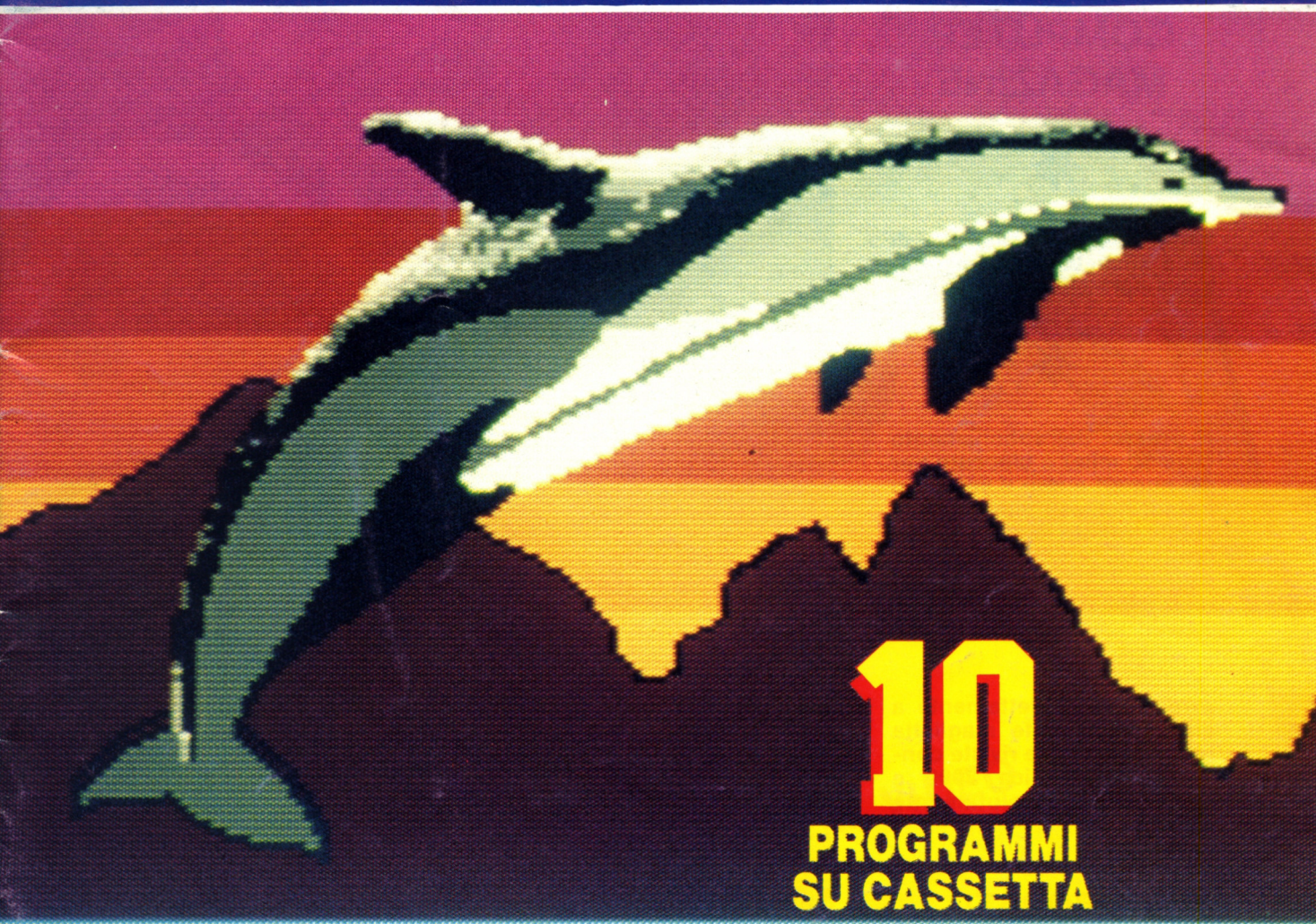
NEXT STOP SUMMER

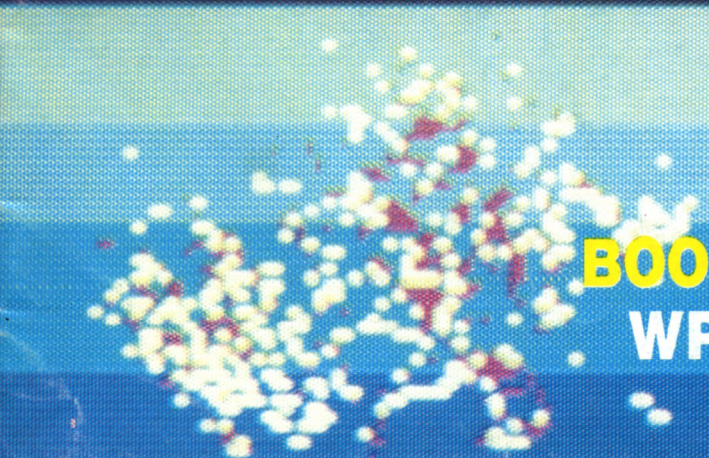
MSX COMPUTER MAGAZINE

N.14

Sped in abb. post. Gr. III L. 9.000



10 PROGRAMMI SU CASSETTA



SCROLL & SOUND

BOOLE NELL'ISTRUZIONE IF

WP: LA GESTIONE VIDEO

UNA MAGLIETTA IN REGALO!

a chi si abbona a

MSX COMPUTER MAGAZINE

sei magnifiche cassette di programmi di gioco e di utilità, sempre più belle e ricche!

il prezzo dell'abbonamento (Lire 50 mila) è bloccato per sei numeri e non ti verranno quindi richiesti aumenti (già subito intanto risparmi 4 mila lire)!

avrai subito, direttamente a casa, un'elegante maglietta (realizzata con le riviste consorelle Elettronica 2000 e Load'n'Run) assolutamente gratis!

ABBONATI OGGI STESSO

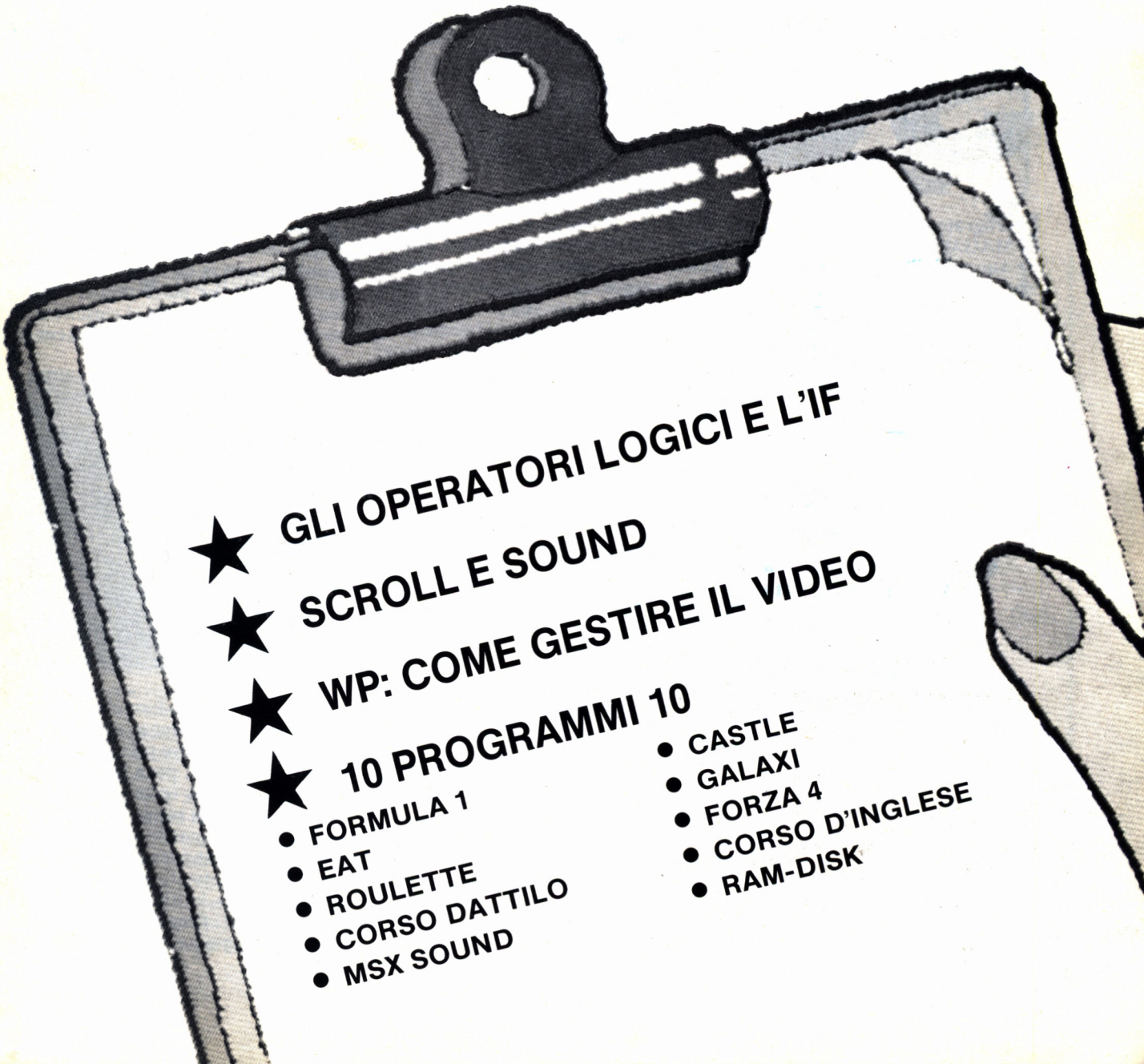
Basta inviare un vaglia ordinario (quello rosa, da richiedere in un qualunque ufficio postale) di lire 50 mila.
Indica esattamente da quale fascicolo desideri l'abbonamento ed i tuoi dati chiari e precisi. Indirizza a MSX Computer Magazine, C.so Vitt. Emanuele, 15 - 20122 Milano.

