO NUMERO E' "Y#
1350 PRINT PRINT PRINT PRINT#1"VOI GIOCARO ANCORA" I1 G=1G+1
1360 INPUT#1 IFD=1 THEN100
1370 END

```

chiede la mossa del giocatore. Il numero introdotto come stringa viene scomposto in cifre e memorizzato temporaneamente nel vettore P%. Poi vengono calcolati in P e M rispettivamente il numero di + e di - per confronto con X%. Se P=5 viene settato il flag F1.

1200 - 1260 Subroutine di conteg-

gio. Effettua i confronti tra le cifre contenute in L% e quelle contenute nella D.esima riga della matrice A%, contando i + ed i - nelle variabili P1 ed M1. È analoga alla parte 1040-1090.

1300 - 1370 Sanciscono la fine della partita con la parità o la vittoria di uno dei 2 contendenti. ■

è in edicola il nuovo numero di

INFORMATICA  
 INFORMATICA  
 INFORMATICA  
 INFORMATICA  
 INFORMATICA  
 INFORMATICA



UNA RIVISTA  
 DEL GRUPPO EDITORIALE JACKSON

---

# Programmare grafici con il TI 99/4A

---

Anche con il Texas si  
possono tracciare  
grafici ad alta  
risoluzione

---

di Roberto Del Giudice

**I**l presente programma permette la rappresentazione grafica di una o più funzioni di una o due variabili (cioè del tipo:  $F(x)$  e  $F(x,y)$ ), ottenendo una buona risoluzione.

Il programma utilizza le funzioni e le istruzioni del Basic standard, risultando perciò piuttosto lento; la mancanza della funzione  $HEX(n)$ , che permette di trasformare un numero dal formato intero in quello equivalente in formato esadecimale e delle funzioni AND, OR e NOT sono la causa primaria di tale inconveniente. Il programma potrebbe risultare ulteriormente più veloce se si potessero inserire più istruzioni per ogni linea di programma.

## Descrizione del programma

- 110 - 120 Inizializzazione parametri.  
130 - 160 Ciclo entro il quale viene definita la funzione da tracciare. I valori X ed Y da essa assunti dovranno essere compresi tra i seguenti estremi:  $0 < X \leq 96$   $0 < Y \leq 80$ ; valori esterni a tali intervalli non vengono elaborati.  
990 - 1050 Inizio della subroutine di costruzione della matrice  $A(N, NY)$  di transcodifica.

Viene controllato che le coordinate del punto da plottare siano nell'intervallo consentito, al-

trimenti ritorna direttamente al programma principale.

1060 - 1080 Trasformazione delle coordinate X, Y in "coordinate carattere".

N = numero del carattere su cui operare; NX e NY = coordinate del pixel da settare all'interno del carattere N-esimo.

1090 - 1130 Operazioni che consentono di poter fare a meno delle funzioni AND e OR. In ogni vettore riga della matrice  $A(N, NY)$  si genera la maschera del relativo N-esimo carattere grafico, utilizzando una rappresentazione binaria in cui i pixel settati sono rappresentati come 1.

Vedi esempio in fondo all'articolo.

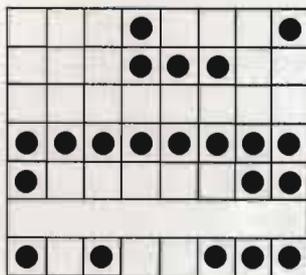
1200 - 1460 Subroutine per la trasformazione della matrice  $A(N, NY)$  in N stringhe C\$ nel formato richiesto dalla CALL CHAR. Si noti come le istruzioni dalla linea 1280 alla 1380 potrebbero essere sostituite, con un conseguente notevole risparmio di tempo macchina, dalla funzione  $HEX$(A$)$  che però non è disponibile in Basic standard.

1500 - 1530 Inizio della subroutine finale che costruisce il diagramma della funzione

$F(x)$  o  $F(x,y)$  sul video.

- 1540 - 1580 Viene "pulito" il video e quindi viene evidenziata l'area destinata al grafico tramite dei punti equidistanti.
- 1590 - 1610 I punti neri sono "trasformati" in punti bianchi.
- 1620 Viene cambiato il colore dello sfondo in nero.

NOTA Il passo di programma 180 ha lo scopo di mantenere il computer in elaborazione, evitando così l'immediata distruzione del grafico, una volta eseguito il programma.



A(6,1)=10001  
 A(6,2)=11100  
 A(6,3)=0  
 A(6,4)=1111111  
 A(6,5)=10000011  
 .....  
 A(6,8)=10100111

Esempio di codifica dei pixel di un carattere.

«PER ACCORCIARE I TEMPI»

il numero di TELEX

GRUPPO EDITORIALE JACKSON

è il seguente:

333436 GEJITI

```

100 OPTION BASE 1
110 DIM A(120,8)
120 E$="10000000"
130 FOR X=0 TO 96
140 Y=(SIN(X/30)+1)*38 (Funzione di esempio)
150 GOSUB 1020
160 NEXT X
170 GOSUB 1500
180 GOTO 180
990 REM SUBROUTINE COSTRUZIONE MATRICE DI TRANSCODIFICA
1000 REM
1010 REM
1020 IF Y>=80 THEN 1140
1030 IF Y<0 THEN 1140
1040 IF X>=96 THEN 1140
1050 IF X<0 THEN 1140
1060 N=INT(X/8)+(INT(Y/8)*12+1)
1070 NX=8-INT(X-INT(X/8)*8)
1080 NY=8-INT(Y-INT(Y/8)*8)
1090 A$=STR$(A(N,NY))
1095 IF (LEN(A$)-NX+1)<=0 THEN 1120
1100 IF SEG$(A$,LEN(A$)-NX+1,1)="1" THEN 1140
1110 REM
1120 D$=SEG$(E$,1,NX)
1130 A(N,NY)=A(N,NY)+VAL(D$)
1140 RETURN
1200 REM SUBROUTINE PER LA CREAZIONE DEI CARATTERI GRAFICI
1210 FOR K=1 TO 120
1220 C$=""
1230 FOR J=1 TO 8
1240 B$="00"
1250 IF A(K,J)=0 THEN 1420
1260 A$=STR$(A(K,J))
1270 H=0
1280 FOR Z=1 TO LEN (A$)
1290 H=H+VAL(SEG$(A$,Z,1))*2 ↑ (LEN(A$)-Z)
1300 NEXT Z
1310 B$=""
1320 A$="0123456789ABCDEF"
1330 FOR Z=INT(LOG(H)/LOG(16)) TO 0 STEP -1
1340 B=INT(H/16 ↑ Z)
1350 B$=B$&SEG$(A$,B+1,1)
1360 H=H-(16 ↑ Z)*B
1370 NEXT Z
1380 REM
1390 IF LEN(B$)<>1 THEN 1420
1410 B$="0"&B$
1420 C$=C$&B$
1430 NEXT J
1440 CALL CHAR(32+K,C$)
1450 NEXT K
1460 RETURN
1500 REM SUBROUTINE DI PRESENTAZIONE DEL GRAFICO
1505 CALL CLEAR
1510 FOR T=33 TO 153
1520 CALL CHAR(T,"18")
1530 NEXT T
1540 FOR I=1 TO 10
1550 FOR J=1 TO 12
1560 CALL VCHAR (17-I,10+J,20+J+I*12)
1570 NEXT J
1580 NEXT I
1590 FOR I=1 TO 16: CALL COLOR(I,16,1): NEXT I
1620 CALL SCREEN(2)
1630 GOSUB 1200
1640 RETURN
    
```

# Picture

---

---

Con il joystick collegato al vostro VIC 20, questo programma vi permette di tracciare disegni in alta risoluzione

---

---

di Matteo Minischetti

**Q**uesto programma permette di creare disegni in alta risoluzione tramite il joystick che si può collegare al VIC 20.

Gira con la memoria base e si compone di due parti:

- listato 1: carica in memoria delle subroutine in linguaggio macchina che gestiscono tutte le opera-

zioni grafiche;

- listato 2: controlla il movimento del joystick e lo fa disegnare.

Il listato 1 va caricato per primo e bisogna salvarlo subito su cassetta, altrimenti quando lo si fa girare si autocancella per fare posto al listato 2.

Dopo aver caricato anche il listato 2 ed averlo salvato, il video pre-

```
100 REM PICTURE PARTE 1
110 POKE 52,28: POKE 51,0
120 FOR K=7168 TO 7389: READ A
130 POKE K,A
140 NEXT: POKE 52,20: POKE 56,20: NEW
200 DATA 160,20,169,0,170,140,10,28,157,0
210 DATA 20,232,208,250,200,192,28,208,241,96
220 DATA 160,1,162,0,130,157,0,30,152,157
230 DATA 0,150,232,208,245,96,0,0,169,0
240 DATA 133,0,133,1,173,1,28,133,2,169
250 DATA 135,56,237,37,28,201,136,176,49,72
260 DATA 41,7,32,109,28,104,41,248,170,160
270 DATA 15,32,109,28,138,136,208,249,173,36
280 DATA 28,281,120,176,23,72,41,248,32,107
290 DATA 28,104,41,7,170,152,56,106,202,16
300 DATA 252,133,0,232,1,1,129,1,96,216
310 DATA 24,101,1,133,1,169,0,101,2,133
320 DATA 2,96,169,0,141,36,28,32,38,20
330 DATA 238,36,28,208,248,96,169,0,141,37
340 DATA 28,32,38,28,238,37,28,208,240,96
350 DATA 165,0,73,255,162,0,33,1,76,100
360 DATA 28,32,20,28,169,143,162,162,160,253
370 DATA 32,202,28,169,19,162,50,160,8,32
380 DATA 212,28,96,169,150,162,174,160,240,32
390 DATA 202,28,169,12,162,38,160,25,32,212
400 DATA 28,96,141,2,144,142,3,144,140,5
410 DATA 144,96,141,0,144,142,1,144,140,15,144,96
```

Listato 1. Routine di caricamento delle subroutine in linguaggio macchina.

senterà il quadrato grafico con al centro il cursore che in questo caso è un pixel. Muovendo il joystick, ci si lascia dietro una linea luminosa, che, sfruttata adeguatamente, compone il disegno. Premendo contemporaneamente al joystick il fire-button, ci si può muovere senza disegnare si possono cancellare i pixel indesiderati col solo passaggio del pixel-cursore sopra di essi. Se si esce dai bordi del quadrato grafico, il pixel-cursore viene riposizionato al centro dello schermo. Buon divertimento!

### Principali routine del listato 2

- 10 - 50 Riempie il vettore JS con le possibili posizioni del joystick.
- 60 - 100 Cancella la mappa dei punti, disegna i bordi del quadrato, pone il pixel-cursore al centro.
- 110 - 900 In base allo spostamento del joystick muove il cursore sullo schermo.
- 1000 - 1080 Determina se il joystick è stato usato.
- 2000 - 2020 Se il cursore esce dai bordi, viene riposizionato, viene stampata la nuova posizione, se FB è premuto, viene cancellato il pixel corrispondente all'attuale posizione. ■

```

5 REM PICTURE PARTE 2
10 DIM JS(2,2)
20 J4="791682540"
30 FOR I=0 TO 2: FOR J=0 TO 2
40 JS(J,I)=VAL(MID$(J4,3*I+J+1,1))
50 NEXT J,I
60 AX=7204: OY=7205: OYS 7108: OYD 7029
70 POKE OY,0: OYS 7200: POKE OY,105: OYS 7020
75 POKE AX,0: OYS 7004: POKE AX,110: OYS 7004
80 POKE AX,00: POKE AY,00
100 OYS 7200: GOSUB 1000
110 IF P=0 THEN 100
120 ON P GOTO 300,400,500,600,700,800,900
200 Y=Y+1: GOSUB 2000: GOTO 100
300 X=X+1: Y=Y+1: GOSUB 2000: GOTO 100
400 X=X+1: GOSUB 2000: GOTO 100
500 Y=Y-1: X=X+1: GOSUB 2000: GOTO 100
600 Y=Y-1: GOSUB 2000: GOTO 100
700 Y=Y-1: X=X-1: GOSUB 2000: GOTO 100
800 X=X-1: GOSUB 2000: GOTO 100
900 X=X-1: Y=Y+1: GOSUB 2000: GOTO 100
1000 POKE 37100,0: POKE 37104,12
1010 C=PEEK(37107)
1020 S0=((RAND4)=0)
1030 S1=((RAND8)=0)
1040 S2=((RAND16)=0)
1050 FB=((RAND32)=0)
1060 S=PEEK(37102)
1070 C0=((RAND120)=0)
1080 POKE 37104,200: A=1+S2+S3: B=1+S0+S1: P=JS(A,B): RETURN
2000 IF X=0 OR X=110 OR Y=0 OR Y=100 THEN P=00: Y=00
2010 POKE AX,X: POKE AY,Y: IF FB=1 THEN OYS 7010
2020 RETURN

```

Listato 2. Programma che controlla il movimento del joystick e lo fa disegnare.

leggete  
**VIDEO  
 Giochi**

**la prima rivista di videogames - computer  
 giochi elettronici**

# Quattro programmi per il Commodore 64

Ognuno di questi programmi illustra una particolare caratteristica di questo potente personal

di Carlo Sintini

Ringraziamo la Kiber Italia che ci ha permesso la pubblicazione di questi programmi

## Compilazione automatica degli sprites

**I**l Commodore 64 ha, fra gli altri pregi, la possibilità di realizzare fino ad 8 figure ad alta definizione, chiamate *sprites*, che possono essere gestite autonomamente una dall'altra e permettono, per esempio, di ottenere videogame assolutamente favolosi.

Però il programma in Basic è piuttosto laborioso, a causa dei DATA che devono essere calcolati con operazioni semplici ma noiose.

Questo programma va considerato come una utility: visualizza un reticolato entro il quale, fornendo le coordinate opportune, possono essere inseriti i punti che costituiscono il disegno dello sprite.

L'operatore può cancellare a volontà il disegno per eventuali correzioni e alla fine, premendo la chiocciolina, il Commodore passa al calcolo dei DATA.

Si può scegliere tra due possibilità: avere il semplice elenco dei DATA, o un programma completo da battere sulla tastiera (per esempio per realizzare un gioco).

Per rendere più comodo e facile l'uso del programma ho usato solo comandi GET (linee 240, 510, 740, 990, 1140, 1280, 1370, 1510), in modo da eliminare la necessità di premere il tasto RETURN dopo ogni risposta. Il cursore che lampeggia è fasullo, ed è generato dalla routine 460-580, che contiene anche le istruzioni necessarie per impedire l'accettazione di dati illogici.

### Listato 1. Compilazione automatica degli sprites.

```
100 REM CARLO SINTINI
110 CLR:PRINT"Q":B=7
120 POKE53280,B:POKE53281,B
130 PRINT"*****"
140 PRINT"  "
150 PRINT"  COMPILAZIONE AUTOMATICA DEGLI  "
160 PRINT"  "
170 PRINT"  S P R I T E S  "
180 PRINT"  "
190 PRINT"*****"
200 PRINT"  "
210 PRINT"  P.LE ASIA N.21 - 00144 ROMA EUR"
220 PRINT"  TEL. 06/5916435"
230 PRINTTAB(4>)"  "SOLO L'ELENCO DEI DATA ?"
240 GETQ$:IFQ$=""THEN240
```

(segue)

## Segue Listato 2.

```
1070 POKETV=47+60*R1+18+40.32
1080 FL(R1)=1:M=M+1
1090 FORK=0TOLEN(V*(R1))-1
1100 KK=TV-41+44+40*R2+R3+k*R2(R1)+k*40*R1(R1)+40
1110 IF PEEK(KK)<30 THEN POKKK,PEEK(KK)+128
1120 NEXT
1130 IFA=10 THEN I1210
1140 GOTO 940
1150 REM FINE GIOCO
1160 FORK=11010:PRINT0#"HAI PERSO !!!"
1170 FORJ=110300:NEXT:PRINT0#"HAI PERSO !!!":FORJ=110300:NEXT:NEXT:PRINT0#H#
1180 C4=""
1190 FORK=0109:H(K)=0:NEXT:M=0
1200 GOTO 1270
1210 I=11/60
1220 PRINT0#H#:FORK=11010:PRINT0#"HAI VINTO !!!"
1230 FORJ=110300:NEXT:PRINT0#"HAI VINTO !!!":FORJ=110300:NEXT:NEXT:PRINT0#H#
1240 PRINT0#"HAI VINTO IN "INT(K)/"SECONDI"
1250 GOTO 1320
1260 REM USCITA PER TEMPO SCADUTO
1270 PRINT0#"ECCO LA SOLUZIONE !"
1280 FORR1=0T09:FORK=0TOLEN(V*(R1))-1
1290 KK=TV-41+44+40*R3(R1)+R4(R1)+k*R2(R1)+k*40*R1(R1)+40
1300 IF PEEK(KK)<30 THEN POKKK,PEEK(KK)+128
1310 NEXT:NEXT
1320 PRINT#"BUONI RIPROVARE ?"
1330 GETY#:IFY#="" THEN I330
1340 IFY#="S" THEN RUN
1350 PRINT#"OK - CIAO !!!"
1360 END
1370 REM VOCABOLI DI BASE
1380 DATAGINEVRA,PORPORA,WEEKEND,SUSANNA,LINDA,VIOLETTA,FRANCESCO
1390 DATA COMODURE,AZALEA,CHARLO,BUCAREST,MARRONE,PAPERINO,CAMOSCIO
1400 DATA KAMIKAZE,TORONTO,RAFFAELE,HANDICAP,CHIOCCIA,AMARANTO,TOKYO
1410 DATA NEWYORK,MIRELLA,BUDAPEST,TIZIANO,CANGURO,PRIMULE,CATULLO
1420 DATA ARANCIONE,COMPUTER,TOFOLINO,QUADRO,MADRID,ANGELO,ANTILOPE
1430 DATAGIUSEPPE,SNOOPY,TULIPANO,VIENNA,ANTONIO,SQUADRA,INDACO
1440 DATA JUVENTUS,PANDA,ANEMONI,AZZURRO,PECHINO,REATTORE
1450 DATA MOUQUETTE,SAHOVAR
```

## Agenda telefonica

I calcolatori della Commodore hanno tutti una particolare prerogativa: quando il cursore si trova su una linea dello schermo (nuova o modificata) e si preme il tasto RETURN, la linea di programma passa dalla memoria di schermo alla memoria centrale.

Questo meccanismo può essere sfruttato vantaggiosamente in molti modi. Per esempio, come in questo programma, per inserire automaticamente nel programma stesso nuove linee contenenti degli indirizzi sotto forma di DATA.

Normalmente i programmi di questo tipo adoperano i file per immagazzinare i dati, ma i file su cassetta (la maggior parte degli utenti del Commodore 64 si serve di regi-

stratore a nastro) sono lunghi e macchinosi.

La ricerca degli indirizzi con questo programma è invece velocissima e pratica. Naturalmente, dopo ogni aggiornamento degli indirizzi, il programma deve essere nuovamente registrato con le modifiche. La chiave del meccanismo risiede nella linea 730.

Nelle linee 700, 710, 720 vengono stampate sullo schermo:

- una nuova linea di programma contenente il nuovo indirizzo;
- la modifica della linea 1000 che contiene il numero N dell'ultima linea del programma;
- il comando diretto di saltare alla linea 540.

Finalmente nella linea 730 vengono "forzati" automaticamente tre RETURN successivi, e il pro-

gramma si è allungato con una nuova linea contenente il nuovo indirizzo.

Per la ricerca del nome potete anche battere le prime due o tre lettere: verranno visualizzati *tutti i nomi* che iniziano con quelle lettere.

#### Segue Listato 4.

```

1040 DD=DD+1
1050 IF YY=0 THEN DN=INT(16*311):GOTO1080
1060 IFT=3 THEN DN=5:GOTO1080
1070 DN=INT(3+(CT-2))
1080 D(0)=D(0)+1
1090 IF D(0)<=DN*(D-7) AND YY=0 THEN 1150
1100 IF D(0)<=DN*D AND YY=1 THEN 1150
1110 D(0)=1:D(1)=D(1)+1
1120 FOR W=1 TO 10
1130 IF D(W)>2 THEN D(W)=1:D(W+1)=D(W+1)+1
1140 NEXT W
1150 PP*(J)=D*(D*DD):J:GOTO1410
1160 REM TRE TRIPLE
1170 NT=NT+1
1180 IF NT=1 THEN READ T#
1190 IF LEN(T#)>3 THEN RESTORE:GOTO1180
1200 Q0#=MID$(T#,NT,1)
1210 PP*(J)=Q0#:GOTO1410
1220 REM QUATTRO TRIPLE
1230 NT=NT+1:IF NT>4 THEN NT=NT-4:GOTO1310
1240 IF NT=1 THEN READ T#
1250 IF LEN(T#)<4 THEN 1240
1260 IF LEN(T#)>4 THEN RESTORE:GOTO1240
1270 Z#=MID$(T#,NT,1)
1280 PP*(J)=Z#:GOTO1410
1290 REM TRIPLE INTEGRALE
1300 TT=TT+1
1310 IF YY=0 THEN TN=16:GOTO1330
1320 IFT>3 THEN TN=9
1330 T(0)=T(0)+1
1340 IF T(0)<=TN*(T-4) AND T>3 THEN 1400
1350 IF T(0)<=(TN*T) AND T<3 THEN 1400
1360 T(0)=1:T(1)=T(1)+1
1370 FOR W=1 TO 10
1380 IF T(W)>3 THEN T(W)=1:T(W+1)=T(W+1)+1
1390 NEXT W
1400 PP*(J)=T*(T(TT))
1410 P#=P#+PP*(J)
1420 NEXT J
1430 IF D#="XXXXXXXX" THEN RESTORE
1440 ND=0:NT=0:DD=0:TT=0:GOSUB1640
1450 IF C=S THEN PRINT:PRINT:END
1460 NEXT C
1470 END
1480 REM BARRA ORIZZONTALE
1490 FOR K=1 TO 40:PRINT" ";:NEXT:RETURN
1500 REM INGRESSO GET
1510 GET W#:IF W#="" THEN 1510
1520 W=ASC(W#):RETURN
1530 REM USCITA PER CALCOLO BANALE
1540 PRINT"LO SVILUPPO E' TROPPO ELEMENTARE!!!"
1550 PRINT"NON VALE LA PENA ADOPERARE UN CERVELLO"
1560 PRINT"ELETTRONICO PER UN CALCOLO COSI' SEN-"
1570 PRINT"PLICE ; BASTA UNA MENTE UMANA !"
1580 PRINTTAB(25)"CIAO!!!"
1590 END
1600 REM RIEMPIMENTO D*(1,K) D*(2,K)
1610 FOR K=1 TO 13:IF V(K)=1 OR V(K)=3 THEN 1630
1620 D*(1,K)=LEFT$(R*(K),1):D*(2,K)=RIGHT$(R*(K),1)
1630 NEXT:RETURN
1640 REM SVILUPPO COLONNE SULLO SCHERMO
1650 IF N=0 OR N=8 THEN N=0:CV=CV+1:GR=GR+1:GOTO1680
1660 N=N+1:CC=CC+1
1670 GOTO1710

```

(segue)

#### Segue Listato 4.

```
1680 IFCV>4THENCV=1:CC=0:PRINT"PREMI UN TASTO":PRINT:GOSUB1500:PRINT"0"  
1690 PRINT"XXXXXXXXXXXXXXXXXXXX"SPC(CV*9-8)"GR."█  
1700 GOTO1660  
1710 FORR=1TO13  
1720 P*(R,CC)=MID*(P*(R,1)  
1730 PRINTX*(R)SPC(N+CV*9-10)P*(R,CC)  
1740 NEXT  
1750 RETURN  
1760 REM DATA  
1770 DATA"X21","21X","2X2","112","1XX","1111","XXX1","2221","1X21"  
1780 DATA"X21X","21XX","12X2","X122","2X12"  
1790 DATA"111111","1X1111","111X1X","1XX1XX","111X1XX","1X1XX1X"  
1800 DATA"11XXX1","1XXX11","X111XX","XX111X","X1X11X1","XXX1X11"  
1810 DATA"X11XX11","XX1X1X1","X1XX11X","XXXXXXX"
```

L'unico mass-media  
di strumenti musicali  
e audio-registrazione.

strumenti  
**MUSICALI**



**GRUPPO  
EDITORIALE  
JACKSON**

QUANDO JACKSON NON È SOLO ELETTRONICA E INFORMATICA.

## Quando il computer parla il linguaggio delle immagini

La computer grafica rappresenta un campo di applicazione dell'informatica relativamente nuovo, ma suscettibile di imprevedibili sviluppi. Questo volume, nato in collaborazione con alcune delle più specializzate istituzioni del settore, esamina tutte le possibilità di questa scienza nuova e affascinante: dall'animazione cinematografica e televisiva ai business graphics; dalla

progettazione in architettura a quella in elettronica e in meccanica; dalla mappazione alla manipolazione tridimensionale delle immagini... Realizzata in modo da permettere un rapido, ma esauriente approccio all'argomento, l'opera si rivolge a quanti (lettori-utenti) siano alla ricerca dei necessari chiarimenti per una corretta e proficua utilizzazione delle tecniche di Computer grafica.

**Mauro Salvemini**

# COMPUTER GRAFICA

176 pagine. Lire 29.000  
Codice 519 P

**GRUPPO  
EDITORIALE  
JACKSON**



Per ordinare il volume utilizzare  
l'apposito tagliando inserito in fondo alla rivista



# 8080 HARDWARE E SOFTWARE

**SCONTO 10%**  
per gli abbonati  
alle riviste JACKSON

## 8080A/8085 PROGRAMMAZIONE IN LINGUAGGIO ASSEMBLY

Il libro esamina il linguaggio assembly. Spiega la programmazione in linguaggio assembly, descrive le funzioni di assembler e le istruzioni assembly, tratta i concetti di sviluppo del software di base. Esamina esempi di programmazione da un semplice ciclo di caricamento della memoria a un completo progetto di programma. Offre, gli strumenti di debugging, la relativa procedura di base, i tipi più comuni di errori. Fornisce, inoltre, esempi di programmi pratici.

**Sommario**  
Introduzione alla programmazione in linguaggio assembly - Assembler - Set d'istruzioni per il linguaggio assembly - Semplici programmi - Semplici cicli di programma - Dati codificati come caratteri - Conversione del codice - Problemi aritmetici - Tabelle e liste - Subroutine - Ingresso/ Uscita - Interrupt - Definizione del problema e progetto del programma - Debugging e testing - Documentazione e riprogetto - Progetto campione.

Pagg. 490      Formato 14,5 x 21  
Prezzo L. 27.500      Codice 323P

## ESPERIMENTI CON TTL E 8080A VOL. 1

Senza alcuna particolare esperienza in elettronica il libro, l'edizione riveduta e corretta del famoso Bugbook V, insegna come programmare un microcomputer, come interfacciarlo verso dispositivi esterni e come questi dispositivi operano da un punto di vista digitale.

**Sommario**  
Codici digitali - Introduzione alla programmazione dei microcomputer - Alcune istruzioni del microcomputer 8080 - Il microcomputer MMD-1 - Alcuni semplici programmi per il microcomputer 8080 - Registri e istruzioni relative ai registri - Porte logiche e tabelle della verità - Istruzioni logiche - Introduzione al breadboarding - Circuiti integrati - Flip-flop e latch - Decodificatori - Contatori - Gating di segnali digitali - Multivibratori monostabili e astabili.

Pagg. 490      Formato 15 x 21  
Prezzo L. 22.000      Codice 005A

## TEA UN EDITOR ASSEMBLER RESIDENTE PER L'8080/8085

Il programma TEA, il cui listing è riportato nell'ultima Appendice del libro, una volta caricato su un supporto di memoria, dà la possibilità di scrivere e modificare programmi sorgente scritti in assembler secondo i codici mnemonici di questi due microprocessori.

**Sommario**  
Modalità d'impiego dell'editor - Modalità d'impiego dell'assembler - Convenzioni relative ai dispositivi di I/O - Formato della banda per il programma TEA - Modifica del comando Z - Aggiunta a TEA di un comando speciale - Elenco riassuntivo dei comandi di TEA - Listing di TEA.

Pagg. 252      Formato 15 x 21  
Prezzo L. 14.000      Codice 322P

## DEBUG

Un programma interprete per la messa a punto del software 8080

Il libro presenta il programma DEBUG che permette di inserire e cambiare i passi di programma, procede attraverso una istruzione completa e non passo passo, è in grado di perforare e leggere un nastro di carta. Nelle appendici, poi sono riportati due listing uno in codice ottale che contiene subroutine ottali di I/O, l'altro in codice esadecimale.

**Sommario**  
Un'introduzione alla programmazione - DEBUG: Un'introduzione - I comandi di DEBUG - Come si legge e si perfora un nastro - Come si esegue un programma - L'uso del breakpoint - La tecnica del passo passo - Che cosa non fare con DEBUG - Salti, richiami, restart e rientri - Le operazioni relative allo stack - Come dare inizio al programma DEBUG nelle memorie di lettura scrittura - Come dare inizio al programma DEBUG in PROM - Come cambiare DEBUG - Subroutine generali e disposizione dell'utente - Format della banda per DEBUG - Sommario del set di istruzioni - Riassunto dei comandi - Listing ottale di DEBUG - Listing esadecimale di DEBUG.

Pagg. 106      Formato 15 x 21  
Prezzo L. 7.000      Codice 313P

## 8080A/8085: PROGRAMMAZIONE IN LINGUAGGIO ASSEMBLY

ESPERIMENTI CON  
TTL e 8080A

ESPERIMENTI CON  
TTL e 8080A

PROGRAMMAZIONE  
DELL'8080  
E PROGETTAZIONE LOGICA

IL BUGBOOK VII  
MICROCOMPUTER-  
CONVERTITORI ANALOGICI

## ESPERIMENTI CON TTL E 8080 VOL. 2

Il libro che rappresenta l'edizione riveduta e corretta del famoso Bugbook VI, completa la trattazione del Vol. 1 (già Bugbook V)

**Sommario**  
Cosa vuol dire interfacciare - Impulsi di selezione dispositivo - Il set di istruzioni 8080A - Le tecniche di bus dati nell'uso di dispositivi tri-state - Introduzione alle tecniche di I/O tramite l'accumulatore - Introduzione alle tecniche di Ingresso/Uscita memory-mapped - L'ingresso/uscita del microcomputer: alcuni esempi - Flag e interruzioni.

Pagg. 495  
Prezzo L. 22.000

Formato 15 x 21  
Codice 006A

## IL BUGBOOK III Interfacciamento e programmazione del microcomputer 8080

Il libro verte su quattro punti fondamentali dell'interfacciamento di microcomputer: le generazioni degli impulsi di selezione dispositivo, le uscite dei microcomputer, la gestione d'interrupt e le associate tecniche software di I/O nell'ambito di un sistema a microprocessore basato sull'8080.

**Sommario**  
Cosa è un microcomputer? - Un piccolo microcomputer basato sull'8080 - Un'introduzione alla programmazione dei microcomputer - Come si genera un impulso di selezione dispositivo - Cicli di clock e loop di timing - Generazione delle informazioni di stato - Input/Output del microcomputer - Subroutine, interrupt, flag esterni e stack - Riferimenti - Set d'istruzione dell'8080  
Sommario del set di istruzione dell'8080  
Sommario del set di istruzione dell'8080 (Intel Corp.)

Pagg. 417  
Prezzo L. 19.000

Formato  
Codice 003A

## PROGRAMMAZIONE DELL'8080 E PROGETTAZIONE LOGICA

Il libro descrive l'implementazione della logica sequenziale e combinatoria con l'uso del linguaggio assembly all'interno di un sistema a microcomputer basato sull'8080. Un capitolo, poi contiene il set completo di istruzioni dell'8080.

**Sommario**  
Introduzione - Linguaggio assembly e logica simulazione diretta della logica digitale - Un semplice programma - Prospettiva del programmatore - Set di istruzioni - Alcune subroutine impiegate comunemente - Codici di caratteri ASCII.

Pagg. 296  
Prezzo L. 19.000

Formato 14,5 x 21  
Codice 325P

## IL BUGBOOK VII

Il libro fa capire come un sistema a microprocessore si interfaccia al mondo esterno, e con particolare riguardo all'interfacciamento di convertitori, sia digitali-analogici che analogici-digitali, con microelaboratori basati su 8080, 8080A, 8085 e Z80. Vengono presentati, inoltre molti esempi di interfacciamento, completi di schemi elettrici e listing dei programmi.

**Sommario**  
Interfacciamento dei convertitori da digitale ad analogico - Interfacciamento con convertitori analogico-digitali - Convertitori analogico digitali a doppia pendenza e pannelli di misura digitali - Miscelanea di tecniche di conversione - Circuiti di sample-and-hold e dispositivi multiplexer - ... dei bit, schede e scatole nere - Esperimenti con convertitori digitali-analogici e analogico-digitali - Specifiche dei convertitori A/D e D/A - Data sheets di una selezione di prodotti, convertitori.

Pagg. 270  
Prezzo L. 17.000

Formato 15 x 21  
Codice 007A



**GRUPPO EDITORIALE JACKSON**  
Divisione Libri

---

# Gestire file con il PET/CBM

---

Facciamo un po' di luce sui misteri della gestione file del PET/CBM

---

di Stefano De Monte

**U**n problema che spesso assilla un possessore di personal vate programmi e dati su disco magnetico.

Per fare ciò esistono delle istruzioni particolari da dare all'unità magnetica tramite l'interprete Basic: per il primo caso quelle conosciute sono LOAD (per caricare da disco) e SAVE (per conservare su disco). Tuttavia se volete conservare dei dati (le vostre spese, gli incassi ecc.) queste due istruzioni non basteranno. Scopo di questo articolo sarà quello di illustrare il funzionamento di un'unità a disco e dei principali comandi a cui questa unità obbedisce.

## I comandi del Basic 3.0

Una volta inserito il disco, per usare l'unità magnetica bisognerà che il nostro sistema si metta in comunicazione con la periferica desiderata: a ciò serve (ad esempio per la configurazione PET 3032, unità 3040) l'istruzione OPEN 1,8,15 che apre il file logico di comunicazione (1) verso l'unità floppy (8) e che riserva questo file per dare comandi o segnalare errori (15). Da notare che, una volta data questa istruzione nel corso di un programma, non ci sarà più bisogno di ripeterla, pena la segnalazione di "FILE OPEN

ERROR" che, automaticamente, chiuderà il file.

Se il disco inserito sarà nuovo dovremo formattarlo per mezzo dell'istruzione

PRINT#1,"N dr:dfn,xx"

dove *dr* sta per il drive (0 o 1), *dfn* è il nome del disco e *xx* è il suo numero di identificazione. Attenzione però a non formattare un disco contenente già dati o programmi: cancellereste tutto! (Infatti la N sta per NEW.) Invece se il nostro disco è già stato usato altre volte bisognerà inicializzarlo: faremo così conoscere all'unità floppy i contenuti del disco.

L'istruzione è PRINT#1 "I *dr*", dove *dr* è il numero del drive in cui è contenuto il disco.

A questo punto possiamo finalmente trasferire dei dati (numeri o stringhe) sul disco magnetico. Ma, per poterlo fare, bisogna prima riservare una parte di memoria del disco nella quale mettere i nostri dati; inoltre a questa parte dobbiamo dare un nome che ci permetta di individuarla ogni qualvolta vogliamo, per recuperare i dati contenuti in essa.

L'istruzione completa è OPEN *fc,8,sa,"dr:fn.ft,mode"* (ma su un manuale in italiano è riportato *solo* OPEN *fc,8,sa* e ciò è errato!). Il *fc* è il file logico riservato dall'unità per comunicare; 8 è il codice dell'unità floppy; *sa* è il "secondary address" (indirizzo secondario); *dr*

è il numero del drive; *ft* è il tipo di file (Sequential, User, Relative, Program); il *mode* può essere Read (lettura) o Write (scrittura). Bisogna cioè dire all'unità "intelligente" floppy attraverso quale file comunicare, su quale drive è diretta la comunicazione, il nome ed il tipo di file che interessa ed, infine, se si vuol leggere o scrivere su di esso.

Per trascrivere un dato (numero A o stringa A) su di un file, precedentemente aperto e predisposto per la scrittura, l'istruzione è PRINT#*fc*, A. Se invece vogliamo scrivere una serie di dati, le cose si complicano: bisognerà separare i dati (record) gli uni dagli altri, altrimenti scriveremmo sul disco un dato unico (non 1; 2; 3 ma 123). Per separarli il PET usa il CHR\$(13) che corrisponde ad un carriage return.

La forma finale sarà

```
PRINT#fc,A;CHR$(13);B;
CHR(13);...
```

Abbiamo così inserito una serie di dati con un certo ordine in un certo file, ma è importante notare che ora il dato immesso (numero o stringa) non possiede più un nome (A o A\$) ma la sua identificazione dipende solo dall'ordine in cui i vari dati vengono scritti nel file. Questo è essenziale per sapere come "recuperare" i dati; ma ciò si potrà fare solo dopo aver chiuso il file con un CLOSE *fc* ed averlo riaperto specificando che ora si vuol leggere su quel file (selezionando cioè il "mode" dell'OPEN). A questo punto basterà dare un INPUT#*fc*,A,B,... ed al primo dato contenuto nel disco verrà assegnata la variabile A, al secondo la variabile B e così via (per le stringhe useremo A\$,B\$ ecc.). Ancora una volta, prima di procedere col programma, dobbiamo chiudere il file con un CLOSE *fc*.

Da notare che, con queste istruzioni, una volta scritto nel file ed averlo chiuso, non è possibile aggirare l'ostacolo, ci si può servire dell'operazione di concatenamento di file (tabella 1).

#### Basic 3.0

```
SAVE "dr:fn",8
LOAD "dr:fn",8
OPEN 1,8,15
PRINT#1,"I dr"
PRINT#1,"S dr:fn"
PRINT#1,"C ddr=sdr"
PRINT#1,"C dr:dfn=
dr:sfn 1,dr:sfn 2,..."
PRINT#1,"N dr:dnome,
xx"
```

dr=numero drive  
fn=nome file  
sdr=dal drive 0 o 1  
ddr=al drive 0 o 1  
sfn=dal file

```
OPEN fc,8,sa"dr:fn,ft,mode"
OPEN 3,8,3 "0:PROVA,S,W"
```

#### Basic 4.0

```
DSAVE, "fn",D dr
DLOAD "fn",D dr
?DS
PRINT#1,"I dr"
SCRATCH"fn",D dr
COPY Dsdr TO Dddr
CONCAT Ddr,"sfn" TO
Ddr,"dfn"
HEADER"nome",D dr,I xx
```

a cui seguono:

inizializza  
cancella  
copia il disco  
concatenamento  
di file  
formatta il disco

dfn=al file  
sa=indirizzo secondario  
fc=file di comunicazione  
ft=tipo di file

in pratica:

NOTA: I caratteri in maiuscolo sono obbligatori, quelli in minuscolo debbono essere sostituiti con dati dal programmatore.

Tabella 1.

#### Listato 1. Il programma "Gestione grafico + pellicole".

