MSX DISK

OTTOBRE 1990
ANNO III

N 20

- 1942
- Circus
- Skooter
- Boxing
- Unicopy
- DBase
- Finance Plus 3

giochi
news
utilità

£ 14.000

L'UNICA DISK MAGAZINE
DEDICATA ALLO STANDARD MSX

MSX DISK - REGISTRAZIONE N. 775 DEL 20/11/87
PRESSO IL TRIBUNALE DI MILANO - GRUPPO
EDITORIALE INTERNATIONAL EDUCATION
S.R.L., VIALE FAMAGOSTA 75, 20142 MILANO -
DIRETTORE RESPONSABILE: GRAHAM JO-
HNSON - DISTRIBUZIONE: MESSAGGERIE
PERIODICI S.p.A. ADERENTE A.D.N., V.LE
FAMAGOSTA, 75 - MILANO - TEL. 84.67.545

SOMMARIO

2 Sommario – Sul disco
Caricamento – Avvertenze
3 Editoriale – Abbonamenti
4 Finance Plus 3
5 DBase
6 1942
7 Circus – Skooter
8 Boxing – Unicopy
9 News
10 Basic (Parte V[a])
15 Il Pascal (Parte I[a])
19 Computer & Programmazione (Parte III[a])
21 Telematica ed educazione
22 Dentro l'MSX (Parte II[a])
25 Il libro del mese
26 Posta

SUL DISCO

1 1942
2 Circus
3 Skooter
4 Boxing
5 Unicopy
6 DBase
7 Finance Plus 3

CARICAMENTO

A computer spento inserite il disco nel driver. Tenendo premuto il tasto CTRL accendete il computer e tenetelo inserito fino alla comparsa sul video del sommario. Per caricare un programma premete il numero corrispondente (dall'1 all'8). Il caricamento avverrà automaticamente.

AVVERTENZE

Questo disco è stato registrato con cura e con i più alti standard di qualità. Leggete con attenzione le istruzioni per il caricamento. Nel caso in cui, per una ragione qualsiasi, trovaste difficoltà nel caricare i programmi, telefonate alla nostra redazione al numero (02) 89502256 oppure spedite il disco al seguente indirizzo:

Gruppo Editoriale International Education srl - viale Famagosta, 75 - 20142 Milano.

Testeremo il prodotto e, nel caso, lo sostituiremo con uno nuovo senza aggiunta di costi supplementari.

EDITORIALE

Credo che questo numero, se ora Msx Disk esce regolarmente, sia quello di Ottobre, l'autunno, un pò di tristezza, rimpianto per l'ultima estate, bei ricordi.
Problemi. Problemi. Problemi.
Risolti. Risolti. Risolti?
Sì, forse sì. Ora credo che possiamo finalmente uscire regolarmente, forse un sollievo per chi ci segue sempre, per chi crede in noi, nell'MSX. Per chi continua a credere in uno standard affondato dal mercato.
Uno standard vivo.
Errori, sì, errori, ma siamo uomini e non macchine e visto che anche noi crediamo nell'MSX dovete aiutarci a crescere accettando anche i nostri errori e consigliandoci. Proprio come le lettere e le richieste che ci mandate e che, lentamente, sono sempre di più.
Un inizio strano ma pur sempre un inizio, ma ora passiamo al disco. Oltre al nuovo corso di programmazione che parte da questo numero, le novità stanno anche nelle utilità. Infatti, da questo numero parte una serie di originali programmi didattico-finanziari che vi aiuteranno sia nel campo della gestione finanziaria che nell'apprendimento di utili tecniche di programmazione in Basic.
Sempre nelle utilità un programma per gestire le librerie e un programma richiestoci da molti di voi: WORD, uno dei più famosi e potenti word processor.
Ma facciamo un pò di anteprima: dal prossimo Msx Disk un nuovo corso dedicato al linguaggio C e, necessariamente, un compilatore C su disco; le prove del vostro software, dei programmi che ci inviate.
Questo è tutto, non dimenticate di leggere la posta che risponde alle richieste dei lettori e che può spiegarvi perché trovate o non trovate i programmi eventualmente richiesti. Buon Lavoro...

LA REDAZIONE

ABBONAMENTI

FINANCE PLUS 3

Continuando la serie Finance Plus vi ricordo quello che è stato già detto in precedenza: questi programmi hanno il duplice scopo di fornire un aiuto nell'ambito finanziario e offrire uno spunto didattico ai principianti che entrano nel mondo della programmazione in Basic. Per utilizzare il programma sarà sufficiente premere il tasto [7] quando compare il menù principale di Msx Disk. Il file 7.BAS è il file sorgente pronto per essere esaminato e rielaborato. Questa volta il programma è dedicato all'Acquisto-Leasing e all'Analisi degli investimenti comuni. Il programma è, come le volte scorse, diviso in tre parti e offre un menù principale con le seguenti opzioni :

1 - ACQUISTO/LEASING;
2 - ANALISI DEGLI INVESTIMENTI COMUNI;
3 - FINE LAVORO.

Nel file sorgente potrete trovare che la prima opzione viene espletata dalla parte che va dalla linea 1000 alla linea 1660 mentre l'analisi degli investimenti comuni è compresa tra la linea 2000 e la linea 2980. Come sempre, dalla linea 3000 in poi trovano posto le routine di servizio. Terminando l'uso del programma tramite la terza opzione del menù principale - che una volta selezionata richiede una conferma S/N l'esecuzione ritornerà al menù principale di Msx Disk 20.

ACQUISTO/LEASING

Questo algoritmo calcola il costo relativo a due delle possibili scelte : acquisto o leasing. La differenza tra i due costi permetterà di stabilire quale sia la scelta più vantaggiosa. Si intende che il tempo sul quale deve essere distribuita la spesa nel caso dell'acquisto, verrà preso come base del calcolo del costo che deriverebbe dalla scelta del leasing. Utilizzando questa opzione dovrete inserire i seguenti dati : prezzo della merce in questione, tasso di interesse, durata del prestito, valore di realizzo della merce a termine prestito, percentuale di tasse, ammontare annuale del mutuo, ammontare annuale del leasing. Il programma fornisce il costo relativo alla scelta dell'acquisto, il costo relativo alla scelta del leasing ed infine la differenza tra questi due valori. Questo algoritmo può rivelarsi istruttivo dal punto di vista finanziario in quanto pone l'accento su fattori decisionali che potreste aver trascurato, ma ovviamente non è fatto per sostituire il vostro giudizio. La vostra decisione dovrà essere guidata da fattori come la pianificazione degli investimenti e la durata del prestito/leasing. In generale, l'ammortamento ed il valore di realizzo aumentano la convenienza dell'acquisto; al contrario se una merce è soggetta a rapida obsolescenza, il leasing si rivelerà la scelta più conveniente. Oltre a questo, il programma è effettivamente una versione ampiamente modificata dell'algoritmo "Valore Attuale Di Un Investimento", pubblicata in Finance Plus 2. Per questo potrete trovare istruttivo vedere come si possa modificare un programma allo scopo di adattarlo alle proprie esigenze specifiche.

ESEMPIO - La società 'Terapia Tapioca Spa' ha bisogno di un furgoncino per uso quotidiano e si sta considerando la possibilità di comprarne uno al prezzo di 6.000.000 di lire. Il valore di realizzo al termine di quattro anni è stimato in 2.000.000 di lire. La banca presta 6.000.000 di lire al 16% di interesse da rifondere in quattro rate da 2.145.000 lire. Il leasing costerebbe 2.000.000 di lire l'anno. Le tasse ammontano al 40% e l'ammortamento consiste in 1.000.000 all'anno. Qual'è attualmente il costo dell'acquisto? Qual'è il costo del leasing? Quale scelta deve fare la società?

Risposta : il costo del prestito è di 3.011.900 lire (all'anno). Il costo del leasing è di 3.357.820 lire. La società dovrebbe comprare l'automezzo.

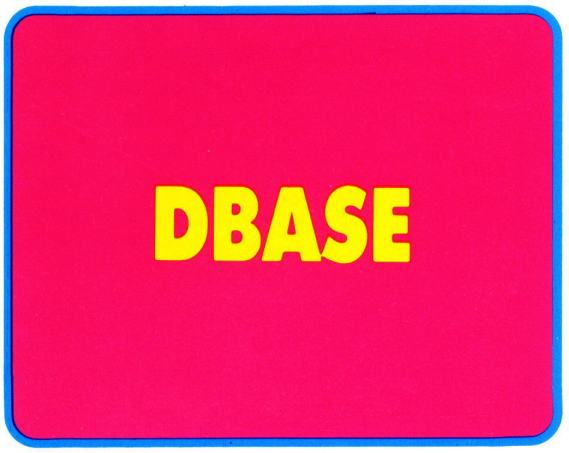
ANALISI DEGLI INVESTIMENTI COMUNI

Questo secondo algoritmo valuta il risparmio in tasse e gli introiti netti di una persona che investa all'interno di un gruppo di finanziatori. Il programma considera l'aliquota fiscale del finanziatore considerato, la proporzione in cui si è impegnato nell'investimento iniziale, la partecipazione agli utili, le entrate tassabili e i crediti fiscali. Per usare il programma inserite la durate dell'analisi in anni e il primo anno di investimento comune. Per ogni anno inserire le entrate comuni seguite dalle entrate tassabili. Inserire l'anno (1,2 e così via) e l'investimento totale del gruppo per quell'anno. Inserire poi l'anno e l'ammontare dell'investimento o di altri crediti fiscali (da inserirsi come numeri negativi), o il ritiro di crediti (da inserirsi come quantità positive). Fatto questo, inserite le percentuali del finanziatore: percentuale sull'investimento totale, sulle entrate, sulle entrate tassabili e sui crediti. L'ultimo valore richiesto è l'aliquota fiscale del finanziatore, da inserirsi in percentuale. Il programma stampa poi la sua analisi mostrando al finanziatore il proprio investimento iniziale, le entrate, le entrate tassabili, il risparmio in tasse (i risparmi sono indicati come quantità negative, le tasse pagate sono indicate come quantità positive), le entrate nette a fine anno e le entrate cumulative. Potete ripetere l'analisi per aliquote differenti quando questo vi viene chiesto dal programma (tutti gli altri parametri dell'investimento rimangono tali e quali). Inserite una aliquota di 999 per ridefinire le percentuali nell'investimento. Inserite una percentuale di partecipazione all'investimento di 999 per concludere l'uso dell'opzione.