MSX Computer Magazine è edita da Arcadia srl,
C.so Vitt. Emanuele 15, Milano.
Tel. 02/706329 (solo giovedì h. 15-18).
Una copia L. 9.000.
Fotocomposizione: Composit.
Stampa: Garzanti,
Milano. Distribuzione: SO.DI.P. Angelo
Patuzzi srl, Via Zuretti 25, Milano.
Registrato Trib. Milano N. 52 del 2/2/85.
Resp. Sira Rocchi.
Sped. in abb. post. Gr. III/70.

Manoscritti, disegni, fotografie
e programmi inviati non si
restituiscono anche se non pubblicati.

MSX COMPUTER MAGAZINE 14

IN QUESTO NUMERO



★ GLI OPERATORI LOGICI E L'IF

★ SCROLL E SOUND

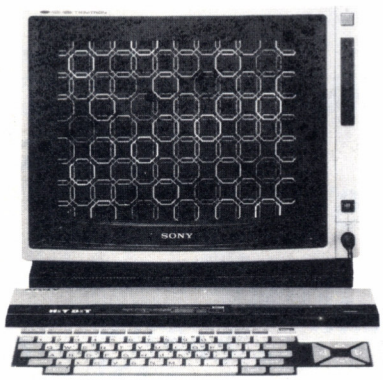
★ WP: COME GESTIRE IL VIDEO

★ 10 PROGRAMMI 10

- FORMULA 1
- EAT
- ROULETTE
- CORSO DATTILO
- MSX SOUND
- CASTLE
- GALAXI
- FORZA 4
- CORSO D'INGLESE
- RAM-DISK

MSX TAPE SOFT

I programmi contenuti in questo fascicolo di MSX COMPUTER MAGAZINE sono tutti compatibili con qualsiasi sistema MSX. Ecco per voi ancora 10 bei programmi! Ricordate di collegare la spina del controllo motore alla presa REM del vostro registratore se quest'ultimo la possiede. Assicuratevi che la spina nera sia collegata alla presa EAR del registratore e che la spina rossa sia inserita nella presa MIC. Se il vostro mangiacassette non possiede la presa REM, fate particolare attenzione a quando un programma è stato caricato o deve esserlo, affinché il nastro scorra per il giusto tempo. Appena vedete apparire sul video, dopo un comando di caricamento, la scritta OK, spegnete il registratore. Nelle istruzioni che seguono troverete, accanto al titolo ed al codice di caricamento di ogni programma, la notazione della memoria minima necessaria per vedere il programma.



Nella cassetta allegata a questo fascicolo troverete, sul lato A:
Formula 1, Eat, Roulette, Corso di dattilo, Msx Sound Adventure.
Sul lato B ci sono:
Castle, Galaxi, Forza 4, Corso d'inglese, Ram-disk.

1

FORMULA 1

(CLOAD "F. 1" - 16 K)

di R. Mesiti

Alla guida della vostra auto da corsa, dovete raggiungere i due traguardi volanti in un tempo inferiore a quello limite per poter tagliare la linea d'arrivo finale. Se ci riuscirete vi verranno assegnati dei punti secondo il criterio della vera formula 1.

Alla partenza bisogna stare attenti al semaforo e partire solo all'apparire del segnale verde, pena la squalifica. Avete tre bolidi a vostra disposizione per completare la prova. Se rilasciate i comandi l'auto comincerà a rallentare fino a scomparire in basso sullo schermo. Dovete stare attenti a non scontrarvi con le altre auto e a non uscire di pista. Ricordate che più alta è la velocità, più sarà difficile pilotare

la vostra auto; tenete quindi d'occhio l'indicatore di velocità (espressa in km/h).

Nella spiacevole eventualità di un incidente, verranno visualizzati il tempo impiegato fino ad allora ed i chilometri percorsi. Per cominciare una nuova gara, premete la barra spaziatrice o il tasto fire del joystick.

È possibile giocare sia con la tastiera che con il joystick. Il significato di ogni tasto è riportato all'inizio del gioco.

EAT

(CLOAD "EAT" - 32 K)

di M. De Iuliis

Avete molta fame ma, prima di abbuffarvi, dovete raccogliere otto oggetti: cucchiaio, forchetta, coltello, piatto, tovagliolo, bottiglia, bicchiere e brocca.

Il gioco si presenta in due schermi. Nel primo bisogna colpire gli oggetti nascosti in altrettante anfore: queste anfore si muovono orizzontalmente e si rompono se vengono colpite tre volte. Nel secondo schermo si raccolgono gli oggetti colpiti nella precedente schermata.

Il passaggio da uno schermo all'altro avviene quando tutte le anfore sono state colpite; attenti, esse non sempre si muovono orizzontalmente ma, occasionalmente, possono precipitare dalla loro posizione verticale: in questo caso dovranno essere colpite tre volte altrimenti si perderà l'oggetto che esse contengono.

Il secondo schermo si articola nel seguente modo: bisogna recuperare gli oggetti nello stesso ordine con il quale sono stati colpiti (l'oggetto da recuperare diventa di colore giallo). Il recupero avviene tramite un puntatore che non deve toccare le asce tra le quali sono posti gli oggetti, altrimenti si perde una vita. Il gioco finisce quando si hanno tutti gli oggetti oppure quando si esauriscono le vite a disposizione.

Per sparare utilizzate la barra spaziatrice, per muovere il cursore usate i tasti cursore, oppure attaccate il joystick.

Le parti più interessanti del programma sono le due routine in linguaggio macchina. Una consente lo spostamento degli sprites indipendentemente dallo svolgersi del programma, l'altra verifica se è avvenuta una collisione tra sprites.

La routine di spostamento inizia all'indirizzo F100h ed utilizza una tavola delle velocità a partire da F000h che contiene, per ogni sprite, rispettivamente un byte per la velocità verticale, uno per la velocità orizzontale e due byte riservati.

Se il byte per la velocità ha un valore da 1 a 127 lo sprite si muove in basso (o a destra), mentre se ha un valore da 255 a 128 si muove in alto (o a sinistra).

La routine è attivata con il comando "jump" del linguaggio assembler all'indirizzo di memoria FD9Fh (tale indirizzo corrisponde ad una CALL che lo Z80 effettua ogni cinquantesimo di secondo), a cui far seguire l'indirizzo di inizio della nostra routine. Perciò noi avremo: FD9Fh,C3h — FDA0h,0 — FDA1h,F1h.

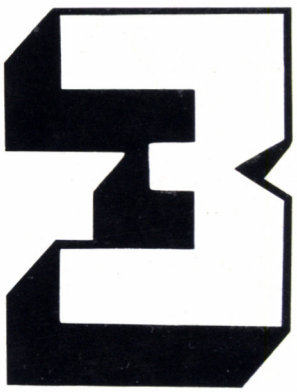
Prima di attivare la routine, bisogna porre in EFFFh il numero di sprites da mettere in movimento.

Per disattivare la routine basta porre in FD9Fh un RET (codice C9h) come normalmente è memorizzato.

La seconda routine in linguaggio macchina verifica se un certo sprite è entrato in collisione con altri (secondari). Per utilizzarla bisogna dare: DEFUSR=F19Ah e poi inserire i seguenti quattro valori a partire da F080h:

F080h = numero sprite principale
F081h = numero del primo sprite del gruppo dei secondari
F082h = numero dell'ultimo sprite del gruppo dei secondari
F083h = tolleranza in pixel: distanza alla quale due sprites sono considerati coincidenti.

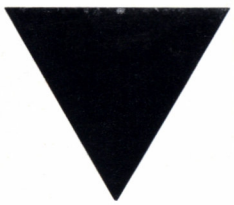
La routine viene chiamata con Z=USR(0). "Z" assume il valore —1 se non vi è stata collisione, altrimenti contiene il numero dello sprite secondario che l'ha causata.



ROULETTE

(CLOAD "ROULET" - 32 K)

di A. Stringhi



Se non puoi andare al casinò a tentare la fortuna, invita un tuo amico e vivi ugualmente le emozioni del gioco d'azzardo utilizzando questo programma che ripropone il gioco della roulette. Il regolamento è il seguente:

1 - si gioca in tre: due giocatori più il computer;
2 - il computer tiene il banco ed ha un

5

MSX SOUND ADVENTURE

(CLOAD "SOU-AD" - 32 K)

di C. Vianello

capitale iniziale di L. 2.500.000;
3 - il capitale iniziale dei giocatori è di L. 500.000;
4 - è ammessa una sola puntata per volta con gettoni da L. 10, 100, 1000, 10.000, 100.000;
5 - vince chi termina con il maggiore capitale;
6 - la partita termina quando uno dei due giocatori, o il computer, resta senza capitale.

Le vincite avvengono nel seguente modo: numero singolo (pleine) vince 35 volte quanto scommesso; due numeri contigui (cheval) 17 volte; tre numeri trasversali (transv. 3) 11 volte; quattro numeri tangenti (carre) 8 volte; sei numeri su due righe successive (transv. 6) 5 volte. Le puntate che vincono 2 volte sono: prima dozzina da 1 a 12 (12 P); seconda dozzina da 13 a 24 (12 M); terza dozzina da 25 a 36 (12 D); colonna di sinistra (col. 1); colonna di centro (col. 2); colonna di destra (col. 3). Quelle che vincono una volta: numeri pari (pair); numeri dispari (impair); numeri rossi (rouge); numeri neri (noir); prima metà da 1 a 18 (passe); seconda metà da 19 a 36 (manque). Buona fortuna!