```
50 REM*****
52 REMGESTIONE+GRAFICO PELLICOLE
55 REM*****
60 GOSUB5000
61 DIMA$(20)
70 PRINTTAB(10)"PROCEDURA PELLICOLE"
75 PRINTTAB(5)"DISCO IN DRIVE 0":FORK=1101000:NEXT
80 REM INIZIALIZZA DISCO
81 OPEN1,8,15
82 PRINT#1,"I0"
85 PRINT CHR$(147)
90 PRINTTAB(5)"SCRIVI LE PRIME 2 LETTERE"
91 PRINTTAB(5)"DEL MESE:1MAGGIO=MG"
95 INPUTG$:I=1
99 REMROUTINE CONTROLLO MESE
100 DATA GE,FE,MA,AP,MAI,GI,LU,AG,SE,OT,NO,DI
110 READA$(I)
120 IFA$(I)=$GOTO150
130 IFLA$(I)THENI=I+1:GOTO110
140 PRINT"ISCRIVI I":FORUU=110500:NEXTUU:RESTORE:GOTO55
150 INPUT"NUMERO DEL GIORNO":N
160 PRINT"SE IMMETTI DATI PREMI I"
161 PRINT
162 INPUT"PER IL RESOCONTO 2":A
170 IFA=26GOTO450
175 PRINTCHR$(147):REM IMPIAGAZZINAMENTO DATI
180 PRINTTAB(10)"PELLICOLE SECONDO IL FORMATO"
190 INPUT"FORMATO 110":B
195 INPUT"FORMATO 126":C
200 INPUT"FORMATO 135":D
210 INPUT"DIAPOSITIVE":F
220 B1$="0":B2$="S,W":B3$="S,R"
225 C1$=B1$+G$:C2$=B1$+G$+B3$
230 PRINT"PRIMA OPERAZIONE DEL MESE ? S/N"
235 INPUTF$
251 PRINT:PRINT:PRINT
253 INPUT"CONFERMA S/N":LA$
255 IFLA$="N"THENRESTORE:GOTO90
256 IFF$="S"THENZ=1:GOTO320
```

(segue)

vengono costruite come somme di stringhe nelle righe 220 e 360.

Per il resoconto il programma, dopo aver richiesto il mese desiderato, va a leggere nel file contatore il numero Z di operazioni fatte (480-490), quindi richiede i Z file con il nome: MEse richiesto 1; ...; MEse richiesto Z, ne legge i contenuti (500-550) e ad ogni dato assegna una variabile indicizzata (ME = le prime due lettere del mese).

Non resta ora che utilizzare i dati per disegnare un istogramma con la stampante. Si può scegliere un

### Segue Listato 1.

```
4999 END:REM SDM
5000 FORLL=1 TO 38:PRINT "#":NEXT:PRINT
5010 FORLL=1 TO 18:PRINT TAB(38) "#":NEXT
5020 FORLL=38 TO 1 STEP -1:PRINT TAB(ALL) "#":
PRINT CHR$(145):CHR$(145):NEXT
5030 FORLL=1 TO 19:PRINT "#":
PRINT CHR$(145):CHR$(145):CHR$(145):NEXT
5040 RETURN
```

formato singolo di pellicole da visualizzare e verrà così fornita, numericamente e graficamente, giorno per giorno la quantità sviluppata nel mese richiesto; oppure con

lo stesso criterio verrà fornita la somma totale di pellicole trattate ogni giorno nell'arco di un mese. ■



# AUTOMAZIONE OGGI

## Finalmente una rivista dedicata all'automazione



Una pubblicazione  
**GRUPPO EDITORIALE JACKSON**

## Scrivi, suona, gioca, entusiasma

Gaetano Marano

# 66 PROGRAMMI PER ZX 81

## E ZX 80 CON NUOVA ROM + HARDWARE

Per le sue qualità e il suo modestissimo prezzo lo ZX 81 della Sinclair è il computer più venduto nel mondo.

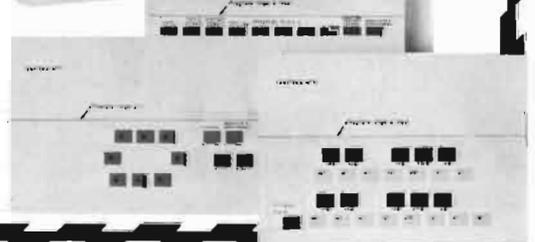
Oggi, sempre con una modestissima spesa, si può imparare a sfruttare questo eccezionale strumento al limite delle sue capacità. Basta scorrere questo libro per scoprire quante cose lo ZX 81 può fare con l'aggiunta di alcuni semplici ed economici componenti. Ad esempio, tramite un semplice circuito musicale può riprodurre 50 note su 4 ottave e, sempre grazie a una modifica hardware da poche migliaia di lire, lo ZX 81 diventa anche l'unico computer in grado di conferire effetti sonori ai giochi inseriti tra i suoi programmi. Ma non è tutto. Un'altra novità di quest'opera, preziosa anche per chi possiede lo ZX 80 con ROM, è il regalo di alcune tastiere disegnate da sovrapporre a quella sensitiva dell'apparecchio, per ricavarne altre, speciali funzioni.

**136 pagine. Lire 12.000 Codice 520 D**

Per ordinare il volume  
utilizzare l'apposito tagliando  
inserito in fondo alla rivista



**GRUPPO  
EDITORIALE  
JACKSON**



---

# Archivio scacchistico

## per TI 99/4A

---

Con questo programma potete memorizzare su nastro le partite che più vi interessano, rivedendole a vostro piacere

---

di Sergio Borsani

**N**on molto tempo fa lessi in una rivista che in un questionario proposto ai candidati per un corso di programmazione, tra le altre domande, veniva chiesto se sapevano giocare a scacchi. Come è risaputo questo gioco presuppone in chi lo pratica capacità di analisi e di concentrarsi a lungo su un problema. Dalle premesse si può arguire che anch'io sono un giocatore di scacchi, forse al livello della terza categoria, anche se non ho mai partecipato ad un torneo di qualificazione. Da molto tempo raccolgo ritagli di riviste dove sono riportate partite di scacchi e le vado ordinando in un piccolo archivio. Dall'idea di computerizzare le informazioni raccolte è nato questo primo programma che, senza eccessive velleità, assolve in modo semplice e piacevole al suo compito.

Il programma è scritto in TI Basic e necessita del registratore come memoria di massa per la creazione di un file, cioè è richiesta la sola configurazione base.

Si distinguono due sezioni principali. Le considereremo separatamente.

La prima parte viene usata per memorizzare una partita. Selezionato il numero 1, appare la scritta REGISTRARE UNA PARTITA, e vengono chiesti in input una serie di dati: nome del giocatore con il bianco, nome del giocatore con il

nero, luogo e data ed infine il tipo di apertura. Luogo e data formano un'unica stringa, pertanto non devono essere separati da una virgola. Tale segno serve, infatti, a separare le variabili in input. Al termine di ogni dato premete ENTER.

Tutti questi dati vanno a formare il primo record del file. Poiché nella OPEN non è stata specificata la lunghezza del record, il suo valore di default è di 64 caratteri e bisognerà stare attenti a non superare tale limite. La ragione per cui più variabili vengono riunite a formare un unico record è che si può sfruttare meglio la memoria di massa e rendere inferiori i tempi di registrazione e di caricamento. Se creiamo un record con un solo nome, ad es. Rossi, il computer completerà il record, con degli spazi in formato DISPLAY o con degli zeri binari in formato INTERNAL, fino a fargli raggiungere la lunghezza minima di 64 caratteri. Inoltre il computer, tra un record e l'altro, crea un gap, cioè uno spazio privo di registrazione; così, tra una sosta e l'altra, si dà modo al nastro di raggiungere l'adeguata velocità di registrazione. Insomma, se non fossero riuniti in modo ottimale, i dati nella memoria potrebbero occupare uno spazio anche 10 volte maggiore, allungando conseguentemente anche i tempi di utilizzazione.

I dati che costituiscono la notazione delle mosse hanno una lunghezza fissa di 12 caratteri, 6 per la mossa del bianco e 6 per il nero. Tale notazione, pur essendo abbastanza aderente a quella comunemente usata, presenta rispetto ad essa qualche differenza. Il primo carattere rappresenta il pezzo che viene mosso; questo dato non ha alcuna influenza nell'elaborazione tuttavia deve essere presente altrimenti si altera la lunghezza della stringa (uno spazio all'interno di una stringa è calcolato come un carattere mentre se è posto all'inizio no). Ad esempio, la mossa "e4" si deve trascrivere PE2E4-, indicando che un pedone si è spostato dalla casa e2 alla casa e4. Il numero di mossa appare automaticamente e quindi non deve essere trascritto. "2. Cf3 - Cg6" diventa:

CG1F3-CB8C6; G1 e B8 sono le rispettive case di partenza. La notazione deve essere esatta perché il programma non è in grado di rilevare eventuali errori, se scrivete RE1E4- ecc., durante la riproduzione della partita, il computer farà eseguire al Re un balzo ignorando completamente le regole degli scacchi. L'ultimo carattere di ogni semistringa, il sesto e il dodicesimo del record, normalmente non interviene nell'elaborazione, non può tuttavia essere una virgola, altrimenti compare il messaggio di errore e dovete riscrivere l'ultimo dato input. Questo carattere può essere utilizzato come commento; ad esempio, se c'è la presa di un pezzo, può essere ":", se c'è uno scacco al re, può essere "+", oppure "!" se la mossa è buona e ancora "#" se c'è lo scacco matto. Normalmente si può usare il segno "-" o "=". Ci sono tuttavia alcuni casi in cui questo sesto carattere della semistringa influisce nell'elaborazione. Innanzi tutto se è ":", nella seconda semistringa, vogliamo indicare che la partita è finita; pertanto, quando si verifica questa eventualità e l'ultima mossa è del bianco, si completerà la stringa con degli spazi e si terminerà con il punto. Un altro caso particolare è la presa al passaggio; questa si se-

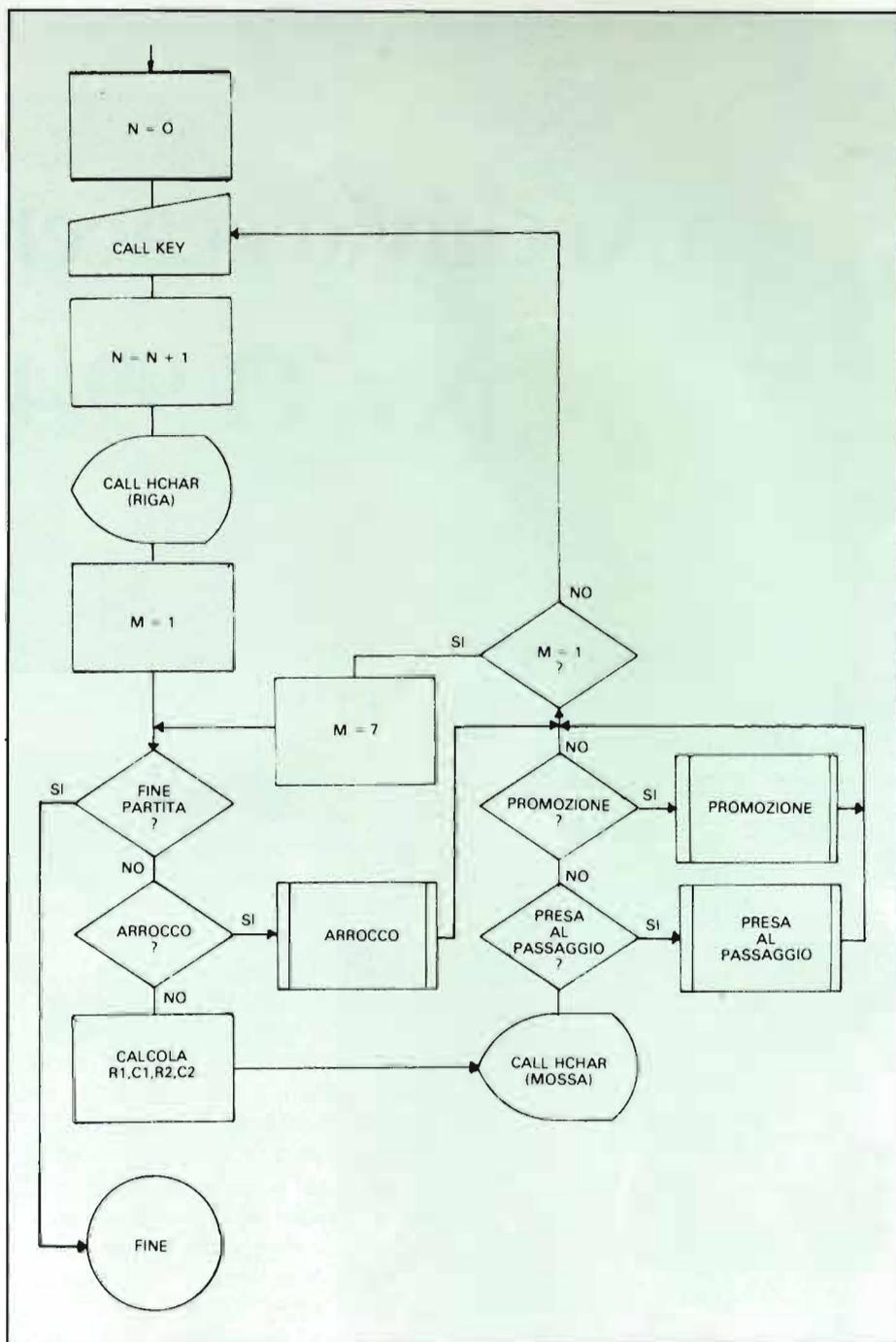


Figura 1. Diagramma di flusso della parte riguardante la stampa. Il ciclo viene percorso due volte, la prima, con  $M=1$ , in cui viene letta la prima semistringa (mossa del bianco), la seconda, con  $M=7$ , in cui vengono letti gli ultimi sei caratteri (mossa del nero).

gnala posponendo alle coordinate il simbolo "%". La promozione di un pedone esige che si specifichi il pezzo con il quale viene cambiato; D è la regina, T la torre, A l'alfiere e C il cavallo. Il simbolo va inserito come sesto carattere della semistringa. Inoltre si indicherà con -0-0- l'arrocco corto e con 0-0-0 l'arrocco lungo.

Come vedete la notazione delle mosse è quasi uguale a quella convenzionale, la quale risulta soltanto più semplificata.

Le partite possono essere registrate su nastro una di seguito all'altra oppure a partire da numeri prefissati del contatore del vostro registratore.

Le partite trascritte su nastro

possono essere in seguito richiamate ed eseguite sullo schermo; questo avviene selezionando il numero 2. Quando appare la scritta REWIND CASSETTE TAPE CS1, dovete posizionare il nastro all'inizio di un file. Il caricamento avviene in un tempo che dipende dalla lunghezza della partita e generalmente dovrebbe durare uno o due minuti.

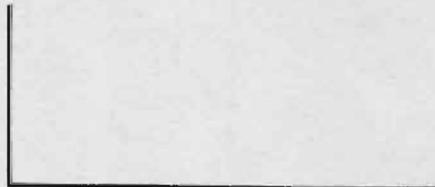
Caricata la sequenza di mosse, sul video appare una scacchiera, i nomi dei giocatori, il tipo di apertura, ecc. Ogni volta che viene premuto un tasto appare la notazione scritta della mossa e quindi questa viene eseguita sulla scacchiera. Per i dati scritti ho creato uno scroll verticale, come se occupassero una "finestra" e questo per non coinvolgere nello scrolling anche la scacchiera. Così appaiono scritte fino a dodici mosse; quando si passa alla tredicesima, scompare la prima. La possibilità di vedere, oltre ai movimenti sulla scacchiera, anche la notazione scritta è particolarmente utile quando si crea un archivio di aperture. Alla pressione di un tasto la partita prosegue fino a quando viene incorporato il punto al termine della stringa. A questo momento è possibile rivedere tutta la partita o passare ad un'altra.

Si consiglia di apportare alcune modifiche al programma per ottenere uno scrolling verso l'alto e verso il basso di tutte le mosse della partita e di programmare un tasto per poter riprendere l'esecuzione della partita dall'inizio senza dover attendere il suo completamento.

### Descrizione del programma

Alla linea 210 viene dimensionata la matrice uni-dimensionale A\$, ogni suo elemento conterrà una mossa. In 220-280 compare il menù, in base alla scelta si passa alle linee 340 per l'output, alla riga 790 per l'input del file e alla linea 2810 per uscire dal programma. Per i principianti aggiungo che si può uscire in ogni momento dal programma premendo FCTN(4).

Alla linea 350 iniziano le istruzioni INPUT per l'introduzione dei dati e le istruzioni relative. La linea 580 fa in modo che i numeri progressivi vengano incolonnati in corrispondenza delle unità. Alla linea 610 si rileva se la stringa in input ha lunghezza diversa da 12 caratteri e, a 640, se questa termina con un punto, si esce dal ciclo di input. Se si ritiene di aver trascritto correttamente i dati si può passare alla registrazione; l'OPEN viene data alla linea 700, su nastro il file non può che essere organizzato in modo sequenziale, l'apertura è in output poiché i dati devono passare ad un mezzo esterno, INTERNAL si riferisce al formato di registrazione, infine FIXED indica che tutti i record hanno la stessa lunghezza (64 byte). Come si può notare ogni record comprende 4 mosse. In 800-1190 si definiscono i caratteri ed i colori che formeranno la scacchiera. Le successive variabili con indice YY serviranno in seguito nella subroutine "promozione di un pedone". In 1250 si trova la OPEN in input per richiamare i dati in memoria centrale. Notate che i record vanno richiamati con riferimento alla stessa



struttura usata per l'output, cioè, se il record è formato da quattro campi alfanumerici, anche se si usano nomi diversi, devono sempre essere richiamate quattro variabili di stringa. Interessanti sono i cicli delle linee 1450 e seguenti, il loro scopo è quello di creare il grafico della scacchiera senza ricorrere a 64 istruzioni CALL HCHAR, tante sono le case della scacchiera. Nella linea 1740 una CALL KEY dà il via all'esecuzione di una mossa. I REM inseriti nel listato vi permetteranno di comprendere l'ordine in cui vengono eseguite le varie operazioni. Il clou del programma è la parte adibita a riconoscere il pezzo presente nella casa di partenza, spostarlo nella casa d'arrivo e rimettere una casella vuota nella precedente.

Queste operazioni non sono così semplici come potrebbe apparire in un primo istante, infatti un pezzo bianco su sfondo scuro potrebbe spostarsi su una casella chiara ed in questo caso il pezzo non può essere definito dallo stesso carattere. In questo programma il Re bianco su fondo scuro è il carattere 128, mentre il RE bianco su fondo chiaro è il carattere 136. Alle linee 1960-1990 avviene la decodifica delle coordinate di ogni mossa che vengono trasformate in numeri di riga (R1 ed R2) e di colonna (C1 e C2) dello schermo. La seguente CALL GCHAR riconosce il pezzo della casa di partenza (PX). Si possono verificare quattro situazioni condizionate dai colori dello sfondo: da chiaro a chiaro, da chiaro a scuro, da scuro a chiaro ed infine da scuro a scuro. Il numero di carattere sarà invariato se resterà invariato lo sfondo (PX=PX) ma cambierà al variare di quello (PX=PX+8 oppure PX=PX-8). Il computer sa se una casa è chiara o scura sommando il numero di linea con il numero di colonna, se la somma è pari la casa è chiara, se è dispari è scura. In 2150-2240 c'è il controllo su un'eventuale presa al passaggio o sulla promozione di un pedone; le relative routine si trovano dopo la linea 2250 che chiude il ciclo.

## VIDEO Giochi

LA PRIMA E UNICA  
RIVISTA DI VIDEOGAMES  
COMPUTER  
GIOCHI ELETTRONICI



Una pubblicazione  
del Gruppo Editoriale Jackson

Listato 1

```

100 REM PERSONAL SOFTWARE
110 REM *****
120 REM *
130 REM * ARCHIVIO *
140 REM * SCACCHISTICO *
150 REM *
160 REM *****
170 REM DI SERGIO BORSANI
180 REM 23.6.83
190 REM TI-99/4A TI BASIC
200 CALL CLEAR
210 DIM A$(100)
220 PRINT TAB(11);"ARCHIVIO"
::
230 PRINT TAB(9);"SCACCHISTI
CO"::::::
240 PRINT "PREMI:"::
250 PRINT "1.: REGISTRARE UN
A PARTITA"::
260 PRINT "2.: RICHIAMARE UN
A PARTITA"::
270 PRINT "3.: STOP"::
280 INPUT "SCELTA? ":S$
290 IF S$="1" THEN 340
300 IF S$="2" THEN 790
310 IF S$="3" THEN 2810
320 GOTO 280
330 REM *** SAVE ***
340 CALL CLEAR
350 PRINT " REGISTRARE UNA
PARTITA"::
360 PRINT "SCRIVI I NONI DEI
GIOCATORI"
370 INPUT "BIANCO ":G1$
380 INPUT "NERO ":G2$
390 PRINT
400 PRINT "LUOGO E DATA"
410 INPUT LD$
420 PRINT
430 INPUT "APERTURA: ":AP$
440 PRINT
450 N=0
460 PRINT "SCRIVI LE MOSSE U
NA AD UNA"
470 PRINT "ES.: PE2E4-PC7C5-
"::
480 PRINT "PER FINIRE TERMIN
A CON UN PONTO"::
490 PRINT "AL PASSAGGIO: %":
:
500 PRINT "PROMOZIONE: D,T,A
OPPURE C"::
510 PRINT "ARROCCO: -0-0- OP
PURE 0-0-0"::
520 PRINT TAB(9);"-----
"
530 N=N+1
540 IF N<101 THEN 580
550 PRINT "* MEMORY FULL"
560 CALL SOUND(1000,440,2)
570 GOTO 200
580 PRINT TAB(6-LEN(STR$(N))
);STR$(N);".":
590 INPUT A$(N)
600 CALL HCHAR(23,9,32)
610 IF LEN(A$(N))=12 THEN 64
0
620 PRINT "ERRORE! RIPROVA"
630 GOTO 580
640 IF SEG$(A$(N),12,1)=". "
THEN 660
650 GOTO 530
660 CALL CLEAR
670 INPUT "VUOI REGISTRARE?
(S/N) ":RS$

```

```

680 IF SEG$(RS$,1,1)="S" THE
N 700
690 GOTO 340
700 OPEN #1:"CS1",SEQUENTIAL
,OUTPUT,INTERNAL,FIXED
710 K=1
720 PRINT #1:G1$;G2$;LD$;AP$
730 PRINT #1:A$(K);A$(K+1);A
$(K+2);A$(K+3)
740 K=K+4
750 IF K>N THEN 770
760 GOTO 730
770 CLOSE #1
780 GOTO 200
790 CALL CLEAR
800 RE$="0B1C087F493E1C00"
810 DA$="000049497F7F3E00"
820 AL$="000818323E087F00"
830 CA$="107C7E0E1E0C3E00"
840 TO$="002A3E1C1C1C3E00"
850 PE$="00000808081C3E00"
860 CALL CHAR(128,RE$)
870 CALL CHAR(129,DA$)
880 CALL CHAR(130,AL$)
890 CALL CHAR(131,CA$)
900 CALL CHAR(132,TO$)
910 CALL CHAR(133,PE$)
920 CALL CHAR(134,"0")
930 CALL CHAR(136,RE$)
940 CALL CHAR(137,DA$)
950 CALL CHAR(138,AL$)
960 CALL CHAR(139,CA$)
970 CALL CHAR(140,TO$)
980 CALL CHAR(141,PE$)
990 CALL CHAR(144,RE$)
1000 CALL CHAR(145,DA$)
1010 CALL CHAR(146,AL$)
1020 CALL CHAR(147,CA$)
1030 CALL CHAR(148,TO$)
1040 CALL CHAR(149,PE$)
1050 CALL CHAR(152,RE$)
1060 CALL CHAR(153,DA$)
1070 CALL CHAR(154,AL$)
1080 CALL CHAR(155,CA$)
1090 CALL CHAR(156,TO$)
1100 CALL CHAR(157,PE$)
1110 CALL CHAR(158,"0")
1120 CALL CHAR(34,"000000000
0000OFF")
1130 CALL CHAR(36,"010101010
1010101")
1140 CALL CHAR(64,"808080808
0808080")
1150 CALL CHAR(38,"FF")
1160 CALL COLOR(13,16,6)
1170 CALL COLOR(14,16,11)
1180 CALL COLOR(15,2,6)
1190 CALL COLOR(16,2,11)
1200 YY(1)=129
1210 YY(2)=132
1220 YY(3)=130
1230 YY(4)=131
1240 REM *** OLD ***
1250 OPEN #1:"CS1",INPUT,SE
QUENTIAL,INTERNAL,FIXED
1260 K=1
1270 INPUT #1:G1$,G2$,LD$,AP
$
1280 INPUT #1:A$(K),A$(K+1),
A$(K+2),A$(K+3)
1290 FOR J=K TO K+3
1300 IF SEG$(A$(J),12,1)=". "
THEN 1340
1310 NEXT J
1320 K=K+4
1330 GOTO 1280

```

(segue)

Segue Listato 1.