ESEMPIO - Consideriamo il seguente investimento comune: una proprietà acquistata con un investimento iniziale di £35.000.000 genera nei quattro anni a venire un'entrata di £4.500.000, un'entrata di £5.200.000 nei quattro anni che seguono ed un'entrata di £5.500.000 per ognuno dei rimanenti cinque anni. Per l'investimento è prevista una riduzione fiscale di £3.500.000 per il primo anno. Le entrate tassabili sono di –£3.800.000 il primo anno e aumentano di £1.100.000 all'anno per tutta la durata dell'investimento. Il finanziatore ha un'aliquota fiscale del 55% e contribuisce per il 30% dell'investimento iniziale. Come verrà usato il programma?

Risposta: alla fine della proiezione le entrate cumulative del finanziatore sono di £4.432.000 e l'investimento è al riparo da tasse fino alla fine del periodo, quando si dovranno pagare 109 dollari.

DBASE



Come il WordStar propostovi negli scorsi numeri, anche questo è uno dei programmi più famosi e venduti nel Mondo: dBASE II. Il programma è un potente DBMS destinato alla gestione anche di complessi database, o archivi.
Il programma richiede un Msx 2 poiché funziona su 80 colonne.
Su disco non abbiamo potuto inserire il file di Help che, se possibile, verrà inserito in uno dei prossimi numeri di Msx Disk.
Il programma richiede un vero e proprio manuale che non siamo in grado di darvi, ma in ogni caso cominceremo, dal prossimo numero, un vero e proprio corso all'uso di questo programma. Per ora dovrete accontentarvi di una breve introduzione ai comandi più importanti, altrimenti potrete procurarvi un manuale dedicato al dBASE II. Il titolo del libro e': MANUALE DEL DBASE II -di A. Pipitone & M. Pipitone, Edito da Gruppo Editoriale Jackson. E' composto da 224 pagine e costa 24.000 lire. Il codice è 546P.
Ma ecco una breve lista dei comandi più importanti:

CREATE
Crea una struttura di file archivio

USE
Apre un archivio già creato

APPEND
Aggiunge record al file aperto

EDIT
Permette di Analizzare/Modificare il record indicato

APPEND BLANK
Aggiunge un record vuoto

MODIFY STRU
Modifica la struttura dell'archivio creato

LIST
Lista il record indicato

DISPLAY
Come LIST ma fà una pausa alla fine della pagina

LIST ALL
Mostra tutti i record

DISPLAY ALL
Come LIST ALL ma con la pausa di fine pagina

DELETE RECORD
Cancella il record indicato

DELETE ALL
Cancella tutti i record

TO PRINT
Aggiungendolo al LIST o al DISPLAY permette di mandare i dati alla stampante

PACK
Elimina fisicamente i record cancellati

ZAP
Azzera l'archivio

LOCATE FOR ‹campo› = ‹dato›
Effettua una ricerca sequenziale del record in cui il campo verifica la condizione data

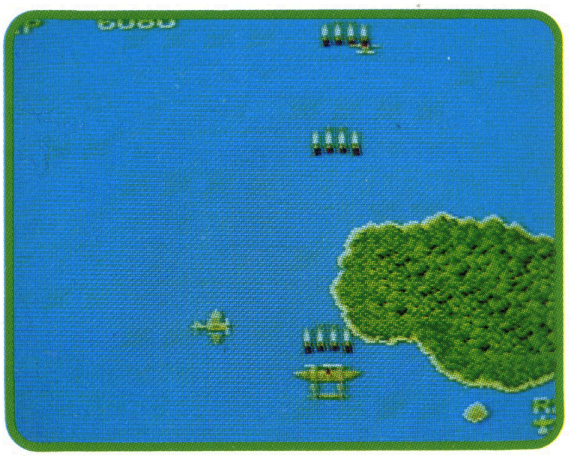
CONTINUE
Continua la ricerca dal record in cui si era interrotta

QUIT
Fine lavoro con dBASE II

HELP
Aiuto (Solo quando c'è il file di aiuto)

Vi ricordiamo che è bene usare un'altro disco come archivio in quanto potreste danneggiare il disco di Msx Disk e comunque non avreste abbastanza spazio sul disco.
Buon Lavoro!

1942

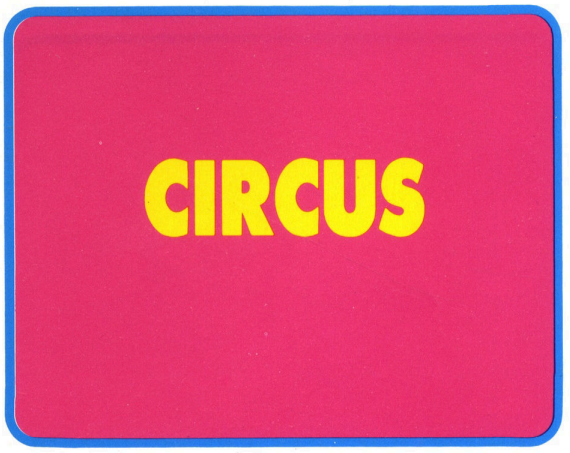


Direttamente dalla sala giochi ecco uno shoot'em up che ha riscosso un grandissimo successo.
Il gioco è ambientato nell'Oceano Pacifico durante la Seconda Guerra Mondiale, nel 1942, e vede fronteggiarsi giapponesi e americani in mare e in acqua.
E' un gioco a scorrimento (SCROLL) verticale in cui sarete ai comandi di un caccia P47 Lightning che, partendo da una superpotente portaerei, dovrà fronteggiare tutto solo i caccia e bombardieri nemici.
Il gioco è articolato in 32 stages che comprendono decollo, combattimento e atterraggio.
In volo dovrete affrontare una miriade di caccia-kamikaze nemici che cercheranno di eliminarvi in tutti i modi.
Periodicamente incontrerete anche dei bombardieri di diverse dimensioni, che rappresentano il vostro vero obbiettivo.
Distruggerli è difficile ma danno un sacco di punti. In determinati punti della rotta che seguirete verrete attaccati da uno stormo di aerei rossi: distruggendoli vedrete comparire la scritta POW che, se raccolta, aumenterà notevolmente il vostro volume di fuoco, rendendovi la vita molto più facile.
In volo potrete spostarvi nelle quattro direzioni standard e sparare con i soliti tasti, cioè Cursori e Spazio, ma alcune volte sarà davvero difficile evitare i caccia troppo numerosi così potrete fare affidamento sul Looping.
Entrando in Looping il vostro aereo compirà un giro della morte completo cambiando immediatamente quota di volo e poi tornando alla posizione iniziale.
I looping sono limitati, tre per ogni missione, e quando non vengono utilizzati danno un bonus in punti al completamento di ogni stage. Per usare i looping da tastiera dovrete usare il tasto per i caratteri grafici, cioè [GRAPH], mentre per avere il looping usando il joystick dovrete utilizzarne uno fatto apposta per lo standard Msx che preveda due differenti tasti di fuoco.
Nella fase di caricamento del gioco vi verrà chiesto che tipo di video usate e dovrete premere [1] per il Monitor e [2] per la Televisione.
Una volta completato il caricamento sullo schermo comparirà il menù principale seguito dalla dimostrazione.
Interrotta la dimostrazione premendo la barra spaziatrice, potrete selezionare l'opzione desiderata tramite i cursori o la leva del joystick e poi far iniziare il gioco premendo ancora lo spazio oppure il primo tasto di fuoco.
Le opzioni disponibili inizialmente sono due: 1 Giocatore e 2 Giocatori.
A queste opzioni se ne aggiunge un'altra al termine di ogni partita: Continue, che permette di continuare la partita appena terminata.
Nello schermo di gioco troverete il punteggio in alto, a sinistra per il primo giocatore e a destra per il secondo, mentre in basso a destra troverete l'indicazione degli aerei e dei looping rimasti, che inizialmente sono rispettivamente 5 e 3.

Non rimane altro che giocare...