4

DATTILOGRAFIA (1ª PARTE)

(CLOAD "CORSO 1" - 32 K)

di F. Baccani

Dopo il corso di inglese (che termina con questo numero), MSX COMPUTER MAGAZINE presenta un nuovo programma didattico in quattro puntate che vi insegnerà le tecniche migliori per diventare esperti dattilografi. In questa prima puntata sono presenti le prime cinque lezioni (dalla n. 1 alla n. 5) oltre alla lezione n. 21 e n. 0. La lezione n. 21, detta di esercitazione, presenta a caso dei caratteri studiati nelle cinque lezioni. La lezione n. 0, invece, è sempre una sezione d'esercitazione ma più complessa della precedente, quindi è consigliabile farne uso soltanto dopo essere passati dalla lezione n. 21.

Ogni esercitazione è composta dalla successione di 122 lettere disposte con metodo. L'inizio di ogni serie di caratteri è segnalato da due BEEP. In questa prima puntata imparerete ad usare ed a memorizzare le posizioni sulla tastiera dei seguenti caratteri: A S D F J K L; M E I G H R U. Per interrompere il corso di ogni esercitazione bisogna premere il tasto funzione n. 1 (F1) con il quale si torna al menu principale. Con il tasto F5 si ha il proprio indice di apprendimento mentre con il tasto F3 si ricomincia la lezione corrente.

Per quanto riguarda l'indice di apprendimento esso verrà esposto sotto forma di 6 contatori aventi il seguente significato:

ERRATE = indica il numero di battute errate

GIUSTE = indica il numero di battute corrette

TOTALE = indica il totale delle battute

BAT/MIN = riporta il totale delle battute corrette eseguite in un minuto

LEZIONE = visualizza il numero

Non si tratta di un gioco, nè di un'avventura, bensì di un'utility. Non la solita utility musicale ma qualcosa di speciale. Infatti, oltre che poter sperimentare con il processore sonoro inserendo i dati in modo immediato, questo programma fornisce tutte le indicazioni per utilizzare l'effetto sonoro da noi provato in un altro programma.

Per usare l'utility attenetevi alle seguenti istruzioni:
è importantissimo digitare le varie cifre richieste come visualizzato dal programma stesso. Per esempio, se l'utility chiede d'inserire il valore di un registro da 00 a 13 e si vuole modificare il registro 7, si dovrà digitare 07. Oppure, se il programma richiede un numero da 000 a 255 e si vuole immettere il valore 18, bisognerà digitare 018.

Al registro n. 7 bisogna fornire sei valori binari 0 o 1, 0 significa aperto (ON), 1 significa chiuso (OFF).

Il programma provvede da solo a correggere eventuali errori di battitura.

Per uscire dal programma con le istruzioni necessarie per ottenere quell'effetto sonoro in qualsiasi altro programma Basic bisogna digitare, alla richiesta del canale da modificare, ££.

Per riazzerare tutti i parametri tenere premuto il tasto CONTROL e battere due volte il tasto STOP.

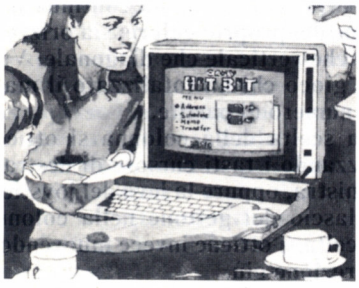
UN ESEMPIO: EFFETTO ELICOTTERO

Alle varie domande, digitare in sequenza i valori separati dal trattino (—). Non bisogna battere RETURN dopo ogni valore:
01-03-00-200-06-27-07-000011-08-16-11-200-12-002-13-12

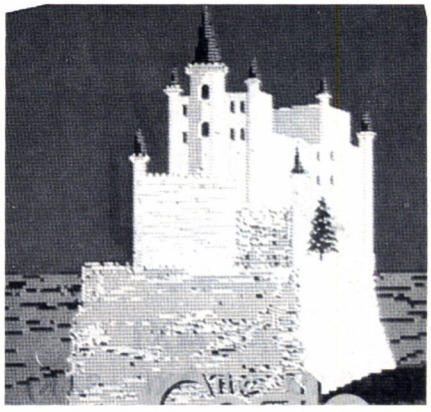
della lezione in corso
MAX = è il record stabilito fino a quel momento.

PER CHI HA IL DRIVE

Quando ci si trova al menu, si può premere il tasto F2 seguito da RETURN ed accedere ad una sezione differente del corso, cioè ad una delle quattro puntate in cui è articolato l'intero programma di dattilografia, potendo così esercitarsi con una delle 20 lezioni previste dal corso. In realtà F2 consente di accedere ad un sottomenu per la gestione dei files su disco. Ovviamente tutto questo sarà completamente operativo soltanto quando si avranno le quattro puntate del corso. Inoltre bisogna ricordarsi di salvare ogni programma con lo stesso nome con il quale è stato caricato.



6



CASTLE

(CLOAD "CASTLE" - 16 K)

di M. Nicolosi

Il gioco si articola in tre quadri ed ha come scopo il raggiungimento del castello e quindi la liberazione della principessa. Lo scenario del primo quadro è una palude: si dovrà tentare di attraversarla saltando sulle piazzole che man mano compariranno fino al raggiungimento dell'altra sponda. Il tutto verrà ostacolato dalla comparsa di insetti giganti che si precipiteranno sull'omino. Il contatto con essi penalizza di una vita e provoca l'addormentarsi dell'omino. Per riprendere il movimento dell'eroe bisogna lasciare prima che gli insetti passino sopra di esso e quindi premere i tasti di con-

trollo cursore (destra o sinistra).

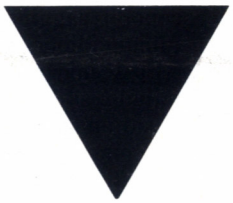
L'avvenuto arrivo sull'altra sponda della palude provocherà l'apparire di un messaggio ed il suono di una musichetta. Si passerà quindi al secondo schema. Qui l'omino dovrà lottare contro delle palle infuocate che tentano di atterrarlo per impedirgli di attraversare la caverna per raggiungere il castello dove si trova la principessa. Ad aumentare la difficoltà di questo quadro ci sono dei precipizi che provocano la caduta dell'eroe. Anche in questo caso il contatto con le palle infuocate provoca la morte dell'omino.

A completamento del secondo quadro si passa finalmente al terzo ed ultimo schermo: lo scenario è un campo con il castello nel quale è tenuta prigioniera la principessa. All'eroe il compito di liberarla schivando frecce e palle infuocate (non si spengono mai!). Non è finita: attenzione anche al fossetto che potrebbe aprirsi sotto i piedi del nostro prode! Usate i tasti cursore destra e sinistra per spostare l'omino, la barra spaziatrice per farlo saltare.

GALAXI

(CLOAD "GALAXI" - 32 K)

di A. Vecchio



Eccoci alla conquista dello spazio a combattere contro alieni e navicelle nemiche con armi a tecnologia laser.

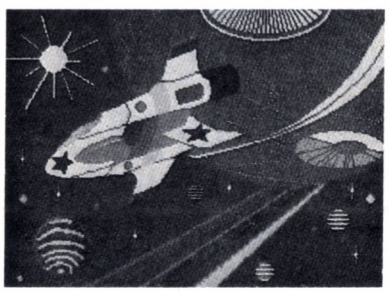
Per giocare si può usare la tastiera

(in tal caso premete la barra spaziatrice dopo aver dato il RUN) oppure il joystick (premete il tasto fire).

Il video è suddiviso così:
- il quadro grande a sinistra ospita l'azione;
- il riquadro in alto a destra è il radar;
- nella parte in basso a destra vi sono gli indicatori del gioco: punteggio, livello e colpi.

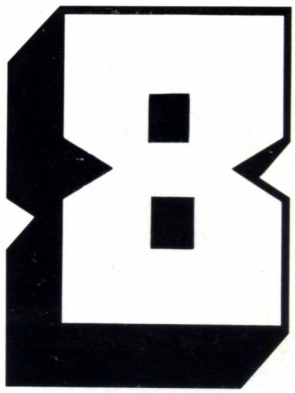
Per terminare il gioco bisogna colpire le basi nemiche individuate sul radar dal simbolo di due quadratini uno interno all'altro. Oltre che le basi nemiche, il radar è in grado di visualizzare la presenza di città (rappresentate da un cerchietto), quella del mare e delle foreste (quadratino); "vede" inoltre i corsi d'acqua (segmenti lineari, a croce e ad angolo retto) e l'astronave propria (un puntino).

Quando atterrate in una base nemica non potete più andarvene e dovete colpire al centro un prisma, posto nella parte alta dello schermo. Fatto questo vi aspetterà una dura prova di prontezza di riflessi consistente nel premere il tasto fire (o spazio) quando appare la scritta "FIRE". Da questa prova dipenderà il funzionamento del-



l'astronave sulla quale viaggiate.

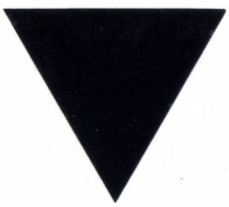
Il programma per eseguire il test della collisione tra sprite non usa il comando ON SPRITE GOSUB ma va a testare direttamente il bit n. 5 del registro del VDP. Se capitasse che il programma non dovesse quindi riconoscere l'urto tra due sprite, andate a vedere la linea 8390.