```
2600 GOTO 2250
2610 REM *** PROMOZIONE ***
2620 ORD=(R2+C2)/2
2630 IF ORD=INT(ORD) THEN 269
0
2640 IF M=1 THEN 2670
2650 PEZZO=YY(PRO)+16
2660 GOTO 2730
2670 PEZZO=YY(PRO)
2680 GOTO 2730
2690 IF M=1 THEN 2720
2700 PEZZO=YY(PRO)+24
2710 GOTO 2730
```

```
2720 PEZZO=YY(PRO)+8
2730 CALL HCHAR(R2,C2,PEZZO)
2740 GOTO 2250
2750 REM *** FINE PARTITA *
**
2760 CALL HCHAR(23,1,32,64)
2770 PRINT "VUOI RIVEDERE LA
STESSA"
2780 INPUT "PARTITA? (S/N) "
:RIS$
2790 IF SEG$(RIS$,1,1)="S" T
HEN 1350
2800 GOTO 200
2810 CALL CLEAR
2820 END
```



# È vero: piccolo è bello!

## Alla scoperta dello ZX SPECTRUM

a cura di **Rita Bonelli**

ZX Spectrum è l'ultimo nato della famiglia Sinclair. È un calcolatore a colori di piccole dimensioni, ma di grandissime possibilità. Imparare a usarlo bene può essere fonte di molte piacevoli scoperte. Questo libro vi aiuta a raggiungere lo scopo. In 35 brevi e facilissimi capitoli non solo imparerete tutto sulla programmazione in BASIC, ma arriverete anche a usare efficientemente il registratore e a sfruttare al meglio le stampe. Soprattutto capirete la differenza tra il vostro Spectrum e gli altri computer.

**320 pagine. Lire 22.000 Codice 337 B**

**GRUPPO  
EDITORIALE  
JACKSON**



Per ordinare il volume utilizzare l'apposito tagliando inserito in fondo alla rivista

# Procedura di recupero

---

Ovvero come annullare gli effetti di un NEW nel VIC 20

---

di Luciano Gemme

**A** chiunque si reputi un programmatore sarà certamente capitato almeno una volta di aver dato, in un attimo di follia o di distrazione, il fatidico comando e di essere poi rimasto ad osservare sconsolato quelle tre letterine che magari significavano ore di programmazione buttate via.

Quasi per caso, studiando come il VIC memorizza i programmi, ho scoperto che è invece possibile "riportare a galla" il programma presente in memoria all'attimo del NEW.

## Effetti del comando NEW

Nella sua configurazione base, il VIC memorizza il programma a partire dall'indirizzo 4097 (il cui valore è contenuto nei byte 43 e 44) secondo le modalità indicate in tabella 1.

All'attimo del NEW il calcolatore si limita a porre a zero i primi byte (4097 e 4098) lasciando intatto il resto del programma. Il brutto è che il puntatore di variabile si posiziona automaticamente subito dopo ovvero all'indirizzo 4099 ( $\text{peek}(45)=3, \text{peek}(46)=16$ ).

Indirizzo	Contenuto
43	1 puntatore inizio programma $16 \times 256 + 1 = 4097$
44	16
45 } 46 }	X+2 puntatore inizio variabili
4096	0
4097 } 4098 }	N+1 link, contiene l'indirizzo della prossima istruzione
4099 } 4100 }	n. linea
4101	codifica linea programma
....	
N	0 fine linea programma
N+1	
N+2	altra linea
....	
X	0 fine programma
X+1	0
X+2	inizio variabili

Tabella 1.

### Procedura di recupero: prima parte

La prima condizione che dobbiamo realizzare è di rimettere i giusti valori nei due byte azzerati; procediamo così:

```
POKE 4098,16  
LIST
```

Vedrete apparire la prima linea del defunto programma: perché questo?

La risposta si trova se ci ricordiamo che i byte 4096 e 4097 sono ancora a zero, il calcolatore lista la prima linea poi salta a  $16 \cdot 256 + 0 = 4096$  dove trova due zeri e si arresta. Ora contate il numero di comandi Basic contenuti nella frase e sommatelo agli altri caratteri presenti (tranne il numero di linea), sia N il numero ottenuto:

```
POKE 4097,6+N  
LIST
```

Esempio:

```
10 REM PROVA: PRINT "PROVA"  
POKE 4097,6+16=22
```

Se avete contato bene vedrete apparire il listato di tutto il programma. Badate bene: ora il programma è solo listabile, qualsiasi tentativo di modifica o di attuazione manderà la macchina in tilt, questo perché il puntatore inizio variabili non è ancora al suo posto ma è ancora puntato a 4099.

### Procedura di recupero: seconda parte

Battete ora:

```
POKE 44,29  
POKE 46,29
```

Abbiamo ora spostato il puntatore di programma e di variabile rispettivamente a 7425 e 7427. Andremo, quindi, ad operare in una zona di memoria lontana dal programma.

Premete RUN/STOP, RESTORE e battete:

```
0 REM (non omettetela mai!)  
10 I=4097  
20 I=PEEK(I)+PEEK(I+1)*256  
30 PRINT I:IF I<>0 THEN 20  
GOTO 10 (non date RUN!)
```

Appariranno una serie di numeri progressivi poi uno zero e il programma si arresterà: ponete N uguale all'ultimo numero prima di zero più due.

Sia  $X = \text{INT}(N/256)$  e  $Y = N - X \cdot 256$ ; scrivete poi:

```
POKE 45, valore di Y  
POKE 46, valore di X
```

Siamo alla fine, non manca che rimettere al proprio posto il puntatore inizio programma:

```
POKE 44,16
```

E il gioco è fatto.

Naturalmente chi è in possesso di qualche espansione deve variare alcuni indirizzi; tenga comunque presente lo schema in tabella 2.

Indirizzo	Conf. base	Superespansione	Descrizione
43	1	1	byte meno significativo puntatore inizio programma
44	16	4	byte più significativo puntatore inizio programma
45	3	3	byte meno significativo puntatore inizio variabili
46	16	4	byte più significativo puntatore inizio variabili
55	0	0	byte meno significativo indirizzo fine memoria
56	30	30	byte più significativo indirizzo fine memoria

Tabella 2.



# Guida e cacciatore

---

---

## Due divertimenti matematici per lo ZX81

---

---

di Paolo Ferrari

Questi due programmi sono ispirati ad un semplice gioco matematico pubblicato sul libro di Carlo Sintini *Quiz e giochi matematici*, della Longanesi.

Girano sullo ZX81 nella versione da 1 K e consentono di fare dei solitari, di giocare contro i vostri amici o... contro il vostro ZX.

ro "sparato" dal cacciatore le consentirà di raggiungere più velocemente l'obiettivo. Il numero di partenza è sempre 1.

Nella tabella 1 è illustrato un esempio di partita. A questo punto si contano le mosse che sono occorse per raggiungere l'obiettivo e si scambiano i ruoli. La guida che utilizzerà il minor numero di mosse vincerà.

### Il gioco

Il gioco originale avviene tra due giocatori. Uno riveste il ruolo (attivo) della guida e l'altro quello (passivo) del cacciatore. La guida fissa un numero-obiettivo che deve restare sconosciuto al cacciatore, il quale si limiterà a "sparare" cifre da 1 a 9 (mai due uguali consecutivamente, altrimenti il gioco diventa banale). La guida deve scegliere tra le quattro operazioni di somma, sottrazione, moltiplicazione e divisione quella che applicata al nume-

### Il primo programma

Questo programma (listato 1) vi consente di giocare nel ruolo di guida. Quindi potete utilizzarlo per fare solitari o per sfidare i vostri amici a chi, per esempio, totalizza il minor numero di mosse sulla distanza di 10 partite. Il computer fa da cacciatore e visualizza sullo schermo tutto ciò che vi serve. Tramite la linea 35 non "spara" mai due volte la stessa cifra e quando

Cifra di partenza	Cifra sparata dal cacciatore	Operazione scelta dalla guida	Risultati parziali
1	3	+	4
	8	*	32
	5	+	37
	2	-	35
OBIETTIVO = 35 NUMERO MOSSE = 4			

Tabella 1.

indovinate l'obiettivo vi indica il numero di mosse impiegate e vi chiede se volete fare un'altra partita.

### Il secondo programma

Il secondo programma (listato 2) assegna il ruolo di guida al computer e a voi quello del cacciatore. Dovete inserire le cifre da 1 a 9 ma ... attenzione, la linea 30 non vi consente di barare! Non potrete quindi inserire la stessa cifra consecutivamente. Il computer sceglie l'operazione più conveniente, esegue i calcoli, visualizza i risultati e vi chiede un nuovo input. Ovviamente ha già scelto l'obiettivo, ma ve lo tiene ben nascosto per poi comunicarvi, quando lo ha raggiunto, il numero di mosse impiegate. Questo programma è preparato per generare obiettivi di due cifre, ma potete facilmente, alterando la linea 5, alzarne l'ordine di grandezza.

In entrambi i programmi i risultati delle divisioni sono arrotondati ai numeri interi, per evitare "impegolamenti" con numeri decimali da cui è difficile uscire.

```
1 REM "GUIDA E CACCIATORE N. 1"
5 LET M=1
8 LET Q=0
10 LET Y=1
15 PRINT "SCRIVI IL NUMERO DA RAGGIUNGERE"
20 INPUT Z
25 CLS
30 LET X=1+INT (RND*9)
35 IF X=Q THEN GOTO 30
40 PRINT AT 0,26;Z;AT 10,0;Y;TAB 8;X
45 INPUT A$
50 IF A$<>"-" AND A$<>"+" AND A$<>"*" AND A$ <>"/" THEN
   GOTO 45
55 IF A$="-" THEN LET Y=Y-X
60 IF A$="+" THEN LET Y=Y+X
65 IF A$="*" THEN LET Y=Y*X
70 IF A$="/" THEN LET Y=INT (Y/X)
75 PRINT AT 10,6;A$;TAB 10;"=" ";Y
80 IF Y=Z THEN GOTO 200
85 LET X=Q
90 LET M=M+1
95 SCROLL
100 GOTO 30
200 PRINT AT 18,0;M; " MOSSE"; AT 21,0;"ANCORA UNA PARTITA ?"
210 INPUT B$
220 CLS
230 IF B$="SI" THEN RUN
```

Listato 1. *Il computer è il cacciatore.*

### La competizione con il computer

Usando entrambi i programmi potrete misurarvi contro il vostro computer e, se lo trovate troppo duro da battere, ricordatevi che quando arriva vicino all'obiettivo

comincia ad "oscillare" (cioè alterna la somma e la sottrazione). In questa fase, se siete bravi, potete anche individuare l'intorno del numero che deve raggiungere e lo potete giocare tenendolo lontano da tale zona.



A proposito di questa competizione, è interessante analizzare più a fondo il gioco. La vittoria è determinata dalla differenza tra le nostre mosse e quelle dell'avversario e dipende quindi da due fattori:

- impiegare poche mosse quando si gioca come guida;
- far impiegare molte mosse all'avversario quando si gioca come cacciatore.

Questi fattori dipendono a loro volta da:

- infallibilità nei calcoli da eseguire per la scelta dell'operazione;
- capacità di elaborare strategie che (come quella sopra indicata) ostacolano l'avvicinarsi all'obiettivo.

È evidente che sul primo punto è avvantaggiato il computer, e lo è tanto più quanto più complessi diventano i calcoli (quindi quanto più alto è l'ordine di grandezza dell'obiettivo). Sul secondo punto, invece, siamo avvantaggiati noi. Tutta la competizione è perciò incentrata su questi due punti e può essere resa più o meno equilibrata variando l'ordine di grandezza dell'obiettivo.

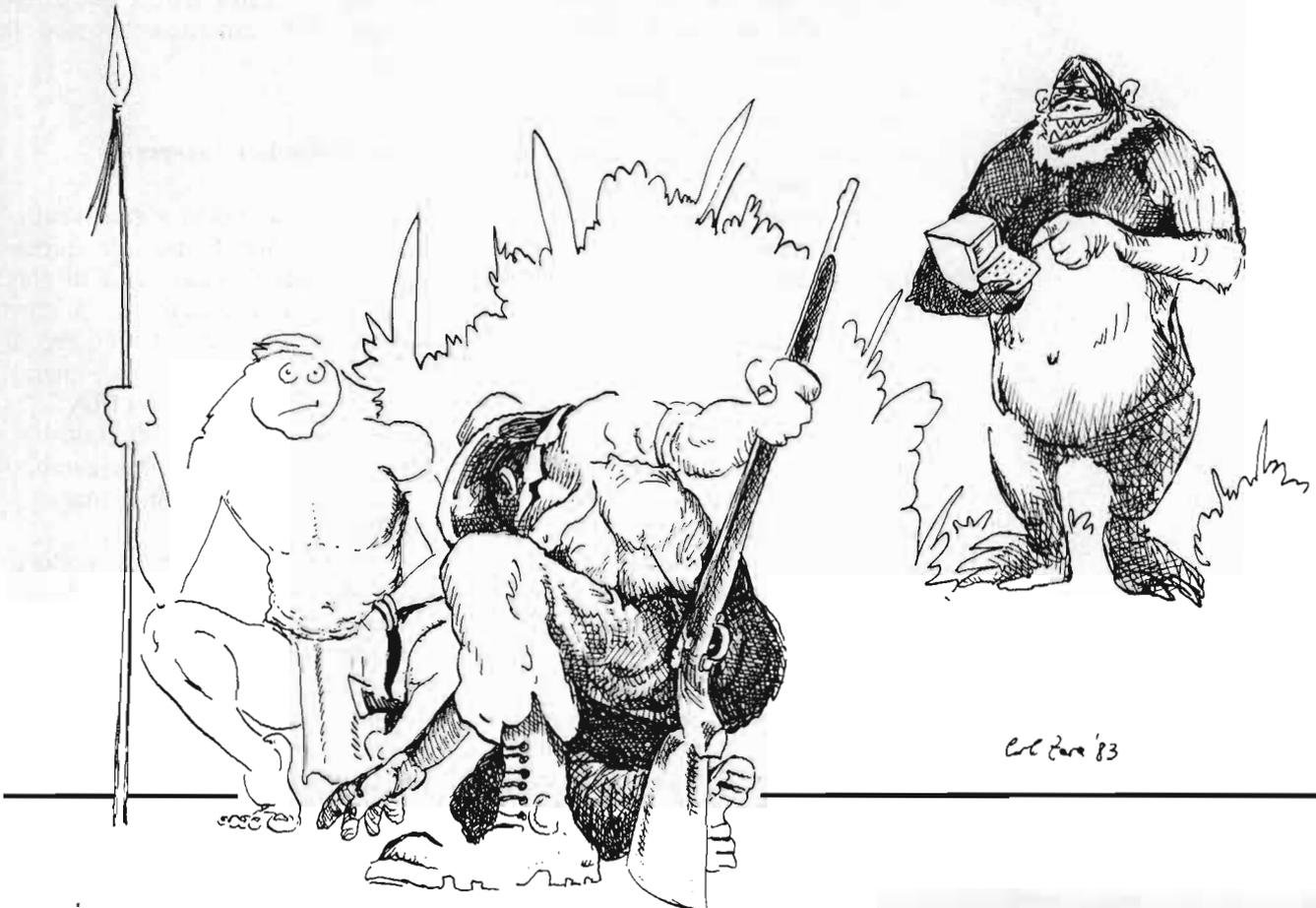
Buon divertimento! ■

```

1 REM "GUIDA E CACCIATORE N. 2"
5 LET Z=INT (RND*100)
10 LET M=1
15 LET J=0
20 LET Y=1
25 INPUT X
30 IF X=J THEN GOTO 25
35 PRINT AT 6,0;Y;TAB 8;X
40 LET A=ABS (Z-(Y+X))
45 LET B=ABS (Z-(Y-X))
50 LET C=ABS (Z-(Y*X))
55 LET D=ABS (Z-(Y/X))
60 IF A<B AND A<C AND A<D THEN GOTO 200
65 IF B<A AND B<C AND B<D THEN GOTO 300
70 IF C<A AND C<B AND C<D THEN GOTO 400
75 IF D<A AND D<B AND D<C THEN GOTO 500
80 PRINT AT 6,6;W$;TAB 10;"=" ";Y
85 SCROLL
90 IF Y=Z THEN GOTO 600
93 LET J=X
96 LET M=M+1
99 GOTO 25
200 LET W$="+ "
210 LET Y=Y+X
220 GOTO 80
300 LET W$="- "
310 LET Y=Y-X
320 GOTO 80
400 LET W$="*"
410 LET Y=Y*X
420 GOTO 80
500 LET W$="/"
510 LET Y=INT (Y/X)
520 GOTO 80
600 PRINT M; " MOSSE"

```

Listato 2. Il computer è la guida.



---

---

# ZX Manager

---

---

## Un gioco di simulazione finanziaria per lo ZX81

---

---

di *Enrico Ferreguti*

I giochi di simulazione finanziaria sono sempre stati dei classici, basti pensare a Monopoli, e solo in questi ultimi tempi hanno preso il sopravvento i grandi giochi di simulazione strategica. Nel nostro caso, trattandosi di un programma, la dotazione tipica di questi giochi come le carte degli imprevisi, le pedine, le banconote, trova comodamente posto sullo schermo sotto forma di numeri.

Il nostro gioco si configura come una gara tra 5 o meno concorrenti che si combattono a colpi di migliaia di dollari. Lo scopo dei giocatori è quello di quintuplicare il capitale iniziale (pari a 3000 \$) investendolo e disinvestendolo nella compravendita di titoli azionari.

La "corsa" si svolge su di un percorso lineare anziché circolare come nei giochi più tradizionali, collocato in alto a destra che contiene le iniziali delle varie caselle e l'asterisco per segnalare la partenza.

Le caselle sono 9: la borsa (B inversa), la banca (B), il telex (T inversa), pagamento debiti (P), dividendi (D), tasse (T), congiuntura (C), inizio (un asterisco) e riposo (uno spazio). Ad ogni mossa viene mostrato al giocatore che possiede il turno il numero ottenuto con il lancio del dado (sullo schermo in basso a sinistra), viene spostata la pedina, e viene domandato se si vuole passare la mossa al prossimo giocatore o se si vuole approfittare

della casella su cui si è capitati. Per le caselle congiuntura, tasse, pagamento debiti le operazioni che le riguardano vengono eseguite automaticamente senza fermare il programma per domandare se il giocatore vuole passare.

Nella parte centrale sinistra trova posto la situazione dei titoli, i dividendi e la tassazione. Sulla parte destra del video trova posto la situazione patrimoniale del concorrente che possiede il turno mentre in fondo a destra c'è lo spazio riservato alle comunicazioni con il programma.

### **Borsa (simbolo B inversa)**

Sulla casella borsa si può vendere o acquistare azioni alla quotazione corrente segnalata sul display nella sezione riguardante la contrattazione dei titoli. Il numero di titoli sul mercato è limitato, inizialmente è pari a 200 azioni FIAT, 75 Montedison, 46 SAI, 60 Italcable. La quantità è oggetto di cambiamento in occasione di acquisto o vendite.

Come in un qualsiasi mercato la penuria di una data merce provoca l'innalzamento del suo valore, e viceversa quando c'è abbondanza il suo valore tende a diminuire. Questa legge economica è fissa per qualunque merce oggetto di scambio (per es. le monete, la frutta,

Listato 1. Il programma "ZX Manager"

```

1 REM *****
2 REM * Z X M A N A G E R *
3 REM *
4 REM * P E R S O N A L   S O F T W A R E *
5 REM *****

100 PRINT "QUANTI GIOCATORI ?".
110 INPUT NUM
112 IF NUM>5 THEN GOTO 110
115 DIM A$(NUM,10)
120 FOR K=1 TO NUM
130 PRINT "NOME DEL GIOCATORE "
K:
132 INPUT A$(K)
150 PRINT " : "A$(K)
160 NEXT K
165 LET TAX=5
170 LET DIV=5
175 LET Z$="*50 P500 CT500"
CD:TC
180 DIM A(NUM,7)
185 DIM B(2,NUM)
190 DIM C(3,4)
192 LET TURNO=0
193 DIM D(NUM,4)
195 FOR K=1 TO NUM
200 LET A(K,5)=3000
210 NEXT K
215 LET C(1,1)=200
220 LET C(1,2)=75
225 LET C(1,3)=46
230 LET C(1,4)=50
235 LET C(2,1)=30
240 LET C(2,2)=50
245 LET C(2,3)=130
250 LET C(2,4)=100
255 DIM S$(4,10)
260 LET S$(1)="FIAT"
265 LET S$(2)="MONTEDISON"
270 LET S$(3)="SAI"
275 LET S$(4)="ITALCABLE"
280 LET C(3,1)=0.2
281 LET C(3,2)=0.5
282 LET C(3,3)=0.0
283 LET C(3,4)=0.6
285 CLS
290 PRINT Z$
300 FOR K=0 TO 5
305 PRINT AT K,25;" "
310 NEXT K
315 PRINT AT 6,0:" "

320 FOR K=7 TO 21
325 PRINT AT K,15;" "
330 NEXT K
335 PRINT AT 7,0:"CONTRATTAZION
I " "DI BORSA"
" "FIAT 50 " "MONTEDISON
" "SAI 130 " "ITALC
ABLE 100 " "
"DIVIDENDI 5%."
340 PRINT " " "T
ASSAZIONE 5%."

350 PRINT AT 20,0:"DADO=( )"
355 PRINT AT 1,26:"TOCCA";AT 3,
26;"A (<)"
360 PRINT AT 7,16:"CAPITALE DI"
370 PRINT AT 15,15:" "

375 FOR K=1 TO NUM
380 PRINT AT K,0:A$(K,1)
385 NEXT K
500 FOR G=1 TO NUM
501 IF A(G,7)=1 THEN GOTO 720
502 LET TURNO=TURNO+1
505 FOR K=8 TO 14
510 PRINT AT K,16:" "

515 NEXT K
520 PRINT AT 8,16:A$(G);": "AT
10,16:INT A(G,5);" DOLLARI"
525 FOR K=1 TO 4
530 IF A(G,K)<>0 THEN PRINT TAB
16;"N."A(G,K);" ";5$(K)
535 NEXT K
539 FOR K=1 TO 10
540 PRINT AT 3,29;" "
541 PRINT AT 3,29:A$(G,1)
542 NEXT K
545 LET DADO=INT (RND*3+1)
550 FOR K=1 TO 3
555 PRINT AT 20,6;CHR$ (DADO+25

```

```

)
560 FOR U=1 TO 5
565 NEXT U
570 PRINT AT 20,6;CHR$ (DADO+15
6)
575 FOR U=1 TO 5
580 NEXT U
585 NEXT K
590 PRINT AT G,A(G,6);" "
595 LET A(G,5)=A(G,6)+DADO
601 DIM L$(1)
602 LET O$="N"
605 PRINT AT G,A(G,6);A$(G,1)
607 LET L$=Z$(A(G,6)+1)
610 IF L$<>"C" AND L$<>"P" AND
L$<>"T" THEN PRINT AT 21,0;"PASS
I ?<N=NO>"
615 IF L$<>"C" AND L$<>"P" AND
L$<>"T" THEN INPUT O$
620 PRINT AT 21,0;" "

625 IF O$="" THEN GOTO 700
627 IF O$<>"N" THEN GOTO 610
630 GOSUB 800
635 IF L$="P" THEN GOTO 2000
640 IF L$="D" THEN GOTO 2500
645 IF L$="S" THEN GOTO 3000
650 IF L$="I" THEN GOTO 3500
655 IF L$="O" THEN GOTO 4000
660 IF L$="C" THEN GOTO 4500
665 IF L$="T" THEN GOTO 5000
700 FOR K=1 TO NUM
705 LET B(1,K)=B(1,K)+B(1,K)*1/
5000
710 LET B(2,K)=B(2,K)+B(2,K)*2/
5000
715 NEXT K
717 GOSUB 800
720 NEXT G
725 GOTO 500
800 FOR J=16 TO 21
805 PRINT AT J,16;" "

810 NEXT J
815 RETURN
2000 REM PAGAMENTO DEBITI
2010 IF B(2,G)=0 THEN GOTO 700
2030 PRINT AT 16,16:"DEVI PAGARE
";TAB 16;"UN DEBITO";TAB 16;"DI
";A(2,G);" CREDITI"
2040 LET A(G,5)=A(G,5)-B(2,G)
2050 IF A(G,5)<0 THEN GOTO 2100
2055 FOR K=1 TO 50
2060 NEXT K
2070 GOTO 700
2100 LET B(2,G)=ABS A(G,5)
2110 LET A(G,5)=0
2120 PRINT AT 20,16:"RESTANO DA
PAG.";TAB 16;B(2,G);" CREDITI"
2130 GOTO 2055
2500 REM --BANCA--
2510 PRINT AT 16,16:"-BANCA-";TA
B 16;"S=SITUAZIONE";TAB 16;"D=DE
POSITO";TAB 16;"P=PRELIEVO";TAB
16;"F=FINANZIAMENTO";TAB 16;"COS
A SCEGLI ?"
2520 INPUT O$
2525 GOSUB 800
2530 IF O$="D" THEN GOTO 2600
2532 IF O$="S" THEN GOTO 2950
2535 IF O$="P" THEN GOTO 2700
2540 IF O$="F" THEN GOTO 2500
2545 GOTO 700
2599 REM DEPOSITO
2600 PRINT AT 16,16:"DEPOSITO IN
C/C"
2605 PRINT TAB 16;"DI $ ";
2610 INPUT SOMMA
2615 PRINT SOMMA
2620 IF A(G,5)<SOMMA THEN LET 50
MMA=A(G,5)
2625 LET B(1,G)=B(1,G)+SOMMA
2630 LET A(G,5)=A(G,5)-SOMMA
2635 PRINT TAB 16;"C/C =$ ";B(1,
G)
2640 FOR U=1 TO 100
2645 NEXT U
2650 GOTO 700
2699 REM PRELIEVO
2700 PRINT AT 16,16:"PRELIEVO";T
AB 16;"C/C =$ ";B(1,G)
2705 PRINT AT 19,16:"PRENDI "

```

(segue)

Segue Listato 1.

```

2710 INPUT SOMMA
2715 IF B(1,G) < SOMMA THEN LET 50
MMA=B(1,G)
2725 PRINT TAB 16;"$ ";SOMMA;TAB
16;"RESTANO $ ";B(1,G)
2730 FOR W=1 TO 100
2735 NEXT W
2740 GOTO 700
2799 REM FINANZIAMENTO
2800 PRINT AT 16,16;"FINANZIAMEN
TO"
2805 PRINT AT 16,16;"QUANTO VUOI
?"
2810 INPUT SOMMA
2812 PRINT TAB 16;SOMMA;" $"
2815 IF RND(.3 OR (SOMMA+B(2,G)
)>2000 THEN GOTO 2900
2820 PRINT TAB 16;"PRESTITO";TAB
16;"ACCORDATO"
2825 LET B(2,G)=B(2,G)+SOMMA
2830 LET A(G,5)=A(G,5)+SOMMA
2835 FOR W=1 TO 100
2840 NEXT W
2850 GOTO 700
2900 PRINT AT 19,16;"NON POSSIAM
O";TAB 16;"ACCORDARE";TAB 16;"IL
PRESTITO"
2910 GOTO 2835
2950 REM SITUAZIONE
2955 IF B(1,G) <> 0 THEN PRINT AT
16,16;"IN ATTIVO ";INT B(1,G);"
$"
2960 IF B(2,G) <> 0 THEN PRINT AT
17,18;"IN PASSIVO ";INT B(2,G);"
$"
2965 FOR W=1 TO 50
2970 NEXT W
2972 GOSUB 800
2975 GOTO 2500
2999 REM BORSA
3000 PRINT AT 16,16;"BORSA";TAB
16;"1)=ACQUISTO";TAB 16;"2)=VEND
ITA";TAB 16;"COSA SCEGLI ?"
3001 INPUT 0$
3002 IF 0$="2" THEN GOTO 3300
3003 IF 0$<>"1" THEN GOTO 700
3004 GOSUB 800
3005 PRINT AT 16,16;"-BORSA-";
3010 PRINT "QUANTITA""
3015 FOR K=1 TO 4
3020 PRINT TAB 16;K;" ) ";S$(K,1
TO 4);" ";C(1,K)
3025 NEXT K
3030 PRINT TAB 16;"COSA VUOI?"
3035 INPUT SC
3040 IF SC<1 OR SC>4 THEN GOTO 3
035
3045 GOSUB 800
3050 PRINT AT 16,16;"QUANTE AZIO
NI";TAB 16;S$(SC);TAB 16;"VUOI ?"
3055 INPUT AZ
3060 IF C(1,SC) < AZ THEN LET AZ=C
(1,SC)
3065 IF AZ * C(2,SC) > A(G,5) THEN L
ET AZ=INT (A(G,5)/C(2,SC))
3070 PRINT TAB 16;AZ
3072 IF A(G,SC)=0 THEN LET D(G,5
C)=TURNO
3075 LET C(1,SC)=C(1,SC)-AZ
3080 LET A(G,5)=A(G,5)-INT (AZ * C
(2,SC))
3081 PRINT AT 21,16;"SPESA = ";A
Z * C(2,SC);" $"
3082 LET A(G,SC)=A(G,SC)+AZ
3085 LET C(2,SC)=C(2,SC)+C(3,SC)
 * AZ
3095 PRINT AT 9+SC,10;" "
3100 PRINT AT 9+SC,10;C(2,SC)
3105 FOR W=1 TO 100
3110 NEXT W
3115 GOTO 700
3300 REM VENDITA AZIONI
3305 GOSUB 800
3310 PRINT AT 16,16;"-BORSA-";TA
B 16;"CHE AZIONI";TAB 16;"VENDI?"
3314 DIM L$(4)
3315 INPUT L$
3320 FOR K=1 TO 4
3325 IF L$=S$(K,1 TO 4) THEN GOT
O 3350
3330 NEXT K

```

```

3340 GOTO 700
3350 PRINT TAB 16;"QUANTE AZIONI
";TAB 16;S$(K);TAB 16;"VENDI ?"
3355 INPUT AZ
3362 IF AZ * C(2,K) > 1500 THEN GOTO
700
3365 IF AZ > A(G,K) THEN LET AZ=A
(G,K)
3370 LET C(1,K)=C(1,K)+AZ
3375 LET A(G,K)=A(G,K)-AZ
3377 IF A(G,K)=0 THEN LET D(G,K)
=0
3380 GOSUB 800
3385 PRINT AT 16,16;"GUADAGNI";T
AB 16;AZ * C(2,K);" $"
3390 LET A(G,5)=A(G,5)+AZ * C(2,K)
3395 LET C(2,K)=C(2,K)-AZ * C(3,K)
3400 LET SC=K
3410 GOTO 3095
3500 REM TELEX
3505 IF A(G,5) < 250 THEN GOTO 700
3510 PRINT AT 16,16;"TELEX"
3515 PRINT TAB 16;"LA TARIFFA E"
";TAB 16;"DI 250 $"
3520 PRINT TAB 16;"1)=BANCA";TAB
16;"2)=BORSA"
3525 INPUT L$
3527 GOSUB 800
3532 LET A(G,5)=A(G,5)-250
3532 IF L$="1" THEN GOTO 2500
3535 IF L$="2" THEN GOTO 3000
3540 GOTO 700
4000 REM DIVIDENDI
4005 LET SOMMA=0
4010 PRINT AT 16,16;"I DIVIDENDI
"
4020 PRINT TAB 16;"AMMONTANO A"
4025 FOR K=1 TO 4
4030 IF D(G,K)=0 THEN GOTO 4070
4040 LET PERIODI=TURNO-D(G,K)
4045 LET AZ=C(2,K) * (DIV/5000) * PE
RIODI
4050 LET SOMMA=SOMMA+AZ * A(G,K)
4060 LET D(G,K)=TURNO
4070 NEXT K
4080 PRINT TAB 16;SOMMA;" $"
4085 LET A(G,5)=A(G,5)+SOMMA
4100 FOR W=1 TO 100
4110 NEXT W
4120 GOTO 700
4500 REM CONGIUNTURA
4505 FOR K=1 TO 4
4510 LET C(2,K)=C(2,K)+INT (RND *
50-25)/10
4515 NEXT K
4520 FOR K=1 TO NUM
4530 LET CAP=A(K,1) * C(2,1)+A(K,2
) * C(2,2)+A(K,3) * C(2,3)+A(K,4) * C
(2,4)+A(K,5)
4535 IF CAP >= 15000 THEN GOTO 800
0
4540 IF CAP <= 2000 THEN LET A(K,7
)=1
4545 NEXT K
4550 LET TAX=TAX+INT (RND * 3-1)
4555 IF TAX < 3 THEN LET TAX=3
4560 LET DIV=DIV+INT (RND * 3-1)
4565 IF DIV < 0 THEN LET DIV=0
4570 IF DIV > 12 THEN LET DIV=12
4580 FOR K=1 TO 4
4585 PRINT AT 9+K,10;" "
4590 PRINT AT 9+K,10;C(2,K)
4595 NEXT K
4600 PRINT AT 15,11;DIV
4605 PRINT AT 17,11;TAX
4610 FOR K=1 TO 5
4615 PRINT AT 16,16;"CONGIUNTURA
"
4620 FOR W=1 TO 5
4630 NEXT W
4635 PRINT AT 16,16;"
"
4640 FOR W=1 TO 5
4645 NEXT W
4650 NEXT K
4655 GOTO 700
5000 REM TASSE
5010 LET CAP=0
5015 FOR K=1 TO 4
5020 LET CAP=CAP+A(G,K) * C(2,K)
5025 NEXT K
5030 LET CAP=CAP+A(G,5)
5035 LET TS=CAP * TAX / 100
5040 PRINT AT 16,16;"LE TASSE";T

```

(segue)

Segue Listato 1.

```
AB 16; "AMMONTANO"; TAB 16; "A "; TS  
; " $"  
5045 LET A(G,5)=A(G,5)-TS  
5050 IF A(G,5)<0 THEN LET B(2,G)  
=ABS A(G,5)  
5055 IF A(G,5)<0 THEN PRINT TAB  
16; "ADDEBITATI"; TAB 16; ABS A(G,5  
); " $ IN BANCA"  
5060 IF A(G,5)<0 THEN LET A(G,5)  
=0  
5070 GOTO 4100  
6001 DIM L$(1)  
8000 REM FINE
```

```
8005 CLS  
8010 PRINT "HA VINTO "; A$(K); "CO  
N "; A(K,5); " DOLLARI"  
8015 PRINT AT 7,9; "I CONCORRENTI  
"; "VOGLIONO"; "RICOMINCIARE ?"  
8020 INPUT L$  
8025 IF L$(1)="S" THEN GOTO 165  
8030 CLS  
8035 PRINT AT 9,9; "ARRIVEDERCI"; TAB 9; "L"  
8040 STOP  
9000 SAVE "MANAGE"  
9010 RUN
```

l'oro e i valori immobiliari tra cui le azioni).

Anche in questo programma la legge è stata simulata inserendo un algoritmo che fa variare le quotazioni in ragione del numero di titoli sul mercato (più titoli ci sono e meno vengono quotati). Non possono essere vendute azioni per un valore superiore a 1500 \$.

### Banca (simbolo B)

La banca si compone di 3 operazioni: di deposito in c/c, di prelievo da c/c, di finanziamento oltre ad una opzione di situazione per conoscere i fondi in banca. Per le funzioni di prelievo e deposito non ci sono problemi: il programma addebita o accredita sul conto corrente le somme richieste a seconda dell'opzione scelta. L'operazione di finanziamento invece viene eseguita solo se il numero casuale ottenuto con RND è minore di 0,3 e se la somma richiesta è inferiore a 2000 \$. Logicamente questa operazione costa, difatti ne vengono calcolati i corrispondenti interessi passivi ciclicamente. Vengono calcolati anche gli interessi attivi che vengono accreditati in conto corrente, ed è questo che rende convenienti le operazioni di deposito.

### Telex (simbolo T inversa)

Il telex è un servizio a pagamento per mezzo del quale si può accedere alla banca o alla borsa pagando però una tariffa di 250 \$ che dà la possibilità di compiere tutte le operazioni di banca e di borsa.

### Pagamento debiti (simbolo P)

Alla casella "pagamento debiti" si è costretti a saldare tutti i debiti contratti con la banca, maggiorati degli interessi passivi maturati durante il periodo in cui si è usufruito di queste somme.

### Tasse (simbolo T)

In questa casella si adempiono agli obblighi fiscali versando una somma proporzionale al capitale posseduto (non quello in banca ma solo liquidità e titoli) con un valore percentuale che inizialmente è del 5% ma che è soggetto a variazioni.

### Congiuntura (simbolo B)

È la fase di variazione dell'andamento economico. In questo particolare momento i fattori variabili del gioco sono soggetti ad una variazione (dividendi, tasse, azioni). Questa operazione varia le quota-

zioni di borsa delle azioni, varia i dividendi sulle azioni e la percentuale di tassazione oltre a stabilire la bancarotta o la vittoria del concorrente capitato su quella casella qualora il suo patrimonio (attenzione: solo liquidità e titoli, non i fondi in banca) sia minore di 2000 \$ o sia maggiore di 15000 \$.

### Dividendi (simbolo D)

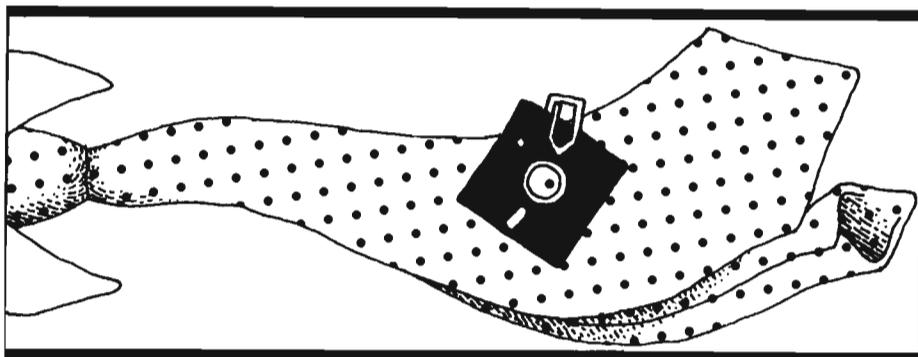
Quando si capita su questa casella si possono riscuotere i dividendi maturati dal momento in cui si è acquistato il titolo, l'operazione e una capitalizzazione semplice del valore delle azioni, quindi sarebbe più giusto parlare di interesse sulle azioni.

### Semplice strategia

I concorrenti non possono battersi direttamente tra di loro ma possono lo stesso danneggiarsi manovrando il prezzo delle azioni con oculate manovre di compravendita; i mezzi più remunerativi per raggiungere l'obiettivo finale sono: l'acquisto e la vendita di azioni, la riscossione dei dividendi, gli interessi di deposito.

Non esiti il giocatore a contrarre un debito con la banca poiché permette il finanziamento per l'acquisto di partite di azioni maggiori.

In questo programma quello che più dà nell'occhio è l'uso del dollaro come unità monetaria. Abbiamo scelto questa moneta per usare il simbolo "\$" di immediato significato invece di "L.", che oltre ad occupare due caratteri costringeva all'uso di numeri a sette-otto cifre (pena la sensazione di essere in ben altri tempi, cioè quando si comprava una casa con 10.000 lire).



# Paroliamo

---

## Giocare con le parole con uno ZX81

---

di Fausto Manfredini

Esistono altri programmi che portano lo stesso nome di questo, ma si limitano ad estrarre a sorte le lettere con cui formare le parole. Invece questo programma, piccolo esempio di intelligenza artificiale, stampa le parole che si possono formare con le lettere estratte.

All'utilizzatore è richiesta una certa pazienza nella preliminare immissione di dati, cioè di parole di cinque lettere. Con minime modifiche, il programma gira anche per parole con numero inferiore o maggiore di lettere e la versione in altri dialetti Basic è molto facile da eseguire.

L'idea originale è stata quella di confrontare tra loro non le singole

parole, ma i numeri che si ottengono sommando i codici delle lettere di ciascuna parola. Ad esempio, se memorizziamo MARIO, ORMAI, STORIA, MOIRA, RIAMO, AMORI, i codici delle cui lettere danno come somma 241 e se poi, in fase di ricerca digitiamo AIORM, anch'esse equivalenti a 241, il programma farà apparire sul display le sei parole suddette.

Avrei potuto rendere più sintetico il listato, ma, per ragioni di chiarezza, ho preferito usare qualche riga in più. Bastano, quindi, alcune parole di commento.

Si noti la simulazione della funzione ON...GOTO di riga 80; la variabile di comodo di riga 105 serve per delimitare il ciclo di ricerca

Listato 1. Il programma "Paroliamo".