COMANDI
Joystick in porta 1
Tasti:
[1] = Monitor
[2] = Televisore
[CURSORI] = Selezione opzione / Movimenti
[SPAZIO] = Inizio gioco / Fuoco
[GRAPH] = Looping
[STOP] = Pausa
Joystick in porta 1

CIRCUS



Agli albori dell'era videogame c'erano giochi che, per le loro caratteristiche innovative, attiravano schiere di agguerriti giocatori con conseguente grande incasso. A quei tempi Circus Charlie fu un gioco decisamente innovativo che contribuì all'entrata del mondo del fumetto in quello del fumetto elettronico: un vero e proprio capostipite. Tra i diversi giochi riproposti in versione Msx, uno dei migliori programmi Konami è proprio questo. La storia è abbastanza semplice ma originale: un simpatico clown deve esibirsi in diverse prove corrispondenti ai diversi livelli di gioco e voi, naturalmente, dovrete guidarlo. Nel primo livello sarete sulla groppa di un leone e con esso dovrete saltare attraverso dei cerchi infuocati, ovviamente senza scottarvi. Nel secondo livello dovrete percorrere una corda sospesa in aria senza cadere e saltando le scimmiette dispettose che corrono sulla corda. Il terzo livello vede il nostro simpatico clown in equilibrio su un grosso pallone da cui saltare su un altro, un altro ancora e così via fino alla fine. Nel quarto livello poi sarete in piedi sulla groppa di un potente cavallo bianco che correrà lungo un percorso composto da piattaforme elastiche su cui dovrete saltare per poterle evitare. Del quinto livello non vi diciamo nulla, anche perché è più bello giocare con un pò di mistero su ciò che ci aspetta. Una volta caricato il gioco vedrete la presentazione iniziale e quindi il menù principale, subito seguito dalla dimostrazione. Quest'ultima può essere interrotta premendo la barra spaziatrice. Il menù principale offre quattro opzioni: la prima e la terza permettono di giocare da soli con il joystick o con la tastiera, mentre la seconda e la quarta opzione consentono di giocare in due, rispettivamente con joystick o con tastiera. Con lo spazio o con il tasto fire del joystick è possibile saltare: premendo poco saltate poco, premendo di più saltate di più! (Ma no?) Questo è particolarmente importante perché è sempre necessario saltare con una certa precisione, evitando gli ostacoli sempre mortali. Il gioco, sempre in scroll orizzontale verso sinistra, ha un tempo limite per ogni livello di gioco: se finisce il tempo perdete una vita ma se arrivate alla fine il tempo rimasto si trasforma in punti bonus. Come di consueto nei giochi Konami, avrete a disposizione tre omini che perderete sia allo scadere del tempo sia colpendo un ostacolo. Quando finite la partita avrete alcuni secondi di tempo per decidere se continuarla oppure no.

COMANDI
Tasti:
[CURSORI] = Scelta opzione nel menù / Movimenti Clown
[SPAZIO] = Inizio gioco / Salto
Joystick in porta 1 per il primo giocatore, in porta 2 per il secondo

SKOOTER

Ecco un gioco che unisce la logica allo stile arcade, un gioco che ci ha fatto letteralmente perdere la testa. Skooter contiene diversi schermi consecutivi, tutti diversi tra loro e nei quali l'unico comun denominatore è il protagonista che deve vedersela con i soliti quattro mostri nel tentativo di raccogliere i soliti quattro oggetti. Oggetti che possono essere frutti, dolci o i personaggi di Pac Man e Frogger, a seconda del livello raggiunto. Anche se i diversi schemi presentano blocchi e disegni simili non crediate che lo siano davvero: ogni schermo richiede una soluzione logica diversa dal precedente e vedrete che non sarà facile completare tutti i livelli. Con questo vi sfidiamo a superare il quarto livello dandovi un piccolo aiuto per superare il primo: distruggete più blocchi che potete fino a che non compare un cuoricino lampeggiante. Proprio come in Pac Man, questo bonus vi permette di eliminare i mostriciattoli del primo livello facilitando notevolmente il vostro compito. Una volta caricato il programma sullo schermo comparirà un messaggio che invita a scegliere il tipo di schermo utilizzato: premendo il tasto [SELECT] dite al computer che state usando un monitor monocromatico, mentre con la barra spaziatrice dite che il vostro schermo è un televisore oppure un monitor a colori. Dopo questo comparirà la presentazione con l'attesa della pressione dello spazio o del tasto di fuoco del joystick per cominciare a giocare. Può capitare che mentre giocate rimaniate bloccati e non riusciate a uscire più: in questo caso l'unica soluzione è l'autodistruzione che potete attivare con il tasto funzione [F5]. Con gli altri tasti funzione potete attivare o disattivare il suono e la pausa.

COMANDI
Tasti:
[SELECT] = Monitor monocromatico o Televisore B/N
[SPAZIO] = Monitor o Televisore a colori / Inizio gioco
[CURSORI] = Movimenti – [F1] = Attiva/Disattiva Suono
[F3] = Attiva Pausa – [F4] = Disattiva Pausa
[F5] = Autodistruzione

BOXING

Tra le simulazione di Boxe esistenti per Msx, questa che vi proponiamo è sicuramente una delle migliori, uno dei pochi programmi che riesce a sfruttare al massimo della sua potenzialità la grafica dell'Msx 1. Una volta completato il caricamento del gioco sullo schermo comparirà la presentazione e quindi il menù principale, subito seguito dalla dimostrazione. Quest'ultima può essere interrotta tramite la barra spaziatrice o un tasto cursore. Il menù principale offre quattro opzioni: le prime due prevedono un giocatore contro il computer con due diversi livelli, le ultime due permettono di giocare in due con due diversi livelli come nelle prime due opzioni. I livelli di difficoltà sono GAME A e GAME B. La selezione dell'opzione può essere fatta muovendo il cursore - che è una "manina" - e poi premendo lo spazio o il tasto di fuoco quando si è sull'opzione desiderata. Nello schermo di gioco troverete il vostro pugile - che si chiama RYU - nella parte destra dello schermo. Nella parte bassa c'è la campanella, il punteggio e l'energia rimasta a voi e al vostro avversario. In alto è posto il timer mentre sullo sfondo vedrete il pubblico attorno al ring. Usando lo spazio contemporaneamente ai diversi tasti del cursore - quelli con le frecce! - potrete sferrare diversi pugni in punti diversi oppure schivare quelli dell'avversario. Con i soli cursori vi spostate a sinistra o a destra. Potete usare due diversi joystick collegati nelle rispettive porte, la uno e la due.

COMANDI
Tasti:
[CURSORI] = Selezione opzione / Movimenti
[SPAZIO] = Inizio gioco / Dà colpo
[CURSORI] + [SPAZIO] = Dà pugno o Schiva colpo
Joystick in porta 1 per il primo giocatore, in porta 2 per il secondo

UNICOPY

UNICOPY

Questo programma è, ancora una volta, un duplicatore di dischetti.

Unicopy è tra i più efficienti programmi di copia che vi abbiamo proposto ed è anche decisamente facile da utilizzare.

Una volta mandato in esecuzione premendo l'opzione corrispondente nel menù principale e poi il tasto funzione

[F1]

compariranno i messaggi del programma.

Rispondendo correttamente ai vari messaggi del tipo: numero di drive, numero di lati, etc., potrete settare il copiatore a vostro piacimento.

Dopo aver risposto alle diverse domande, arriverete passo dopo passo alla fase di copia.

COMANDI
Tastiera & Dos

● Nella rivista di videogames inglese "ACE" abbiamo letto con grande sorpresa che diversi topgames come **Chase HQ**, **Test Driver II The Duel** e **Batman - The Movie** sono disponibili in Spagna anche per lo standard Msx.
Ed è vero!
Ci siamo stati e abbiamo visto molto software, in gran parte a prezzi tra le 20 e le 30 mila lire.
Una soluzione può essere andare in Spagna, ma forse potete ottenere qualcosa rivolgendovi a: LASP - Alfonso I 28 - 50003 ZARAGOZA

● La SNK, che produce videogiochi da bar, importerà in Europa il **Neo Geo**, la nuova potente console con la qualità dei giochi da bar.
Il prezzo dovrebbe aggirarsi sulle 800.000 lire mentre i giochi dovrebbero essere venduti a circa 50.000 lire l'uno.
Prezzi incredibili?
No se pensate che praticamente avete in casa un videogioco da bar che permette di salvare il punteggio e la situazione del gioco per poi riprenderla in qualsiasi sala giochi grazie a una scheda di memoria esterna.
Ma è anche possibile l'operazione inversa, cioè continuare a casa una partita iniziata al bar.

● La **Nintendo** avrebbe già lanciato una sua macchina da 16 bit se la Sega non l'avesse battuta sul tempo col suo Genesis.
A partire da questo autunno comincierà in Giappone la commercializzazione di questa nuova macchina destinata a dare battaglia alla Sega.
Tutto all'insegna dell'evoluzione Msx.

● La società giapponese Fujitsu, che più di trent'anni fa aveva costruito il primo computer giapponese, sta per rientrare alla grande nel mercato giapponese, e forse anche europeo, con suo **FM-Tower Machine**, ultimo discendente dell'Msx.
La macchina è basata di un processore 80386 ed è dotata di un CD-Rom.
In più, la macchina ha un display con una risoluzione di 640 per 480 pixels e 256 colori a 64 toni di grigio, e suono stereo.
La macchina è, naturalmente, una console fantastica e anche totalmente Ms-Dos compatibile.
In Inghilterra è già disponibile, anche se in versione Giapponese.
Il prezzo è di 2250 sterline inglesi, cioè circa 5.000.000 di lire.

● **MSX 3**...un mistero di Agatha Christie?
Nel paese degli Msx (ndr Giappone) corre sempre la voce di un futuro Msx 3 dopo il recente Msx 2 Plus.
E finalmente, forse, abbiamo una notizia fondata: la ASCII - una società Giapponese forse già conosciuta da alcuni di voi per alcuni giochi - ha annunciato che nella primavera del 1991 inizierà a commercializzare un Msx 3. Comunque, finché non lo vediamo non ci creadiamo!

● Tra i nuovi programmi usciti in Giappone e in alcuni paesi europei c'è **SOLID SNAKE** (= Metal Gear 2).
Quando è uscito, Metal Gear era assolutamente magnifico, ma il suo successore sarà ancora meglio, non ci sono dubbi.
Il 21 luglio 1990 è iniziata la commercializzazione in Giappone ed è andato subito tutto esaurito in un giorno, almeno la prima tiratura.