FORZA 4

(CLOAD "FORZA 4" - 32 K)

di A. Stringhi



Un gioco piacevole, semplice e colorato dove bisogna giocare d'intelligenza. Ci si cimenta in due con la tastiera oppure usando la leva del joystick. Ogni giocatore ha a disposizio-

ne 21 pedine che deve inserire in una specie di raccoglitore, in modo da formare una particolare combinazione.

All'inizio del gioco occorre inserire il nome dei giocatori, il colore dei gettoni, la scelta (individuale) tra tastiera e joystick ed il numero di partite che si vogliono fare.

Il gioco consiste nell'infilare, ad ogni turno, una pedina in una delle 7 colonne della grata.

Vince la partita chi, per primo, riesce a formare una fila continua di 4 pedine del proprio colore: sia orizzontale, che verticale che diagonale. Vince il gioco chi ha totalizzato il maggior numero di partite.

La scelta della colonna si ottiene utilizzando i tasti cursore di destra o di sinistra, oppure la leva del joystick. Il rilascio del gettone nella colonna prescelta si ottiene invece muovendosi in direzione giù.

9

CORSO DI INGLESE (4ª parte)

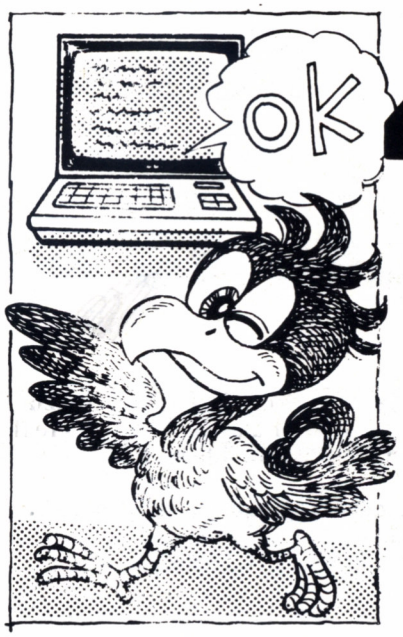
(CLOAD "PARTE4" - 32 K)

di G. Bellomusto

Siamo arrivati all'ultima puntata del corso. La lezione, come sempre, è composta da argomenti ed esercitazioni. In ogni pagina sarete chiamati a rispondere a delle domande con un sì (s) o un no (n). Alla fine, in fondo ad ogni pagina, comparirà un sottomenu che servirà a passare ad un altro argomento. Le istruzioni sugli esercizi

vi saranno proposte quando sceglierete la voce "esercizi". Tutti gli esercizi e gli argomenti saranno trattati con le contrazioni, perché in inglese parlato si usa spesso la contrazione.

In questa lezione si parlerà di: presente semplice, avverbi di frequenza, pronomi possessivi, verbo + oggetto diretto + oggetto indiretto, progressivo + futuro, verbo dovere, esercizi. Good lesson!



10

RAM DISK

(CLOAD "RAMDIS" - 64 K)

di D. Montresor

Abbiamo parlato più volte dei sistemi MSX con 64 KRAM e della possibilità di utilizzare quei 32 KRAM non usati dall'interprete Basic. Un primo articolo a riguardo è stato pubblicato sul numero 7 di MSX Computer Magazine, cui sono seguite numerose lettere su come sfruttare al meglio questa risorsa RAM. L'utility che stiamo per presentarvi dovrebbe soddisfare un po' tutti, anche quelli più esigenti. Si tratta di un programma che permette di usare i 32 KRAM inferiori dei sistemi a 64K come disco virtuale in cui si può aprire un solo file.

L'uso del programma si può dividere in inizializzazione o formattazione, scrittura e lettura.

1) Inizializzazione: dopo aver caricato il programma in memoria, definite la partenza della routine in L/M con DEFUSR=61680, dimensionate un vettore stringa con il numero dei campi che si vogliono ottenere per scheda e allo stesso modo un vettore numerico che conterrà le lunghezze dei campi stessi. Scrivete nella locazione 62318 il numero dei campi e nella locazione 62187 il valore 0 poi, a partire dall'indirizzo 62190, scrivete le lunghezze di ogni campo. Per finire, fate eseguire l'istruzione USR per la formattazione (linea 340 del programma di esempio).

2) Scrittura: registrate in formato a due byte alla locazione 62184 il puntatore scheda compreso tra 0 ed il valore indicato alla locazione 62320 e seguente meno uno, cioè, nel caso in cui l'archivio contenga 10 schede, si potrà scegliere come puntatore un numero da 0 a 9. Azzerate la locazione 62186, quindi caricate il vettore stringa con i campi della scheda facendo attenzione che la loro lunghezza sia uguale o inferiore a quella precedentemente registrata; fate eseguire la subroutine in L/M come alla linea 580 del programma Basic.

3) Lettura: dopo aver registrato il puntatore scheda come nel caso precedente, scrivete alla locazione 62186 un numero diverso da 0, cancellate le singole variabili dell'array stringa come alla riga 700 del programma facendo uso del vettore numerico e, anche in questo caso, eseguite l'istruzione USR come spiegato precedentemente.

Nel caso vogliate nuovamente inizializzare la memoria, scrivete un valore diverso da C7 esadecimale alla locazione 62187. Le linee del programma Basic da 230 a 760 sono un esempio di utilizzo dell'utility. Volendo si può ampliare l'archivio a piacere aggiungendo nuove funzioni. Ogni scheda può avere da 1 a 128 campi ed il numero totale di schede disponibili dipende dall'occupazione di memoria delle stesse.

UTILITY

SCROLL E SOUND

SEI ROUTINE INDISPENSABILI PER LA CREAZIONE DI PROGRAMMI D'EFFETTO. COME FAR SCORRERE LE SCRITTE SUL VIDEO E PRODURRE SUONI DA VIDEOGIOCO

a cura della Redazione

Programmare in Basic è facile con l'MSX perché si tratta di un linguaggio completo di tutte le istruzioni ed in particolare di quelle dedicate agli interrupt con le quali è possibile realizzare effetti che su altre macchine sono ottenibili soltanto in linguaggio macchina. Ma come spesso accade anche il Basic MSX non dà tutto quello che noi vorremmo, così dobbiamo anche con questa macchina ricorrere ai ripari prevedendo nei nostri programmi piccole routine in linguaggio macchina. Scopo di questo articolo è proprio quello di dare al lettore quelle routine che servono quasi sempre quando si vuole realizzare un programma condito di effetti grafici e sonori, per esempio i videogiochi. Vediamo insieme di capire come son organizzate le routine presentate in queste pagine. I risultati non tarderanno per voi!

```
5000 DATA 3A,50,C3,01,20,00,21,1F,18,3C,
3D,CA,52,9C,09,C3,4A,9C,CD,4A,00,F5,01,1
F,00,2B,CD,4A,00,23,CD,4D,00
5010 DATA 2B,0B,79,B0,C2,59,9C,F1,CD,4D,
00,C9
5020 RESTORE 5000:FOR N=40000! TO 40044!
:READ A$:POKE N,VAL("&H"+A$):NEXT N:RETU
RN
```

Listato n. 1 - Caricatore Basic della routine di scroll del testo a destra.

```
6000 DATA 3A,51,C3,01,20,00,21,00,18,3C,
3D,CA,B6,9C,09,C3,AE,9C,CD,4A,00,F5,01,1
F,00,23,CD,4A,00,2B,CD,4D,00
6010 DATA 23,0B,79,B0,C2,BD,9C,F1,CD,4D,
00,C9
6020 RESTORE 6000:FOR N=40100! TO 40144!
:READ A$:POKE N,VAL("&H"+A$):NEXT N:RETU
RN
```

Listato n. 2 - Caricatore Basic della routine di scroll del testo a sinistra.

ROUTINE N. 1: SCROLL DEL TESTO A DESTRA

La prima routine che presentiamo è dedicata allo screen in modo 1, cioè allo schermo per il testo di 32 caratteri per linea.

La routine è in grado di eseguire uno scroll di una qualsiasi delle 24 righe verso destra. Nella locazione di memoria 50000 deve essere memorizzato un numero compreso tra 0 e 23 il quale indica la riga che dovrà slittare verso destra. La routine in l/m è allocata a partire dall'indirizzo 40000 fino all'indirizzo 40044. Il listato n. 1 è il caricatore Basic della routine.