```
10 DIM A$(100,5)
20 DIM A(100)
25 CLS
30 PRINT "          MENU"
40 PRINT "1 MEMORIZZAZIONE PA
ROLE (MAX 5 LETT.) 0 PER FINIRE"
50 PRINT "2 RICERCA"
60 PRINT "3 AGGIUNTA PAROLE"
70 PRINT "4 REGISTRA (PAR PAR
TIRE IL REGISTRATORE)"
75 INPUT MEN
75 CLS
80 GOTO 100#MEN
90 REM MEMORIZZA PAROLE
100 FOR I=1 TO 100
105 LET CON=I
110 INPUT A$(I)
120 IF A$(I,1)="0" THEN GOTO 25
130 PRINT A$(I); " "
140 LET B=0
150 FOR E=1 TO 5
160 LET B=B+CODE A$(I,E)
170 NEXT E
```

(segue)

e anche come indice del vettore A\$ in fase di memorizzazione di altre parole. (Quindi nelle aggiunte che si potranno fare al programma, ad esempio una subroutine per l'estrazione a sorte delle lettere, potremo tranquillamente usare la solita variabile I per altri cicli.) La variabile B riceve la somma dei codici delle lettere della parola, somma che viene memorizzata nel vettore A. Siccome non si può escludere a priori che due parole composte da lettere diverse possano avere lo stesso valore numerico, si è aggiunto al ciclo di ricerca un controllo casuale molto veloce, che confronta una delle lettere date in input con tutte le lettere della parola ricercata. La ricerca, che avviene speditamente raffrontando fra loro dei numeri, passa poi attraverso una fase di controllo che prende in considerazione le lettere che formano le parole.

Segue Listato 1.

```

1000 LET A(I)=B
2000 PRINT A(I)
3000 NEXT I
4000 CLS
5000 PRINT AT 20,0;"SCRIVI LE LE
TTERE ESTRATTE (N/L PER FINIRE)"
6000 INPUT B$
7000 IF B$="" THEN GOTO 25
8000 CLS
9000 LET C=0
10000 FOR N=1 TO LEN B$
11000 LET C=C+CODE B$(N)
12000 NEXT N
13000 FOR M=1 TO CON-1
14000 IF C<>A(M) THEN NEXT M
15000 LET CAS=INT (RND*LEN B$)+1
16000 FOR Z=1 TO LEN B$
17000 IF B$(CAS)=A$(M,Z) THEN GOT
O 18000
18000 NEXT Z
19000 IF C=A(M) THEN PRINT A$(M)
20000 NEXT M
21000 GOTO 210
22000 LET I=CON
23000 GOTO 210
4000 SAVE "P00"
4100 GOTO 25
4200 REM (C)1983 FAUSTO MANFRE
DINI - MODENA

```

## non perdetevi il nuovo numero di

- **Bitest Microprofessor II**
- **Acquisire dati con il VIC 20**
- **Anatomia dello ZX81**
- **Assembly Z80 rivisitato**
- **Trasmissione dati**
- **DBASE II**
- **Riservato personal:  
64 pagine di programmi  
per il vostro personal**



33	54	5	62	11	22	7	20
4	63	32	35	6	19	12	23
55	34	53	18	61	10	21	8
64	3	38	31	36	51	24	13
39	56	43	52	17	60	9	50
2	45	40	37	30	27	14	25
57	42	47	44	59	16	49	28
46	1	58	41	48	29	26	15

Figura 2. Una soluzione su una scacchiera  $8 \times 8$ ; il cavallo parte dalla casella 1 e arriva alla casella 64.

caselle in modo che quelle d'angolo siano nere; è immediato riconoscere che non ci possono essere soluzioni se la casella A di partenza è bianca. Ciò è dovuto al fatto che, mentre una casella e la successiva sul percorso devono essere di colori differenti, il numero di caselle bianche è inferiore di uno a quello delle caselle nere; così, una volta raggiunte tutte le caselle bianche, ne restano due nere non ancora toccate dal percorso, e queste non potranno essere raggiunte una di seguito all'altra.

Fatta eccezione per casi come i precedenti, non esiste un modo semplice per stabilire a priori se, per una data scelta di  $n$  e della casella iniziale A, il problema abbia soluzioni. La sola via per saperlo sembra essere quella di esaminare tutte le possibili sequenze di mosse fino a che

- o si trova una soluzione;
- o, avendo esplorato tutte le sequenze senza trovare una solu-

zione, ne segue che la soluzione non esiste.

Questa sarà perciò la strategia usata dal programma.

### Come generare tutte le sequenze di mosse

Uno dei requisiti necessari per poter realizzare un programma di questo tipo, che deve esaminare un numero enorme di tentativi per trovare una soluzione, è quello di trovare un modo semplice per generare tutti questi tentativi e per ricordare quali sono già stati esaminati e quali sono ancora da provare. Si tratta, in altre parole, di riuscire ad *ordinare* questo grande numero di sequenze di mosse, in modo tale che sia semplice passare da una sequenza alla successiva; in particolare non si deve essere costretti a tenere in memoria una "tabella dei tentativi già fatti" e a confrontare ogni nuovo tentativo

con essa per vedere se è già stato esaminato, perché ovviamente ciò sarebbe poco pratico!

Il metodo che il programma utilizza, e che adesso verrà chiarito nei dettagli, si basa sul concetto di *struttura ad albero*.

In figura 3 è mostrata una piccolissima parte di quello che chiamerò l'"albero dei tentativi"; ogni nodo di questa struttura rappresenta una casella della scacchiera e gli 8 "rametti" che partono dal nodo per raggiungere altri 8 nodi rappresentano le 8 mosse possibili (in generale) a partire da quella casella. Così un ramo che parte dal nodo iniziale A (che rappresenta la casella di partenza) e, traversando l'albero lungo i nodi B e C, raggiunge la casella D, rappresenta la sequenza di mosse A B C D. La classificazione delle sequenze di mosse che sarà usata è basata sulla seguente utile osservazione: se numero le otto mosse possibili a partire da una generica casella secondo un ordine standard specificato una volta per tutte, il numero d'ordine della mossa determina automaticamente quale casella viene raggiunta dopo la mossa. L'ordine standard adottato è quello mostrato in figura 1; il numero scritto dentro ognuna delle caselle raggiungibili da A indica il numero d'ordine della mossa fatta per passare da A a quella casella. Ad esempio, tutte le volte che parto da una certa casella della scacchiera ed eseguo la mossa "1", arrivo in una casella due righe sopra e una colonna a sinistra di quella di partenza. In questo modo ogni rametto che parte da un qualsiasi nodo può essere specificato senza ambiguità con un numero da 1 a 8. (L'ordine standard descritto è adottato per numerare le mosse a partire da *qualsiasi* casella; nel caso di caselle vicino al bordo della scacchiera la sola particolarità è che alcune delle 8 mosse non saranno ammesse.)

Si può ora passare ad analizzare in dettaglio l'albero dei tentativi.

Se da ogni nodo partissero sempre 8 rametti (cioè se ci fossero sempre 8 mosse possibili), i rami

do iniziale (radice) e arriva al nodo finale (foglia) è lungo  $n \times n$  nodi (dove  $n$  è il lato della scacchiera), allora tutte le caselle sono state toccate e la sequenza di mosse rappresentata dal ramo è una soluzione del problema; se, invece, il ramo è più corto, non è una soluzione.

Quello che il programma deve fare è esaminare i rami di questo albero per trovare la soluzione del problema: l'algoritmo con il quale esso svolgerà questo compito è il seguente:

- 1 Parti dalla radice dell'albero (la casella iniziale data)
- 2 Se puoi scendere lungo un ramo non ancora esplorato, vai al punto 3; altrimenti vai al punto 5
- 3 Scendi al nodo successivo non ancora esaminato il più a sinistra possibile
- 4 Torna al punto 2
- 5 Se il ramo (dalla radice alla foglia) è lungo  $n \times n$ , hai trovato una soluzione: stop
- 6 Se non puoi risalire (perché ti trovi alla radice), non ci sono soluzioni: stop
- 7 Risali al nodo precedente e torna al punto 2

L'ordinamento delle mosse, di cui ho parlato prima, rende molto semplice la scelta del ramo successivo da esaminare; infatti basta che ogni volta che da un nodo si passa al successivo si annoti l'indice della mossa fatta: quando si risalirà fino a quel nodo, si dovrà esaminare la mossa con l'indice immediatamente superiore.

Il procedimento illustrato nell'algoritmo esposto può anche terminare senza aver trovato una soluzione; in tal caso il processo di risalita dell'albero ci avrà riportato fino alla radice dopo che si sono esaminati tutti i rami, e quindi si sono esaurite tutte le possibili sequenze di mosse.

Prima di passare a considerare un altro aspetto del problema, riassumo brevemente quanto detto fino ad ora: l'albero dei tentativi è molto grande, ma per fortuna il programma deve solo memorizzarne un ramo alla volta, e ricordare

le decisioni prese ad ogni nodo, perché ciò gli è sufficiente per proseguire lungo il ramo (senza ripassare per caselle già raggiunte in precedenza) o per risalirlo ed imboccare il ramo successivo nel caso che sia necessario.

### Alcuni criteri per ridurre il tempo necessario a trovare una soluzione

L'albero dei tentativi è molto grande, ed esplorarlo tutto richiede molto tempo, come ci si può accorgere nei casi che non hanno soluzione. Fortunatamente si possono trovare dei criteri che permettono di stabilire a priori quando è impossibile che un certo ramo porti alla soluzione, permettendo così di non esaminare quel ramo. Illustrerò alcuni di questi criteri che, alla prova dei fatti, si sono rivelati molto vantaggiosi, ma prima vorrei fare notare una cosa: l'introduzione di questi criteri rende più complesso e lungo il processo di decisione se imboccare un certo sottoramo o no, tuttavia questa dilatazione dei tempi delle singole decisioni è giustificata dal fatto che, scartando a priori molti sottorami, il numero totale delle decisioni da prendere si riduce considerevolmente.

Innanzitutto, sono necessarie alcune definizioni:

- *casella libera*: è una casella non toccata dalla sequenza di mosse che porta dalla radice al nodo che stiamo esaminando;
- *casella occupata*: casella già toccata dalla sopradetta sequenza di mosse;
- *vicini di una casella*: sono le caselle che ne distano una mossa;
- *casella isolata*: casella che non può più essere raggiunta dal percorso.

Passo adesso ad enunciare i criteri:

1. Se l'effettuazione di una mossa (occupazione della casella C) lascia un vicino di C senza vicini liberi, l'unica mossa successiva da prendere in considerazione è quella che occupa quel vicino di C, che altrimenti rimarrebbe isolato; inol-

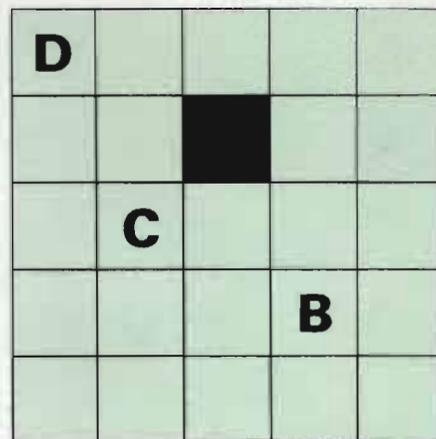


Figura 5. Il cavallo si trova in B; se ora passa in C, lascia la casella D senza vicini liberi. Perciò, se D non è l'ultima casella del percorso, ora il cavallo non deve passare in C (1° criterio).

tre l'unico caso in cui questa mossa porta ad una soluzione si ha quando essa è l'ultima mossa di quella soluzione (vedi figura 5).

2. Se un vicino di C resta con un solo vicino libero, lo si occupi subito; infatti se una soluzione non lo occupa adesso potrà occuparlo solo con l'ultima mossa, e se esiste una soluzione S che lo occuperebbe solo con la mossa finale, esiste anche una soluzione S' che lo occupa adesso e che segue a ritroso le mosse di S (figura 6).

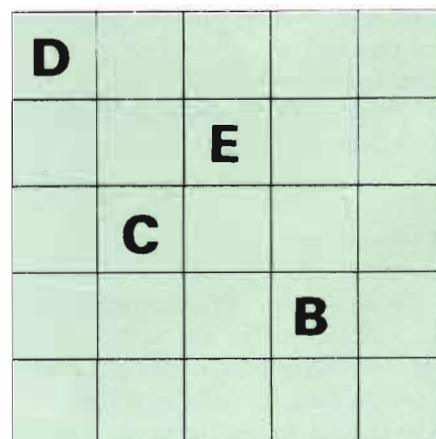


Figura 6. Il cavallo è passato da B a C; D è rimasto con un solo vicino libero (E). Allora il cavallo deve occupare D, perché se esiste una soluzione S che occupa D come ultima casella (via E), esiste anche la soluzione S' che occupa subito D e ripete all'inverso il percorso di S (2° criterio).

CLEAR 10000	Riserva 10000 byte di memoria per le matrici usate dal programma.
MODE 6	Introduce la grafica ad alta risoluzione, con $336 \times 256$ punti indirizzabili sullo schermo, e 4 colori.
COLORG 15 0 3 5	Sceglie i colori che voglio usare (15=bianco, 0=nero, 3=rosso...). Il colore dello sfondo è il primo scelto (quindi il bianco).
DRAW a,b c,d e	Traccia una retta dal punto a,b al punto c,d. Il colore della retta è e.
FILL a,b c,d e	Colora con e un rettangolo avente come vertici opposti di una diagonale i punti a,b e c,d.
MODE 0	Lo schermo ritorna al modo alfanumerico.
CUR x,y	Sposta il cursore sullo schermo nel punto x,y in modo che la prossima scritta parta da quel punto.

Tabella 2. Istruzioni particolari del Basic DAI.

programma si ritira da una casella, il numero viene cancellato.

In tabella 2 riporto il significato delle istruzioni particolari del Basic DAI, per permettere a chi volesse una facile traduzione del programma.

### Risultati e conclusioni

Come già si è visto, l'introduzione dei criteri prima descritti permette di ridurre il tempo necessario a trovare una soluzione, perché consente di scartare a priori molti rami dell'albero. In figura 9 si può ad esempio vedere come viene ridotto l'albero per una scacchiera  $4 \times 4$ . È interessante il confronto con la figura 4, nella quale era riportata una piccola parte dello stesso albero.

I criteri introdotti non sono i soli che possono portare ad un così notevole risparmio di tempo d'esecuzione; il lettore può sicuramente

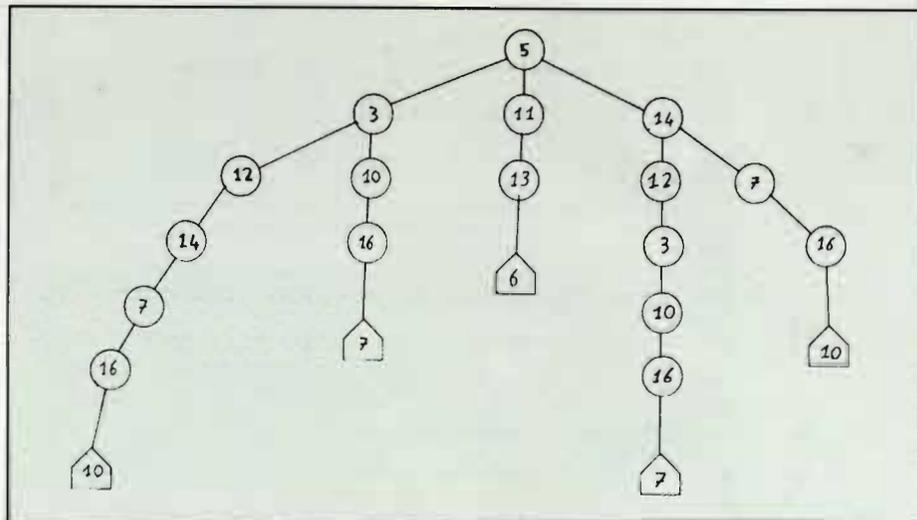
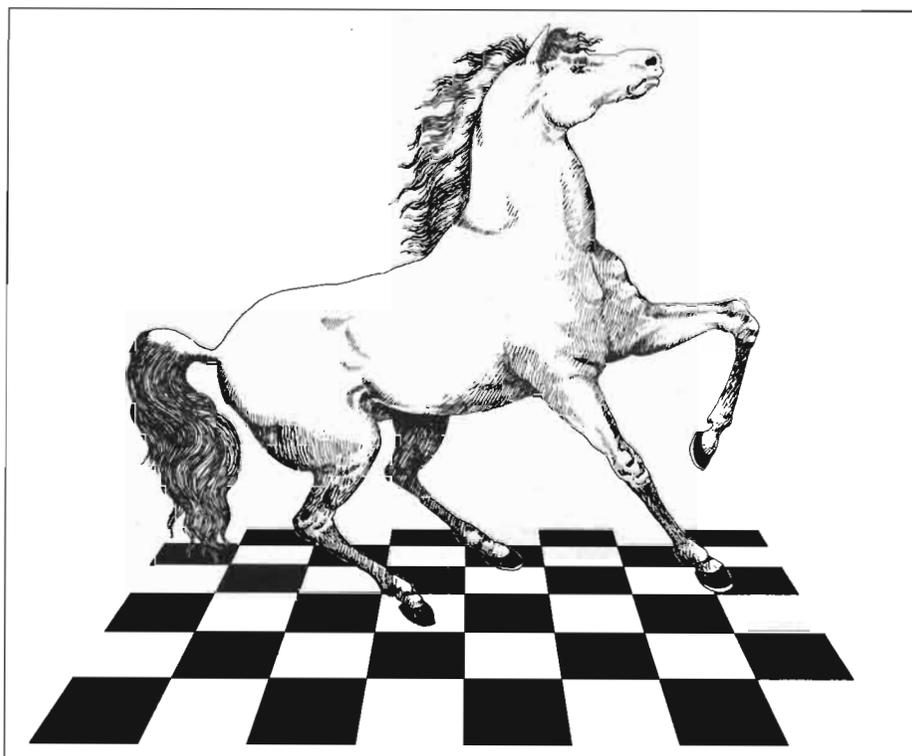


Figura 9. Con l'introduzione dei 4 criteri descritti, l'albero da percorrere per decidere che non ci sono soluzioni sulla scacchiera  $4 \times 4$  si riduce enormemente. Si confronti con la figura 4.

trovarne altri e migliorare il programma. Ad esempio, una condizione introdotta per fare diventare "rientrante" il giro fatto dal cavallo (cioè per fare sì che la casella finale sia una vicina di quella iniziale) ha portato a trovare una soluzione, su una scacchiera  $8 \times 8$ , in meno di 4 minuti, contro gli 8 necessari senza questa ulteriore condizione.

Una vasta classe di problemi è analizzabile e risolubile mediante tecniche e ragionamenti simili a quelli impiegati in questo articolo

(ad esempio il problema delle 8 regine, i problemi di scacchi ecc.). Inoltre queste tecniche sono impiegate anche nell'analisi di giochi nei quali occorra prevedere possibili sequenze di mosse future. Lo scopo dell'articolo non era perciò tanto quello di risolvere il problema esposto, quanto quello di analizzare in dettaglio il procedimento adoperato nel risolverlo, e di fornire quindi al lettore le basi per poter applicare simili metodi nella soluzione di altri problemi.



Listato 1. Il programma "il giro del cavallo".

```

1  REM *****
2  REM *
3  REM *          KNIGHT'S ROUND
4  REM *
5  REM *          by GIULIO MORPURGO
6  REM *
7  REM *****
10 REM -----  inizializzazione  -----
11 CLEAR 10000:COLORT 8 0 0 0
12 INPUT "dimensione della scacchiera":D%:PRINT
13 DIM BOARD%(D%+3.0,D%+3.0),NEAR%(D%+3.0,D%+3.0)
14 G%=D%*D%
15 DIM MOSSAI%(G%),MOSSAJ%(G%),M%(G%),PLAY%(G%)
16 MODE 0
17 PRINT " Sto inizializzando dei vettori "
25 FOR I%=0 TO D%+3:FOR J%=0 TO D%+3
26 BOARD%(I%,J%)=0:NEAR%(I%,J%)=0
27 NEXT J%:NEXT I%
30 FOR I%=0 TO 1:FOR K%=0 TO D%+3
35 BOARD%(I%,K%)=1:BOARD%(K%,I%)=1
40 BOARD%(D%+3.0-I%,K%)=1:BOARD%(K%,D%+3.0-I%)=1
45 NEAR%(I%,K%)=9:NEAR%(K%,I%)=9
50 NEAR%(D%+3.0-I%,K%)=9:NEAR%(K%,D%+3.0-I%)=9
55 NEXT K%:NEXT I%
60 FOR I%=2 TO D%+1:FOR J%=2 TO D%+1
65 FOR K%=-2 TO 2 STEP 4:FOR L%=-1 TO 1 STEP 2
70 NEAR%(I%,J%)=NEAR%(I%,J%)+2.0-BOARD%(I%+K%,J%+L%)-BOARD%(I%+L%,J%+K%)
75 NEXT L%:NEXT K%
80 NEXT J%:NEXT I%
90 PRINT "Scacchiera grafica o alfanumerica ?<G/A>"
91 A=GETC:IF A=0.0 THEN 91
92 GRAPH%=1:IF A=ASC("A") THEN GRAPH%=2
100 ON GRAPH% GOSUB 1500,2000
160 INPUT "posizione iniziale":E%,F%:PRINT
161 IF E%<1 OR F%<1 OR E%>D% OR F%>D% THEN 160
170 N=1.0:MOSSAI%(N)=E%+1:MOSSAJ%(N)=F%+1
180 X%=E%+1:Y%=F%+1
185 BOARD%(X%,Y%)=1
186 REM -----
200 REM -----  Discesa lungo un ramo  -----
201 ON GRAPH% GOSUB 1600,2100
210 C1=0.0:C2=0.0
215 FOR I%=-2 TO 2 STEP 4:FOR J%=-1 TO 1 STEP 2
220 NEAR%(X%+I%,Y%+J%)=NEAR%(X%+I%,Y%+J%)-1.0
225 IF BOARD%(X%+I%,Y%+J%)<>0.0 THEN 240
226 IF NEAR%(X%+I%,Y%+J%)=0.0 THEN C1=C1+1.0:AX=X%+I%:BY=Y%+J%
227 IF NEAR%(X%+I%,Y%+J%)=1.0 THEN C2=C2+1.0:TX=X%+I%:VY=Y%+J%
240 NEAR%(X%+J%,Y%+I%)=NEAR%(X%+J%,Y%+I%)-1.0
245 IF BOARD%(X%+J%,Y%+I%)<>0.0 THEN 260
246 IF NEAR%(X%+J%,Y%+I%)=0.0 THEN C1=C1+1.0:AX=X%+J%:BY=Y%+I%
247 IF NEAR%(X%+J%,Y%+I%)=1.0 THEN C2=C2+1.0:TX=X%+J%:VY=Y%+I%
260 NEXT J%:NEXT I%
270 ON (C1+1) GOTO 275,395,500,500,500,500,500,500
275 ON (C2+1) GOTO 300,280,280,500,500,500,500,500,500
280 AX=TX:BY=VY:GOTO 400
300 M%(N)=1
310 ON M%(N) GOTO 320,325,330,335,340,345,350,355
320 P%=X%-1:Q%=Y%+2:GOTO 360
325 P%=X%+1:Q%=Y%+2:GOTO 360
330 P%=X%+2:Q%=Y%+1:GOTO 360
335 P%=X%+2:Q%=Y%-1:GOTO 360
340 P%=X%-1:Q%=Y%-2:GOTO 360
345 P%=X%-1:Q%=Y%-2:GOTO 360
350 P%=X%-2:Q%=Y%-1:GOTO 360
355 P%=X%-2:Q%=Y%+1
360 IF BOARD%(P%,Q%)=0.0 THEN 370
365 IF M%(N)=8.0 THEN 500
366 M%(N)=M%(N)+1.0:GOTO 310
370 N=N+1.0
371 BOARD%(P%,Q%)=1
375 MOSSAI%(N)=P%:MOSSAJ%(N)=Q%
380 X%=P%:Y%=Q%
385 IF N=G% THEN 1000
390 GOTO 200
395 IF N<G%-1 THEN 500
400 PLAY%(N)=1:N=N+1.0
405 BOARD%(A%,B%)=1
406 IF N=G% THEN P%=AX:Q%=BY:GOTO 1000
410 MOSSAI%(N)=AX:MOSSAJ%(N)=BY:X%=AX:Y%=BY:GOTO 200
499 REM -----
500 REM -----  Risalita lungo un ramo  -----

```

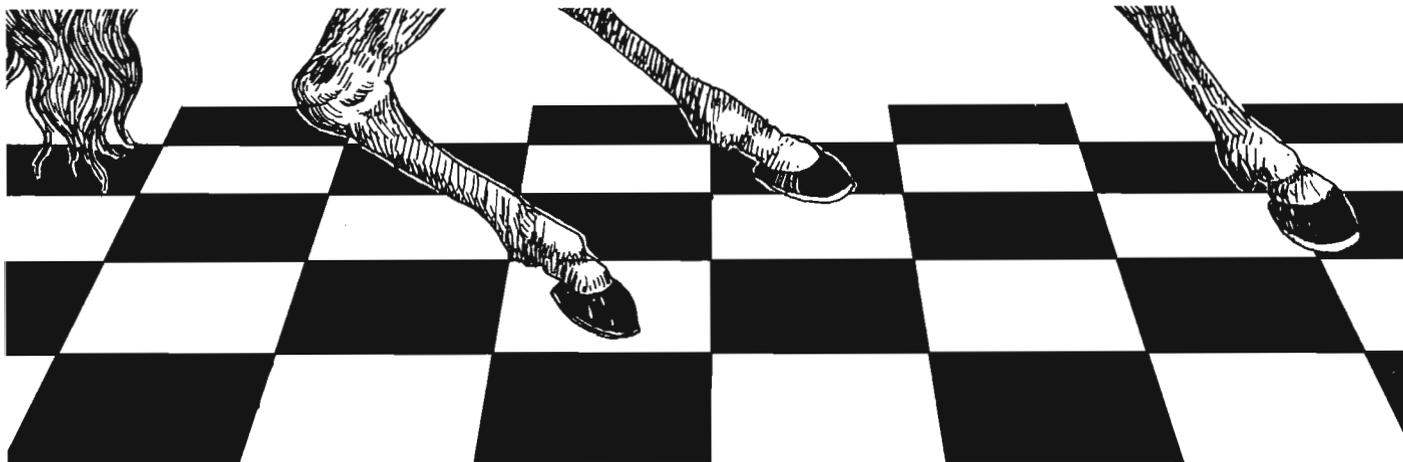
(segue)

Segue Listato 1.

```

501 ON GRAPH% GOSUB 1700,2200
510 BOARD%(X%,Y%)=0:PLAY%(N)=0
520 FOR I%=-2 TO 2 STEP 4:FOR J%=-1 TO 1 STEP 2
530 NEAR%(X%+I%,Y%+J%)=NEAR%(X%+I%,Y%+J%)+1.0
540 NEAR%(X%+J%,Y%+I%)=NEAR%(X%+J%,Y%+I%)+1.0
550 NEXT J%:NEXT I%
560 N=N-1.0:IF N=0.0 THEN PRINT:PRINT "non si puo'":END
570 X%=MOSSAI%(N):Y%=MOSSAJ%(N)
580 IF PLAY%(N)=1.0 OR M%(N)=8.0 THEN 500
590 M%(N)=M%(N)+1.0:GOTO 310
600 REM -----
999 REM ----- Soluzione trovata -----
1000 ON GRAPH% GOTO 1005,1015
1005 FILL 1+(P%-2)*20,1+(Q%-2)*20 19+(P%-2)*20,19+(Q%-2)*20 0
1006 GOTO 1020
1015 X%=P%:Y%=Q%:GOSUB 2100
1016 COLORT 0 15 0 0
1017 CURSOR 0,0
1020 END
1500 REM ----- Disegno scacchiera grafica -----
1501 MODE 6:COLORG 15 0 3 5
1502 FOR I%=0 TO D%
1503 DRAW 0,I%*20 D%*20,I%*20 0
1504 DRAW I%*20,0 I%*20,D%*20 0
1505 NEXT I%
1506 RETURN
1507 REM -----
1600 REM ----- Disegno mossa in modo grafico ---
1601 FILL 1+(X%-2)*20,1+(Y%-2)*20 19+(X%-2)*20,19+(Y%-2)*20 3
1602 RETURN
1603 REM -----
1700 REM ----- Cancellazione mossa in modo grafico ---
1701 FILL 1+(X%-2)*20,1+(Y%-2)*20 19+(X%-2)*20,19+(Y%-2)*20 15
1702 RETURN
1703 REM -----
2000 REM ----- Disegna scacchiera alfanumerica -----
2001 MODE 0:PRINT CHR$(12)
2002 CURSOR 0,D%*2-1
2005 B$="I" :C$="I"+CHR$(95)+CHR$(95)+CHR$(95)+CHR$(95)
2006 D$=" "+RIGHT$(C$,4)
2007 FOR Z%=1 TO D%:PRINT D$::NEXT Z%:PRINT
2008 FOR Z%=1 TO D%
2010 FOR W%=1 TO D%:PRINT B$::NEXT W%:PRINT "I"
2011 FOR W%=1 TO D%:PRINT C$::NEXT W%:PRINT "I"
2012 NEXT Z%
2013 RETURN
2014 REM -----
2100 REM ----- Disegno mossa alfanumerica -----
2101 CURSOR 1+5*(X%-2),2*(Y%-1)+1
2102 PRINT LEFT$(STR$(N),3)
2103 RETURN
2104 REM -----
2200 REM ----- Cancella mossa alfanumerica -----
2201 CURSOR 1+5*(X%-2),2*(Y%-1)+1
2202 PRINT " "
2203 RETURN
2204 REM -----

```



---

---

# Dal Basic al Pascal

---

---

## Seconda parte: procedure e subroutine

---

---

di Ronald W. Anderson

**I**n Basic permette le subroutine, sebbene la sua struttura non ne incoraggi l'uso. In Basic si chiama una subroutine con una istruzione GOSUB. Per esempio, GOSUB 1000 farà in modo che il vostro programma prosegua l'esecuzione dalla riga 1000, continuando con le righe successive fino a che non incontra un RETURN. A quel punto tornerà all'istruzione che segue immediatamente il GOSUB, e riprenderà da lì.

In Pascal una subroutine si chiama *procedura*. Una procedura ha un *nome* che si considera a tutti gli effetti come un identificatore, esattamente come i nomi delle variabili. Per chiamare una procedura in Pascal, basta semplicemente nominarla. Supponete di avere una procedura che trasmette al terminale il carattere di controllo per l'azzeramento dello schermo e il posizionamento del cursore in alto a sinistra. Se questo carattere fosse il codice ASCII 26, la procedura potrebbe essere quella del listato 1.

Per chiamare questa procedura dal programma principale basta usare il suo nome (vedi listato 2).

Il motivo principale per cui si usa una procedura, anziché mettere semplicemente una istruzione WRITE per azzerare lo schermo è che potreste usare la procedura in molti posti nel programma. Un altro motivo, è che è immediato a chiunque legga il programma lo

scopo della procedura; le istruzioni WRITE richiederebbero una spiegazione della loro presenza. Naturalmente questo è vero solo se chi ha scritto il programma ha usato un nome che descrive abbastanza bene cosa la procedura fa.

Se per esempio la procedura si chiamasse FATUTTO, si perderebbe la chiarezza. Un terzo motivo per cui si usa la procedura è che questa può essere facilmente modificata, per permetterne l'uso con un altro terminale. Si può per esempio cambiare il 26 con un 12, se il nuovo terminale lo richiede. Naturalmente, se una certa procedura è richiesta in molti posti in un programma e se essa è lunga, si riducono sensibilmente le dimensioni del programma facendone una procedura. Non dimenticatevi che *procedura*=*subroutine*.

Paragonate le stesse istruzioni in Basic (listato 3).

Mentre le istruzioni Pascal si spiegano da sole, per rendere chiara la versione Basic ci vogliono alcune spiegazioni, mentre un programma Pascal è abbastanza chiaro anche senza spiegazioni. Prima di poter interpretare un programma Pascal, tuttavia, bisogna conoscerne abbastanza per poter riconoscere le parole che non sono istruzioni, e quindi rendersi conto che si tratta di chiamate di procedure.

## Il passaggio dei parametri alle procedure

Abbiamo fatto solo qualche accenno all'uso dei parametri. Ora parliamo dell'uso principale del passaggio dei parametri in Pascal. Nel Basic il suo uso è limitato a funzioni come nel caso di  $Y=SQR(A*B)$ . Supponete di voler inserire nel vostro programma Basic una subroutine che calcoli l'ipotenusa di un triangolo rettangolo, noti i due cateti. Il teorema di Pitagora dice che l'ipotenusa è uguale alla radice quadrata della somma dei quadrati dei cateti. Supponete di usare *A* e *B* per rappresentare i cateti e *H* per l'ipotenusa (vedi listato 4).

Vedete che le variabili *A*, *B* e *H* vengono usate dalla subroutine. Inoltre, è necessario fissare i valori dei cateti *A* e *B*, e il risultato viene restituito nella variabile *H*. Ora supponete che altrove nel programma, le dimensioni dei cateti siano rappresentate con due altre variabili, per esempio *P* e *Q*, e che il risultato debba essere contenuto in *R* (listato 5).

Ci sono tre istruzioni per assegnare le variabili. Se si potesse rendere più generale la subroutine, queste istruzioni potrebbero essere eliminate. In Pascal si può farlo (listato 6).

Pascal usa *SQR* per il quadrato e *SQRT* per la radice quadrata. Il programma Pascal assume che *P*, *Q* e *R* siano state dichiarate variabili reali all'inizio del programma.

### Parametri formali

Le variabili indicate tra parentesi nella definizione di procedura si chiamano *parametri formali*. Hanno significato solo all'interno della procedura e si possono considerare come variabili "fittizie" alle quali saranno sostituite quelle vere più avanti nel programma.

Me variabili fra parentesi (nel programma principale) possono essere dei nomi di variabili come in questo caso, delle espressioni come  $P+3.7$ , o dei numeri come  $5.3$ . Ci

#### Listato 1.