● Conoscete sicuramente quei giochi con i quali si spara con una pistola ottica allo schermo.
Sì, proprio come quello della Nintendo. Ebbene, adesso c'è qualcosa di similare per il MSX.
Sempre in Giappone è uscito **PLUS-X TERMINATOR LASER**, una pistola ottica fornita con un gioco per MSX 1 e 2 su ROM.
Gli screenshots danno una bella impressione di questo gioco un gioco normale con eccellenti effetti 3D.

Le istruzioni di controllo (salto, salto condizionato, esecuzione condizionale, ecc.) consentono di alterare la sequenza di svolgimento del programma, a seconda delle condizioni che si presentano. Sono le istruzioni che danno "intelligenza" al computer.
Nei diagrammi di flusso abbiamo visto come il percorso logico del programma non sia quasi mai costituito da un'unica linea, ma possa seguire diverse strade, con diramazioni e ritorni. Introduciamo adesso le principali istruzioni BASIC che consentono di realizzare questo controllo della sequenza di esecuzione delle istruzioni.

SALTO INCONDIZIONATO - Istruzione GOTO

Osservate questo programma:

```
10 PRINT "CIAO! IO SONO UN COMPUTER."
20 GOTO 10
```

La linea 10 fa stampare sullo schermo la stringa racchiusa tra virgolette (qualunque cosa ci sia scritto). La linea 20 dice: GOTO 10. "Go to" in inglese significa "vai a" ed in questo caso dice al computer: Invece di eseguire la linea successiva (con numero maggiore di 20), vai alla linea 10, e poi prosegui da lì. Come risultato, il programma "torna" alla liena 10, la esegue (stampando), prosegue alla 20, viene rimandato alla 10, e continua a stampare la stringa all'infinito. Questo è un esempio di LOOP (ciclo o anello chiuso), in cui il programma ripete (in questo caso per sempre) una certa sequenza di istruzioni.
A proposito, per fermarlo bisogna premere i tasti [CTRL] e [C] (tenendo premuto il tasto [CTRL], premere [C]) oppure [CTRL] e [STOP] o [BREAK] o quello specificato dal manuale.
Un programma arrestato in questo modo può essere fatto ripartito dallo stesso punto dove si era fermato scrivendo il comando CONT [RETURN] oppure CONTINUE [RETURN].

SALTO CONDIZIONATO - Istruzione IF...GOTO e IF...THEN

I calcolatori sono macchine stupide. I calcolatori sono macchine intelligenti. Entrambe le affermazioni sono vere, basta indendersi sul significato di "intelligenza". Il computer esegue senza capire, dunque è stupido, ma è in grado di prendere delle decisioni, quindi deve possedere una qualche forma di intelligenza. Questa non è altro, in realtà, che quella del programmatore, il quale gli ha spiegato in dettaglio (scrivendo il programma) come comportarsi in tutte le possibili situazioni di scelta. Le istruzioni che effettuano queste scelte sono fondamentali: senza di esse il calcolatore eseguirebbe sempre la stessa sequenza di istruzioni e non sarebbe quella macchina potente e versatile che conosciamo. Vediamo subito in pratica un esempio di "intelligenza":

```
10 INPUT "SCRIVI UN NUMERO MINORE DI 9: ";N
20 IF N>=9 GOTO 10
30 PRINT "OK, VEDO CHE CI SIAMO CAPITI"
```

Cosa succede? Il computer stampa il messaggio alla linea 10 e resta in attesa di un numero dalla tastiera. Quando lo riceve passa alla linea successiva (20). Alla linea 20 c'è scritto IF. In inglese, IF significa SE; il calcolatore si chiede SE il numero introdotto (N) è MAGGIORE(>) o UGUALE(=) a 9. Se questo è vero (quindi non è minore di 9), esegue il GOTO 10 che rimanda alla linea 10, dove viene chiesto di introdurre nuovamente il numero.
Se invece la CONDIZIONE (N maggiore o uguale a 9) non è verificata, allora non succede niente ed il programma prosegue alla linea seguente (30). Questo si chiama SALTO CONDIZIONATO: il programma "salta" a un'altra istruzione solo se la condizione è vera. Abbiamo insegnato al computer a "capire" quello che gli viene dato e ad accettare soltanto i numeri validi. In un certo senso, gli abbiamo dato un pò di "intelligenza".
Una volta erano pochissimi e ora sono ormai estinti i BASIC che consentono soltanto IF...GOTO. Tuttu i dialetti Basic esistenti permettono una forma più completa, che non consente soltanto il salto condizionato, ma anche l'ESECUZIONE CONDIZIONALE di una o più istruzioni.
Facciamo un esempio:

```
10 INPUT "DAMMI UN NUMERO: ";N
20 Q=N*N:REM CALCOLA QUADRATO
30 PRINT "QUANT'E' IL QUADRATO DI ";N
40 INPUT X:REM TENTATIVO
50 IF X < Q THEN PRINT "TROPPO BASSO!":
GOTO 30
60 IF X > Q THEN PROMT "TROPPO ALTO!":
GOTO 30
70 PRINT "CENTRATO! ERA PROPRIO ";Q
```

Il programma chiede un numero (N) e ne calcola il quadrato (Q). Poi chiede un altro numero (X), che rappresenta un tentativo di indovinare il quadrato di N. A questo punto (linea 50) si verifica se il numero importato è minosre del valore da indovinare. Se si, ALLORA (in inglese THEN) viene eseguito tutto il resto della linea 50. Se no, il resto della linea viene slatato e si prosegue alla linea 60.
Questo si chiama ESECUZIONE CONDIZIONALE di un'istruzione (nel nostro caso di due istruzioni, un PRINT ed un GOTO), in quanto le istruzioni vengono eseguite soltanto se la condizione è vera.
Il resto della linea 50 è composto da due istruzioni,

separate dal simbolo ":" (due punti). In BASIC due istruzioni separate da due punti sono eseguite una dopo l'altra, come se fossero su due linee successive (qualche BASIC usa un simbolo diverso e pochissimi, come quello dell'ormai estinto ZX81, non lo permettono affatto). Dato che entrambe le istruzioni si trovano dopo il THEN, sulla stessa linea, o vengono eseguite entrambe (se la condizione è vera) o non ne viene eseguita nessuna (se la condizione è falsa).
L'effetto sul funzionamento del programma esempio è che, se il numero impostato (X) è minore della soluzione (Q), viene stampato un "TROPPO BASSO!" e viene poi eseguito un GOTO 30, che trona a richiedere un tentativo. Se il numero non è minore della soluzione (quindi è uguale o maggiore), il programma prosegue ignorando completamente il seguito della linea 50.
Alla linea 60, tutto come sopra, con la differenza che viene verificato se il numero è maggiore del risultato. Se è così, messaggio e ritorno alla linea 30. Infine, se il numero non è né maggiore né minore deve essere uguale: la risposta è esatta. Messaggio e fine del programma (linea 70).
Notate come la stessa istruzione PRINT possa stampare, ad esempio, una stringa (racchiusa tra virgolette) e, di seguito (punto e virgola), il valore di una variabile numerica.
Negli esempi di cui facciamo uso per illustrare il funzionamento delle istruzioni BASIC, useremo sempre nomi di variabili di una o due lettere al massimo (o una lettera ed un numero, come A3). Questo perché in molti BASIC le variabili sono identificate soltanto dai primi due caratteri. E' sempre comunque utile inserire qualche REM che spieghi il signigicato delle variabili, come abbiamo fatto alla linea 40 (nelle altre linee le stringhe dei PRINT erano già abbastanza esplicative).
Per concludere, ormai tute le versioni di BASIC permettono anche la clausola ELSE (ALTRIMENTI), che specifica le istruzioni da eseguire se la condizione è falsa. Ad esempio:

```
IF A=3 THEN PRINT "A VALE 3" ELSE PRINT "NON VALE 3"
```

Questa costruzione è molto comoda perché se i gruppi IF...THEN...ELSE sono NIDIFICABILI, cioè possono essere inseriti uno dentro l'altro (in genere un numero limitato di volte), consente una costruzione più "pulita" del programma. Ma qui stiamo sconfinadno dal BASIC verso i linguaggi strutturati come Pascal o C.

ESPRESSIONI LOGICHE

Un notevole passo avanti nella capacità di programmare si ha quando possiamo chiedere al calcolatore di prendere delle decisioni. Le "espressioni logiche", simile a quelle numeriche, permettono appunto al calcolatore di fare questa scelta. "Utility" e funzioni di "Debug" alleviano la fatica del programmatore e oggi non sono più patrimonio solo dei grandi calcolatori, ma anche dei piccoli personal e home computer. Grandezze numeriche e "stringhe di caratteri" sono i dati su cui può lavorare un personal computer. Abbiamo visto come nelle istruzioni condizionali, per esempio IF...THEN, la scelta dipenda da una CONDIZIONE. In realtà la condizione può essere un'intera ESPRESSIONE LOGICA, il cui risultato viene usato ai fini della scelta. Un'espressione logica, come una normale espressione numerica, è composta da OPERANDI (variabili o costanti) separati da OPERATORI (per esempio, nell'espressione numerica A+3, A e 3 sono gli operandi e + è l'operatore). A differenza delle espressioni numeriche, pero', il risultato è un VALORE LOGICO, o BOOLEANO (dal nome del matematico George Boole), che può assumere due soli valori: vero o falso. Quindi, diamo una definizione più completa dell'istruzione di salto condizionato:

```
IF espressione logica GOTO numero linea
```

Se il risultato dell'ESPRESSIONE LOGICA è VERO, viene eseguito il GOTO; se è FALSO, l'esecuzione prosegue normalemnte con la linea successiva. Lo stesso vale, naturalmente, per gli altri tipi di espressione condizionale.
Nelle espressioni logiche si usano gli operatori relazionali e logici.