ROUTINE N. 2: SCROLL DEL TESTO A SINISTRA

Esegue la stessa funzione della routine n. 1 con la sola differenza che il testo è fatto slittare verso sinistra. Altra differenza è la locazione di memoria dove memorizzare il numero della linea su cui agire, che in questo caso è la locazione 50001. Il valore da inserire qui deve sempre essere compreso

tra 0 e 23. La routine inizia all'indirizzo 40100 e termina all'indirizzo 40144. Il listato n. 2 è il caricatore Basic.

```
7000 DATA 3A,52,C3,3C,21,00,18,3D,CA,7B,
9D,23,C3,73,9D,CD,4A,00,F5,01,17,00,11,2
0,00,19,CD,4A,00,ED,52,CD,4D
7010 DATA 00,19,0B,79,B0,C2,85,9D,F1,CD,
4D,00,C9
7020 RESTORE 7000:FOR N=40300! TO 40345!
:READ A$:POKE N,VAL("&H"+A$):NEXT N:RETU
RN
```

Listato n. 3 - Caricatore Basic della routine di scroll del testo verso l'alto.

```
8000 DATA 3A,53,C3,3C,21,E0,1A,3D,CA,17,
9D,23,C3,0F,9D,CD,4A,00,F5,01,17,00,11,2
0,00,ED,52,CD,4A,00,19,CD,4D
8010 DATA 00,ED,52,0B,79,B0,C2,21,9D,F1,
CD,4D,00,C9
8020 RESTORE 8000:FOR N=40200! TO 40246!
:READ A$:POKE N,VAL("&H"+A$):NEXT N:RETU
RN
```

Listato n. 4 - Caricatore Basic della routine di scroll del testo verso il basso.

```
10 CLEAR 200,39999!
20 GOSUB 5000
100 SCREEN 1
110 VDP(2)=6
120 X=65
130 FOR A=6144 TO 6880 STEP 32
140 Y=X
150 FOR B=0 TO 31
160 VPOKE (A+B),Y
170 Y=Y+1
180 NEXT B
190 X=X+1
200 NEXT A
210 POKE (50000!),10
220 DEFUSR=40000!
230 A=USR(1):FOR N=1 TO 1000:NEXT
240 GOTO 230
```

Listato n. 5
Programma Basic per provare le routine di scroll.

```
10 CLEAR 200,39999!
20 GOSUB 9000
30 DEFUSR=40600!
40 A=USR(0)
50 END
9000 DATA 3E,08,1E,0F,CD,93,00,3E,07,1E,
FE,CD,93,00,3E,00,1E,6E,CD,93,00,3E,01,1
E,00,CD,93,00,1E,6E,3E,00,CD
9010 DATA 93,00,3E,C8,F5,3E,0A,3D,C2,C0,
9E,F1,3D,C2,BD,9E,7B,C6,06,CA,D4,9E,3D,5
F,C3,B6,9E,3E,07,1E,FF,CD,93,00,C9
9020 RESTORE 9000:FOR N=40600! TO 40667!
:READ A$:POKE N,VAL("&H"+A$):NEXT N:RETU
RN
```

Listato n. 6
Caricatore Basic con demo della routine di simulazione del rumore emesso da un cannone a raggio laser.

ROUTINE N. 3: SCROLL DEL TESTO VERSO L'ALTO

Ci troviamo sempre a lavorare con lo screen in modo 1. Questa volta la routine esegue lo scroll di una delle 32 colonne verso l'alto. La locazione 50002 dovrà contenere un valore compreso tra 0 e 31 indicante la colonna su cui far agire la routine. Il linguaggio macchina si colloca tra l'indirizzo 40300 e l'indirizzo 40345. Nel listato n. 3 trovate il caricatore Basic.

ROUTINE N. 4: SCROLL DEL TESTO VERSO IL BASSO

Così come abbiamo presentato entrambi gli scroll orizzontali vediamo anche lo scroll verticale del testo verso il basso. La locazione di memoria dove inserire il valore della colonna (tra 0 e 31) da far slittare è la 50003. La routine si estende dall'indirizzo 40200 all'indirizzo 40246. Il listato n. 4 carica da Basic i codici macchina.

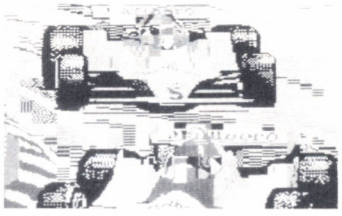
IL LISTATO PER PROVARE

Il listato n. 5, scritto anch'esso in Basic, serve a provare le routine fin qui descritte. Le linee 210 e 220 vanno cambiate a seconda della routine che si vuole provare. Per la routine n. 1 vanno bene così come sono scritte nel listato originale, per le altre tre routine bisognerà inserire alla linea 210, al posto di 50000, la locazione di memoria relativa alla riga o alla colonna su cui eseguire lo scroll. Inoltre la linea 220 dovrà contenere, al posto di 40000, il valore dello start delle altre tre routine. Per esempio per la routine n. 2 il valore è 40100, per la routine n. 3 è 40300 mentre l'ultima routine vista è 40200. Alla linea 230 il ciclo FOR-NEXT da 1 a 1000 serve soltan-

```
10 CLEAR 200,39999!
20 GOSUB 10000
30 DEFUSR=40400!
40 A=USR(0)
50 END
10000 DATA 3E,07,1E,FE,CD,93,00,3E,0A,1E
,0F,CD,93,00,1E,28,3E,00,CD,93,00,3E,0A,
F5,3E,FF,3D,C2,EA,9D,F1,3D,C2
10010 DATA E7,9D,7B,D6,96,CA,FF,9D,C6,97
,5F,C3,E0,9D,3E,00,1E,00,CD,93,00,3E,07,
1E,F7,CD,93,00,3E,00,F5,5F,CD
10020 DATA 93,00,3E,32,F5,3E,FF,3D,C2,19
,9E,F1,3D,C2,16,9E,F1,D6,1F,CA,2D,9E,C6,
20,C3,0F,9E,3E,64,F5,3E,FF,3D
10030 DATA C2,32,9E,F1,3D,C2,2F,9E,3E,07
,1E,FF,CD,93,00,C9
10040 RESTORE 10000:FOR N=40400! TO 4051
4!:READ A$:POKE N,VAL ("&H"+A$):NEXT N:R
ETURN
```

Listato n. 7 - Caricatore Basic con demo della routine di simulazione del rumore della caduta di una bomba.

to per rallentare l'esecuzione del programma e per vedere meglio che cosa succede sullo schermo. Provate ad eliminare questo ciclo e vi sorprenderete della velocità del linguaggio macchina.



ALTRE DUE ROUTINE: SUONI DA VIDEOGIOCO

Gli effetti grafici sono molto importanti, soprattutto nei programmi di gioco ma lo sono anche gli effetti sonori. Le prossime due routine simulano rispettivamente il suono di un cannone a raggio laser e quello della caduta di una bomba.

La prima routine (listato n. 6; caricatore Basic) è allocata dall'indirizzo 40600 all'indirizzo 40667 ed è attivata semplicemente chiamando il suo start (40600), cioè senza passare alcun parametro come avviene per le routine già descritte.

La seconda routine (listato n. 7; caricatore Basic) deve essere usata come la precedente sapendo che il suo start è l'indirizzo 40400.

Va ricordato che per entrambe le due routine le linee dalla 10 alla 50 servono semplicemente per chiamare la subroutine che carica i codici macchina e per attivare la routine tramite l'istruzione A=USR(0).

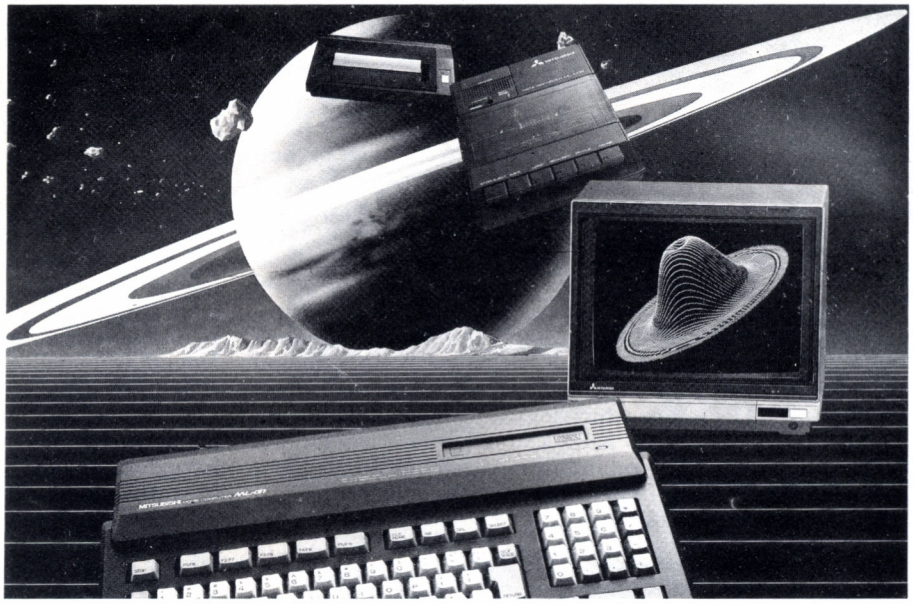
TUTTE E SEI IN UN UNICO LISTATO

Le sei routine (quattro grafiche e due sonore) che vi abbiamo presentato possono essere tutte quante utili in un solo programma. Infatti sono state studiate per poter essere inserite insieme in un listato. Le subroutine che caricano i codici macchina iniziano tutte con numeri di linea diversi (da 5000 a 10000) e lo start, in memoria RAM, è per ognuna di esse differente (40000, 40100, 40300, ecc.). L'importante è riservare la RAM necessaria, cioè inserire come prima linea del listato l'istruzione CLEAR, la quale definisce la massima locazione di memoria riservata al Basic. Oltre tale locazione verranno inseriti i dati in linguaggio macchina. Nel caso delle sei routine di questo articolo la locazione limite è 39999 perché a 40000 inizia la prima routine di scroll. Scriveremo allora come prima istruzione del programma CLEAR 200, 39999. ■

LOGIC

ALGEBRA BOOLEANA

GLI OPERATORI LOGICI ALL'INTERNO DELLA STRUTTURA IF. DAI CALCOLI MATEMATICI ALLA GRAFICA D'EFFETTO. VEDIAMO INSIEME QUALCHE ESEMPIO...