```
PROCEDURE PULISCI SCHERMO;  
BEGIN  
  WRITE(CHR(26))  
END;
```

#### Listato 2.

```
PULISCI SCHERMO;  
WRITE('SCHERMO AZZERATO');
```

#### Listato 3.

```
100 GOSUB 1000: REM AZZERA LO SCHERMO  
110 PRINT "ORA LO SCHERMO VIENE AZZERATO"  
990 REM QUESTA SUBROUTINE AZZERA LO SCHERMO  
1000 PRINT CHR*(26)  
1010 RETURN
```

#### Listato 4.

```
100 A=3: B=4  
110 GOSUB 1000  
120 PRINT H  
1000 H=SQR(A*A+B*B): RETURN
```

#### Listato 5.

```
100 A=P: B=Q  
110 GOSUB 1000  
120 R=H
```

#### Listato 6.

```
PROCEDURE IPOTENUSA (A,B: REAL;VAR H: REAL);  
BEGIN  
  H:=SQRT(SQR(A)+SQR(B));  
END;  
(OUT SOTTO IL PROGRAMMA PRINCIPALE)  
BEGIN  
  IPOTENUSA (P,Q,R);  
END;
```

sono tre variabili nella definizione della procedura, e tre gliene vengono passate dalla chiamata. I valori delle tre espressioni fra parentesi separate da virgole sono passati ai tre parametri formali nella procedura. Qui viene il bello. Il terzo parametro nella definizione della procedura è preceduto dalla parola *VAR*. Questo dice al compilatore di non passare il valore contenuto nel terzo parametro della chiamata alla procedura, ma di passare la sua posizione, cioè il posto dove il risultato sarà immagazzinato. Sarebbe dunque sbagliato esprimere il terzo parametro come qualcosa di diverso da un nome per variabile. I parametri *VAR* nella procedura si usano per dare i risultati della procedura. Naturalmente ciò significa che il contenuto della terza variabile nell'elenco dei parametri verrà cambiato come risultato della procedura.

Forse è ovvio, ma per comple-

tezza dirò che i parametri formali nell'elenco dei parametri della procedura prendono i valori dei parametri della chiamata nell'ordine in cui questi sono elencati. Cioè, i parametri della chiamata devono essere nello stesso ordine e dello stesso numero e tipo dei parametri formali.

La procedura riceve due variabili (i parametri del tipo non *VAR* si chiamano parametri di valore) e un indirizzo al quale mettere i risultati del calcolo. I nomi dei parametri nella procedura *IPOTENUSA* hanno un significato solo all'interno di quella procedura, e delle variabili con gli stessi nomi possono apparire altrove nel programma senza conflitto. Per chiarire ulteriormente, vedete nel listato 7 delle chiamate valide per la procedura *IPOTENUSA*.

Queste sono valide se le dichiarazioni esatte delle variabili sono già state fatte in precedenza. Se le

BOOLEAN implica la sostituzione del suo valore), non deve essere messa fra parentesi:

```
IF ALPHA AND (C>D) THEN...
```

### Operatori aritmetici

Tutti gli operatori aritmetici standard (\*, /, + e -) si usano in maniera identica nel Pascal e nel Basic, e le regole di precedenza degli operatori del Basic sono applicabili anche al Pascal. Cioè la moltiplicazione e la divisione hanno la precedenza rispetto alla somma e alla differenza. Poiché il Pascal distingue fra le variazioni reali e intere, ci sono altri due operatori nel Pascal. DIV si usa per dividere con risultati interi. La risposta non è approssimata, bensì troncata, nello stesso modo in cui funziona la funzione INT nel Basic. 3 DIV 4 dà come risposta 0, 4 DIV 3 dà 1. Assieme a questo operatore per dividere con risultati interi, c'è un operatore che dà il resto intero. Si chiama MOD. 4 MOD 3 fornisce il resto dopo la divisione cioè 1. 12 MOD 7 è 5. Potrebbe essere di aiuto indicare che  $A \text{ DIV } B * B + A \text{ MOD } B = A$ . Per la divisione normale si usa la sbarra.

### L'aritmetica mista

Poiché il Basic distingue fra le variabili reali e intere, potreste aspettarvi della confusione quando questi due tipi di variabili vengono mescolati in un'espressione aritmetica. Pascal è molto amichevole in queste circostanze. È permessa qualunque combinazione per la quale non vada persa dell'informazione. Cioè, si può dare un valore intero a una variabile reale oppure usare una costante intera in un'espressione la quale usa delle variabili reali. Nel Pascal, la cifra 2 che appare in un'espressione è compilata come un valore intero mentre 2.0 è compilato come valore reale. REAL/2 è un'espressione valida. Pascal converte l'intero in un numero reale e il risultato sarà un numero reale. Contrariamente, se si cercasse di dare un valore reale a una variabile intera, e ciò avesse successo, la parte frazionaria della cifra sarebbe persa. Pascal vi costringe a pensarci e vi dà due scelte che riguardano il da farsi per la parte frazionaria. Si può dare un valore reale a una variabile intera come:

```
VARIABILEINTERA:=  
TRUNC(VARIABILEREALE);
```

oppure come

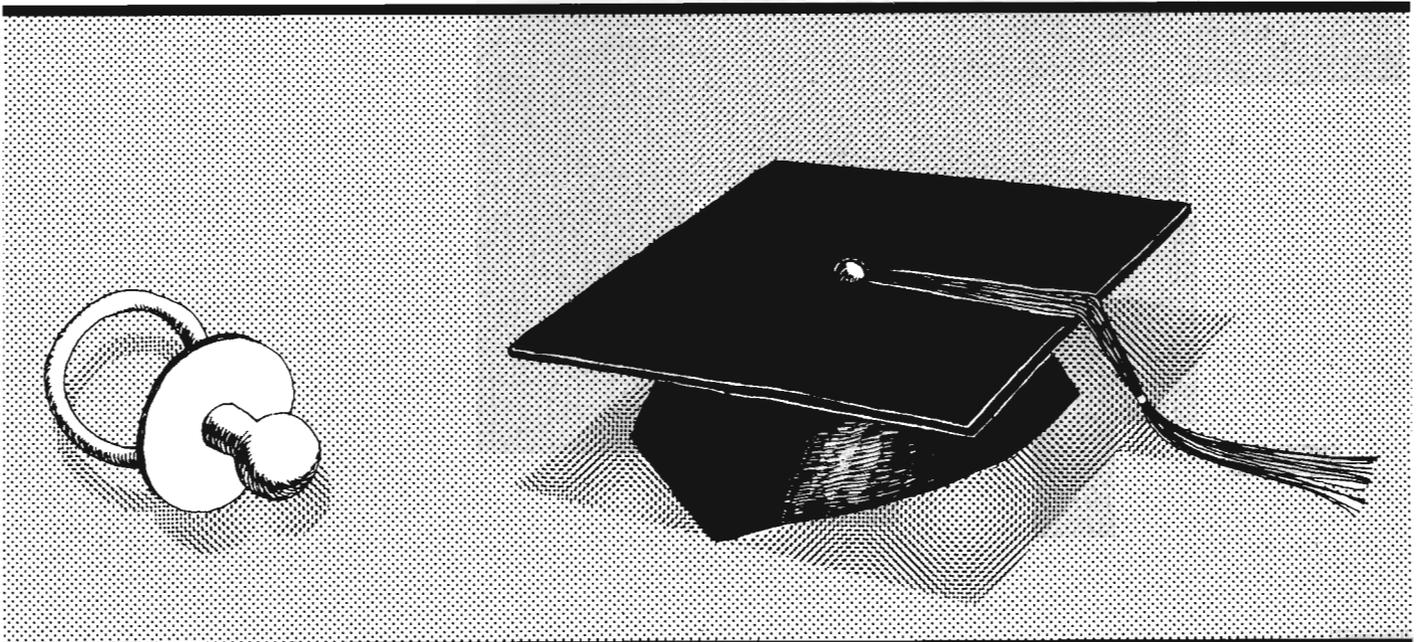
```
VARIABILEINTERA:=  
ROUND(VARIABILEREALE);
```

È evidente ciò che ognuna di queste dichiarazioni comporta. TRUNC semplicemente butta via la parte frazionaria del valore del numero reale, e ROUND arrotonda al valore intero più vicino. Voi dovette solo scegliere quale usare. Il compilatore segnerà un errore se dimenticherete di usare una delle due.

### Le funzioni standard

Il Pascal e il Basic condividono alcune funzioni standard. Poiché in alcuni casi i nomi differiscono leggermente, compaiono nella tabella qui sotto.

Basic	Pascal
ABS(X)	ABS(X)
ATN(X) o ATAN (X)	ARCTAN(X)
CHR\$(X)	CHR(X)
COS(X)	COS(X)
LOG(X) (logaritmo) base 10)	LN(X) (logaritmo base e)
SIN(X)	SIN(X)
SQR(X) (radice quadrata)	SQRT(X) (radice quadrata)
	SQR(X) (numero al quadrato)
EXP(X)	EXP(X)



---

# Matematica aerobica

## con il TRS 80

---

Controllate i vostri  
esercizi aerobici con  
questi programmi

---

di Bruce Douglas

**I**l cuore funziona come una pompa che muove il sangue (che trasporta ossigeno e nutrimento ai muscoli e ritira diossido di carbonio e prodotti di scarto). Ci sono due principali controlli del cuore: quello nervoso e la cosiddetta "legge di Sarling" del cuore.

Il controllo nervoso del cuore è mediato dal sistema nervoso simpatico e parasimpatico. Il sistema simpatico passa attraverso la spina dorsale, ha delle sinapsi in un gruppo chiamato gangli, e quindi continua a innervare il cuore. Se eccitate queste fibre stimolano il battito cardiaco (tachicardia) a seconda della forza della contrazione (controllata alterando la contrattilità miocardiale).

Le innervazioni parasimpatiche passano per il nervo craniale X, detto vago, che ha origine in una parte del midollo detto centro vasomotorio. L'inibizione del nodo seno-atriale, all'inizio del battito, rallenta il cuore (bradicardia).

Queste due vie nervose controllano il ritmo cardiaco per periodi relativamente brevi. Per esempio, la sveglia suona e voi saltate giù dal letto. Che cosa fa in modo che il sangue venga pompato al cervello e non ai piedi?

Molti sono i riflessi che inviano informazioni nervose al cervello. Uno di questi è il barorecettore. Certi recettori nel sistema circolatorio, come quelli situati nella ca-

rotide e nell'aorta, sono sensibili alla pressione; quando la pressione del sangue si abbassa troppo, i loro impulsi diventano meno frequenti: ciò significa che la pressione è troppo alta e il cervello farà in modo di diminuirla.

I nervi di accelerazione cardiaca secernono noradrenalina e la sostanza midollare surrenale secerne adrenalina, aumentando il ritmo cardiaco e la pressione sanguigna. Il nervo vago secerne acetilcolina, che invece rallenta il ritmo cardiaco.

La legge di Sarling del cuore è un approccio meccanico: dice che tutto il sangue che arriva al cuore deve lasciarlo. Se il ritorno venoso, cioè la quantità di sangue che arriva al cuore attraverso le arterie, aumenta, aumentano pure il ritmo cardiaco e la pressione sanguigna: allora il cuore deve pompare tutto questo sangue, altrimenti le vene si dilaterrebbero molto presto. Questo è un riflesso molto importante per il controllo nel lungo periodo dei battiti cardiaci.

I polmoni hanno il compito di scambiare i gas usati del sangue con gas nuovi presi dall'aria. L'unità funzionale del polmone, alveolo, ha la forma di un piccolissimo grappolo. Una membrana sottilissima circonda gli alveoli; essa ha la proprietà di diffondere il diossido di carbonio dai capillari agli alveoli; l'ossigeno, al contrario, diffonde

dagli alveoli ai capillari. Una volta nel sangue l'ossigeno viene trasportato ai tessuti dall'emoglobina, mentre il diossido di carbonio viene riportato indietro.

La distribuzione del sangue cambia a seconda dei bisogni. Quando si mangia, ad esempio, vengono irrorati molto di più lo stomaco e il fegato; quando si corre, invece, la quantità di sangue in questi organi si riduce, mentre aumenta incredibilmente quella che va ai muscoli. Di solito le vene, il fegato e la milza servono come riserva di sangue per quando se ne ha bisogno. Il cervello, invece, riceve costantemente il 25% del sangue, indipendentemente dall'attività che si sta svolgendo.

### Gli effetti dell'esercizio fisico

Quando si inizia un esercizio fisico, di solito, aumentano un po' il ritmo cardiaco e la pressione sanguigna, poiché aumenta molto la quantità di sangue che ritorna al cuore (vengono utilizzate le riserve di sangue). Anche la frequenza di respirazione influisce sul ritmo cardiaco. Man mano che si va avanti col lavoro, poi, anche il sistema nervoso inizia ad avere un ruolo nel controllo dell'affluenza del sangue ai muscoli, togliendolo da organi come il fegato e lo stomaco (questo processo si chiama derivazione del sangue). La sostanza midollare surrenale ha ora il compito di secernere adrenalina, che aumenterà ancora la frequenza cardiaca e causerà la vasocostrizione

dei vasi sanguigni periferici. Aumentando anche la derivazione, come si può vedere, sono molti i meccanismi che incrementano l'afflusso del sangue ai muscoli per portare ossigeno e nutrimento e ritirare i rifiuti.

Il corpo umano è molto pigro e lavora e migliora la propria meccanica e la capacità dei propri organi solo se costretto. Se il corpo è sottoposto a pochi sforzi, si abituerà a questo stato di calma. Se invece esso è sottoposto a un'attività continua, si adatterà, nei limiti del possibile, migliorando le sue funzioni.

Durante un esercizio, il cuore risponde agli sforzi aumentando la frequenza e l'intensità di ogni contrazione (in modo da assicurare un più completo svuotamento dei ventricoli). L'aumento della frequenza cardiaca può passare da 40 a 200, cioè aumentare del 500%. Per quanto riguarda la quantità di sangue pompata in ogni contrazione, invece, l'aumento è minore e si aggira normalmente intorno al 10-35%. In effetti quando la frequenza cardiaca supera i 180 battiti al minuto, il volume del sangue pompato diminuisce, perché il cuore non ha il tempo per riempirsi completamente prima della successiva contrazione. In atleti bene allenati, la gettata cardiaca (volume pompato per frequenza) può aumentare fino a sei-sette volte durante esercizi molto faticosi.

Nel lungo periodo il cuore diventa più forte. Le pareti dei ventricoli si ispessiscono, fornendo più potenza muscolare per mandare il sangue nell'aorta. Così il cuore non

deve battere tanto spesso per fornire abbastanza sangue al corpo.

Il ritmo cardiaco a riposo di molte persone (anche se non di tutte) diminuisce dopo pochi mesi di esercizio regolare. Quando sono in forma il ritmo cardiaco si stabilizza sui 40-45 battiti al minuto, ma torna a 65-70 dopo poche settimane di inattività.

Per rilevare il vostro ritmo cardiaco, potete usare sia il programma presentato che un orologio. Mettete una o due dita sul polso o, ancora meglio, sulla carotide, sul collo vicino alla trachea. Contate il numero di battiti in 10 secondi e moltiplicate per sei. Per conoscere il ritmo cardiaco a riposo, bisognerebbe misurarlo la mattina prima di alzarsi.

Bisogna comunque prendere delle precauzioni. Ad esempio correre può far male a persone che non fanno dello sport. Infatti affatica molto il cuore e il sistema circolatorio, e anche i muscoli, i tendini e i legamenti. Per questo bisogna sempre riscaldarsi prima di correre ed evitare di smettere di correre improvvisamente. Questo serve ad impedire l'insorgenza di crampi, cosa che succede quando ci si sottopone ad uno sforzo breve ed intenso. Per evitare la maggior parte dei problemi che hanno i corridori basta seguire un buon programma di riscaldamento e raffreddamento. Chi corre meno di 60 km alla settimana, di solito ha un ritmo cardiaco che supera i 150 battiti al minuto durante la corsa, ma non fa migliorare la propria forma superare i 180.

```
10 REM ***** AEROBICA *****
20 CLS: PRINT"PROGRAMMA DI AEROBICA"
40 INPUT"SCRIVI IL TEMPO CHE HAI CORSO (MIN,SEC)":M,S
50 INPUT"SCRIVI LA DISTANZA PERCORSO (KM)":D
60 TIME=M+S/60
70 PTS=17.0882*D-.554151*TIME-1.21753
80 PRINT"HAI FATTO ":PTS:" PUNTI AEROBICI"
90 PRINT"TEMPO MEDIO PER CHILOMETRO (MIN/KM) ":TIME/D
100 PRINT"VELOCITA' MEDIA (KM/H) ":D/TIME*60
110 INPUT"INTRODUCI IL TUO RITMO CARDIACO DOPO 5 MINUTI DI RIPOSO":HR
120 IF HR>120 THEN PRINT"NON VA BENE, DEVI CORRERE DI MENO."
    ELSE PRINT"VA BENE, VELOCITA' E DISTANZA PERCORSO NON ECCESSIVE."
130 END
```

Listato 1. Programma per calcolare i parametri aerobici di una corsa.

## Analisi statistica

Consideriamo ora l'analisi delle serie temporali mediante medie mobili e autocorrelazione. L'autocorrelazione è semplicemente la correlazione di una serie temporale con se stessa. Ma può una serie temporale avere un'alta correlazione con se stessa?

Una serie temporale è una collezione di dati, raccolti in vari istanti nel tempo. Un ECG ad esempio (elettrocardiogramma) rileva la frequenza cardiaca (e i vettori elettrici) e produce un insieme di dati, ossia una serie temporale. Anche un grafico dei prezzi di borsa in un determinato periodo di tempo è una serie temporale. Due sono i più frequenti problemi che si pongono sulle serie temporali:

- esiste una caratteristica di periodicità (stagionale) nelle serie?;
- esiste una tendenza generale?

L'autocorrelazione determina le fluttuazioni periodiche delle serie temporali; le medie mobili eliminano le fluttuazioni periodiche e rilevano la tendenza generale. In questo senso le due funzioni sono complementari. Affrontiamo per prima l'autocorrelazione.

Se immaginiamo che una serie temporale abbia due stati (0 e 1) e ne visualizziamo i valori (lo stato 0 è rappresentato da uno spazio bianco), avremo qualcosa come lo schema di figura 1. Se vogliamo effettuare una correlazione, dobbiamo allineare una serie con l'altra e confrontarne gli stati. Se, per un tempo dato, ambedue sono "accesi" o ambedue sono "spenti", si aggiunge un 1 alla somma di correlazione; altrimenti si aggiunge uno 0 (vedi figura 2).

Naturalmente questa è una correlazione perfetta, come ci aspettavamo. Ricordiamo che le equazioni per le correlazioni sono:

$$r = \frac{\sum (xy)}{\sqrt{(\sum x^2 \sum y^2)}}$$

in cui

$$\sum xy = \sum xy - \frac{\sum x \sum y}{N}$$

1 1 111 1 1 111 1 1 111 1 1 111

Figura 1.

1 1 111 1 1 111 1 1 111 1 1 111  
1 1 111 1 1 111 1 1 111 1 1 111

Figura 2.

1 1 111 1 1 111 1 1 111 1 1 111  
1 1 111 1 1 111 1 1 111 1 1 111

Figura 3.

1 1 111 1 1 111 1 1 111 1 1 111  
1 1 111 1 1 111 1 1 111 1 1 111

Figura 4.

$$\sum x^2 = \sum X^2 - \frac{(\sum X)^2}{N}$$

$$\sum y^2 = \sum Y^2 - \frac{(\sum Y)^2}{N}$$

L'autocorrelazione è solo un caso particolare della correlazione prodotto-momento. Da queste equazioni possiamo vedere che la funzione di autocorrelazione, poiché  $X$  e  $Y$  hanno lo stesso valore, si riduce all'unità.

Ma qualcosa di interessante avviene spostando le serie di bit, come in figura 3.

Ora non abbiamo più una correlazione perfetta! Facciamo una copia della serie temporale originale e sommiamo una costante al tempo per ogni intervallo. Guardiamo cosa succede (figura 4) se la spostiamo ulteriormente. La spostiamo e ritroviamo la correlazione perfetta. Cosa è successo?

Lo spostamento che abbiamo effettuato sulla serie è chiamato spostamento di fase della serie stessa. Nell'autocorrelazione si mette in correlazione una serie con se stessa, ma aggiungendo diversi spostamenti di fase alla correlazione per

scoprire elementi di periodicità nei dati. Normalmente l'equazione per l'autocorrelazione ha un formato leggermente diverso rispetto alla normale equazione di correlazione, perché si correla una serie temporale con se stessa, ma a differenti spostamenti di fase:

$$\phi(p) = \frac{1}{2N} + \sum_{i=-N}^{i=N} f(i) (i+p)$$

Possiamo vedere che la funzione di autocorrelazione  $\phi(p)$  (in cui  $p$  è lo spostamento di fase), è una funzione dello spostamento di fase. Cambiando lo spostamento di fase si possono definire le fluttuazioni periodiche nei dati.

Il programma del listato 2 controlla la vostra frequenza cardiaca e poi ne effettua l'autocorrelazione. L'output sullo schermo è un istogramma, chiamato autocorrelogramma. Poiché la frequenza cardiaca è discretamente regolare, si può notare un picco alla frequenza cardiaca fondamentale, ma si può notare anche un secondo picco, più

$$\frac{X(1) + X(2) + \dots + X(n)}{N}, \frac{X(2) + X(3) + \dots + X(n+1)}{N}, \frac{X(3) + X(4) + \dots + X(n+2)}{N}, \dots$$

Figura 5.

## Segue Listato 2

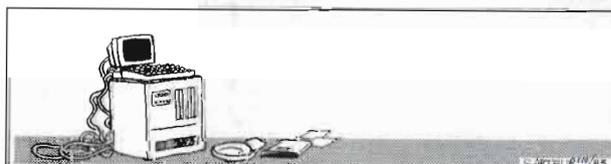
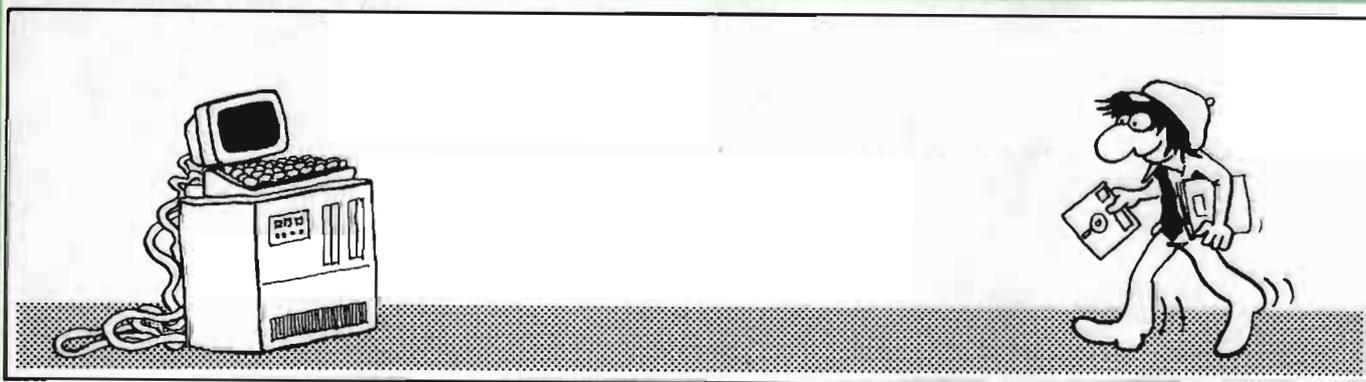
```
1060 NEXT I
1070 LPRINT TP#
1080 CLS: PRINT TAB(20);"ORA PUOI:"
1090 PRINT"1: RIVEDERE I RISULTATI"
1100 PRINT"2: RIFARE LE MEDIE MOBILI"
1110 PRINT"3: FINIRE"
1120 A#=INKEY#: IF A#="" GOTO 1120 ELSE ON VAL(A#) GOTO 410,790,1140
1130 GOTO 1120
1140 END
9000 AD=65280: REM #HFF0
9020 FOR A=AD TO 65370
9030 READ I: POKE A-65536,I
9040 NEXT A
9050 M=65279: MS=PEEK(16561)+PEEK(16562)*256+1
9060 IF MS>M THEN POKE 16562,M/256: POKE 16561,M-INT(M/256)*256
9080 GOTO 65
10000 DATA 243,62,42,50,0,60,33,86,255,205
10010 DATA 13,38,213,221,225,17,0,0,33,0
10020 DATA 0,58,64,56,254,16,40,53,254,1
10030 DATA 40,3,35,24,242,221,117,0,221,35
10040 DATA 221,116,0,221,35,33,0,0,19,213
10050 DATA 229,221,229,1,255,39,205,96,0,221
10060 DATA 225,225,209,58,0,60,254,191,40,4
10070 DATA 62,191,24,2,62,32,50,0,60,24
10080 DATA 196,213,225,195,154,10,90,40,48,41,0
```

medie mobili con un altro ordine. Provate a farlo e guardate l'effetto del periodo del livellamento.

Il programma contiene una serie di routine molto utili. La routine di autocorrelazione scala e disegna un istogramma di valori dati. La routi-

ne delle medie mobili vi permette di scalare la larghezza di un grafico per la stampante e stampa un lungo grafico di punti. (Può lavorare anche sullo schermo, ma un po' meno efficacemente). Non potete usare le istruzioni TAB con argo-

mento superiore a 64 anche se la vostra stampante ha più colonne di questa (a meno che non usiate un Basic adatto, come ad esempio il DOS PLUS Disk Basic). ■



## 6

### I nidi di FOR

Mauro Boscarol

Considerate questo semplice programma:

```
FOR I=1 TO 3
FOR J=1 TO 3
FOR K=1 TO 3
PRINT I,J,K
NEXT K
NEXT J
NEXT I
```

Se lo fate girare, otterrete come risultato questa colonna di terne:

1 1 1	2 1 1	3 1 1
1 1 2	2 1 2	3 1 2
1 1 3	2 1 3	3 1 3
1 2 1	2 2 1	3 2 1
1 2 2	2 2 2	3 2 2
1 2 3	2 2 3	3 2 3
1 3 1	2 3 1	3 3 1
1 3 2	2 3 2	3 3 2
1 3 3	2 3 3	3 3 3

Ognuna di queste terne è un possibile risultato di una giornata ippica, durante la quale vi sono tre corse, e ad ogni corsa partecipano tre cavalli, indicati con 1, 2 e 3. Per esempio la settima terna 1 3 1 indica che la prima corsa è stata vinta dal cavallo 1, la seconda dal cavallo 3, la terza dal cavallo 1.

(Si tratta delle combinazioni ordinate con ripetizione di tre oggetti, a tre a tre, oppure dell'elenco di tutti i punti di  $N^3$  contenuti nel cubo di lato 3).

Se qualcuno vi chiede di rendere più generale il programma, prevedendo che alle tre corse partecipino non sempre esattamente tre cavalli, bensì  $M_1$ ,  $M_2$  e  $M_3$  cavalli rispettivamente, potrete far fronte al problema semplicemente cambiando gli indici finali dei FOR.