OPERATORI RELAZIONALI

Così detti perché verificano una relazione, accettano come operandi le normali variabili e costanti e restituiscono come risultato un valore logico.
Ecco un esempio:

```
IF A ‹3 THEN
"A ‹3" è vera
```

Se A è minore di 3 l'espressione

Tra normali variabili numeriche, gli operatori relazionali hanno il consueto significato.
Ad esempio, 5 è maggiore di 3 (attenzione: -5 è minore di 3). Tra variabili stringa, gli operatori = (uguale) e ‹› (diverso) verificano se le due stringhe coincidono carattere per carattere, spazi compresi.
Ecco un esempio tipico:

```
150 INPUT "ALTRI DATI ? ";A$
160 IF A$ = "NO" GOTO 200
170 IF A$ ‹› "SI" GOTO 150
180 ....
```

La linea 150 stampa
"ALTRI DATI?"
e aspetta che l'operatore scriva una linea, terminata con la pressione del tasto di invio, [RETURN] o [CR] . I caratteri scritti vanno a finire nella variabile stringa A$.
La linea 160 controlla se A$ è uguale a "NO", cioè se era stato scrittp "NO" sulla tastiera. Se è vero, salta alla linea 200.
La linea 170 è una "sicura" contro errori dell'operatore. Se non era "NO", doveva essere "SI". In caso contrario ritorna a chiedere l'INPUT.
Se A$ valeva "SI", il programma prosegue alla linea 180. L'effetto degli altri operatori relazionali (maggiore, minore ecc.) su confronti tra variabili - e costanti - stringa non è uguale per tutti i BASIC. Di solito, comunque, viene effettuato un confronto alfabetico, fino alla lunghezza della stringa più corta. Questo permette di fare ordinamenti alfabetici, ricerche più veloci in un insieme di dati, ecc.

OPERATORI LOGICI

Capitano spesso situazioni più complesse di quelle risolvibili in una sola istruzione con gli operatori relazionali. Ad esempio, per sapere se il valore della variabile A è compreso tra 5 e 12 servirebbero due istruzioni:

```
330 IF A < 5 GOTO 350
340 IF A <= 12 GOTO 900:REM E' COMPRESO
TRA 5 E 12
350 ...
```

L'istruzione alla linea 330 serve a saltare il secondo TEST (prova, controllo) nel caso in cui A sia minore di 5. Si tratta evidentemente di una forma scomoda e poco leggibile. Per fortuna, in quasi tutti i BASIC è possibile scrivere:

```
330 IF (A>=5) AND (A<=12) GOTO 900
```

che significa: se (è vero che) A è maggiore od uguale a 5, e (è anche vero che) A è minore od uguale a 12, allora vai alla linea 900.
AND ("e" in italiano) è un OPERATORE LOGICO. Gli operatori logici accettano come operandi due valori logici (nell'esempio i risultati dei due confronti) e restituiscono come risultato un valore logico. Gli operatori logici usati nel BASIC sono:

AND (PRODOTTO LOGICO, due o più operandi):
il risultato è vero solo se tutti gli operandi sono veri;

OR (SOMMA LOGICA, due o più operandi):
il risultato è vero se almeno uno degli operandi è vero;

NOT (INVERSIONE LOGICA, un solo operando):
il risultato è vero solo se l'operando è falso.

Nel nostro esempio, le parentesi erano probabilmente inutili, in quanto di solito in BASIC la precedenza (o priorita') degli operatori relazionali è maggiore di quella degli operatori logici. In altri termini, prima si eseguono i confronti e poi vari AND, OR, NOT. Un esempio di uso dell'operatore logico OR ("o" in italiano):

```
40 INPUT "DAI UN NUMERO TRA 2 E 18: ";N
50 IF (N < 2) OR (N > 18) GOTO 40
```

A differenza di altri linguaggi di programmazione, in BASIC non esistono VARIABILI LOGICHE (BOOLEANE) speciali, ma si usano le normali variabili numeriche. Una variabile ha valore logico FALSO se vale zero, VERO in tutti gli altri casi. Perciò è possibile scrivere:

```
40 IF A GOTO 100
```

in luogo di

```
40 IF A<> 0 GOTO 100
```

E' anche possibile assegnare ad una variabile numerica il risultato di un'espressione logica. Ad esempio:

```
120 R=(A=B)
```

Viene assegnato ad R il valore logico risultante dal confronto tra A e B (le parentesi sono per chiarezza). Ciò significa che R vale zero se A e B hanno valore diverso (confronto falso). Se A e B sono uguali, R assume il valore corrispondente al valore logico "vero".
In alcuni dialetti BASIC il valore di "vero" e 1 (uno), in altri è -1 (in ogni caso è diverso da zero e può essere usato in altre espressione logicha). Per il BASIC della nostra macchina il valore "vero" e -1 e se un dialetto BASIC usa -1 come valore logico per "vero" vuol dire che consente di manipolare i singoli bit della memoria. Non sviluppiamo ora l'argomento, come pure non parliamo per ora dell'uso avanzato delle espressioni logiche, per non creare inutile confusione. Meglio, prima, conoscere il BASIC "normale".

UTILITY

Nella maggior parte dei calcolatori, la semplice correzione di un carattere in una linea inserita in precedenza (tipica operazione di EDITING) è già un lavoro alquanto fastidioso. Ma le scomodità che il

programmatore incontra aumentano rapidamente con il crescere delle dimensioni del programma. Come faccio ad inserire 15 linee tra le linee 120 e 130? Ho già usato la variabile H3? C'è qualche GOTO alla linea 1380? Eccetera, eccettera. Un programmatore professionista rischia di passare la maggior parte del proprio tempo risolvendo problemi del genere, invece di concentrarsi su una corretta scrittura (stesura, codifica) del programma.
Per questo esistono le UTILITY (funzioni o programmi di utilita'). Si tratta di funzioni già presenti insieme al BASIC (RESIDENTI) o che si possono introdurre leggendole da una memoria di massa (in genere un supporto magnetico o una ROM).
Possono essere state progettate dal costruttore del computer, da una SOFTWARE HOUSE (ditta di software) qualunque, o dal programmatore stesso. Una volta inserite - se non sono già presenti - non aggiungono istruzioni BASIC, ma entrano far parte dei comandi disponibili in MODO DI EDITING (quando si scrive il programma, non quando gira) per rendere il lavoro del programmatore più comodo e veloce.

Le UTILITY INTEGRATE (già comprese) o disponibili per ogni singola macchina sono molto diverse per qualità e quantita'. Il primato spetta indubbiamente all'PC IBM, per il quale la varietà di software disponibile è immensa, seguito da APPLE e COMMODORE. Poiché è impossibile descrivere, anche solo sommariamente, quello che il mercato offre, vediamo almeno i principali tipi di utility che possono facilitare il lavoro di chi scrive programmi e che già in parte si trovano nel BASIC della nostra macchina:

EDIT (MODIFICA)
Le utility di EDIT servono a rendere più comodo il lavoro di modifica delle linee già introdotte. Molto diverse da una macchina all'altra, sono particolarmente utili per i calcolatori che non dispongono del VIDEO EDITING (modifica diretta sul video).

AUTO-NUMBER (Numerazione automatica)
Già presente come comando in quasi tutti i BASIC, consente di introdurre automaticamente il numero della linea successiva all'ultima scritta, evitando errori e cancellazioni accidentali di altre linee. In genere è possibile specificare l'INCREMENTO, cioè di quanto aumentare ogni volta il numero di linea (di solito si usa 10). E' un'utility che può fare comodo, ma può anche creare intralci se non è ben realizzata (numeri di linea che appaiono anche se non voluti).

RENUMBER (RINUMERA)
Anche questa già presente a volte come comando, svolge una funzione importante: permette di RINUMERARE il programma od una sua parte, cioè di cambiare tutti i numeri di linea. Di solito è possibile stabilire inizio e fine della zona interessata e PASSO (Incremento) dei nuovi numeri. Serve quando occorre aggiungere più linee già scritte. Oltre ai numeri di linea, vengono naturalmente cambiati tutti i RIFERIMENTI (GOTO, ecc.) alle linee stesse.

MOVE, COPY (SPOSTA, COPIA)
Permettono di spostare e/o duplicare gruppi di linee da un punto all'altro del programma. A volte sono incluse nel RENUMBER.

FIND, REPLACE (TROVA, SOSTITUISCI)
FIND serve a cercare un'istruzione, una variabile od una qualunque sequenza di caratteri usata nel programma. REPLACE consente di rimpiazzarla con un'altra istruzione, variabile o sequenza specificata.

VARIABLE CROSS REFERENCE (RIFERIMENTO INCROCIATO DELLE VARIABILI)
Indica tutte le variabili usate nel programma, con linee nelle quali ciascuna è impiegata. Molto utile per risolvere problemi derivanti dall'impiego accidentale della stessa variabile con significati diversi.

LINE CROSS REFERENCE (RIFERIMENTO INCROCIATO DELLE LINEE)
Indica tutte le linee che sono riferite da altre linee (ad esempio linee destinazione di GOTO) e, per ciascuna di esse, le linee contenenti il riferimento. Di solito individua anche alcuni errori, come GOTO a linee inesistenti.

OPTIMIZER, CRUNCHER, COMPRESSOR (OTTIMIZZATORE, COMPRESSORE)
Toglie dal programma tutte le istruzioni di REM e le linee di soli REM, riducendo l'occupazione di memoria, aumentando la velocità di esecuzione e rendendo difficile la lettura del programma da parte di persone non autorizzate. Le migliori utility di questo genere riducono ulteriormente le dimensioni del programma, riunendo quante più istruzioni possibile in una sola linea. La lunghezza di un programma compresso puo; essere anche meno di metà dell'originale.