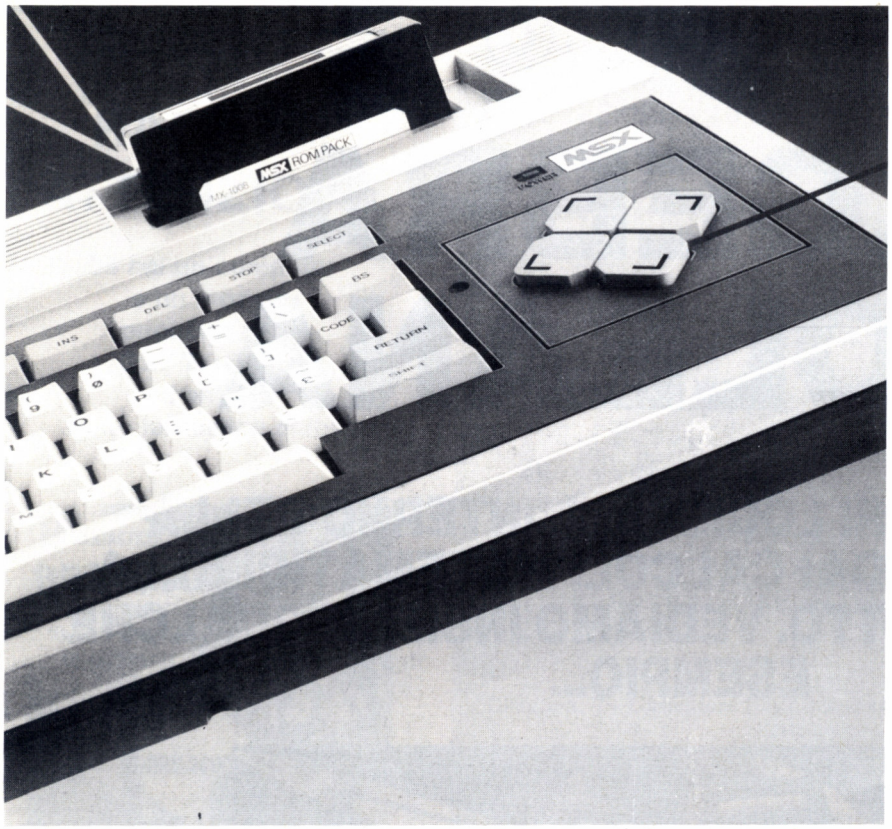


Da quanto abbiamo detto nella scorsa puntata, normalmente gli operatori logici vengono utilizzati con l'istruzione IF/THEN/ELSE ed è proprio di quest'ultima che vogliamo parlare prima di affrontare la risoluzione di alcuni problemi.

La forma sintattica dell'istruzione condizionale è la seguente:
IF <espressione> THEN <istruzione 1> ELSE <istruzione 2> il cui comportamento è il seguente: il gruppo <istruzione 1> è eseguito solo se <espressione> risulta vera altrimenti (ELSE) viene eseguito il gruppo <istruzione 2>. Va precisato che <istruzione 1> e <istruzione 2> possono essere una o più istruzioni. Ma quando <espressione> è vera? In generale tutte le volte che <espressione> dà come risultato non zero ed è falsa in caso contrario. Ecco alcuni esempi di <espressioni> vere e false:
1. 200<10 falsa
2. pippo=23 è vera se pippo=23 e falsa se pippo non è uguale a 23
3. alfa dà come risultato vero se la variabile alfa è diversa da 0 altrimenti il risultato è falso se alfa è uguale a 0.

Per chiarire meglio l'esempio n. 3 supponiamo di aver definito la variabile alfa=3.14, l'esecuzione dell'istruzione:
IF alfa THEN PRINT "vera" ELSE PRINT "falsa"
produrrà la stampa del messaggio "vera", se invece la variabile alfa avesse il valore 0 allora il messaggio stampato sarà "falsa".

Va ricordato che <espressione> può essere formata non solo da variabili numeriche ma anche da variabili



FENNER COURTESY

alfanumeriche il cui risultato (vero/falso) dipende dal valore ASCII di ogni carattere della stringa da testare. Per esempio:
1. "ABC"<"abc" vero perché le lettere minuscole hanno un valore ASCII minore di quelle maiuscole
2. "ABC">"ABD" falso perché la lettera "D" è maggiore della lettera "C" e quindi è vero "ABC"<"ABD".

Per completare l'argomento relativo all'istruzione condizionale IF/THEN/ELSE precisiamo che è possibile utilizzarla più volte sulla stessa linea come l'esempio seguente:

```
10 IF A=18 THEN IF B=20 THEN
      30 ELSE 40 ELSE 80
```

QUANDO C'È LA FLESSIBILITÀ

Decisamente la struttura della linea 10 non è molto semplice da tradurre ma abbiamo voluto così scriverla per farvi osservare quanto sia flessibile tale istruzione. In sintesi se A=18 allora viene eseguito il successivo test (IF B=20 THEN...) altrimenti il programma continua alla linea 80. Se la condizione è vera (A=18) allora viene valutata un'altra espressione, B=20, e se anche quest'ultima è vera il programma continua alla linea 30 altrimenti alla linea 40. Regola: se ci sono meno ELSE dei THEN allora ogni ELSE è accoppiato con il THEN più vicino. Infatti nell'esempio precedente IF B=20 THEN è accoppiato con ELSE 40 mentre IF A=18 THEN con ELSE 80.

Ed ora, dopo una lunga parentesi dedicata all'istruzione IF, ritorniamo all'argomento principale, ovvero gli operatori logici. Come abbiamo già visto nella scorsa puntata e come poi ricordato all'inizio di questo articolo, gli operatori logici vengono utilizzati soprattutto con l'istruzione IF, in particolare per correlare più condizioni in un'unica istruzione. Supponiamo infatti di risolvere il seguente problema: dobbiamo controllare con il computer l'accensione di una caldaia, la cui variabile di controllo è CALDAIA$, a seconda della temperatura rappresentata dalla variabile GRADI e dall'umidità presente nell'aria indicata dalla variabile UMIDO. Cioè CALDAIA$="ACCESA" se GRADI<18 e UMIDO>80. Il problema si può risolvere in due modi: con e senza gli operatori logici. Semplicemente utilizzando istruzioni IF scriveremo:

```
IF GRADI<18 THEN IF UMI-
DO>80 THEN CALDAIA$="AC-
CESA"
```

altrimenti con l'uso degli operatori logici:

```
IF GRADI<18 AND UMIDO>80
THEN CALDAIA$="ACCESA"
```

la cui forma sintattica è decisamente più chiara e comprensibile.

Ovviamente le correlazioni possono essere più di due, dipende dalla complessità del problema e quindi dal numero delle variabili in gioco, ciò vuol dire poter usare più operatori booleani e tra di loro anche differenti. Spieghiamo subito con un esempio quanto appena detto. Data la variabile NUMERO il nostro programma dovrà eseguire la subroutine alla linea 5000 solo se NUMERO è un numero pari ed un multiplo di tre oppure pari a 5. La linea IF/THEN che tiene conto di tutte queste condizioni sarà scritta come segue:

```
IF NUMERO/2=INT(NUMERO/2)
AND NUMERO/3=INT(NUME
RO/3) OR (NUMERO=5) THEN
GOSUB 5000
```

L'espressione NUMERO/2=INT (NUMERO/2) verifica che NUMERO sia pari mentre la condizione NUMERO/3=INT(NUMERO/3) controlla che NUMERO sia un multiplo di tre. Le due espressioni sono correlate dall'operatore AND ed unite logicamente dall'operatore OR alla condizione NUMERO=5.

Anche se nella scorsa puntata abbiamo descritto i sei operatori booleani disponibili sull'MSX, dobbiamo riconoscere che i più usati nelle istruzioni di test condizionali sono solo tre: AND, OR e NOT.

Gli operatori logici vengono usati per testare più condizioni in un'unica istruzione IF/THEN ma risultano allo stesso modo utili nelle istruzioni di assegnazione: applicazione raramente usata, forse perché da pochi conosciuta, ma assai potente. Come ormai è di normale prassi chiariamo subito il concetto con un esempio.

Supponiamo di voler incrementare la variabile A solo se le rispettive variabili B e C hanno un valore maggiore di 10. Con un'istruzione condizionale il problema si risolve semplicemente:

```
IF B>10 AND C>10 THEN A=A+1
```

e così pure con una istruzione d'assegnazione piuttosto particolare:

```
A=A—(B>10 AND C>10)
```

Innanzitutto dobbiamo far presente che l'MSX dà come risultato, in seguito ad una condizione vera, —1 e 0 per una condizione falsa. Così se (B>10

AND C>10) è vero il risultato sarà —1 ciò vuol dire scrivere A=A—(—1) e quindi incrementare il valore di A.

In ultimo vediamo quando applicare gli operatori booleani non come "giuntori" di espressioni condizionali ma come istruzioni binarie, cioé operanti su di una maschera a più bit le cui applicazioni sono ben diverse da quelle studiate fino ad ora.

IL PIXEL, OVVERO IL PUNTO

Le informazioni espresse in forma binaria hanno poco significato per il programmatore di linguaggi evoluti quali il Basic ma per alcuni casi si rivelano l'unica struttura di rappresentazione significativa. È il caso del pixel ovvero la più piccola informazione grafica, in altri termini il punto. Un grafico o un qualsiasi disegno artistico è rappresentato sullo schermo da un insieme di punti; questi sono memorizzati in una zona di memoria dedicata esclusivamente al video. Tale memoria è organizzata in modo tale che ogni byte contiene lo stato di ogni pixel, acceso o spento. Cioè se il bit è a 1 vuole dire che il punto corrispondente sullo schermo è visualizzato. Volendo realizzare un inverse del video è sufficiente settare i punti spenti e resettare quelli accesi, ciò si realizza semplicemente usando l'operatore NOT il quale, come già citato, nega il valore binario 0 in 1 e viceversa. Vediamo quindi le poche istruzioni per realizzare il negativo dello schermo:

```
100 FOR B%=0 TO 6143
110 A%=VPEEK (B%)
120 A%=NOT A%
130 VPOKE B%,A%
140 NEXT B%
```

Dall'indirizzo 0 all'indirizzo 6143 della memoria VRAM (Video-RAM) sono contenuti i dati di tutti i punti dello schermo grafico 2 dell'MSX (256x192 pixel). Ponendo in A% il valore di otto punti per volta (un byte), negando il contenuto del byte (operatore NOT) e successivamente riscrivendolo nella sua posizione iniziale (linea 130) si ottiene l'utile funzione inverse.