```
INPUT M1, M2, M3
FOR I=1 TO M1
FOR J=1 TO M2
FOR K=1 TO M3
PRINT I,J,K
NEXT K
NEXT J
NEXT I
```

La cosa diventa però molto più complicata appena vi si chiede di scrivere un programma che funzioni

per un numero *qualsiasi* di corse. Bisognerebbe allora scrivere un numero *variabile* di FOR, ma ciò è impossibile. La soluzione consiste nel cambiare completamente filosofia, cercando di scrivere un programma senza FOR.

Per semplicità, vediamo il caso in cui gli indici finali sono tutti uguali ad  $M$  (ad ogni corsa partecipano sempre  $M$  cavalli). Quindi il problema è: come scrivere un nido di  $N$  FOR ognuno dei quali va da 1 a  $M$ ? (con  $N$  e  $M$  variabili, diciamo introdotti con una istruzione INPUT).

L'idea di base consiste nel pensare che ogni volta che non è più possibile incrementare un indice perché ha raggiunto il massimo, bisogna andare alla sua sinistra e trovare il primo indice che possa essere incrementato, incrementarlo, porre ad 1 tutti gli indici alla sua destra e quindi il processo ricomincia con l'indice più a destra.

Procuriamoci dunque un vettore di  $N$  elementi  $A(1), A(2), \dots, A(N)$  e un puntatore  $P$  agli elementi del vettore (cioè una variabile che contiene un particolare indice del vettore).

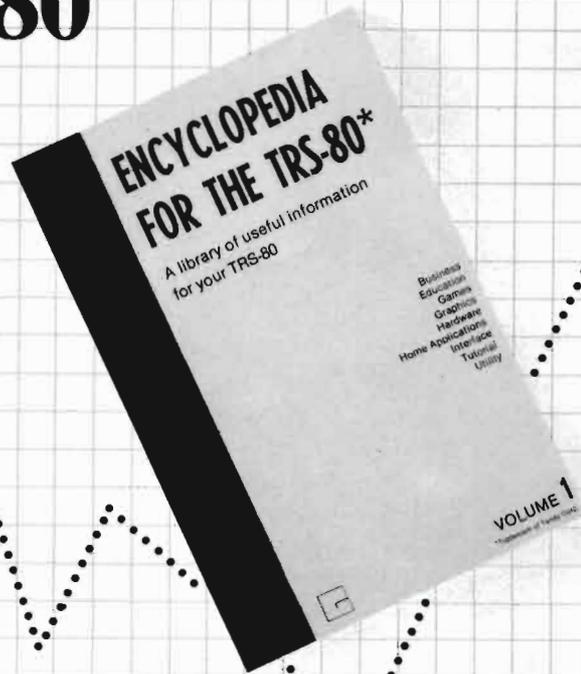
```
10 INPUT N,M
20 FOR I=1 TO N
30 A(I)=1
40 NEXT I
50 FOR I=1 TO N
60 PRINT A(I),
70 NEXT I
80 PRINT
90 P=N
100 A(P)=A(P)+1
110 IF A(P) > M THEN 50
120 A(P)=1
130 P=P-1
140 IF P < 0 THEN 100
150 STOP
```

Questo algoritmo può essere reso più generale apportando le seguenti modifiche:

1. gli indici finali non sono tutti  $M$  ma sono un vettore  $M(1), \dots, M(N)$ ;
2. gli indici iniziali non sono tutti 1 ma un vettore  $L(1), \dots, L(N)$ ;
3. i passi non sono tutti 1 ma un vettore  $P(1), \dots, P(N)$ .

Dopo aver apportato queste modifiche si ha a disposizione un algoritmo che simula un nido di  $N$  FOR, ognuno con un proprio rango e un proprio passo. ■

# Sono usciti i nuovi volumi dell'Encyclopedia for the TRS-80



## Ora la raccolta è completa

L'Encyclopedia for the TRS-80 è la maggior fonte mondiale di informazioni sul TRS-80 mod. I e III. Tutti i dieci volumi contengono programmi pratici e articoli progettati per portare voi e il vostro computer al di là dei limiti imposti dai manuali del costruttore, in un nuovo e affascinante mondo.

Ogni volume contiene tra 15 e 20 articoli accompagnati da diagrammi e listati di programmi. Gli articoli toccano tutti gli argomenti più interessanti: utility, didattica, business, word processing, hardware, grafica, giochi e molti altri.

Pensateci! Più di 150 programmi per 200.000 lire, il prezzo dell'intera raccolta in 10 volumi.

## Cos'è l'Encyclopedia Loader?

Sono 10 cassette del tipo C30, contenenti tutti i programmi dell'Encyclopedia for the TRS-80. L'Encyclopedia Loader vi permette di caricare i programmi velocemente e con facilità, risparmiando le ore di tempo necessarie per la battitura e la correzione degli errori.

Spedire a  
**COMPLETO SOFTWARE**  
 Via Bonporti 36  
 35141 PADOVA

Per ordinare l'intera raccolta a prezzo speciale

Encyclopedia for the TRS-80  
 Volumi 1-10 L. 200.000

Encyclopedia Loader  
 Volumi 1-10 L. 250.000

Per ordinare singoli volumi (indicare i numeri dei volumi)

Encyclopedia for the TRS-80  
 volumi .....  
 L. 23.000 al volume

Encyclopedia Loader  
 volumi .....  
 L. 28.000 alla cassetta

Spese di spedizione e imballo gratuite

Totale lire ..... che pagherò

- contrassegno  
 con un assegno qui allegato  
 versando l'importo su c/cp 16915357 intestato a Completo Software

Nome e cognome .....

Indirizzo .....

Cap, località .....

## COMMODORE PET/CBM

### Un tasto per una istruzione

*Giuseppe Franza di Foggia ha scoperto un piccolo trucco per introdurre un'intera parola chiave del Basic con un solo tasto. Ecco qui di seguito la sua lettera.*

Sono un assiduo lettore sia di *PS* che di *Bit*, che seguo da oltre due anni, tralascio i convenevoli a cui avreste diritto e vi sottopongo un caso particolare che mi successe circa due anni fa.

Io possiedo un CBM 3032, e un giorno per caso mi capitò di battere dei caratteri grafici dopo una REM su una linea di programma, ma dopo aver listato la linea mi accorsi che i grafici furono sostituiti da istruzioni Basic corrispondenti. Da quel momento intuì che forse era possibile usare un solo tasto per dare una istruzione Basic, ma bisognava dare sempre il REM all'inizio della linea e poi bisognava toglierlo, cosa che risulta fastidiosa se unita all'operazione di ricerca del carattere grafico che corrisponde all'istruzione da battere.

A distanza di tempo, mi sono deciso a progettare una routine che mi evitasse l'ingrato compito di andare a togliere tutti i REM che "intestano" le linee; e quindi ho creduto opportuno rendervi partecipi di questa mia curiosità che se pubblicata forse renderà qualche "pettomane" soddisfatto.

I vantaggi della routine:

Sull'area Basic: è possibile l'editing di linee lunghe oltre le due normali di video, risparmiando 5 byte per ogni linea in meno di programma (2 di link, 2 di numero di linea, 1 di End of Line).

Ciò è possibile perché ogni istruzione viene memorizzata con il suo codice e non secondo le sue lettere.

Sul printing: quando bisogna battere un programma per ore, è possibile avere un sensibile risparmio di tempo, battendo per ogni istruzione un solo tasto.

Gli svantaggi:

Sull'area Basic: per ogni linea vi è un byte in più

ABS	6	NEXT	N
AND	/	NOT	(
ASC	F	ON	]
ATN	A	OPEN	
CHR\$	G	OR	0
CLOSE		PEEK	B
CLR		POKE	
CMD		POS	9
CONT		PRINT	
COS	>	PRINT#	
DATA	O	READ	S
DEF		REM	[
DIM	R	RESTORE	X
END		RETURN	Z
EXP	=	RIGHT\$	I
FN	%	RND	;
FOR	M	RUN	V
FRE	8	SAVE	
GET	!	SGN	4
GO	K	SIN	?
GOSUB	Y	SPC	&
GOTO	U	SQR	:
IF	W	STEP	)
INPUT	Q	STOP	\
INPUT#	P	STR\$	D
INT	5	SYS	
LEFT\$	H	TAB(	#
LEN	C	TAN	@
LET	T	THEN	'
LIST		TO	\$
LOAD	←	USR	7
LOG	<	VAL	E
MID\$	J	VERIFY	
NEW	"	WAIT	

*Tabella delle corrispondenze tra le istruzioni e i tasti indicati, intesi in modo Shift (in ordine alfabetico). Non tutte le istruzioni hanno un corrispondente carattere sui tasti.*

del dovuto, se non si batte RETURN su ogni linea listata per eliminare il blank iniziale.

Sull'editing: quando bisogna correggere una linea che va oltre le due righe video, si deve ribatterla per intero corretta sempre secondo la proce-

```
63900 K=1025:J=256:I=098:W=PEEK(42)+J*PEEK(43)-I
63910 FOR L=K TO W:IF PEEK(L+4)=143 THEN POKE L+4,58
63920 L=PEEK(L)+J*PEEK(L+1)-1:NEXT
```

Listato 1. La routine di cui si parla nell'articolo.

# La più diffusa rivista italiana di elettronica pratica allarga l'orizzonte e parla anche di personal computer.

**Sperimentare**, la più autorevole e diffusa rivista di elettronica pratica, tende a perfezionare i suoi contenuti e ad ampliare l'orizzonte. Oltre alle realizzazioni per gli amatori e gli specialisti di elettronica nei più svariati campi, la rivista, da questo numero, presenterà mensilmente degli articoli dedicati al personal computer, con particolare riguardo al più diffuso di essi: **il Sinclair**. Hardware, software, consigli e idee da sviluppare insieme, saranno un contenuto abituale di **Sperimentare**.

Per questo motivo, **Sperimentare** sarà d'ora in poi la rivista non solo del tecnico elettronico e dell'hobbista, ma anche il mensile dell'utente di personal computer. Acquista il numero in edicola con l'inserto **Sinclub**. Un numero stimolante della rivista senza confronti.

**SPERIMENTARE**

UNA PUBBLICAZIONE J.C.E.



## I SEGRETI DEI PERSONAL

### COMMODORE PET/CBM

dura e quindi si potrà eseguire di nuovo la routine che elimina il REM.

#### Note

63900: K = Inizio area Basic del CBM.  
J = Limite superiore di un byte.  
I = Numero di byte che la routine occupa, più di 2 byte di fine testo Basic.  
W = Puntatore dell'ultimo byte del programma-utente.

63910: Loop sui link del programma-utente in area Basic; controllo che il primo byte di linea sia una REM (143), se lo è lo scambio con due punti (58).

63920: Punto di Link successivo, saltando la linea.

Nella routine dovranno essere eliminati tutti gli spazi, altrimenti I=098 non è più valida.

Sulla linea 63920 decremento L perché tale puntatore sarà posizionato sul Link successivo dal NEXT che incrementa L.

Quando L supera l'ultimo link di programma, entra nell'area della routine, quindi si esaurirà il loop.

È importante che il programma-utente non superi i numeri di linea della routine.

Dopo aver battuta la routine per intero, si inizia a battere il testo Basic del programma secondo questa procedura:

- Batto il numero di linea, il REM, e quindi la linea codificata.
- La linea codificata ha per ogni sua istruzione il relativo carattere grafico (vedi tabella delle corrispondenze).
- Ogni carattere grafico al di fuori delle " " verrà interpretato come una istruzione dal sistema Basic del CBM.

In altri termini dopo un REM tutte le istruzioni (quasi tutte), possono essere rappresentate dai corrispondenti caratteri grafici; ma nel listato vedremo le vere istruzioni dopo ogni REM di linea.

Dopo aver completato il necessario editing del programma, per provarlo bisognerà eliminare tutti i REM che non servono, per farlo basta dare il comando diretto GOTO 63900.

È utile chiudere il programma con END se non si vuole eliminare la routine in coda, che potrà servire per altre modifiche in altri momenti. ■

## 1983 : l'inizio

Una grande impresa editoriale  
comincia oggi nella vostra edicola

### Il micro-millennio è cominciato.

Siamo nell'era dell'elettronica e dell'informatica.

Una rivoluzione silenziosa sta cambiando il nostro modo di vivere, pensare, esprimerci.

Una scelta ci sta oggi davanti: subire le novità che ci attendono oppure viverle da protagonisti; impadronirci del futuro o farcene travolgere. Decidiamo!

Varcare le soglie del micro-millennio conoscendone tutti i segreti è oggi possibile.

Oggi c'è E.I. l'enciclopedia dell'elettronica e dell'informatica.

Un'opera unica al mondo, scritta da specialisti per uomini-protagonisti.

È completa, rigorosa, documentata, facile da capire... anche se, parla di elettrotecnica, elettronica allo stato solido, elettronica digitale, microprocessori, comunicazioni, informatica di base, informatica e società.

Tutto quello che volete e dovete sapere sul micro-millennio che ci sta aspettando.



**Enciclopedia**

**di Elettronica e Informatica**

#### 50 fascicoli settimanali

- 12 pagine di elettronica digitale e microprocessori
- 16 pagine di informatica (oppure elettronica di base e comunicazioni)
- 1 scheda (2 pagine) di elettrotecnica

per ottenere in meno di un anno

- 7 grandi volumi
- 1400 pagine complessive
- 1 volume schede di elettrotecnica

L'opera è arricchita da circa 700 foto e 2200 illustrazioni a colori.

Ogni settimana l'elettronica, l'informatica,  
l'elettrotecnica in un unico fascicolo



**Enciclopedia di Elettronica e Informatica**

Oggi in edicola... domani nella vostra biblioteca



**GRUPPO EDITORIALE  
JACKSON**

In collaborazione con il Learning Center

**TEXAS INSTRUMENTS** 

TEXAS TI 99/4A

## PRINT e INPUT in un contesto grafico

Sergio Borsani

In occasione del primo numero di questa rubrica dedicata ai possessori del TI 99/4A desidero rivolgere un particolare saluto tramite il breve programma di listato 1 che vi prego di mandare in esecuzione.

```

100 CALL CLEAR
110 DATA 67,73,65,79,32,84
120 DATA 69,88,65,78,73
130 RESTORE 110
140 FOR I=11 TO 21
150 READ L
160 CALL HCHAR(12,I,L)
170 NEXT I
180 CALL COLOR(2,16,7)
190 CALL HCHAR(10,9,42,15)
200 CALL VCHAR(11,9,42,3)
210 CALL VCHAR(11,23,42,3)
220 CALL HCHAR(14,9,42,15)
230 FOR I=1 TO 50
240 CALL COLOR(2,7,16)
250 CALL COLOR(2,16,7)
260 NEXT I
270 END
    
```

Listato 1.

La scritta è apparsa alla riga 12 e non nella parte bassa del monitor come avviene con l'istruzione PRINT.

La possibilità di accedere ad ogni posizione del video è particolarmente interessante quando si lavora in un ambiente grafico poiché l'esecuzione di ogni PRINT provoca lo spostamento verso l'alto di un disegno già presente. Per chi avesse ancora poca dimestichezza con il Basic, nel nostro esempio le istruzioni DATA contengono i codici ASCII delle lettere che formano la frase. L'istruzione RESTORE non è qui indispensabile, ma lo diventa nel caso di ripetute analoghe istruzioni in un programma articolato. Gli elementi contenuti nei DATA, vengono letti sequenzialmente indipendentemente dalla loro posizione, il RESTORE serve a riportare il puntatore nella posizione desiderata.

Lo stesso risultato può essere ottenuto utilizzando

```

110 A$="CIAO TEXANI"
120 FOR I=1 TO 11
130 L=ASC(SEG$(A$,I,1))
140 CALL HCHAR(12,10+I,L)
150 NEXT I
    
```

Listato 2.

funzioni di stringa. Le righe dalla 110 alla 170 possono essere sostituite da quelle riportate nel listato 2 e in questo modo si evita la conversione delle lettere in codice ASCII.

Il TI Extended Basic mette a disposizione il comando DISPLAY AT(x,y) che svolge le stesse funzioni consentendo un notevole risparmio di tempo e di memoria. Nel primo caso sono stati occupati 129 byte, nel secondo 98, con l'istruzione in Extended Basic solo 33.

In molte applicazioni si rende necessaria anche l'introduzione di dati in ogni posizione video. Anche in questo caso l'Extended Basic offre il comando ACCEPT AT(x,y) ma anche con il Basic residente nella consolle è possibile ottenere un effetto analogo usando l'istruzione CALL KEY. Consideriamo dapprima i suoi elementi di sintassi: CALL KEY(0,K,S). 0 definisce la tastiera, in questo modo il computer associa ad ogni tasto il valore ASCII corrispondente; tale valore viene attribuito alla variabile K; S assume invece i seguenti valori: 0 quando non viene premuto alcun tasto, -1 quando viene premuto lo stesso tasto premuto in precedenza, 1 quando viene premuto un nuovo tasto.

Un uso dell'istruzione CALL KEY, oltre a quello descritto nel manuale è riportato nel listato 3.

```

100 CALL CLEAR
110 C=10
120 CALL KEY(0,K,S)
130 IF S=0 THEN 120
140 A$=A$&CHR$(K)
150 C=C+1
160 CALL HCHAR(12,C,K)
170 GOTO 120
    
```

Listato 3.

Questi passi di programma corrispondono ad un INPUT nella riga 12 a partire dalla colonna 10, se ora digitate una frase, essa viene visualizzata in quella posizione e viene memorizzata come variabile di stringa A\$. Infatti, se interrompete il ciclo premendo FCTN(4)=CLEAR e impostate il comando immediato (senza numero di linea) PRINT A\$, ritorna la frase digitata in precedenza.

A conclusione voglio presentare un gioco che ben sintetizza tutte le considerazioni esposte (listato 4).

Si tratta di colpire un castello a colpi di cannone: vengono messi a disposizione 5 tiri e di volta in volta bisogna specificare l'alzo scrivendo i gradi. Il programma prevede due cifre per gli interi ed una per il valore decimale, pertanto, per scrivere un angolo di 8°, è necessario digitare 08 0.

SINCLAIR ZX80/ZX81

## Scritte giganti, grafici e disegni, messaggi video

Enrico Ferreguti

La prima routine serve a stampare parole a lettere ingrandite. A questo punto qualcuno avrà già storto il naso perché a conoscenza di programmi ben più brevi, che però hanno due punti a sfavore rispetto a "Scritte giganti": prima di tutto il tempo di esecuzione: nel nostro programma è pari a 3 sec. per stampare una parola di 6 lettere, quando negli altri il tempo di esecuzione per la stessa parola sale al minuto, proprio perché usano la mappa dei caratteri residente in ROM. Secondariamente i caratteri stampati dalla nostra routine sono più piccoli, comunque leggibilissimi.

Come si capisce dal listato la parola da ingrandire è contenuta nella stringa G\$. Bisogna inoltre specificare in X ed Y le coordinate d'inizio della stampa.

La routine "riconosce" solo le lettere dell'alfabeto italiano, quindi J, K, W, X, Y sono state tralasciate. L'utente può comunque aggiungerle usando la tabella 1 per ricavare il numero di linea in cui inserire la stampa del carattere. Durante questa operazione bisogna tener presente che il carattere gigante è stato costruito con una matrice di 2x3 caratteri grafici.

A	-->014	R	-->017
B	-->020	T	-->020
C	-->026	U	-->026
D	-->032	V	-->032
E	-->038	W	-->038
F	-->044	X	-->044
G	-->050	Y	-->050
H	-->056	Z	-->056
I	-->062		
L	-->068		
M	-->074		
N	-->080		

Tabella 1.



Figura 1. Esempio di esecuzione.

```

100 *****
110 LET X=0
120 LET Y=0
130 LET G$="SCRITTE"
140 GOSUB 200
150 LET X=X+6
160 LET Y=Y+4
170 LET G$="GIGANTI"
180 GOSUB 200
190 STOP
200 FOR K=1 TO LEN G$
201 GOSUB 300 G$(K)#3+100
202 IF G$(K)="I" THEN LET X=X+2
203 IF G$(K)>"I" THEN LET X=X+4
204 NEXT K
205 RETURN
214 PRINT AT Y,X;" "TAB X;" "
215 RETURN
217 PRINT AT Y,X;" "TAB X;" "
218 RETURN
220 PRINT AT Y,X;" "TAB X;" "
221 RETURN
223 PRINT AT Y,X;" "TAB X;" "
224 RETURN
226 PRINT AT Y,X;" "TAB X;" "
227 RETURN
229 PRINT AT Y,X;" "TAB X;" "
230 RETURN
232 PRINT AT Y,X;" "TAB X;" "
233 RETURN
235 PRINT AT Y,X;" "TAB X;" "
236 RETURN
238 PRINT AT Y,X;" "TAB X;" "
239 RETURN
242 PRINT AT Y,X;" "TAB X;" "
243 RETURN
245 RETURN
250 PRINT AT Y,X;" "TAB X;" "
251 RETURN
253 PRINT AT Y,X;" "TAB X;" "
254 RETURN
256 PRINT AT Y,X;" "TAB X;" "
257 RETURN
259 PRINT AT Y,X;" "TAB X;" "
260 RETURN
262 PRINT AT Y,X;" "TAB X;" "
263 RETURN
265 PRINT AT Y,X;" "TAB X;" "
266 RETURN
268 PRINT AT Y,X;" "TAB X;" "
269 RETURN
271 PRINT AT Y,X;" "TAB X;" "
272 RETURN
274 PRINT AT Y,X;" "TAB X;" "
275 RETURN
277 PRINT AT Y,X;" "TAB X;" "
278 RETURN
280 PRINT AT Y,X;" "TAB X;" "
281 RETURN
283 RETURN

```

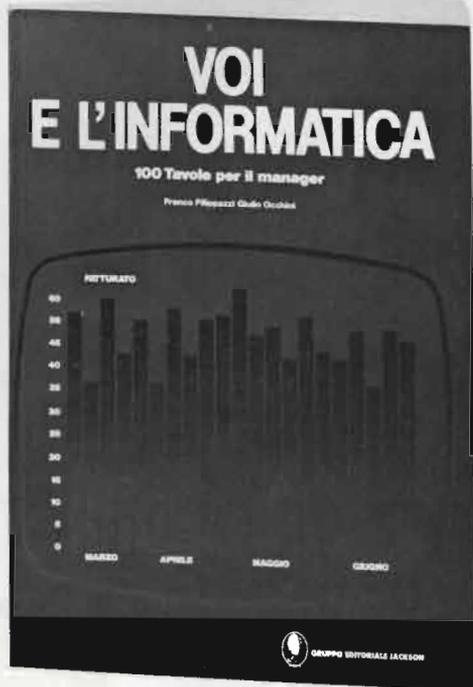
Listato 1. Il programma "Scritte giganti".



Franco Filippazzi - Giulio Occhini

# VOI E L'INFORMATICA

100 tavole per il manager



**Gli strumenti dell'informatica; l'informatica e l'Azienda; prospettive tecnologiche e sistematiche; verso la Società Informatica:**

i temi fondamentali della scienza che sta rivoluzionando il mondo della produzione e della gestione aziendale, in un volume scritto in funzione delle nuove esigenze dei quadri direttivi e manageriali. Un'opera agile ed esauriente, nella quale un testo eminentemente pratico si accompagna a chiarissime tavole commentate, che favoriscono un'immediata comprensione degli argomenti esposti.

**116 pagine. Lire 15.000**



**GRUPPO EDITORIALE  
JACKSON**

**Nelle migliori librerie tecnico-scientifiche**

### SINCLAIR ZX80/ZX81

- 20 - 40 Se la stringa è minore di 32 caratteri si provvede a riempirla con degli spazi fino al trentaduesimo.
- 50 - 70 Avviene la visualizzazione a destra controllata dal loop I (32-1), consistente nello stampare i primi I caratteri.
- 80 - 100 Succede l'inverso: il loop I parte da 1 ed arriva a 32 e si stampano gli ultimi I caratteri.
- 110 Si ripete dal passo 50.

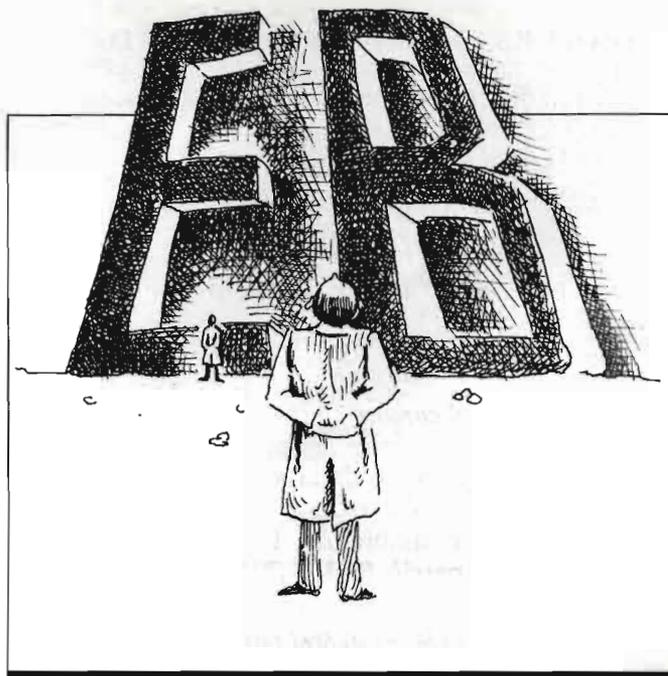
#### *Possibili modifiche*

- 1) Per un loop calcolato, invece che infinito, eseguire queste modifiche:

```
15 INPUT N
17 FOR X=1 TO N
110 NEXT X
```

dove N è il numero di volte che si vuol vedere passare la stringa.

- 2) Per aumentare o diminuire la velocità agire sugli STEP nelle righe 50 e 80.
- 3) La routine non usa l'istruzione CLS, quindi lo schermo si altera solo nella riga puntata da S.
- 4) Gira su 1 K di RAM, ma si presta ad essere inserito in programmi più grandi. ■



## COMMODORE VIC 20

### Gestione del cursore nel VIC 20

Enrico Bossaglia

Alcune istruzioni mancanti sul VIC si ripercuotono sulla compattezza dei programmi. Un esempio è il PRINT AT X,Y che permette di indirizzare il cursore sullo schermo, e la cui mancanza costringe a noiosi cicli con i caratteri di controllo.

Non è, invece, difficile portare direttamente il cursore in un punto qualsiasi dello schermo e, con una sola istruzione, si può avere il cursore lampeggiante senza doverlo simulare in Basic.

In tabella 1 è riportata la mappa delle locazioni più interessanti nella pagina zero di memoria del VIC, con la spiegazione del contenuto di ognuna di esse.

Datele una attenta lettura, è una buona fonte di idee, e vi sarà più chiaro quanto segue.

#### Posizionamento diretto del cursore

Dalla tabella 1 apprendiamo che le locazioni 211 (\$D3) e 214 (\$D6) contengono rispettivamente il numero della colonna e della riga in cui si trova il cursore. Per cambiare la posizione del cursore, però, non è sufficiente modificare queste due locazioni ma anche le celle di memoria 209, 210 (\$D1, \$D2) e 201, 202 (\$C9, CA). Ma di questo non vi dovete preoccupare perché esiste una routine del sistema operativo (58759, \$E587) che aggiorna queste quattro locazioni basandosi sul contenuto delle prime due.

Perciò l'esatta sequenza di istruzione è:

```
POKE 211,X:POKE 214,Y:SYS(58759)
```

X e Y sono il numero di colonna (0-21) e di riga (0-22). Aggiungendo un RETURN si può utilizzare come subroutine.

#### Visualizzazione del cursore

Il cursore è controllato dalla locazione 204 (\$CC). Se questa contiene 0 il cursore è visualizzato lampeggiante, altrimenti è disabilitato. L'istruzione per attivare il cursore è perciò POKE 204,0. E per disattivarlo: POKE 204,1.

Un uso interessante delle due utility viste è la routine Basic riportata nel listato 1.

HEX	DEC	Commento
2B-2C	43-44	Inizio dei programmi Basic
2D-2E	45-46	Puntatore fine programma inizio variabili
2F-30	47-48	Puntatore fine variabili inizio array
31-32	49-50	Puntatore fine array
33-34	51-52	Puntatore inizio zona stringhe
37-38	55-56	Puntatore fine memoria
61-66	97-102	Accumulatore #1 virgola mobile
90	144	Registro di stato ST per l'I/O
98	152	Numero dei file aperti
99	153	Numero della periferica in input (norm. tastiera=0)
9A	154	Numero della periferica di output (norm. schermo=3)
A0-A2	160-162	Contatori della funzione ora
B7	183	Numero dei caratteri nel filename
B9	185	Indirizzo secondario utilizzato correntemente
BB-BC	187-188	Puntatore al filename in memoria
C5	197	Numero del tasto premuto. Si ha un numero da 0 a 64 (64 se nessun tasto è premuto)
C6	198	Numero di caratteri nel buffer della tastiera
C7	199	Flag del reverse on/off (1/0)
C9-CA	201-202	Posizione del cursore (riga e colonna)
CC	204	Lampeggio cursore (0=on, 1=off)
D1-D2	209-210	Indirizzo dell'inizio della riga in cui si trova il cursore
D3	211	Posizione del cursore sulla riga
D6	214	Riga dello schermo sulla quale si trova il cursore. Per spostare il cursore bisogna cambiare le locazioni 201, 202, 209, 210, 211, 214
FB-FE	251-254	Spazio utilizzabile in pagina zero

Tabella 1. Mappa delle locazioni più utili in pagina zero.

La sua funzione è di gestire un input controllato, evitando tutti i problemi che nascono dall'uso dell'istruzione INPUT.

La routine permette di definire:

1. una descrizione del campo;
2. una proposta (default);
3. la lunghezza del campo, indicata sullo schermo da due barre;
4. un intervallo di caratteri accettati;
5. la disabilitazione dei caratteri " , " e SPAZIO indipendentemente dall'intervallo scelto;
6. le coordinate XX,YY di partenza sullo schermo.

Sono abilitate le funzioni di editing: CRSR LEFT, RIGHT e INST, DEL.

# I SEGRETI DEI PERSONAL

## COMMODORE VIC 20

### Listato 1. Routine di input controllato.

```
10 sp$=" "
20 nm$="Citta'":df$="Francavilla F."
30 xx=0:yy=10
40 in$="1803322410"
50 gosub60000
60 nm$="CAP":df$="70121"
70 yy=13:in$="0504705700"
80 gosub60000:ifcrthen20
90 end
100 rem
59720 rem*****
59730 rem*
59740 rem* routine di inPut controllato
59750 rem*
59760 rem*   variabili di inPut:
59770 rem*   xx=colonna di inizio descrizione campo
59780 rem*   yy=ri9a di inizio descr. campo
59790 rem*   nm$=descrizione campo
59800 rem*   df$=ProPosta (default)
59810 rem*   in$="llminmaxsc"
59820 rem*   ll = lunghezza del campo (00-80)
59830 rem*   min =min.cod. ascii accettato (000-255)*
59840 rem*   max =max. cod.ascii accettato (000-255)*
59850 rem*   s = spazio: 1=abilitato 0=disab.
59860 rem*   c = virgola: 1=abilitata 0=disab.
59870 rem*
59880 rem*   sp$=n spazi con n=ll
59890 rem*
59900 rem*   variabili di lavoro:
59910 rem*   cx,cy,ln,mn,mx,sp,cm,cs,a$
59920 rem*
59930 rem*   variabili di ritorno dalla subroutine:
59940 rem*
59950 rem*   r$=stringa digitata
59960 rem*   cr=1 se l'inPut e' stato confermato con
59970 rem*   crsr UP altrimenti cr=0
59980 rem*
59990 rem*****
60000 cx=xx:cy=yy:ln=val(left$(in$,2)):mn=val(mid$(in$,3,3))
60010 Poke211,xx:Poke214,yy:sys(58759)
60020 mx=val(mid$(in$,6,3)):sp=val(mid$(in$,9,1)):cm=val(right$(in$,1))
60030 r$=left$(df$+sp$,ln)
60040 Printnm$:ifln=0then60230
60050 Print" |:in$:"|":cx=cx+len(nm$)+1:cs=1
60060 Poke211,cx:Poke214,cy:sys(58759):Poke204,0
60070 geta$:ifa$=""then60070
60080 ifa$=" "andsp=0then60070
60090 ifa$=" "then60200
60100 ifa$=","andcm=0then60070
```

(segue)

## COMMODORE VIC 20

### Segue Listato 1.

```

60110 ifa$=","then60200
60120 ifa$=chr$(13)ora$="␣"thenora$="":goto60210
60130 ifa$="␣"thenora$="":goto60210
60140 ifa$="!"andcs>1then9osub60260:9osub60210:goto60060
60150 ifa$="!"andcs<1then9osub60280:9osub60210:goto60060
60160 ifa$=chr$(20)andcs>1thenr$=left$(r$,cs-2)+mid$(r$,cs)+" ":a$="!":goto60140
60170 ifa$=chr$(148)andcs<1thenr$=left$(r$,cs-1)+" "+mid$(r$,cs,ln-cs):9osub60210:goto60060
60175 ifa$<chr$(31)ora$>chr$(127)anda$<chr$(160)then60070
60180 ifa$>chr$(mm)anda$<chr$(mx)then60200
60190 goto60060
60200 r$=left$(r$,cs-1)+a$+mid$(r$,cs+1):a$="!":9osub60210:goto60150
60210 poke204,1:poke211,xx+len(nm$):poke214,yy:sys(58759)
60220 print"!"r$!":return
60230 print"!!?":poke211,len(nm$)+2:sys(58759):poke204,0
60240 geta$:ifa$<chr$(mm)ora$>chr$(mx)then60240
60250 poke204,1:printa$:return
60260 cx=cx-1:ifcx=-1thencx=21:cy=cy-1
60270 cs=cs-1:return
60280 cx=cx+1:ifcx=22thencx=0:cy=cy+1
60290 cs=cs+1:return
    
```

La conferma dei dati in ingresso si ha con i tasti RETURN, CRSR DOWN e CRSR UP. I primi due permettono il passaggio al campo successivo e l'ultimo, invece, permette di tornare al precedente.

Se la lunghezza del campo era stata fissata uguale a zero allora viene accettato solo un carattere e non viene attesa la conferma con uno dei tasti visti prima.

La stringa ottenuta da questa subroutine è sempre lunga quanto il campo, eccetto il caso lunghezza = 0.

Le prime righe del programma sono solo un esempio di utilizzazione. Va notato che è necessario definire all'inizio del programma la variabile SP\$ composta di tanti spazi quanto è la lunghezza del campo maggiore. ■

# COMPUTERWORLD ITALIA

## Programma 1983

### UNA TAVOLA ROTONDA AL MESE

La certificazione del software  
 DP Manager, un ruolo in transizione  
 Il personal nell'azienda  
 Le reti locali in Italia  
 Brokeraggio di sistemi usati: quale convenienza?  
 Il package e la banca.  
 Metodologie DP: sono imparabili?  
 La rete pubblica di trasmissione dati  
 Office Automation: autonomia o integrazione?  
 Formazione: è propria l'ultima

### dei problemi?

Il DP come servizio parabanca

### SERVIZI "IN PROFONDITÀ"

Pianificazione strategica dei sistemi informativi  
 Intervista a James Martin, il "guru" dell'informatica  
 Tecnologie delle unità a dischi  
 Scegliere un sistema di data base  
 Capacity Planning  
 Data Encryption Standard: come implementarlo?  
 Cosa aspettarsi dalla Quinta Generazione  
 Intervista a Janey Masuda  
 Programmazione senza errori  
 Data Base Mochiras

Infatuarsi di un androide  
 Una lezione dal caso Wells Fargo Bank  
 Come sopravvivere ad un disastro  
 ... e molti altri ancora!

### SERVIZI SPECIALI

Teletext e Videotex  
 Il nuovo scenario del Data Entry  
 Data Base Management Systems  
 Reti di trasmissione dati  
 Software di base e di utilità  
 ... e molti altri ancora!

### INCHIESTE SULL'ITALIA

DP Italiana: perché tante Associazioni

Il centro DP come centro di profilo  
 DP bancario: quanto si decentra?  
 Viaggio nei "santuari" dell'informatica italiana (su vari numeri)  
 Il DP nelle assicurazioni italiane  
 Il leasing nell'informatica  
 ... e molte altre ancora!

### ... E IN PIU' TUTTE LE RUBRICHE FISSE:

Attualità, Spazio Banche, Brief Room, Software e Servizi, Comunicazioni, Microinforze, Sistemi e Periferiche, Office Automation, Industria e Mercato, Sistemi in Vendita.

**Servizi, rubriche, inchieste, tavole rotonde, attualità, anticipazioni, indiscrezioni, commenti, opinioni, analisi di mercato, di prodotto, novità, ricerca e offerta di personale, di sistemi, di software, interviste esclusive, servizi in profondità, settori emergenti ... e altro ancora!**

Per abbonarsi, inviare assegno di L.78.000 intestato a:  
**COMPUTER PUBLISHING GROUP** - Via Rosellini, 12  
 20124 Milano - Tel. 02/6880951-2-3-4-5 - Tlx 333436-GEJ  
 oppure effettuare versamento su c/c n. 26732204-CPO



SINCLAIR ZX SPECTRUM

## Come programmare uno sfondo scorrevole

Marcello Spero

Capita spesso di essere alle prese con programmi che prevedono uno schermo "virtuale" che si estende ben oltre i limiti di quello reale. È questo il caso dei grandi tabelloni di dati o, meno seriamente, di quei giochi che hanno bisogno di uno sfondo scorrevole (chi non conosce il "Defender"?) per simulare il movimento di oggetti che in realtà stanno fermi rispetto allo schermo.

Quando a dover scorrere è solo una sezione ridotta dello schermo (diciamo non più di 70 od 80 caratteri, che potranno essere raggruppati a formare due o tre righe piuttosto che tre o quattro colonne) una routine in Basic può essere soddisfacente, specie se la velocità non è essenziale; volendo muovere tutto lo schermo è invece necessario usare il linguaggio macchina.

I programmi uno e due presentano uno dei possibili modi per far scorrere orizzontalmente una riga, mentre il programma tre sposta verticalmente una colonna (verso il basso, perché verso l'alto ci pensa già lo scrolling automatico della macchina); la riga, o la colonna, è contenuta in un vettore, e può essere lunga a piacere. Dovendo invece spostare qualcosa che esiste solo sullo schermo si può utilizzare una routine tipo quella del programma quattro, cioè del tipo "leggi e scrivi", dove ogni carattere è letto per mezzo della funzione SCREEN\$ e riscritto nella nuova posizione; l'esempio si riferisce al movimento orizzontale verso destra, ma può essere facilmente modi-

```

20 LET f=0
30 INPUT LINE a$
40 FOR i=1 TO (33-LEN a$ AND L
EN a$ < 28) + (6 AND LEN a$ >= 28)
50 LET a$=a$+" "
60 NEXT i
70 FOR i=1 TO LEN a$
80 PRINT AT 10, (32-i AND i < 32)
; a$( (i-31 OR i < 32) TO i)
90 IF f=1 AND i < 32 THEN PRINT
AT 10,0; a$(LEN a$-31+i TO LEN a$
)
95 PAUSE 10
100 NEXT i
110 LET f=1
120 GO TO 70
    
```

Listato 1. Routine Basic per lo spostamento laterale di una riga verso sinistra.