MERGE (FONDE, RIUNISCE)
Abbiamo già citato questa utility tra i possibili comandi BASIC. Serve a riunire due programmi in uno solo. Di solito, uno dei due è presente in memoria (centrale), mentre l'altro viene letto da un supporto magnetico (dischetto o nastro). Consente di tenere da parte (IN LIBRERIA) una serie di parti di programma già pronte (ROUTINE), da inserire nel programma che si sta scrivendo.

Poiché l'elenco delle possibili utility potrebbe continuare ancora per motlo abbiamo citato soltanto le principali. Come si vede, comunque, il lavoro del programmatore può risultare notevolmente sveltito da questi strumenti. E' quindi sempre consigliabile procurarsi qualche utility per il proprio computer. Di solito, riuntie in un certo numero, sono vendute sotto forma di programmi chiamati TOOL KIT (in inglese: scatola d'attrezzi), TOOL SET, PROGRAMMER'S AID (aiuti al programmatore) o simili, che si installano (introducono, attivano) facilmente leggendoli da dischetto, cassetta o cartuccia.
Il concetto di utility può anche essere esteso a tutti i programmi che non servono direttamente a risolcere un certo problema, ma facilitano il lavoro in generale. Per esempio, programmi per realizzare grafici, per copiare dischetti, per verificare il corretto funzionamento del calcolatore, e così via.

DEBUG

Non sempre un programma funziona perfettamente al primo colpo; anzi, è abbastanza raro. Errori o imperfezioni nell'analisi del problema, nella stesura dell'algoritmo o nella scrittura fisica del programma causano spesso un comportamento diverso da quello voluto. Gli errori o imperfezioni di un programma si chiamano BUG, termine inglese traducibile con pulce o insetto molesto (qualcuno lo chiama anche BACO). Il lavoro di messa a punto ed eliminazione dei problemi prende il nome di DEBUG (letteralmente "spulciare").
Certi errori sono facili da eliminare: se il programma si ferma ed appare il messaggio:

```
? SINTAX ERROR IN 120
```

vuol dire, evidentemente, che la linea 120 contiene un errore di sintassi che la rende incomprensibile per il computer. La situazione è più complessa quando ci sono istruzioni condizionali. Ad esempio, consideriamo queste linee di programma:

```
50 IF (A=B-2) OR (B>C-1) GOTO 70
60 A=A-1 70 ...
```

Quando le cose non vanno nel modo voluto, è difficile stabilire se l'istruzione alla linea 60 è stata eseguita oppure no. Occore allora fare uso di qualche TECNICA DI DEBUG. Vediamo le più comuni.

1 - Inserire delle PRINT per verificare se il programma passa da (esegue o meno) una certa linea. Nel nostro esempio, possiamo aggiungere una linea:

```
61 PRINT 61
```

Facendo GIRARE (RUN) il programma, sapremo se la linea 60 viene eseguita oppure no. Se viene eseguita, il programma prosegue con la 61 che abbiamo inserito e stampa, appunto, il numero 61 (si può far stampare qualunque cosa, ad esempio il valore di una variabile). Se, invece, il programma salta dalla linea 50 alla 70, non esegue nè la 60 nè la 61, dunque non stampa nulla.

2 - Inserire un BREAKPOINT (punto di arresto) ed osservare i valori delle variabili. Nell'esempio, si può scrivere:

```
61 STOP
```

Quando (se) il programma esegue la linea 61, l'esecuzione si arresta con il messaggio:

```
BREAK IN 61
```

A questo punto, scriviamo direttamente:

```
PRINT A, B, C
```

e vediamo i valori delle tre variabili, verificando se sono corretti.

3 - Per seguire l'esecuzione del programma, alcuni computer possiedono il comando TRACE,od altri simili; in altre macchine è possibile aggiungerlo con un'apposita utility letta da una memoria di massa. Il comando TRACE (traccia) di solito racchiuso tra due simboli speciali visualizza il numero di linea di tutte le istruzioni eseguite. E' quindi possibile vedere quali istruzioni vengono eseguite e quali no, ed in quale ordine.

In ogni caso, il miglior metodo per eliminare gli errori più fastidiosi cosiste nell'evitarli fin dall'inizio, spendendo un poco di tempo in più nell'analisi del problema e nella definizione dell'algoritmo. Questo tempo si ripaga abbondantemente nella stesura del programma, che deve soltanto tradurre in BASIC l'algoritmo, e soprattutto nella fase di debug, dove si incontrano solo problemi banali e facilmente risolvibili.
Molto importante è anche una completa DOCUMENTAZIONE del programma, cioè la spiegazione del suo funzionamento, sia su carta sia nel programma stesso (per mezzo delle istruzioni REM). Verso la fine del corso vedremo per esteso una corretta e moderna tecnica di documentazione dei programmi BASIC.

(continua)

IL PASCAL (Parte I[a])

Benvenuti a questa nuova serie di articoli dedicati al Pascal che con semplicità e chiarezza cercheranno di introdurvi in questo affascinante linguaggio, un vero e proprio mondo nuovo, una nuova dimensione. Si tratta di una serie di articoli rivolti a chi ha già qualche cognizione di base, magari grazie al nostro corso sulla programmazione.

INTRODUZIONE

Un calcolatore dispone di per sé di istruzioni estremamente elementari e di istruzioni di controllo che permettono di effettuare salti da un'istruzione a un'altra, senza strutture di controllo. Il meccanismo con cui è possibile usare linguaggi di programmazione di adegato livello di vicinanza del problema è quello della COMPILAZIONE, ovvero della disponibilità di uno strumento, detto appunto COMPILATORE, che traduce dal linguaggio evoluto al linguaggio disponibile sul calcolatore. E' il caso del Turbo Pascal che, se si esclude l'editor integrato, è un compilatore tra i più diffusi insieme al Microsoft Pascal. Un compilatore è realizzato come programma a sua volta, cosicché il suo compito è quello di accettare come dati d'ingresso il testo di un programma scritto in un linguaggio evoluto e fornire come risultato in uscita un programma equivalente, ma scritto in un linguaggio direttamente comprensibile e quindi eseguibile da un calcolatore.

CENNI STORICI

Il pascal è un linguaggio algoritmico per la programmazione sequenziale disegnato da N.Wirth nel 1968 all'ETH (Eifgenossische Technische Hochschule) di Zurigo. L'intento di Wirth era quello di definire un linguaggio che includesse pochi essenziali e fondamentali concetti di programmazione e che gli consentisse di insegnare la disciplina della programmazione in un modo semplice e sistematico.
Il linguaggio generò successivamente crescente interesse, specialmente all'interno del mondo accademico, in virtù della sua semplicità e facilità di apprendimento e insegnamento e delle sue solide basi metodologiche. Rispetto al linguaggio algoritmico ALGOL60, da cui traeva le origini formali, il Pascal aggiungeva un ben nutrito insieme di strutture di controllo e molti meccanismi per la strutturazione dei tipi di dati. Lo studio e la ricerca sulla programmazione strutturata, enorme e diffuso in quegli anni, trovano in questo linguaggio un giusto approdo.
Il compilatore del linguaggio fu utilizzabile nel 1970 e un anno dopo fu pubblicata la definizione del linguaggio, in un famoso articolo su "Acta Informatica". In particolare, la relativa facilità con cui era possibile trasportare il compilatore del linguaggio su un nuovo sistema favoriva enormemente la diffusione del linguaggio all'interno delle università. Molti anni dopo compariva un testo che rappresentò per molti il punto di riferimento e definì in pratica lo standard del linguaggio: "Pascal User Manual and Report", 1976. Per molti anni il mondo industriale delle grandi case costruttrici di calcolatori non mostrò alcun interesse verso il linguaggio, relegando il suo impiego solamente nelle università. Tuttavia negli ultimi anni diversi di fattori hanno contribuito alla rapida crescita di popolarità del Pascal, soprattitto dopo l'avento dei microprocessori.
Infatti, molti nuovi costruttori di microsistemi, non avendo rigidi requisiti di compatibilità con i vecchi sistemi, potevano investire in grossi progetti software realizzandoli con nuovi linguaggi di programmazione, proprio come il Pascal. Questi stessi piccoli sistemi supportavano il Pascal favorendo il diffondersi del linguaggio presso un più vasto pubblico di utenti. Inoltre, la "mentalità Pascal", che si era precedentemente formata e consolidata nell'ambito universitario, cominciava a spargersi anche all'esterno grazie agli stimoli culturali dei nuovi lautreati che entravano adesso nel mondo del lavoro.
Nonostante questa crescita di popolarità, i grandi costruttori sono rimasti fondamentalmente estranei al "fenomeno Pascal". Una delle ragioni risiede probabilmente nella mancanza di uno standard del linguaggio accettato internazionalmente. Negli anni precedenti infatti una proliferazione enorme di realizzazioni estese del linguaggio ha in pratica reso limitativo e obsoleto il vecchio standard accettato. Attualmente un processo di standardizzazione del linguaggio è in corso e questo potrebbe sortire nuovi effetti nel mondo industriale. Il Pascale quindi è nato nella università e qui si è diffuso, evidentemente solo per propri meriti, ovvero del suo ideatore. La validità del linguaggio è sicuramente indiscutibile dal punto di vista didattico e metodologico. Esso definisce una buona astrazione sulla macchina

reale (pur non discostandonese troppo per requisiti di efficienza), permettendo di costruire e esprimere in maniera semplice e sistematica le soluzioni ai problemi. Queste sono anche le basi su cui impiantare un corso sulla programmazione eil linguaggio Pascal sembra essere il veicolo più appropriato per entrare in questo mondo solo apparentamente misterioso.