E con quest'ultimo esempio chiudiamo l'argomento degli operatori booleani che come abbiamo visto si prestano molto bene nella risoluzione di diversi problemi ma che purtroppo poche volte vengono utilizzati anche dall'esperto programmatore.

■

SOFTWARE

WORD PROCESSOR

ORGANIZZAZIONE GENERALE DI UN PROGRAMMA DI ELABORAZIONE TESTI. COME VISUALIZZARE I DATI: LA GESTIONE DEL VIDEO

Seconda puntata

di EMANUELE DASSI

Eccoci alla seconda puntata dedicata alla programmazione del WP, il programma per la stesura dei testi con il computer.

Come abbiamo già detto la scorsa volta un WP si divide in tre parti:
1. Input da tastiera.
2. Output su video.
3. Operazioni sul testo.

In questa puntata tratteremo il punto 2: output su video.

Con il termine "output su video" si intende l'insieme delle operazioni atte a modificare il display file in modo tale da tenerlo sempre aggiornato in seguito alle operazioni decise dall'utente. Vediamo quali sono queste operazioni e come il programma deve comportarsi in merito.

Dobbiamo prima di tutto precisare che il sottoprogramma di output su video non viene chiamato subito dopo quello di input da tastiera ma dopo che siano state chiamate altre subroutine, quelle che modificano l'impostazione del testo ovvero ne migliorano l'estetica: centratura, marginatura, ecc.

Consideriamo quindi che la subroutine di output su video sia attivata soltanto quando nel buffer del testo quest'ultimo risulti già pronto alla stampa, cioè centrato, spaziato correttamente e così via.

Le operazioni che deve assolvere la subroutine di output su video sono:
1 visualizzazione del testo tenendo conto della posizione del cursore;
2 tenendo conto dello spazio che riserva il video, calcolare e visualizzare più caratteri possibili;
3 riservare parte del video per la visualizzazione delle informazioni sulla posizione del cursore;
4 evidenziare la posizione del cursore;
5 copiare le informazioni del buffer al video nel minor tempo possibile.

Queste sono le prestazioni dell'output su video. Lo sviluppo del punto 5 implica l'utilizzo di un linguaggio differente dal Basic, più veloce. Come vedremo più avanti la soluzione sarà l'adozione del linguaggio macchina. Ed ora veniamo al listato.

IL LISTATO

Come si può notare dal listato n. 1 i cinque punti descritti sopra sono sviluppati in poche linee delle quali alcune riservate solo al caricamento del linguaggio macchina. È proprio da quest'ultimo punto che vogliamo iniziare a descrivere l'intero sottoprogramma.

Per trasferire velocemente i dati del buffer (la zona di memoria dove sono stati memorizzati i valori ASCII dei caratteri del testo) al video abbiamo utilizzato una routine del sistema operativo la cui chiamata è all'indirizzo 005CH. I dati di input di questa routine sono:
- il registro BC contenente il numero di byte da trasferire
- il registro HL contenente l'indirizzo da dove prelevare i dati da copiare
- il registro DE contenente l'indirizzo dove copiare i dati.

La funzione svolta da tale routine è quella di copiare tanti byte quanti quelli indicati dal valore del registro BC prelevandoli a partire dall'indirizzo HL e copiandoli a partire dall'indirizzo contenuto in DE.

La routine in linguaggio macchina, caricata in memoria a partire dall'indirizzo 49000, è scritta nel DATA alla linea 2900 mentre le istruzioni che si occupano del suo caricamento in memoria sono alla linea 2910. Tale routine predispone i valori di input (registri BC, HL e DE), chiama la routine del sistema operativo e, una volta eseguita, ritorna al Basic con l'istruzione RET. Riportiamo qui di seguito il breve listato in ASSEMBLER ed i relativi codici macchina dell'intero pro-

```
2000 IF LM<IL THEN IL=IL-LP%:GOTO 2000 '
SCROLL VERSO L'ALTO
2010 IF LM-IL>880 THEN IL=IL+LP%:GOTO 20
10
2020 VA=IL:GOSUB 2970' SETTA IL REG. HL
2030 VA=LM-IL+1
2040 IF VA<880 AND LT%+49999!>LM THEN IF
 LT%+49999!-LM>880-VA THEN VA=880 ELSE V
A=VA+(LT%+49999!-LM)
2050 GOSUB 2950' SETTA IL REG. BC
2060 CLS:NN=USR(0)'AGGIORNA IL VIDEO
2070 LOCATE 0,23:PRINT "PAGINA:";INT((LM
-50000!)/(LP%*NR%))+1;
2080 LY%=INT((LM-IL)/LP%):LX%=LM-IL-LY%*
LP%
2090 LOCATE LX%,LY%,1:VPOKE 0,PEEK(IL)
2100 RETURN
2900 DATA 01,00,00,21,00,00,11,00,00,CD,
5C,00,C9
2910 FOR N%=1 TO 13:READ A$:POKE 48999!+
N%,VAL("&H"+A$):NEXT N%:RETURN
2950 POKE 49001!,VA-INT(VA/256)*256:POKE
 49002!,INT(VA/256)
2960 RETURN
2970 POKE 49004!,VA-INT(VA/256)*256:POKE
 49005!,INT(VA/256)
2980 RETURN
```

La routine in linguaggio macchina, caricata in memoria a partire dall'indirizzo 49000, è scritta nel DATA alla linea 2900 mentre le istruzioni che si occupano del suo caricamento in memoria sono alla linea 2910.

PHILIPS COURTESY

```
10 CLEAR 200,48999!
20 GOSUB 2900 'CARICA L/M
30 DEFUSR=49000!:KEYOFF
40 IL=50000!:LM=IL:LP%=40:NR%=30:LT%=0
50 WIDTH 40
60 GOSUB 1000:BEEP:GOSUB 2000:GOTO 60
```

Alla linea 10 l'istruzione CLEAR 200,48999 definisce il limite massimo di memoria riservata al Basic all'indirizzo 48999.

grammino (13 bytes) di copia veloce del buffer nella memoria video:

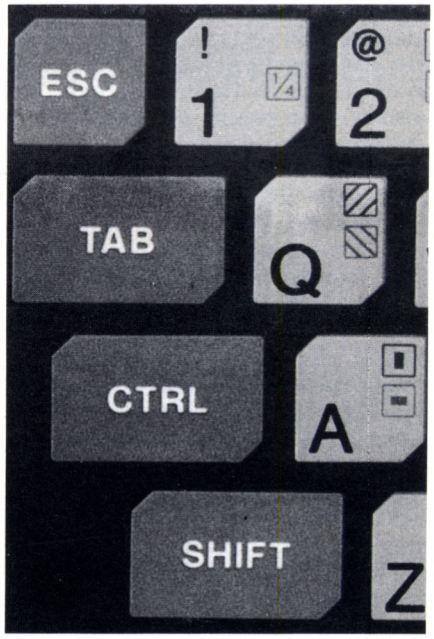
```
01  00  00   LD BC,0000
21  00  00   LD HL,0000
11  00  00   LD DE,0000
CD  5C  00   CALL 005C
C9           RET
```

Per il registro DE il valore da caricarvi è sempre 0 perché, dovendo lavorare in SCREEN 0 e con 40 caratteri per riga (WIDTH 40), la prima locazione di memoria dove copiare i dati è all'indirizzo 0. Per il registro BC e HL invece i valori vengono modificati dalle subroutine alla linea 2950 ed alla linea 2970.

LE ISTRUZIONI UTILIZZATE

Veniamo ora alla descrizione del listato Basic. Per poter funzionare abbiamo definito all'inizio dell'intero programma di word processing le linee del listato 2 delle quali facciamo un breve commento prima di passare alla descrizione della subroutine in questione.

Alla linea 10 l'istruzione CLEAR 200,48999 definisce il limite massimo di memoria riservata al Basic all'indirizzo 48999. La linea seguente esegue la chiamata del sottoprogramma alla linea 2900 dove viene caricato il linguaggio macchina come descritto prima. Segue poi la definizione dell'inizio della routine in l/m all'indirizzo



49000 e viene cancellata dallo schermo la linea dei tasti funzione (KEYOFF). Ora facciamo particolare attenzione alla linea 40 nella quale vengono definite alcune variabili nuove ed utilizzate poi dalla routine di stampa. La variabile IL, inizializzata a 50000, indica l'indirizzo del buffer da dove incominciare a prelevare il testo da visualizzare sul video. In altre parole il carattere in alto a sinistra del video è memorizzato alla locazione puntata da IL. La variabile LM inizializzata anch'essa a 50000 e descritta nella scorsa puntata, indica la locazione di memoria dove memorizzare il carattere digitato da tastiera. LP%=40 definisce la larghezza della pagina del testo, nel nostro caso 40 caratteri; NR%=30 invece rappresenta il numero di righe per ogni pagina ed infine LT%, già trovata nella scorsa puntata, contiene la lunghezza del testo espressa in numero di caratteri.

La linea 50 predispone il video a 40 colonne mentre la linea 60 attiva alternativamente le due subroutine del WP sviluppate fino a questa seconda puntata.