```

20 LET f=0
30 INPUT LINE a$
40 FOR i=1 TO (33-LEN a$ AND L
EN a$ < 28) + (6 AND LEN a$ >= 28)
50 LET a$=" "+a$
60 NEXT i
70 FOR i=0 TO LEN a$-1
80 PRINT AT 10,0; a$(LEN a$-i T
O (LEN a$-i+31 AND i >= 31) + (LEN a
$ AND i < 31))
90 IF f=1 AND i < 31 THEN PRINT
AT 10, i+1; a$( TO 31-i)
95 PAUSE 10
100 NEXT i
110 LET f=1
120 GO TO 70
    
```

Listato 2. Routine Basic per lo spostamento laterale di una riga verso destra.

```

20 LET f=0
30 INPUT LINE a$
40 FOR i=1 TO (23-LEN a$ AND L
EN a$ < 18) + (6 AND LEN a$ >= 18)
50 LET a$=a$+" "
60 NEXT i
70 FOR i=0 TO LEN a$-1
75 FOR j=0 TO (i AND i < 21) + (21
AND i >= 21)
80 PRINT AT j, 16; a$(i+1-j)
85 NEXT j
90 IF f=1 AND i < 21 THEN FOR j=
i+1 TO 21: PRINT AT j, 16; a$(LEN
a$-j+i+1): NEXT j
95 IF f=0 THEN PAUSE 25-i
100 NEXT i
110 LET f=1
120 GO TO 70
    
```

Listato 3. Routine Basic per lo spostamento verticale di una colonna verso il basso.

```

20 PRINT AT 10,15;"Personal So
ftware"
30 FOR i=0 TO 31
40 FOR j=1 TO 31
50 PRINT AT 10, j-1; SCREEN$ (16
; j)
60 NEXT j
70 PRINT AT 10,31;" "
80 NEXT i
    
```

Listato 4. Routine Basic per lo spostamento laterale verso destra di una riga dello schermo.

ficato per gli altri casi. Questo tipo di routine non può essere usata con caratteri grafici, per una limitazione della funzione SCREEN\$, che non è in grado di riconoscerli. Veniamo adesso al movimento dell'intero schermo che, come abbiamo detto, richiede il linguaggio macchina se non altro per ragioni di velocità.

Il primo caso riguarda lo spostamento di un pixel per volta; un pixel è l'unità grafica che compone qualsiasi tipo di carattere appaia sullo schermo ed an-

# I SEGRETI DEI PERSONAL

## SINCLAIR ZX SPECTRUM

che il puntino che otteniamo con l'istruzione PLOT x,y. Questo tipo di movimento (o "scrolling") laterale è molto graduale e preciso ma d'altra parte ancora relativamente lento; è molto indicato per spostare la grafica e in tutti quei casi in cui la lentezza del movimento renderebbe troppo brusco il salto di un intero carattere.

Il meccanismo su cui si basa il programma è molto semplice, caso classico di procedimento che in un linguaggio evoluto avrebbe richiesto strani procedimenti e molte istruzioni mentre in Assembler, grazie alle particolari istruzioni disponibili, risulta estremamente lineare. Si tratta infatti di far "slittare" una intera riga di un solo bit verso destra o sinistra, movimento ottenuto con una serie di istruzione rr (verso destra) o rl (verso sinistra) (listati 5 e 6).

Un solo inconveniente, legato al tipo di soluzione adottata sullo Spectrum per la grafica: sullo schermo non ci devono essere più di due colori diversi, pena

la "migrazione" dei caratteri da un colore all'altro; d'altra parte questa è una limitazione che dobbiamo sopportare ogni volta che usiamo la grafica ad alta risoluzione.

Secondo caso, intermedio: sempre due soli colori per volta, una maggiore velocità di scorrimento con un'impressione visiva ancora buona; si tratta di un tipo di movimento che sfrutta in maniera impropria alcune istruzioni Assembler create per il trattamento delle cifre BCD nelle operazioni con i numeri decimali, dove è necessario scindere un byte nelle due metà che lo compongono: rld e rrd, rispettivamente. Sarebbe interessante esaminare più a fondo questa ed altre particolarità del linguaggio macchina dello Z 80, ma lo spazio non ce lo consente in questa sede. (listati 7 e 8).

Il risultato pratico, comunque, è il movimento di mezzo carattere per volta, con un meccanismo di "slittamento" simile a quello di prima; tenete presente che questa, come del resto tutte le routine che stia-

```

ld hl, 22527      03
                  0003
                  0005
ld c, 192         14
                  1075
ld b, 32          10
                  102
and a             00
rl(hl)            0007
                  0003
dec hl           40
djnz, -5         10
                  1001
dec c            10
ret z            00
jr, -12          04
                  044
    
```

Listato 5. Routine in linguaggio macchina per lo spostamento a sinistra dell'intero schermo un pixel alla volta.

```

ld hl, 22527      03
                  0003
                  0005
ld c, 192         14
                  1075
ld a, 0           00
ld b, 32          10
                  102
rld              00
                  007
dec hl           40
djnz, -5         10
                  1001
dec c            10
ret z            00
jr, -13          04
                  043
    
```

Listato 7. Routine in linguaggio macchina per lo spostamento a sinistra dello schermo mezzo byte alla volta.

```

ld hl, 16384      03
                  00
                  04
ld c, 192         14
                  102
ld b, 32          10
                  100
and a             00
rr(hl)            0007
                  0003
inc hl           10
djnz, -5         10
                  1001
dec c            10
ret z            00
jr, -12          04
                  044
    
```

Listato 6. Routine in linguaggio macchina per lo spostamento a destra dell'intero schermo un pixel alla volta.

```

ld hl, 16384      03
                  00
                  04
ld c, 192         14
                  102
ld a, 0           00
ld b, 32          10
                  100
rrd              00
                  107
inc hl           10
djnz, -5         10
                  1001
dec c            10
ret z            00
jr, -13          04
                  043
    
```

Listato 8. Routine in linguaggio macchina per lo spostamento a destra dello schermo mezzo byte alla volta.

## SINCLAIR ZX SPECTRUM

```

ld d, 0          167
                 222
                 99
ld hl, 22526     33
                 254
                 074
ld c, 192       14
                 192
ld b, 31        6
                 31
ld a, (hl)     126
inc hl         35
ld (hl), a    119
dec hl        43
dec hl        43
djnz, -7      16
                 249
inc hl        35
l (hl), d    114
dec hl        43
dec c         13
jr z, +3      46
                 3
dec hl        43
jr, -18       24
                 238
ret c         216
ld d, 0       22
                 0
ld hl, 23294   33
                 254
                 90
ld c, 24       14
                 24
cf             55
r, -29        24
                 227

```

Listato 12. Routine in linguaggio macchina per lo spostamento a destra di un carattere alla volta, con movimento degli attributi.

quindi sottrarre al valore originale della RAMTOP (che, vi ricordo, è 32599 per il 16 K e 65367 per il 48 K) l'occupazione di memoria della routine prescelta più uno, ed il risultato sarà il numero da fornire in risposta alla richiesta "nuova ramtop?" che vi verrà fatta dal programma di caricamento.

Immediatamente otterrete in risposta l'indirizzo di inizio della routine, che userete nel programma dimostrativo, od in qualsiasi altro programma, per richiamarla.

Il programma 14 è, appunto, un programma dimostrativo; fornitegli una stringa di caratteri e la vedrete spostarsi con direzione e velocità dipendenti dalla routine che avrete scelto. La pausa all'interno del programma serve a diminuire la velocità dello spostamento, e può essere variata a piacere; anzi, provate ad eliminarla del tutto per rendervi conto della effettiva velocità del linguaggio macchina. La riga 40 è adatta alla routine per lo scrolling di carattere, mentre dovrete restituire il 31 con un 62 per il mezzo carattere o con 255 per il pixel; se non lo fate l'unica conseguenza sarà uno spostamento più breve.

A questo punto non mi resta che ricordarvi che il

vero uso di questi programmini in linguaggio macchina è all'interno di programmi Basic: a quanto una vostra applicazione?

```

5 BORDER 0: PAPER 0: INK 7:
  CLS
10 INPUT "nuova ramtop:", r
20 CLEAR r
30 LET r=r+1
40 PRINT AT 10,5;"indirizzo di
  inizio": TAB 8;"della routine"
  TAB 12;r
50 READ i
60 POKE r,i
70 GO TO 30
100 DATA linguaggio macchina

```

Listato 13. Routine di caricamento del linguaggio macchina.

```

5 BORDER 0: PAPER 0: INK 7:
  CLS
10 INPUT TAB 10;"testo e grafi
  ca": TAB 8;"che dovrà scorrere"
  LINE t$
20 INPUT TAB 7;"indirizzo di i
  nizio": TAB 9;"della routine": T
  AB 12;i
30 PRINT AT 10,0;t$
40 FOR j=0 TO 31
50 RANDOMIZE USA j
60 PAUSE 10
70 NEXT j

```

Listato 14. Programma dimostrativo di spostamento di una stringa di caratteri. ■



# MICROPROCESSORI E MICROCOMPUTER

## ELEMENTI DI TRASMISSIONE DATI

Affronta in maniera chiara e facile gli argomenti relativi alla trasmissione dei dati e segnali in genere. In particolare il libro si sofferma anche sui problemi che si incontrano lavorando "on line", soprattutto quelli connessi con la ricerca dei guasti o del miglioramento della trasmissione.

### Sommario

Comunicazioni verbali e visive - I computers e le comunicazioni - Sistemi telefonici - Terminali dei circuiti e modi di funzionamento - Segnali convenzionali di comunicazione - Metodi e tecniche di modulazione - Sistemi per portanti fondamentali - Caratteristiche fondamentali di una linea di trasmissione - Il decibel, un rapporto di potenza - Panoramica sui problemi di trasmissione - Elementi correttivi nei circuiti telefonici - Specifiche dei circuiti - Modems nella trasmissione dei dati - Esame finale del corso di elementi di trasmissione dei dati - Dati di riferimento - Glossario di termini per comunicazioni EDP - Risposte ai quesiti.

Pagg. 178 Formato 15 x 21  
Prezzo L. 10.500 Codice 316D

## IL LIBRO DEI PRINCIPIANTI

Introduzione ai microcomputer Vol. 0

Il libro dà una visione d'insieme su calcolatori ed elaboratori, fornendone nel contempo tutti i concetti generali e la terminologia di base per capire la tecnologia usata. Vengono illustrate anche le singole parti che costituiscono il sistema con le possibilità di espansione e componenti accessori.

### Sommario

Le parti che costituiscono il tutto - Usare un microcomputer e guardarlo crescere - Componenti dei sistemi a microcomputer, quello che si vede non è sempre quello che si ottiene - Gettando le basi - Dentro il computer - Mettiamo assieme il tutto

Pagg. 240 Formato 13,5 x 20,5  
Prezzo L. 16.000 Codice 304A

## I MICROPROCESSORI

Dai chip ai sistemi

Descrivere l'architettura di un sistema microprocessore, le funzioni richieste per allestirlo, i componenti e le loro interconnessioni. Presenta le caratteristiche che qualificano ciascun prodotto, ne analizza vantaggi e svantaggi, fornisce i criteri di valutazione.

### Sommario

Concetti fondamentali - Funzionamento interno di un microprocessore - Componenti del sistema - Valutazione comparativa tra microprocessori - Interconnessioni per la costruzione di un sistema - Applicazioni del microprocessore - Tecniche di interfacciamento - Programmazione di microprocessori - Sviluppo del sistema - Il futuro - Simboli elettronici - Set di istruzioni per il Motorola 6800 - Set di istruzioni per l'Intel 8080-Bus S-100 - Costruttori - Abbreviazioni.

Pagg. 384 Formato 14,5 x 21  
Prezzo L. 25.000 Codice 320P

## INTRODUZIONE AL PERSONAL E BUSINESS COMPUTING

Il testo è stato scritto per il lettore che non conoscendo nulla dei computer vuole addentrarsi in questo mondo affascinante per diventare in un secondo tempo, lui stesso utente. In modo pratico e progressivo, comunque, sono presentati tutti gli elementi di un sistema finanche i metodi di valutazione per una scelta oculata

### Sommario

L'era del microcomputer - Impiego del sistema - Definizioni di base - Come funziona - La programmazione - BASIC e APL - Business Computing - Scegliere un sistema - Le periferiche - Scegliere un microcomputer - Economia di un sistema commerciale - Come fallire con un sistema commerciale - Aiuto - Domani - Logica dei computer - Bits e Bytes - Sistemi di trasmissione base del Computer - Files e records - Alcuni costruttori di piccoli sistemi commerciali - Costruttori di microcomputer.

Pagg. 224 Formato 14 x 21  
Prezzo L. 14.000 Codice 303D



## PRACTICAL MICROPROCESSORS

Hardware, software e ricerca guasti

Primo manuale essenzialmente pratico, in lingua italiana, insegna tutto sui microprocessori: dall'hardware di un sistema, a microprocessore, al software che viene utilizzato per controllare il sistema, a come utilizzare queste informazioni per apprendere le tecniche pratiche, applicabili a qualunque sistema digitale, di ricerca guasti.

### Sommario

Introduzione ai sistemi a microprocessore - Sistemi di Microprocessore Lab - Alcuni concetti di software - All'interno del microprocessore - Concetti fondamentali di hardware - Decodifica degli indirizzi - Memorie periferiche - Circuiti di controllo.

Pagg. 454 Formato 21,5 x 28  
Prezzo L. 35.000 Codice 308B

## PRINCIPI E TECNICHE DI ELABORAZIONE DATI

È una trattazione chiara e conscia dei principi base della numerazione - Elementi di software - Uso del flusso e della gestione dei dati in un sistema di elaborazione elettronica, concepita per l'auto-apprendimento degli argomenti trattati, mediante test ed esercizi da svolgere.

### Sommario

Fondamenti di elaborazione elettronica di dati - Elementi funzionali di base - Sistema di numerazione e codifica dei dati - Manipolazione dei dati - Sistemi di memoria - Criteri operativi relativi al programma, al controllo ed all'elaboratore - Alcuni concetti sui sistemi di elaborazione - Concetti relativi ai sistemi terminali - Test finale - Risposte ai test di riepilogo - Risposte ai test finale.

Pagg. 254 Formato 14,5 x 21  
Prezzo L. 17.000 Codice 309A

## IL LIBRO DEI CONCETTI FONDAMENTALI

Introduzione ai microcomputer Vol. 1

Volume ormai "storico" presenta la struttura logica fondamentale su cui sono basati i sistemi a microcomputer in modo tale che il lettore può imparare a valutare l'applicabilità o meno, del microcomputer ad ogni problema pratico. Il libro sviluppa un quadro dettagliato dall'architettura alla programmazione, di cosa un microcomputer sa fare, come opera, dove si presta ad essere utilizzato.

### Sommario

Che cos'è un microcomputer - Alcuni concetti fondamentali - Come si realizza un microcomputer - L'unità centrale del microcomputer - Logica addizionale della CPU - Programmazione del microcomputer - Un set di istruzioni - Codice caratteristiche standard.

Pagg. 321 Formato 15 x 21  
Prezzo L. 18.000 Codice 305A

## TECNICHE DI INTERFACCIAMENTO DEI MICROPROCESSORI

Questo libro indica i concetti, le tecniche di base, i componenti per assemblare un sistema completo a partire dalla fondamentale unità centrale di elaborazione, per arrivare, ad un sistema equipaggiato con tutte le periferiche comunemente usate.

### Sommario

Tecniche di implementazione dell'unità di elaborazione (CPU) - Fondamenti di trasferimento dati su interfaccia (I/O) - Interfacciamento delle periferiche - Circuitiera analogica - Conversione analogica/digitale (A/D e digitale/analogica D/A) - Standard di interfaccia (BUS) - Studio di un caso: moltiplicatore a 32 canali - Errata funzionalità digitale - Conclusioni - Evoluzioni.

Pagg. 400 Formato 15 x 21  
Prezzo L. 25.000 Codice 314P



**GRUPPO EDITORIALE JACKSON**  
**Divisione Libri**

Per ordinare il volume utilizzare l'apposito tagliando inserito in fondo alla rivista.

# CONVERSIONI

- sostituzione di ogni STRING\$(K," ") con SPC(K).

Ho poi ulteriormente modificato il programma per poter stampare il calendario su quattro colonne con una stampante da 132 caratteri e con la possibilità di inserire un breve messaggio augurale.

```

110 DIM M1$(42),M2$(42),M3$(42),M4$(42),L(12),C$(31),M$(12)
121 DEF FNF(Y)=365*Y+INT((Y-1)/4)-INT(3*INT((Y-1)/100)+1)/4)
142 D$=" LU MA ME GI VE SA DO "
145 FOR I=1 TO 12
150 READ L(I),M$(I)
155 NEXT I
160 FOR I=1 TO 31
165 READ C$(I)
170 NEXT I
175 DATA 31," GENNAIO ",28,"FEBBRAIO ",31," MARZO "
180 DATA 30," APRILE ",31," MAGGIO ",30," GIUGNO "
185 DATA 31," LUGLIO ",31," AGOSTO ",30,"SETTEMBRE "
190 DATA 31," OTTOBRE ",30,"NOVEMBRE ",31,"DICEMBRE "
195 DATA " 1"," 2"," 3"," 4"," 5"," 6"," 7"," 8"
200 DATA " 9"," 10"," 11"," 12"," 13"," 14"," 15"," 16"
210 DATA " 17"," 18"," 19"," 20"," 21"," 22"," 23"," 24"
220 DATA " 25"," 26"," 27"," 28"," 29"," 30"," 31"
230 PRINT""
240 PRINT:PRINT
250 PRINTTAB(11):"*PROGRAMMA JULIA*":PRINT
260 PRINT"QUESTO PROGRAMMA STAMPA UN CALENDARIO":PRINT
270 PRINT"GIULIANO PER OGNI ANNO SPECIFICATO ":PRINT
275 PRINT"POSTERIORE AL 1583"
310 PRINT:PRINT
320 INPUT "PER FAVORE SCRIVERE ANNO":Y
360 IF Y<100 THEN Y=Y+1900
370 IF Y<1583 GOTO 260
372 GOSUB5000
375 OPENS.5:CMD5
376 FORSP=1T06:PRINT:NEXTSP
378 IFGR<>" " THENPRINTTAB(LU)GR$:GOTO382
380 PRINT
382 FORSP=1T07:PRINT:NEXTSP
384 PRINTTAB(61):"*****"
385 PRINTTAB(61):" * ";Y:" *"
387 PRINTTAB(61):"*****"
390 F=FNF(Y)
395 V1=F:V2=7:GOSUB4000
410 D=H
470 IF D=0 THEN500
480 C1=D
490 GOTO505
500 C1=7
505 V1=Y:V2=4:GOSUB4000
510 IFH=0THEN540
520 L(2)=28
530 GOTO550
540 L(2)=29
550 FORB1=1T012STEP4
560 GOSUB3000
570 L1=L(B1)
590 FORI=1TOL1
600 M1$(C1)=C$(I)
610 C1=C1+1
620 NEXT I
640 L2=L(B1+1)
645 V1=C1:V2=7:GOSUB4000
650 C2=H
670 IFC2=0THENC2=7
680 FORI=1TOL2
690 M2$(C2)=C$(I)
700 C2=C2+1
710 NEXT I
730 L3=L(B1+2)
735 V1=C2:V2=7:GOSUB4000
745 C3=H
750 IFC3=0THENC3=7
760 FORI=1TOL3
780 M3$(C3)=C$(I)
790 C3=C3+1
800 NEXT I
801 L4=L(B1+3)
803 V1=C3:V2=7:GOSUB4000
805 C4=H
810 IFC4=0THENC4=7
820 FORI=1TOL4
830 M4$(C4)=C$(I)
840 C4=C4+1
850 NEXT I
860 V1=C4:V2=7:GOSUB4000
870 C1=H
880 IFC1=0THENC1=7
900 PRINT:PRINT

```

```

905 PRINTSPC(16);M$(B1);SPC(22);M$(B1+1);SPC(21);
M$(B1+2);SPC(21);M$(B1+3)
910 PRINT
915 PRINT" "ID$:" "ID$:" "
D$:" "ID$
920 PRINT
925 FORI=1T042STEP7
930 PRINT" "
935 FORJ=1T07
940 PRINTM1$(I+J-1)
945 NEXTJ
950 PRINT" "
955 FORJ=1T07
960 PRINTM2$(I+J-1)
965 NEXTJ
970 PRINT" "
972 FORJ=1T07
974 PRINTM3$(I+J-1)
976 NEXTJ
978 PRINT" "
980 FORJ=1T06
982 PRINTM4$(I+J-1)
984 NEXTJ
986 PRINTM4$(I+6)
988 NEXTI
990 NEXTB1
1000 FORSP=1T011:PRINT:NEXTSP
1005 PRINT#5:CLOSE5
1010 PRINT
1020 INPUT"VUOI UN ALTRO CALENDARIO (SI-NO)?:A$
1040 IF A$="SI" THEN310
1050 STOP
3000 FORI=1T042
3010 M1$(I)=" "
3020 M2$(I)=" "
3030 M3$(I)=" "
3035 M4$(I)=" "
3040 NEXTI
3050 RETURN
4000 REM SUB. SOSTITUZIONE DI FNF
4010 H=V1-INT(V1/V2)*V2
4020 RETURN
5000 REM OPZIONE INPUT MESSAGGIO
5010 PRINT
5012 PRINT"1- VUOI SCRIVERE UN MESSAGGIO ?"
5015 PRINT
5020 PRINT"2- VUOI RIPETERE QUELLO PRECEDENTE?"
5025 PRINT
5030 PRINT"3- NON VUOI SCRIVERE NIENTE ?"
5035 PRINT
5040 PRINTTAB(11)" (PREMERE 1,2 O 3) ":
5050 INPUT R$
5060 IFR$="1" THENGOSUB6000:RETURN
5065 IFR$="2" THENRETURN
5070 IFR$="3" THENGOSUB6050:RETURN
5080 PRINT"":GOTO510
6000 REM INPUT MESSAGGIO
6010 FORI=1T010:GETX$:NEXTI:GOSUB6050
6015 PRINT:PRINT"BATTI IL MESSAGGIO(MAX.79 CARATTERI)"
6020 INPUT GR$
6025 LU=INT((132-LEN(GR$))/2)
6030 RETURN
6050 REM AZZERAMENTO VARIABILI
6060 GR$="":LU=0
6070 RETURN

```

Listato 2. Il programma con le ulteriori modifiche. ■

## DEBUG

### Come realizzare un listato bidirezionale

*Molti lettori ci hanno informato della presenza di un errore sulla routine del listato bidirezionale per il VIC 20 pubblicata su PS4. Ecco la lettera del signor Ercole Palmeri di Torino.*

Scrivo per informarvi della presenza di un errore nel programma "Come realizzare un listato bidirezionale" per il VIC 20, presente nel numero 4 della vostra rivista.

Leggendo il programma sono arrivato alle seguenti linee:

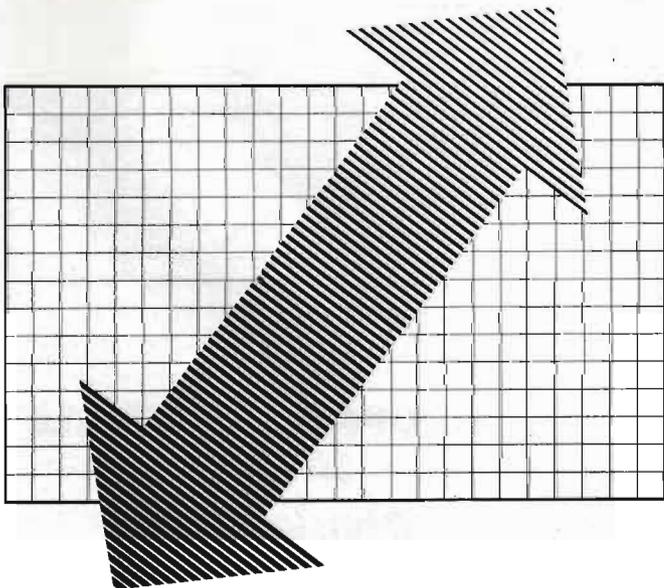
```
63100 IF PEEK(SA+5)<>0 THEN SA=SA+1
63105 GOTO 63100
```

mi sono accorto che il programma va in loop.

La correzione è semplice: basta cancellare la linea 63105 e sostituire la 63100 con:

```
63100 IF PEEK(SA+5)<>0 THEN SA=SA+1:
GOTO 63100
```

Il programma è molto utile e ben fatto. Saluti e complimenti per la vostra eccezionale rivista! ■



## COMPUTER CLUB TI 99



### 200

 programmi disponibili gratuitamente

- convenzioni agevolate per l'acquisto del tuo home computer
- aiuto all'utilizzo dell'home computer e tanti altri vantaggi che scoprirai associandoti

#### RIVENDITORI CONVENZIONATI

COMPUTERWORLD - Tel. 06/460818  
Via del Traforo, 137 - 00100 ROMA  
ESSEMMECI - Tel. 0746/44704  
Via delle Orchidee, 19 - 02100 RIETI  
COMPUDATA - Tel. 02/545560  
Via Botta, 16 - 20135 MILANO  
MED - Tel. 0737/3329  
Via Venanzi, 11-13 - 62032 CAMERINO (MC)  
A TRE - Tel. 0424/25105  
Piazzale Firenze, 23  
36061 BASSANO DEL GRAPPA (VI)  
TECNINOVAS COMPUTER srl - EDP SHOP  
Via Emilia, 36 - 56100 PISA  
Tel. 050/502516  
COMPUTER CENTER - Tel. 010/300797  
Corso Gastaldi, 77R - 16131 GENOVA  
CENTRO DIFFUSIONE MICRO COMPUTER  
Via Trento, 42B - 27029 VIGEVANO (PV)  
MEV system - Tel. 0461/24886  
Via Grazioli, 59 - 38100 TRENTO  
LEUCI SISTEMI - Tel. 080/902582  
Via A. Figliera, 53  
74015 MARTINA FRANCA (TA)  
VISICOM computer - Tel. 0961/41673  
Via Menniti Ippolito, 10 - 88100 CATANZARO  
FRANCO - GIOCHI INTELLIGENTI  
Corso Fogazzaro, 174  
36100 VICENZA - Tel. 0444/42678  
SECA - Tel. 0883/44508  
Via Postumia, 21 - 70059 TRANI (BA)  
C.E.M.E. - Tel. 0963/44655  
Via della Pace, 1<sup>a</sup> Trav. 6  
88018 VIBO VALENTIA (CZ)  
COMPUTER SHOP - Tel. 095/441620  
Via V. E. Orlando, 164-166 - 95127 CATANIA  
IMPEL - Tel. 0522/43745  
Viale Isonzo, 11A - 42100 REGGIO EMILIA  
IMPEL - Tel. 059/225819  
Viale Emilia est, 16 - 41100 MODENA  
F.lli BRENNNA snc - Tel. 031/540096  
Via Giordano Bruno, 3 - 22100 COMO  
MASH COMPUTER SYSTEM - Tel. 0382/37300  
Via Strada Nuova, 86 - 27100 PAVIA

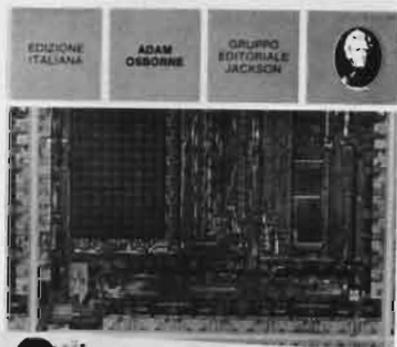
Entra anche tu a far parte  
della famiglia  
internazionale  
degli utenti di  
Home Computer TI

Computer Club TI 99  
Via delle Orchidee n. 19  
Tel. 0746/44704-5  
02100 RIETI

TI-99/4A  
Nome e cognome  Sono interessato a  
Via  «Computer Club TI 99»  
Città \_\_\_\_\_ cap. \_\_\_\_\_  
Telefono \_\_\_\_\_  
Ritagliare e spedire a  
«Computer Club TI 99»  
02100 RIETI - Via delle Orchidee n. 19  
Tel.: 0746/44705

# 8080 PROGRAMMAZIONE E PROGETTAZIONE LOGICA

Programmazione  
dell'8080  
e progettazione logica



Il libro descrive l'implementazione della logica sequenziale e combinatoria con l'uso del linguaggio assembly all'interno di un sistema a micro-computer basato sull'8080.

Lo scopo è quello di insegnare ai progettisti logici come eseguire in modo nuovo un vecchio lavoro mediante la creazione di programmi e ai programmatori come la programmazione abbia trovato uno scopo nuovo nel progetto logico.

I concetti tradizionali di programmazione in linguaggio assembly non sono utili né attinenti per usare i microprocessori in applicazioni logiche digitali: l'uso delle istruzioni in linguaggio assembly per simulare il packages digitale è in tutti i casi errato.

Il libro chiarifica questi concetti per prima cosa simulando sequenze logiche digitali, poi illustrando alcune efficienti soluzioni per spiegare l'uso corretto dei microcomputer. Un capitolo, infine, contiene il set completo di istruzioni dell'8080.

Sommario

Introduzione - Linguaggio assembly e logica digitale - Una simulazione diretta della logica digitale - Un semplice programma - Prospettiva del programmatore - Set di istruzioni - Alcune subroutine impiegate comunemente - Codici di caratteri ASCII.

Pagg. 296  
Prezzo L. 19.000

Formato 14,5 x 21  
Codice 325P

Per ordinare il volume  
utilizzare l'apposito tagliando  
inserito in fondo alla rivista.



**GRUPPO EDITORIALE  
JACKSON**  
Divisione Libri

## CONTRIBUTI DEI LETTORI

*Il signor Giorgio Cortelazzo di Padova aggiunge queste precisazioni sul controllo del codice fiscale.*

Nell'articolo sul controllo del codice fiscale si afferma che alcuni caratteri sono numeri. In realtà, "normalmente" sono numerici. Infatti bisogna tener conto dei codici fiscali omonimi.

Ad esempio io e il signor Cortese Giorgio, nato a Padova il mio stesso giorno, avremmo come codice fiscale CRTGRG58C30G224E.

Per evitare questa ambiguità, la legge prevede che vengano sostituiti ai caratteri numerici dei caratteri alfabetici.

Se ricordo bene le sostituzioni vanno effettuate da destra utilizzando la tabella seguente:

0	L	5	R
1	M	6	S
2	N	7	T
3	P	8	U
4	Q	9	V

Non conosco esattamente tuttavia il numero di caratteri che vanno sostituiti: mi pare che dovrebbero essere mantenute le sostituzioni fino a rendere i codici diversi.

Ad esempio:

CRTGRG58C30G2NP + carattere di controllo  
e non

CRTGRG58C30G2N4 + carattere di controllo.

Se così non fosse bisognerebbe controllare che, dopo il primo carattere non numerico che sostituisce un carattere numerico, tutti i successivi siano per forza alfabetici.

Da notare che queste sostituzioni non alterano il carattere di controllo e permettono sempre una semplice decodifica del codice (ad es. G22EE... G224E) univocamente. ■

# Per 'lavorare' al meglio con il Pet e l'M20

Paolo e Carlo Pascolo

## IL BASIC DEL PET E DELL'M20

Il personal computer rappresenta oggi, oltre che un valido aiuto nel lavoro, anche un'irresistibile tentazione. Può capitare, così, che qualcuno si trovi a disporre di un Commodore o di un M 20 Olivetti senza conoscerne appieno il linguaggio e le possibilità. Questo volume vuol rappresentare proprio un prezioso supporto per chi debba, o voglia imparare a programmare in Basic su questi strumenti di lavoro, gioco o studio: comandi, istruzioni, informazioni, consigli... fino a diventare davvero 'padroni' di due dei più diffusi Personal Computer.

226 pagine. Lire 16.000  
Codice 336 D

Per ordinare il volume  
utilizzare l'apposito tagliando  
inserito in fondo alla rivista



GRUPPO EDITORIALE  
JACKSON



## Utilizzare le routine della cartridge del VIC

*Il signor Moncecchi Giovanni di Tresenda (SO) ci invia questa esperienza. In fondo alla sua lettera pubblichiamo il nostro commento.*

Quando inseriamo qualche cartridge-gioco nel connettore del VIC e poi lo accendiamo, avremo il nostro computer trasformato in una consolle videogame. Questo succede perché la routine di inizializzazione del sistema operativo verifica se all'indirizzo 40960 è presente un programma; in caso affermativo passa ad eseguire il programma indirizzato dai primi due byte di questa memoria.

Se però inseriamo la ROM a VIC acceso, il sistema operativo non è più in grado di verificarne la presenza e quindi possiamo utilizzare le subroutine presenti tramite il funzionamento in Basic. In particolare voglio presentare le possibilità che ci offre il cartridge VIC-1908 cioè la ROM con il gioco del poker, che negli esempi verrà sfruttata per creare e disegnare le carte da gioco, così che possiamo disegnare una qualsivoglia carta in un punto stabilito dello schermo.

Vi è però il problema che riguarda l'inserimento del cartridge che deve essere effettuato a VIC acceso e che a volte può bloccare il controllo tramite tastiera; se ciò avviene il cursore smette di lampeggiare e in questo caso non rimane che ripetere l'operazione.

Una volta ottenuto il funzionamento normale con cartridge inserito dobbiamo spostare il puntatore Basic per far posto al nuovo generatore di caratteri:

POKE 44,24:

POKE PEEK(44)\*256+PEEK(43)-1,0: NEW

La seguente istruzione creerà il nuovo generatore di caratteri, riposizionerà il puntatore relativo e posizionerà lo schermo in maniera corretta:

SYS 40975: POKE 36864,12

Ora abbiamo a disposizione il nuovo set grafico, che non crea problemi in quanto ha sempre i numeri ed i caratteri maiuscoli, ma che possiamo controllare provando a stampare i caratteri semigrafici con i tasti SHIFT e Commodore. Ora carichiamo questo programma che ci memorizza una piccola subroutine a partire dalla locazione 830:

Il numero che stabilisce la carta, cioè quello che bisogna mettere nella locazione 1002 è abbastanza semplice da capire se trasformato in binario come nell'esempio:

<i>bit più significativi</i>	<i>bit meno significativi</i>
0000 Picche	0000 nullo
0001 Cuori	0001 asso
0010 Quadri	0010 due
0011 Fiori	0011 tre
	0100 quattro
	0101 cinque
	0110 sei
	0111 sette
	1000 otto
	1001 nove
	1010 dieci
	1011 fante
	1100 regina
	1101 re
	1110 nullo
	1111 nullo

Se abbiamo il numero 18 che in binario è 00010010  
 corrisponde a      0001      0010      due di cuori  
                               cuori      due

Figura 1.



# CONTRIBUTI DEI LETTORI

10 FOR V=830 TO 849  
 20 READ N: POKE V,N  
 30 NEXT  
 100 DATA 173,232,3,133,97,173,233,3,133,98  
 110 DATA 173,234,3,141,12,3,32,171,160,96

Dopo aver fatto girare questo programma, caricando i seguenti puntatori potremo disegnare la carta voluta nella posizione desiderata:

- POKE 1000, (tra 0 e 18) Posiziona la carta in orizzontale
- POKE 1001, (tra 0 e 16) Posiziona la carta in verticale
- POKE 1002, (figura 1) Decide la carta

Per economizzare lo spazio di memoria disponibile possiamo cancellare il precedente programma in quanto la routine in linguaggio macchina rimarrà disponibile al richiamo SYS 830.

Questo programma ci stampa sul video un mazzo di 52 carte:

```
10 FOR G=0 TO 16 STEP 8
20 FOR T=0 TO 18
30 D=D+1
40 A=D AND 15: IF A>13 THEN D=D+3:
   IF D>63 THEN END
50 POKE 1000,T: POKE 1001,G: POKE 1002,D:
   SYS 830
60 NEXT
70 NEXT
```

Ora che abbiamo le carte possiamo giocare o no? Dimenticavo di dire che le carte vengono stampate con il colore corrispondente.

*L'idea è molto buona, meno il sistema usato: inserire la cartuccia a VIC acceso. Poiché quasi tutti i circuiti integrati sono del tipo MOS, questo può comportare facilmente la rovina della cartuccia o peggio del VIC stesso. I problemi nascono infatti dal fatto che se una espansione, generalmente di giochi, è inserita nello slot, questa all'accensione intercetta la routine di COLD START, sostituendone una propria che non restituisce più il controllo al Basic. Questo però avviene solo se la cartuccia presenta la sequenza di caratteri A0CBM a partire dal byte \$A00A. La soluzione perciò sarebbe non quella di inserire la cartuccia col VIC acceso, ma di spostare in un'altra zona di memoria l'allocazione della ROM-gioco, servendosi dei ponticelli presenti nel circuito stampato.*

In questo modo l'idea ci sembra interessante e soprattutto applicabile senza rischi per il computer. ■

## CEDOLA DI COMMISSIONE LIBRARIA

Da inviare a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Nome Cognome

Indirizzo

Cap.  Città  Provincia

Partita I.V.A. (indispensabile per le aziende)

Si richiede l'emissione della fattura

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione:

- Allego assegno n° ..... di L. ....
- Non abbonato  Abbonato sconto 20%  Elettronica Oggi  Automazione Oggi  Elektor  Informatica Oggi  Computerworld  Bit  Personal Software  Strumenti Musicali  Videogiochi

Data .....

Firma .....

## CEDOLA DI COMMISSIONE LIBRARIA

Da inviare a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Nome Cognome

Indirizzo

Cap.  Città  Provincia

Partita I.V.A. (indispensabile per le aziende)

Si richiede l'emissione della fattura

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione:

- Allego assegno n° ..... di L. ....
- Non abbonato  Abbonato sconto 20%  Elettronica Oggi  Automazione Oggi  Elektor  Informatica Oggi  Computerworld  Bit  Personal Software  Strumenti Musicali  Videogiochi

Data .....

Firma .....

Vendo CBM 4032 con floppy 2031, registratore e sonoro usati pochissimo. Con programmi vari e giochi tra cui la ricezione visiva dei giornali commerciali. Telefonare ore serali a Alessandro Consonni, Via Cà Grandà 18, 20162 Milano, tel. 02/6438566.

Vendo o scambio programmi per Commodore GL. Telefonare ore ufficio o scrivere a Claudio Gambiè, Via Fatebenefratelli 26, 20075 Lodi, tel. 0371/52402.

Vendo VIC 20 + cabinet espansioni VIC 1020 + 3K super expander + 8 K RAM + 6 cartidges di giochi: Alien; Road Race; Adventureland; Jelly Monsters; Star Battle; Sargon II a L. 800.000 trattabili. Ugo Merlone, P.za Galimberti 1, 10131 Torino, tel. 011/364756.

Compro per VIC-20 cabinet di espansione TC12055-00 solo vera occasione. Scrivere con offerta a Lauro Michelotti, Via Boboli 1a, 51017 Pescia (Pt).

Software VIC 20. Vantaggiosi prezzi vendo vasto assortimento. Richiedere documentazione e listino prezzi. Assicuro massima serietà. Alberto Locatelli, Via Sardegna 31, 20146 Milano, tel. 02/496576.

Vendo VIC 20 6 K RAM + Joystick + Superexpander con 3 K RAM + Programmer AID + Vicgraf + CN2 + Cabinet espansione 6 schede + 2 cartucce giochi + testi vari o permutato tutti gli accessori con cartucce giochi. Opipari Giuseppe, Via Saint Bon 16, 20100 Milano, tel. 4121196.

Vendo VIC 20 + registratore + 3 Kramsopere + 16 K RAM + Joystick + 4 giochi su cartuccia e 10 giochi su nastri + manuali e testi vari per imparare a programmare. Il tutto nuovo, a sole L. 1.250.000 (oltre 25% in meno del costo). Stefano Guarneri, Via Bembo 14, 26100 Cremona, tel. 0372/27902 ore pasti.

Vendo per VIC (3 K byte - 8 K byte) e PET 2001 (word processing) utilities varie, giochi in cassetta o disco. A richiesta si invia ampia documentazione. Si produce software anche su commissione. Carlo Zanini, Via Largo Paolo Sarpi 17, 26100 Cremona, tel. 27498.

Vendo VIC 20 a prezzo eccezionale di L. 450.000 incluso manuale in inglese ed italiano + trasformatore + 2 cartidges (Scacchi, Space Invaders), prezzo trattabile. Telefonare dopo le 20 a Daniel Palmacci, Via Stringher 31, 00191 Roma, tel. 06/3274976.

Per VIC 20, vendo-compro-scambio molto software solo per posta e su cassetta. Liste gratis a chi manda le sue, oppure a L. 1.000. Cerco 8 K a buon prezzo. Ho anche molti programmi in LM anche originali inglesi. Giorgio Ferrario, Via Adua 1, 21052 Busto Arsizio (Va).

Vendo prog. magazzino art. 1500 per CBM 64. Per informazioni Elisabetta Lovino, Via S. Saffi 7, 70037 Ruvo di Puglia (Bari), tel. 080/812832.

Vendo fantastici programmi in L.M. per VIC 20 in configurazione base o espanso fino ad 8 K. Il tutto a prezzi stracciatissimi!!! Per ricevere la lista inviare L. 500. Contatto solo in zona Milano. Rispondo a tutti. Massimiliano Bastoni, Via S. Sofia 8, 20122 Milano, tel. 806502.

Vendo VIC 20 completo alimentatore interfaccia video + unità cassette C2N + espansione 16 K + manuale italiano-inglese + introduzione al Basic + 2 software utility. Tutto garantito a L. 750.000 trattabili. Daniele Ghiselli, Via Enrico Toti 54, 51016 Montecatini T. (Pt), tel. 0572/71765 ore ufficio.

Vendo per VIC 20: Busi Board 7 slot a L. 100.000; luci rotanti a L. 40.000; Tool-kit a L. 30.000; Assembler a L. 30.000; Hi-Res a L. 45.000; programmatore di Eprom (2516-2716) a L. 30.000; tutte le cassette giochi a L. 10.000 l'una; Hi-Res con Basic in italiano a L. 55.000. Nicola Pedrolli, Via Vigevano 36, 20010 Bareggio (Mi), tel. 9013314.

Compro per PET-CBM 3032 scheda grafica VG 32. Cerco inoltre numero di gennaio 1982 (N° 24) di BIT. Telefonare ore pasti o scrivere a Giorgio Raffagli, Via Veneziano 5, 20139 Milano, tel. 5397853.

Acquisto espansioni 8/16 K per VIC 20, Cabinet o Memory Expansion Board, Stampante ad aghi. Solo se in buono/ottimo stato. Lorenzo Giacchello, Via Marchese della Rocca 4, 10074 Lanzo Torinese (To), tel. 0123/28621.

Vendo VIC 20; Cabinet espansione; cartucce 16 K; Programmeraid; Superexpander; Vicgraf; 2 cartucce giochi; libri e manuali vari. Valore acquisto 2 milioni, vendo a L. 1.200.000. Usato pochissimo. Opipari Giuseppe, Via Saint Bon 16, 20100 Milano, tel. 4121196.

Vendo (zona Palermo) programma per VIC 20 che riduce sistema integrale totocalcio composto di doppie e triple. Il programma accetta qualunque condizione. La prova si fa a casa dell'inserzionista. Bruno Cardella, Via Calabria 4, Lotto 43, 90100 Palermo, tel. 512302.

Vendo Commando per i mod. 4000 e 8000, su nastro L. 80.000, su disco 9000, su EPROM L. 100.000. Word processor con manuale in italiano per 3032 e 3032 modificato a 4032 su disco L. 150.000. Vendo inoltre programmi corretti e funzionanti apparsi su qualsiasi rivista del settore personal su nastro L. 15.000, su disco L. 20.000. Spese postali escluse. Massimo Rignoni Savio, Via A. Dia 47, 35031 Abano T. (Pt), tel. 049/811389.

Scambio programmi per VIC di vario genere e di capacità. Cercasi programma per l'anti-copy e l'anti-list. Inviare vostra lista dei programmi, io invierò la mia. Risponderò a tutti. Roberto Oselladore, Via Fausta, 136/A-5, 30010 Ca' Savio (Venezia), tel. 966923 da settembre.

Vendo/cambio programmi per VIC 20 a prezzi bassissimi. Alcuni esempi: Boss-Traxxamol Gridrunner Bonzo ecc. Richiedere liste gratuite. Giuseppe Mascali, Via R. Margherita 573, 98028 S. Teresa Riva (Me), tel. 791692.

Scambio software ingegneria civile per Commodore 3-4-8032 anche zone sismiche. Ing. Alvaro Albani, Via Castelfidardo 17, 47037 Rimini, tel. 0541/25765.

Vendo programma contabilità 16 K per ZX 81 a L. 20.000 + cartuccia Programmer's Aid per VIC 20 a L. 30.000. Antonio Riccardelli, Via Osoppo 5, 01100 Viterbo, tel. 0761/224410.

Cambio/vendo programmi VIC 20, traduzione integrale istruzioni d'uso cartucce VIC 1211 - VIC 1212, introduzione al Basic (parte I). Rispondo a tutti. Luigi De Negri, Via Puggia 22, 16131 Genova.

Vendo VIC 20 Commodore + registratore per lo stesso + manuale, ancora tutto imballato a L. 450.000. Telefonare solo se in Firenze o provincia. Alberto Cencetti, Via Delle Montalve 14, 50141 Firenze, tel. 450064.

Vendo causa regalo indesiderato VIC 20 + registratore + Joystick + manuali e cavi di connessione, il tutto nuovissimo a L. 450.000 intrattabili. Telefonare ore 20 e chiedere di Franco Bolli, Via Pascoli 4, 20031 Cesano Maderno, tel. 0362/501204.

Gruppo di utenti Commodore 64 cerca altri utilizzatori disposti a scambiare software ed esperienze. Luciano Cuneo, Via Emilio Lepido 46, 00175 Roma, tel. 06/7491542.

Cerco professori CMB 67 per scambio informazioni e programmi. Giochi (Gridrunner ecc.) e utilità (grafica). Osvaldo Stark, Piazza S. Vigilio 7, 39012 Merano (Bz).

Scambio programmi per VIC 20 e contatto persone interessate all'hardware, possesso libro "VIC 20 Interfacing BLBE Book". Lucio Rota, Via U. Levi 5, 42100 Reggio Emilia, tel. 0522/30155.

VIC 20 vendo oltre 100 giochi, programmi cartuccia grafica, per gestione casa, rubrica telefono e appuntamenti, utility (gratis per chi acquista) con espansione o senza. VIC Revealed in italiano. Pietro Scalzitti, Via Del Ridotto 9, 10147 Torino, tel. 011/250930.

Vendo-scambio software su nastro per VIC 20. Inviare liste per L. 1.200 o gratis se inviate vostre liste. Cerco cartucce VIC 20. Ho circa 250 programmi anche LM e originali. Vendo tutte le riviste Videogiochi. Giorgio Ferrario, Via Adua 1, 21052 Busto Arsizio (Va).

Per VIC 20 vendo alta risoluzione a L. 45.000, Bus Board 7 Slot a L. 100.000, programmatore di Eprom a L. 70.000, luci rotanti programmabili 14 vie a L. 60.000. Programma Eprom 2516. Nicola Pedrolli, Via Vigevano 36, 20010 Bareggio (Mi), tel. 9013314.

## QUANDO JACKSON NON È SOLO ELETTRONICA E INFORMATICA.

strumenti  
**MUSICALI**

**Strumenti Musicali** è un'altra rivista Jackson, ma con la stessa autorevolezza e professionalità delle riviste specializzate in elettronica e informatica.

È il mensile più letto dai professionisti della musica, dagli appassionati di strumenti musicali, tradizionali ed elettronici. Servizi speciali e test su singoli strumenti dai più diffusi ai più strani; articoli didattici, consigli pratici, analisi di mercato, interviste, novità e tante altre rubriche costituiscono il prezioso contenuto di **Strumenti Musicali**.

La rivista che è ormai un simbolo per tutti coloro che amano non solo la musica, ma i mezzi per studiarla, comporla, eseguirla. Giunta al quinto anno di vita, **Strumenti Musicali** è la rivista del Gruppo Editoriale Jackson con tirature da vero e proprio mass-media.

**Strumenti Musicali**: l'appuntamento mensile per chi esige professionalità, nell'editoria musicale.

strumenti  
**MUSICALI**



**GRUPPO EDITORIALE  
JACKSON**

Vendo per VIC 20 16 K RAM a L. 140.000; 8 K RAM a L. 85.000; Modulo espansione Arfom 7 Slot a L. 200.000; Programmer's Aid Pack a L. 35.000; Forth Cartridge a L. 75.000. A chi acquista tutto regalo Vic Reviled e Vicreference guide. Vittorio Godio, Via Roma 87, 47042 Cesenatico, tel. 0547/82036.

CMB 64 e VIC 20 vendo interessantissimi programmi. Assicuro una risposta a tutti. Chiedere la lista allegando L. 300 in francobolli. Claudio Marchiondelli, Via Libertà 3, 33010 Raspano (Ud), tel. 0432/852343.

Vendo per VIC 20 giochi su cartuccia; Jupiter Lander e Star Battle e programmi su nastro di cui alcuni originali americani in linguaggio macchina. Telefonare dopo le ore 17.30 a Paolo Lambri, Via Alfieri 60, 20099 Sesto San Giovanni, tel. 2421130.

Vendo CBM-64 nuovissimo + Joystick + interfaccia registratore memoria 64 K + varie capacità grafiche e sonore a L. 750.000 (risparmio su prezzo listino = L. 300.000). Telefonare ore pasti ad Alessandro Fogar, Via Venezia 26, 34073 Grado (Go), tel. 0481/768655.

Cerco possessori di VIC 20 per cambio di programmi. Cerco programma di listino prezzi codice fornitori e con possibilità di registrazione di fatture. Il tutto per un piccolo negozio (200 articoli). Roberto Oselladore, Via Fausta 136/a int. 5, 30010 Ca' Savio (Ve), tel. 966923.

Vendo VIC 20 in perfette condizioni completo di manuali ed imballaggio originali a L. 450.000. Telefonare ore serali a Tarcisio Colawiz, Via Vitt. Veneto 54, 39100 Bolzano, tel. 0471/43179.

Vendo per VIC 20 espansione video 40 colonne con 32 K RAM memoria compatibile Videotex, mantiene colori e grafica VIC con ulteriori possibilità. L. 500.000 intrattabili. Stampante VIC 1515 a L. 500.000. Aldo Albergucci, Via Brigata Marche 11, 31015 Conegliano (Tv), tel. 0438/23512.

VIC 20 software possiedi un VIC e sei un principiante? Mandami il tuo indirizzo e ti invierò una cassetta in contrassegno a L. 15.000 contenente 35 programmi di giochi utili e dimostrativi. Grassi G. Carlo, Via Vasto 81, 46044 Goito (Mn), tel. 0376/607239.

Cerco nuovi utenti VIC 20 per formazione gruppo acquisto programmi in società e scambio consigli. Gennaro Galante, Via Caravaggio 65, 80043 Agropoli (Sa).

Cambio software per VIC 20. Contatto "Vichinghi" per creazione Vic-club. Scrivere ad Amedeo Fasano, Res. Sagittario MI 2, 20090 Segrate (MI).

Vendo PET 2001 8 K byte unità a cassette e video incorporati in perfetto stato con manuali e cassette giochi a L. 650.000. Massimo Caracciolo, Via Monsignor Pini 54, 00125 Roma, tel. 06/6055696.

Vendo programmi per VIC 20 su cassetta. Richiedere elenco allegando francobollo o telefonare ore pasti. Alessandro Commisso, Via Montebianco 12, 20090 Cesano Boscone (Mi), tel. 02/4500698.

Vendo stampante VIC 1515 nuovissima, completa di imballaggio e libretto di istruzioni a L. 400.000. Telefonare ore pasti (solo zona di Milano) a Alessandro Commisso, Via Monte Bianco 12, 20090 Cesano Boscone (Mi), tel. 4500698.

VIC 20 software scambio o vendo programmi per VIC 20 + espansioni. Inviatemi il vostro indirizzo oppure le vostre liste, invierò le mie. Giancarlo Grassi, Via Vasto 81, 46044 Goito (Mantova), tel. 0376/607239.

Vendo listati per VIC 20 a L. 2.000 ogni 5 acquistati uno in regalo. Progr.: Star War, grafica, matematica, battaglia navale, rinocerente, VIC medico, NIM, orologio, tastiera musicale, briscola, tiro al drago. Massimo Gusso, Viale Felissent 32, 31020 Lance-nigo (Tv), tel. 62969.

Vendo VIC 20 + unità cassette C2N + 2 manuali + 50 programmi + cartuccia scacchi a L. 600.000 trattabili. Telefonare a Mosca Valentino, Via Cassia 1216 00189 Roma, tel. 06/3765394.

Cerco programmi di ogni genere listati o cassetta per Commodore 64. Fabrizio o Gianluca Verona, Via Lorenzo Rocci 14, Roma, tel. 06/5376146.

Vendo programmi per VIC 20 su cassetta. Massima serietà. Per informazioni e lista programmi scrivere inviando L. 350 per spese postali a Ivan Gaboli, Via XXIII Marzo 228/A, 28100 Novara, tel. 0321/400451.

Scambio software per VIC 20. Biblioteca di oltre 500 programmi. Astenersi per tempo e non interessati. Amos Aimi, Via Zanella 11, 43015 Nocetto (Pr).

Vendo VIC 20 + registratore 2CN + superespansione (3 K + grafica) + cartridge gioco poker + manuali e tutto il software fatto su cassetta a L. 600.000. Stefano Albanesi, Via Leopardi 10, 20123 Milano, tel. 8055804.

Vendo VIC 20 + interfaccia registratore + 2 manuali in italiano (manuale d'uso e impariamo a programmare), 6 mesi di vita, costo ufficiale (L. 580.000) L. 400.000 intrattabili. Emilio Ziliani, Via Toscana 19, 20061 Carugate (Mi), tel. 02/9251429.

Vendo VIC 20 nuovo con registratore C2N a L. 490.000. Turi Dina, Via S. Polo 1811, 30100 Venezia, tel. 041/38584.

Vendo per passaggio a sistema superiore PET 3001-8c con monitoria fosfori verdi in perfette condizioni come nuovo a L. 700.000 inoltre stampante CBM 3022 usata solo per una prova a lire 700.000. Gianni Pavan, Via Arsa 13, 30172 Mestre, tel. 041/911367.

Grafici ad Alta Risoluzione con il Commodore PET: trasformate la vostra stampante 4022, in un plotter! Con questa routine potrete creare grafici, tracciare linee, scrivere commenti, il tutto con una risoluzione di 0.35 mm. I grafici possono essere registrati su disco o su nastro per poter essere in seguito riutilizzati, o corretti. Tutto quello che serve è 32K di memoria, e la stampante 4022. La routine è disponibile su cassetta corredata di chiare istruzioni per l'utilizzo, al prezzo di L. 50.000. Scrivere o telefonare a Massimo Rossi, Corso Porta Romana 121 Milano, tel. 590727 (al mattino o dopo ore 21.30).

Vendo programmi di ingegneria civile per PET/CBM. Telefonare negli orari di ufficio a Gianluca Luoni, Via Della Concordia 8/bis, 21050 Busto Arsizio (Va), tel. 0331/679128.

Per VIC 20 vendo espansioni 16 K a L. 150.000, Superexpander a L. 50.000, praticamente nuove. Telefonare ore pasti a Ugo Cartia, Via Monte Cervialto 80, 00139 Roma, tel. 06/8186838.

Vendoper VIC 20 Super Expander a L. 50.000; Programmer's Aid a L. 25.000; 8 K RAM a L. 40.000; Vic Forth a L. 50.000; Giochi: Avancer, Sargonii Chess, Omega Race, Alien a L. 20.000 l'uno solo zona Torino. Claudio Coccato, Via Pacinotti 9, 10036 Settimo Torinese (To), tel. 8002485.

Scambio software CBM 3032/4032 per ingegneria civile, speciale zona sismica, POR, muri di sostegno, telaio sismico (con inviluppo delle sollecitazioni e calcolo armature) e tanti ancora. Ing. Alvaro Albani, Via Castelfidardo 7, 47037 Rimini (Fo), tel. 0541/25765.

Vendo VIC + 5 K Hi-Res + 16 K RAM + registratore C2N + manuali italiano inglese 50 programmi vari su cassette + copertine coprisistemi. Tutto usato circa 10 ore a L. 800.000 (possibilità di dilazioni). Giuseppe Di Pasqua, Via E. Donadoni 1, 56100 Pisa, tel. 050/574605.

## Sharp

Sharp PC 1211: cerco programmi di termotecnica, "373", impianti condizionamento, riscaldamento aria calda, fabbisogni termici. Rispondo a tutti. Dino Fornaciari, Villaggio Dante 30, 52100 Arezzo, tel. 0575/351451 ore pasti.

Vendo/cambio programmi Sharp HZ80B (ingegneria civile 1 e 2, word processing, giochi, pronostici totocalcio). Scrivere per la lista o per ulteriori informazioni. Annuncio sempre valido. Stefano Lazzaro, Via Monte Sabotino 2, 35100 Padova, tel. 049/22675.

Vendo per Sharp MZ80 a routine Assembler per l'esecuzione di somme e sottrazioni algebriche con 15 cifre più segno. Consentono un notevole aumento di velocità nell'esecuzione dei programmi. Piergiorgio Cresto, Via Circonvallazione 27/3, 10018 Pavone, tel. 51480.

Vendo programmi di ingegneria per Sharp HZ80B a lire 50.000 cadauno. Dispongo inoltre di vario software. Per informazioni scrivere all'indirizzo seguente, inviando L. 500 per contributo spese spedizione. Stefano Lazzaro, Via Monte Sabotino 2, 35100 Padova.

Cambio programmi per Sharp MZ80B. Possiedo giochi: Star Trek, Missione aerea, Gioco delle rane, Prova riflessi, Totocalcio, Gioco del NIM ecc. Inviatemi i vostri listati o cassette ed io vi manderò i miei. Emanuele Matarazzo, Via Castello 10, 75100 Matera, tel. 0835/211626.

## PICCOLI ANNUNCI PERSONAL SOFTWARE

Sel un lettore di PERSONAL-SOFTWARE e vuoi entrare in contatto con tutti gli altri lettori per comperare, cambiare o vendere software? Spedisci questo tagliando a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Nome..... Cognome.....  
Via..... C.A.P.....  
Città..... Tel.....

# Servizio programmi

Per alcuni dei programmi pubblicati, *Personal Software* mette a disposizione dischi e nastri già registrati, realizzati in collaborazione con l'autore. Potete ottenerli in contrassegno, pagando direttamente al postino la cifra indicata, spedendo il tagliando pubblicato in fondo alla pagina.

N.	Sistema	Programmi	Supporto	pubblicato in <i>Personal Software</i> n.	Prezzo
1	Apple II+	La carta del cielo Collisione	floppy 5" DOS 3.3	3 pag. 83 3 pag. 93	30.000
2	TRS-80 mod. I	Backgammon	floppy 5" DOS 2.3	3 pag. 89	25.000
3	PET/CBM 3032/4032	Editor/Assembler in Basic	floppy 5" 3032/4032+3040/4040	2 pag. 33	40.000
4	Apple II+	Interi in precisione multipla Grafica 3D	floppy 5" DOS 3.3	4 pag. 17 4 pag. 47	40.000
5	PET/CBM 3032/4032	Gioco del calcio	floppy 5" 3032/4032+3040/4040	4 pag. 67	25.000
6	Apple II +	Pretty Printer Shape Table	floppy 5" DOS 3.3	5 pag. 27 5 pag. 58	30.000
7	Apple II +	Data base modulare	floppy 5" DOS 3.3	7 pag.	25.000

Spedire in busta  
chiusa a

PERSONAL SOFTWARE  
Servizio Programmi  
Via Rosellini 12  
20124 Milano

Inviatemi i seguenti dischi di *Personal Software*

n. \_\_\_\_\_

per un totale di lire \_\_\_\_\_ + L. 2.000 come contributo fisso  
spese di spedizione che pagherò al postino alla consegna del pacco.

\_\_\_\_\_  
Cognome e nome

\_\_\_\_\_  
Indirizzo

\_\_\_\_\_  
Cap., Località

\_\_\_\_\_  
Firma



UNA PUBBLICAZIONE  
DEL GRUPPO EDITORIALE JACKSON

# PERSONAL SOFTWARE

ANNO 2 N. 8-9 LUGLIO-AGOSTO 1983

DIRETTORE RESPONSABILE: Giampietro Zanga

DIRETTORE: Mauro Boscarol

REDAZIONE: Completo Software - Padova

HANNO COLLABORATO A QUESTO NUMERO:

R.W. Anderson, S. Borsani, E. Bossaglia, R. Del Giudice,  
B. Del Medico, S. De Monte, B. Douglas, P. Ferrari,  
E. Ferreguti, N.E. Fischer, L. Gemme, F. Manfredini,  
M. Minischetti, G. Morpurgo, P. Pulli, M. Redolfi,  
C. Sintini, M. Spero.

*Copertina e illustrazioni:* MORGANA artefatti comunicativi - Padova

*Fotocomposizione:* Composizioni Grafiche - Padova

CONTABILITÀ: Franco Mancini, Roberto Ostelli,  
Mariella Luciano, Franca Anelli, Sandra Cicuta,  
Gabriella Napoli

DIFFUSIONE E ABBONAMENTI: Luigi De Cao,  
Adela Bel Lozano, Ombretta Giannetto

AUTORIZZAZIONE ALLA PUBBLICAZIONE: Tribunale  
di Milano n. 69 del 20/2/1982

PUBBLICITÀ: Concessionario per l'Italia e l'Estero  
Reina s.r.l. Via Washington, 50 - 20146 Milano  
Tel. (02) 4988066/7/8/9/060 (5 linee r.a.)  
Telex 316213 REINA I

STAMPA: Arti Grafiche "La Cittadella" S.p.a.  
Pieve del Cairo (PV)

Concessionario esclusivo per la DIFFUSIONE in Italia  
e all'Estero:

SODIP - Via Zuretti, 25 - 20125 Milano

Spedizione in abbonamento Postale Gruppo III/70  
Prezzo della rivista L. 3.500. Numero arretrato L. 6.000.  
Abbonamento annuo (10 numeri) L. 30.000; per l'Estero  
L. 48.000

I versamenti vanno indirizzati a: Gruppo Editoriale Jackson -  
Via Rosellini, 12 - 20124 Milano - mediante emissione di  
assegno bancario, cartolina vaglia o utilizzando il c/c  
Postale numero 11666203.

Per i cambi di indirizzo, indicare, oltre naturalmente al  
nuovo, anche l'indirizzo precedente, ed allegare alla  
comunicazione l'importo di L. 500, anche in francobolli.

© TUTTI I DIRITTI DI RIPRODUZIONE O TRADUZIONE  
DEGLI ARTICOLI PUBBLICATI SONO RISERVATI



GRUPPO EDITORIALE JACKSON Srl

DIREZIONE, REDAZIONE, AMMINISTRAZIONE:  
Via Rosellini, 12 - 20124 Milano - Telefoni: 68.03.68 - 68.00.54

SEDE LEGALE: Via Vincenzo Monti, 15 - 20123 Milano

DIREZIONE EDITORIALE: Giampietro Zanga e Paolo Reina  
COORDINAMENTO EDITORIALE: Daniele Comboni

# ZX Spectrum

## Lo trovi anche nel tuo BITSHOP PRIMAVERA

**ALESSANDRIA** Via Savonarola, 13  
**ANCONA** Via De Gasperi, 40  
**BARI** Via Capruzzi, 192  
**BASSANO DEL GRAPPA** Via Jacopo Da Ponte, 51  
**BERGAMO** Via S. F. D'Assisi, 5  
**BIELLA** Via Italia, 50A  
**BOLOGNA** Via Brugnoli, 1  
**CAGLIARI** Via Zagabria, 47  
**CALTANISSETTA** Via R. Settimo, 10  
**CAMPOBASSO** Via Mons. Il Bologna, 10  
**CATANIA** Via Muscatello, 6  
**CESANO MADERNO** Via Ferrini, 6  
**CESENA** Via Flli Spazzoli, 239  
**CINISELLO BALSAMO** V.le Matteotti, 66  
**COMO** Via L. Sacco, 3  
**COSENZA** Via Dei Mille, 86  
**CREMA** Via IV Novembre, 56/58  
**CUNEO** C.so Nizza, 16  
**FAVRIA CANAVESE** C.so G. Matteotti, 13  
**FIRENZE** Via G. Milanese, 28/30  
**FOGGIA** Via Marchiano, 1  
**FORLÌ** P.zza Melozzo, 1  
**GALLARATE** Via A. Da Brescia, 2  
**GENOVA** Via Domenico Fiasella, 51/R  
**GENOVA** C.so Gastaldi, 77/R  
**GENOVA-SESTRI** Via Chiaravagna, 10/R  
**GENOVA-SESTRI** Via Ciro Menotti, 136/R  
**IMPERIA** Via Delbecchi, 32  
**LECCE** V.le Marche, 21  
**LECCO** Via L. Da Vinci, 7  
**LIVORNO** Via San Simone, 31  
**LUCCA** Via S. Concordio, 160  
**MACERATA** Via Spalato, 126  
**MERANO** Via S. Maria del Conforto, 22  
**MESSINA** Via Del Vespro, 71  
**MESTRE** Piazza Ferretto, 78  
**MILANO** Via G. Cantoni, 7  
**MILANO** Via E. Petrella, 6  
**MILANO** Via Altaguardia, 2  
**MILANO** P.zza Firenze, 4  
**MILANO** V.le Corsica, 14  
**MILANO** V.le Certosa, 91  
**MIRANO-VENEZIA** Via Gramsci, 40  
**MODENA** Via Fonteraso, 18  
**MONZA** Via Azzone Visconti, 39  
**MORBEGNO** Via Fabiani, 31  
**NAPOLI** Via Lugia Sanfelice, 7/A  
**NAPOLI** C.so Vittorio Emanuele, 54  
**NOVARA** Baluardo Q. Sella, 32  
**PADOVA** Via Fistomba, 8  
**PALERMO** Via Libertà, 191  
**PARMA** Via Imbriani, 41  
**PAVIA** Via C. Battisti, 4/A  
**PERUGIA** Via R. D'Andreotto, 49/55  
**PESCARA** Via Tiburtina, 264 bis  
**PESCARA** Via Trieste, 73  
**PIACENZA** Via IV Novembre, 60  
**PISA** Via XXIV Maggio, 101  
**PISA** Via Emilia, 36  
**PISTOIA** V.le Adua, 350  
**POMEZIA** Via Roma, 39  
**POTENZA** Via G. Mazzini, 72  
**POZZUOLI** Via G.B. Pergolesi, 13  
**PRATO** Via E. Boni, 76/78  
**RIMINI** Via Bertola, 75  
**ROMA** L.go Belloni, 4 (Vigna Stelluti)  
**ROMA** P.zza San Donà Di Piave, 14  
**ROMA** V.le IV Venti, 152  
**ROMA** Via Cerreto Da Spoleto, 23  
**ROMA** Via Panzio Cominio, 46  
**ROMA** Via Del Traforo, 136  
**SAVONA** Via G. Scarpa, 13/R  
**SONDRIO** Via N. Sauro, 28  
**TERAMO** Via Martin Fennesi, 14  
**TORINO** C.so Grosseto, 209  
**TORINO** Via Tripoli, 179  
**TORINO** Via Nizza, 91  
**TRENTO** Via Sighele, 7/1  
**TREVIGLIO** V.le Buonarroti, 5/A  
**TRIESTE** Via F. Saverio, 138  
**TRIESTE** Via Torrebianca, 18  
**UDINE** Via Tavagnacco, 89/91  
**VARESE** Via Carrobbio, 13  
**VENEZIA** Cannaregio, 5898  
**VERCELLI** Via Dionisotti, 18  
**VIAREGGIO** Via A. Volta, 79  
**VOGHERA** P.zza G. Carducci, 11



La prima e la più grande catena  
di computer in Italia.

Telefono 02/6120848-6120795