ESEMPIO DI PROGRAMMA IN PASCAL

Introduciamo subito un semplice esempio di un programma scritto in Pascal, che ci permetta di esaminarne in modo immediato l'aspetto: nel seguito tratteremo con estremo dettaglio tutte le caratteristiche che qui introduciamo in modo informale. Il seguente programma calcola le soluzioni di un'equazione di secondo grado:

PROGRAM radici (input, output);
(* Calcolo delle soluzioni di un'equazione di secondo grado nella forma a*X−+b*X+c=0 *)

```
VAR a, b, c, delta, x, x1, x2: real;
BEGIN
●●●writeln('inserire i valori di a, b e c:');
●●●readln(a, b, c);
●●●delta:=b*b-4*a*c;
●●●IF delta‹0 THEN
●●●●●●writeln('nessuna soluzione reale')
●●●ELSE
●●●●●●IF delta=0 then
●●●●●●● BEGIN
            x:=-b/(2*a);
            writeln('soluzioni coincidenti:',x);
      END
  ELSE
      BEGIN
            x1:=(-b-sqrt(delta))/(2*a);
            x2:=(-b+sqrt(delta))/(2*a);
            writeln('due soluzione:');
            writeln('x1 = ',x1,'x2 = ',x2);
      END
END.
```

Scandendo il testo del programma, facciamo le seguenti osservazioni:

il programma inizia con la parola chiave PROGRAM e finisce con il simbolo "."; la parola chiave PROGRAM è seguita dal nome del programma, e dall'indicazione, tra parentesi, che si richiede l'uso dei files standard di input e di output, che corrispondono normalmente a tastiera e schermo video di un terminale;

subito dopo compare un commento, racchiuso dalla coppia di simboli (* e *);

la dichiarativa VAR permette di dichiarare esplicitamente i nomi delle variabili che vengono adottate dal programma, associamo a esse il tipo real;

il corpo del programma, cioè la parte di istruzioni operative, è racchiusa tra una coppia di "parentesi" BEGIN e END;

le operazioni di scrittura, ovvero di visualizzazione su video, sono effettuate mediante il richiamo della procedura WRITELN, che, ogni volta che viene invocata, produce una linea sullo schermo: a fronte di argomenti che sono stringhe, la stessa stringa è riportata sullo schermo, mentre a fronte di variabili, il loro valore è visualizzato; una procedura può avere contemporaneamente più argomenti da visualizzare, anche di natura diversa (stringhe e variabili), separati da una virgola;

in una sequenza di istruzioni, ciascuna è separata dalla succesiva dal carattere ";";

la lettura dei dati avviene, simmetricamente alla scrittura, mediante il richiamo della procedura READLN, che acquisisce tute le informazioni che, opportunamente separate da almeno uno spazio bianco, sono fornite su una linea, completata dal carattere di "a capo" (Carriage Return=Ritorno Carrello);

il simbolo := indica l'operazione di assegnamento;

la selezione IF..THEN..ELSE è considerata una sola istruzione composta, in grado di scegliere una tra due istruzioni da eseguire; se su una delle due vie le istruzioni da eseguire sono più di una, esse vengono raggruppate da una coppia BEGIN..END.

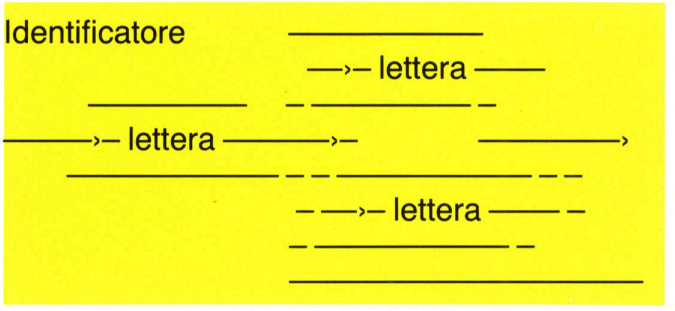
NOTAZIONE E STRUTTURA LESSICALE

Un linguaggio di programmazione è una nota-

zione formale per descrivere algoritmi che devono essere eseguiti da una macchina.
L'aspetto formale del linguaggio si manifesta soprattutto nella definizione di rigide regole per la composizione di programmi a partire da semplici caratteri.
L'insieme di queste regole costituisce la SINTASSI o GRAMMATICA del linguaggio.
Un limitato e basilare insieme di queste regole definisce la STRUTTURA LESSICALE del programma come costituito da unità elementari, cioè le PAROLE del linguaggio.
Questo livello di interpretazione logica consente di strutturare il limitato insieme di caratteri in uno più vasto, fatto di simboli elementari aventi anche una notazione più leggibile e comprensibile al programmatore.
Sopra il livello lessicale, il testo di un programma può essere visto come costituito di FRASI, ovvero di sequenze di parole strutturate secondo le rimandenti regole della sintassi.
Questa successiva interpretazione consente di definire una struttura al testo di un programma, la STRUTTURA TESTUALE.
L'attenzione riposta su questi aspetti nella definizione di un linguaggio nasce dalla convinzione, ormai diffusa, delle analogie esistenti tra struttura logica del frogramma e forme testuali.
In particolare, la leggibilità, la facilità di insegnamento e di apprendimento del linguaggio dipendono strettamente e fortemente da tali analogie.

CARTE SINTATTICHE

Tutta la sintassi di un linguaggio di programmazione può esser specificata mediante un insieme di regole sintattiche o grammaticali.
Trattandosi di linguaggi formali, le cui sentenze devono essere successivamente riconosciute dal traduttore del linguaggio, è necessario che tali regole siano rigorosamente definite.
Un particolare formalismo, quello delle CARTE SINTATTICHE, permette di descrivere convenientemente tali regole sintattiche e si è dimostrato un utile strumento con cui insegnare e apprendere la sintassi specifica del linguaggio.
Una carta sintattica è un grafo in cui linee uniscono delle scatole a spigoli arrotondati e rettangolari.
Qui vi chiederete cosa sia un grafo: in parole molto, molto semplici si tratta di un grafico che serve per illustrare, in questo caso, il funzionamento dell'istruzione.
Una scatola arrotondata descrive un SIMBOLO NON TERMINALE, ovvero simboli che non appaiono direttamente nelle sentenze, ma indicano "categorie sintattiche".
Essi si riferiscono a un'altra carta sintattica descritta altrove e che virtualmente la rimpiazza.
Qualsiasi traiettoria percorsa nel verso delle frecce specificate definisce un valido costrutto sintattico del linguaggio.
Tutti i simboli terminali incontrati nel percorso e in quella data sequenza costituiscono una SENTENZA legale del linguaggio.
Di seguito, come esempio, è data la carta sintatica completa della categoria sintattica ‹identificatore›, basilare nel Pascal e che ritroveremo successivamente:



Alcune sentenze corrette per ‹identificatore› sono:

alfa12 a b1 za

Così si conclude questo primo articolo dedicato al Pascal.
Nel prossimo numero in cui parleremo dell'Alfabeto, del Vocabolario e vedremo la lista delle parole riservate del Pascal.

L' INTRAMONTABILE MITO DEL ROCK'N'ROLL RIVIVE PER VOI IN QUESTA INCREDIBILE RACCOLTA

PARTY CD 88001

HEARTBREAKER CD 70108

HEARTBREAK HOTEL CD 88005

Desidero ricevere l'offerta "ELVIS PRESLEY" **codice CD1**
Allego assegno ☐ ricevuta versamento ☐
+ L. 2.500 quale contributo spese postali

NOME COGNOME

VIA N.

C.A.P. CITTÀ

Firma

Compilare il coupon allegando ricevuta (o fotocopia) del versamento effettuato sul C/C n. 11319209 intestato a **Gruppo Editoriale International Education srl** oppure assegno non trasferibile e spedire a:

Gruppo Editoriale International Education srl
viale Famagosta 75
20142 Milano

ALGORITMI E DIAGRAMMI DI FLUSSO

Un algoritmo è una sequenza di operazioni che devono essere svolte per arrivare alla soluzione ottimale di un problema. Un esempio chiarirà meglio le idee:

Algoritmo per la somma di due numeri

1) Incolonnare a destra i due addendi
2) Considerare le cifre della colonna piu' a destra
3) Assumere inizialmente il riporto uguale a 0
4) Sommare le cifre della colonna in considerazione ed aggiungere il riporto
5) Se il risultato ottenuto eseguendo l' istruzione 4) è minore di 10 scrivere la cifra risultante nella colonna in considerazione, porre il riporto uguale a 0 e andare all' istruzione 7)
6) Se il risultato ottenuto eseguendo l' istruzione 4) è maggiore o uguale a 10 sottrarre 10. Scrivere la cifra risultante nella colonna in considerazione, porre il riporto uguale a 1 e andare all'istruzione 7).
7) Spostarsi sulla cifra contenuta nella colonna immediatamente a sinistra.
8) andare all'istruzione 4).

Come abbiamo potuto notare, l'algoritmo è stato scritto con una forma ordinata e precisa di esposizione che ci mette al riparo da inevitabili ambiguità come potrebbero essere quelle derivanti da un Normale Linguaggio, nel nostro caso, l'italiano.
Descrivere comunque gli algoritmi in NL comporta indubbiamente un vantaggio poiché esso è patrimonio comune di un vasto numero di persone, ma ha lo svantaggio di essere generalmente:

A) poco preciso
B) complesso
C) a volte inadeguato per certe problematiche.