IL SOTTOPROGRAMMA DI OUTPUT SU VIDEO

Dato che il sottoprogramma deve sempre visualizzare una parte del testo nella quale si trova il cursore, inizialmente (linea 2000) si controlla che la posizione di questo, indicata dalla variabile LM, si trovi all'interno del testo visualizzato tenendo presente il valore della variabile IL. Se LM risulta minore di IL vuol dire che il cursore si trova nella parte "superiore" del testo visualizzato quindi si decrementa il valore di IL tanto quanto la larghezza della pagina del testo, ovvero si esegue uno scroll del testo verso l'"alto" fino all'apparire del cursore, cioè LM>=IL.

Allo stesso modo (linea 2010) bisogna controllare che il cursore non si trovi "sotto" la parte del testo visualizzata. Per controllare questa condizione bisogna confrontare il valore di LM-IL con 880. Il valore 880 rappresenta il numero massimo di caratteri visualizzabili sullo schermo, pari a 22 righe da 40 colonne anche se è possibile visualizzare 24 righe, ma si riservano le ultime 2 righe del video per la visualizzazione di alcuni messaggi come vedremo più avanti. Come dicevamo prima, se LM-IL è maggiore di 880 allora il cursore non risulta essere visualizzato sullo schermo perché non rientra nella parte del testo presente sul video. Ecco allora rendersi necessario uno scroll verso il basso del video; ciò si realizza come per lo scroll verso l'alto con la sola differenza che anziché decrementare il valore di IL lo si incrementa sempre di un valore pari a LP% tante volte quanto basta a far "rientrare" nel video il cursore.

LA LUNGHEZZA DEL TESTO

Dopo aver trovato il valore corretto di IL lo si memorizza nella routine in l/m come valore del registro HL passandolo nella variabile VA alla subroutine della linea 2970 (chiamata alla linea 2020). Dopodiché bisogna determinare il valore da inserire nel registro BC, ovvero la lunghezza del testo da visualizzare. Questa è calcolata alla linea 2030 tenendo presente come estremi il valore di IL e di LM. La lunghezza così trovata però non è quella definitiva ma viene modificata, eventualmente, alla linea 2040 in una complessa struttura IF/THEN/IF/THEN/ELSE che ricalcola il valore di VA (dove questa volta indica la lunghezza del testo da visualizzare) in modo da visualizzare più testo possibile compatibilmente allo spazio disponibile sul video. Infine, alla linea 2050, la lunghezza del testo da visualizzare memorizzata in VA viene inserita nella routine in l/m come valore del registro BC tramite la subroutine alla linea 2950. A questo punto, dopo aver definito gli estremi del testo da

visualizzare e preparata la routine in l/m che esegue il trasporto del testo sul video, si esegue la chiamata della stessa con la funzione USR (linea 2060). Segue alla linea 2070 la stampa della pagina corrente; alla linea 2080 si calcolano in base al valore di LM le coordinate del cursore nel video (LY% e LX%) e lo si visualizza con l'istruzione LOCATE LX%, LY%,1. Il terzo parametro dell'istruzione LOCATE, che nel nostro caso è 1, serve a visualizzare o meno il quadratino del cursore, se è 1 lo visualizza se invece è 0 posiziona la stampa a quelle coordinate senza visualizzare il cursore. L'istruzione VPOKE 0, PEEK (IL) successiva a LOCATE alla linea 2090 serve a ripristinare il primo carattere in alto a destra dello schermo. Infine l'istruzione RETURN alla linea 2100 completa l'esecuzione dell'intera subroutine.

Nella prossima puntata continueremo con il punto 3 del wp: operazioni sul testo, insieme di più prestazioni dalle quali dipende la validità di un programma di elaborazione testi.

LETTERE

DUPLI PROBLEMS

Ho riscontrato alcune anomalie nell'uso del programma DUPLI pubblicato nel numero 7 della vostra rivista. Ecco quanto ho riscontrato:
1) nelle istruzioni date da video si parla di stampa su carta degli indirizzi di memoria del programma da copiare: nel listato non è prevista questa funzione;
2) non è chiaro se una volta letti gli indirizzi bisogna riavvolgere il nastro;
3) dopo la duplicazione si scopre che non è stato trasferito il nome del programma ma una semplice a.

Sul n. 10 della vostra rivista, in risposta ad un altro lettore, affermate che il programma funziona e se non si riesce ad utilizzarlo eseguire il comando CLEAR 50, inizio e poi BLOAD "CAS:". Tutto bene se non si usano indirizzamenti inferiori a 33517. Un'altra domanda: è possibile che un programma possa avere tre indirizzi di memoria uguali? Inoltre, con quale procedura è possibile visualizzare i listati scritti in linguaggio macchina?

Mauro Riegler - Milano

Vediamo di rispondere ai quesiti che ci hai posto:
1) La scritta che appare su video consiglia, più che altro, di scrivere i valori ricavati dall'header su carta.
2) Una volta letti gli indirizzi, bisogna riavvolgere il nastro da copiare e poi procedere alla fase di copy.
3) Sì, il programma viene sempre copiato con il semplice nome "a" (vds. linea 596).

Il programma era una delle prime versioni di copiatori. È per questo che, per blocchi in L/M con indirizzi inferiori a 33517 e superiori a 62336, il copiatore non funziona.

È probabile che un programma abbia 3 indirizzi uguali. Il caso però è unico: un programma di un solo byte (assai improbabile!).

Per analizzare i programmi scritti in L/M bisogna utilizzare un disassemblatore, ovvero un programma che decodifica i valori numerici in mnemonici ASSEMBLER.

STAMPA VELOCE

Ho da chiedervi le soluzioni a tre miei problemi:
1) sono in grado di far muovere uno sprite in l.m. ma non so come inserire il tutto in un video game. Infatti il movimento dello sprite è determinato da un ciclo che quando viene eseguito blocca il resto del programma Basic. Come posso fare perché questo non avvenga?
2) Ho deciso di usare il programma di stampa veloce in screen 2 pubblicato sul n. 7 della vostra rivista. Dovendola usare in unione ad un altro programma in l.m. come devo operare perché tale routine non introduca bytes oltre l'indirizzo 49980?
3) Come posso fare per inserire scritte suo bordi?

Guido Iacono - Livorno

Certo che con le domande non scherzi! Ecco la risposta alle ultime da te poste:
1) Il movimento dello sprite può essere gestito da una istruzione del tipo:
10 ON INTERVAL=50 GOSUB 1000
20 INTERVAL ON
ovvero ogni secondo il programma salta alla linea 1000 dove si trova il ciclo gestione sprite. La linea 1000 conterrà l'istruzione /i per lo sprite e terminerà con l'istruzione RETURN.
2) Modificare il programma di scrittura veloce in SCREEN 2 non è cosa facile, perché in realtà i programmi sono due. Bisogna spostare puntatori vari. Consigliamo di modificare invece il tuo programma in L/M, visto che lo stai progettando.
3) Impossibile darti una risposta semplice sul quesito per agire sui bordi. E poi... non ti sembra di chiedere l'impossibile al tuo MSX?

RAM BASIC

Ho un Philips 8000 che a suo tempo ho espanso con una cartuccia da 16K RAM. Adesso questa memoria non basta più e vorrei comperare o aggiungere una cartuccia più potente. Da quanto ho letto, però, sembrerebbe che abbia già raggiunto il massimo. Cioè più di 28815 bytes non posso utilizzare. È vero? Un'altra cosa: nel numero 2 della vostra rivista c'è un ottimo programma l'"Agenda telefonica". Ho un problema ad usarlo: nonostante modifichi il valore delle variabili come da istruzioni, dopo aver inserito 12 o 13 nominativi il programma si blocca dando l'errore "out of string space" in 220. Si può fare qualcosa?

Carlo Fiorani - Roma

Se si programma in basic, più di 32K RAM di sistema (cioè 28815 byte free) non si possono utilizzare. Quindi ti sconsigliamo di espandere ulteriormente la memoria RAM del tuo computer.

AGENDA TELEFONICA: se l'errore è "OUT OF STRING SPACE" in linea 220, è sufficiente cercare nel programma l'istruzione CLEAR n (dove n è un numero) ed aumentare il valore di n.

ERRATA CORRIGE

Ci scusiamo con i lettori ma abbiamo riscontrato alcuni errori nel programma GOLF 18 HOLE (vedi MSX Computer Magazine n. 12). Le linee da modificare sono le seguenti:

```
1530 IF POINT(X1,I)=12 THEN FOR F=1 TO 14 : SOUND
     9,15:SOUND 3,6 : SOUND 9,0 : NEXT F:Y=INT(I) :
     X=X1:GOTO 4000
2120 IF STRIG(0)<>1 THEN 2120 ELSE FOR I=1 TO 200: NEXT I
2130 IF STRIG(0)=—1 THEN 2150
```

SPLENDIDO

UN LOOK
COLORATO
PER LA TUA
CASSETTA

RITAGLIA
LUNGO
IL BORDO
SEGNATO
IN NERO
E
PIEGA
SEGUENDO
IL
TRATTEGGIO
INDICATO

PERSONALIZZA
LA
CASSETTA
CON IL
TUO NOME

MSX COMPUTER
MAGAZINE
PER LA TUA
SOFT-TECA

RIVISTA PROGRAMMI PER MSX

MSX Computer Magazine 14

Per caricare digitare CLOAD

SOMMARIO

LATO A:

- FORMULA 1
- EAT
- ROULETTE
- CORSO DATTILO
- MSX SOUND ADVENTURE

LATO B:

- CASTLE
- GALAXI
- FORZA 4
- CORSO DI INGLESE
- RAM DISK

QUESTA CASSETTA
È DI

NOME

COGNOME

VIA N.

CITTÀ

MSX COMPUTER MAGAZINE