Dobbiamo comunque ammettere che durante le fasi preliminari di analisi (primordiale studio) di grossi problemi il NL è tuttora usatissimo. E' risaputo che un grave errore dei programmatori alle prime armi, è quello di essere così ansiosi di scrivere il programma da non preoccuparsi di stendere prima una dettagliata specifica del programma stesso.
Questo è un approccio claudicante. Il metodo più corretto per scrivere un programma comprende tre fasi essenziali:

A) Il problema deve essere capito (questo comporta la conoscenza della natura delle strutture dei dati in ingresso).
B) Il programma deve essere predefinito con un algoritmo che ne delinei i punti principali.
C) L'algoritmo deve essere trasformato in un diagramma di flusso (per una migliore e schematica comprensione dei punti da sviluppare), per poi essere tradotto (codificato) nel linguaggio di programmazione specifico (es. Fortran Pascal, basic, C, ecc).

L'omissione di uno di questi passi può portare a gravi errori. Un programma che comporti un piccolo numero di istruzioni, non richiede un diagramma di flusso perché la mente umana è in grado di memorizzare e sviluppare un certo numero di dati.
Quando però l'uomo deve sviluppare un programma con numerose istruzioni, allora è necessario aiuto.
In questi casi si ricorre al diagramma di flusso.
Il Diagramma di Flusso è una rappresentazione bidimensionale in forma di programma dell'intero problema che deve essere risolto dal calcolatore. Siccome i calcolatori possono svolgere solamente quattro operazioni elementari, il diagramma di flusso le deve rispettare.

Operazione di ingresso-uscita

Operazione di confronto e decisione

Procedura da sviluppare

La freccia indica la direzione di flusso dell' algoritmo.

Inizio/fine (start/stop) del DdF

Un semplice esempio chiarirà le idee sull'utilizzo dei DdF. Immaginiamo che un meccanismo debba segnalare le macchine che passano in un cancello mediante un dispositivo a fotocellula. L'algoritmo in NL sarebbe:

1) Poni il contatore del meccanismo uguale a zero.

2) Un'automobile è passata dalla fotocellula ? Se NO ripeti l'istruzione numero 2

3) Se SI allora incrementa il contatore di uno e vai all'istruzione numero due.

Tradotto in DdF diventerebbe:

START

Contatore = 0

macchina passata ?

contatore=contatore=1

Con questo esempio abbiamo concluso l'introduzione dei diagrammi di flusso, vediamo ora le costanti e le variabili.

COSTANTI, VARIABILI, ESPRESSIONI E LORO VALUTAZIONE

Come si sa esistono vari tipi di dati, come ad esempio i numeri, le stringhe di caratteri, ecc ecc. In questo paragrafo faremo riferimento a numeri interi grandi a piacere e tutte le considerazioni che seguono possono essere generalizzate a qualsiasi tipo di dato.
Il tipo di dato "intero" è stato scelto sia perché è già sicuramente a conoscenza di chi legge, sia per la ricchezza delle sue operazioni. Iniziamo con l'osservare la seguente scrittura :

```
(5x-y)(x+3)
```

Tutti sanno riconoscere l'espressione numerica dove x e y sono due VARIABILI e i numeri 5 e 3 sono dette COSTANTI. Sappiamo anche che l'espressione non ha un valore particolare fino a che non si attribuiscono dei valori specifici alle variabili come ad esempio se :

```
x = 1 e y = 3 in (5x-y)(x+3)
```

possiamo dire dopo un processo di valutazione e di calcolo che l'espressione vale 8. Questi concetti di costante, variabile, espressione e valutazione di una espressione costituiscono degli ingredienti essenziali della programmazione e verranno ripresi ed estesi più volte nelle nostre lezioni.
D'ora in avanti supporremo che l'esecutore disponga di un dispositivo particolare chiamato VALUTATORE che riceve una espressione aritmetica intera e, dopo un processo di valutazione,restituisce il risultato.
Per fare ciò egli deve poter prelevare i valori associati alle variabili che compaiono nell'espressione come ci si può rendere conto dall'esempio appena fatto. Chiameremo AMBIENTE un insieme di variabili con i valori loro associati.
La valutazione di un'espressione richiede quindi la presenza di un ambiente in cui valutarla come :

```
(5x-y)(x+3)-------›VALUTATORE--------›Valore
della espressione

^
!
!
!
AMBIENTE DI VALUTAZIONE
x = 1     y = 3
```

L'OPERAZIONE DI ASSEGNAMENTO

Come già sappiamo, una variabile è un nome a cui è associato un valore che viene reperito nel processo di valutazione delle espressioni.
Nel suddetto esempio abbiamo implicitamente fatto riferimento a questa proprietà in istruzioni del tipo:

assegna a x il valore 1

Non a caso questa istruzione viene detta assegnamento di una variabile : essa ha l'effetto di costruire un'associazione tra x ed il nuovo valore specificato cancellando ogni eventuale associazione precedente.
La simbologia lega l'operazione di assegnamento con il segno matematico di uguaglianza (=) oppure, in alternativa, il simbolo -->.

```
x ‹-- 1 equivale a dire x = 1
```

L'espressione a sinistra del simbolo

```
"‹--"
```

di assegnamento viene valutata ed assegnata alla variabile di sinistra del simbolo

```
"‹-"
```

nel senso appena descritto.
Più semplicemente possiamo dire che nel caso

```
x = 1 ( o x ‹-- 1)
```

il vecchio valore di x viene sostituito da quello nuovo.
Per intenderci, possiamo pensare ad una variabile come una scatola etichettata che può contenere un unico valore che può essere sostituito dopo una successiva operazione di assegnamento.
Per questo motivo si parla di "contenuto" di una variabile invece di "valore" di una variabile anche se si possono usare scambievolmente le due espressioni.
La lettura di una variabile, cioè il processo di valutazione di una variabile non è distruttiva; la variabile, in questo caso x, continua ad essere legata allo stesso valore 1.
La scrittura di una variabile, cioè l'assegnamento di un valore alla variabile, è invece distruttivo : il suo valore non può essere reperito in quanto sostituito dal nuovo valore assegnato.

(continua)

TELEMATICA ED EDUCAZIONE

Le risorse tecnologiche dei mezzi di comunicazione telematica investono direttamente i processi educativi, proprio perché innovano le modalità della comunicazione modificando i processi di appendimento e conseguentemente, le procedure di insegnamento.
La comunicazione mediata da questi strumenti diventa, marcatamente, da individuale, collettiva e viene percepita in contemporanea da un numero di persone che moltiplicano N volte l'unità classe.
L'unità classe tradizionale si frantuma e si riaggrega in nuove composizioni in base ai livelli di comprensione raggiunti.
Il rapporto comunicativo si decanta da ogni interferenza di origine emotiva che lo rende indubbiamente meno nitido e chiaro, ma certamente più partecipato.
L'allievo può dare segnali di risposta certi e chiari che consentono la verifica dell'esatta comprensione del messaggio e questo fatto permette successivi interventi di correzione della comunicazione tali da renderla chiara ed efficace per il destinatario.
Risultano altresì possibili ulteriori arricchimenti di quanto è oggetto della comunicazione con una grafualità controllabile e funzionale all'apprendimento.
Tuttavia non tutti i messaggi in uscita e in entrata, di cui si compone il processo educativo, sono efficacemente comunicabili attraverso i canali telematici, poiché tale processo non è riducibile ad una pura comunicazione di contenuti, ma esso nella complessità degli interventi è costituito anche da segno che trasmettono valutazioni comportamentali, anche di ordine etico, di funzioni estetiche, di moduli partecipativi emozionali, soggetivi e interpersonali che superano l'ambito della comunicazione oggettiva.
Un primo problema è quindi di individuare gli ambiti in cui sia opportuno utilizzare i canali telematici; in tutti quei casi in cui la comunicazione richiede chiarezza e oggettività e sia di alto valore scientifico, il ricorso a tali canali è utile, opportuno e potrà divenire anche indispensabile.
Un secondo problema è quello di prefigurare una comunicazione in contemporanea di informazioni uguali per tutti gli studenti e di alto valore scientifico, e verificarne l'esatta comprensione; sembra di dover ipotizzare presenze di mediazione da parte di docenti che, dopo aver percepito il messaggio con gli allievi, guidino un comune lavoro di rielaborazione al fine di favorire il processo di interiorizzazione critica e di osservare e valutare le procedure di apprendimento e verificare l'efficacia dell'insegnamento.
Un'operazione di tal genere presuppone docenti preparati all'uso della risorsa tecnica, capaci di cogliere le incidenze psicologiche del mezzo di comunicazione e di correlarle tra loro per coordinare e finalizzare gli interventi nel processo educativo.
Questo cenno ai docenti ed alla loro funzione ci riconduce ad un discorso che è prioritario e preliminare in qualsivoglia processo educazionele, quello dei fini e degli obiettivi di tale processo, senza di che gli interventi, rimanendo del tutto occasionali e casuali, non hanno alcun senso educativo.
L'impiego in tutti i campi del lavoro di risorse tecnologiche che, consentendo la comunicazione in tempi reali, accelerano il cambiamento, impone dall'esterno alcune finalizzazioni dell'attività curricolare scolastica.
Il "sapere" è sempre meno riducibile ad una costruzione statica edificata attraverso una sovrapposizione di elementi definiti e bene incastrati tra loro. Ma è sempre più chiaramente un processo in cui gli elementi base acquisiti consentono l'inserimento di elementi nuovi correttamente compresi e criticamente valutati. Né può essere definito sapere la pura acquisizione di conoscenze scientifiche e di abilità tecnologiche accettate come se fossero dati determinati, certi ed immutabili. Atteggiamenti culturali del genere condannerebbero l'allievo divenuto lavoratore, ma prima ancora il sistema educativo stesso nel suo complesso, ad una rapida obsolescenza.
In realta è quanto sta avvenendo nei diversi settori del lavoro, ma nella crisi sono pienamente coinvolte tutte le agenzie educative, fra cui anche la scuola che viene richiamata ad una rigorosa revisione dei suoi interventi nel processo educazionale.

DENTRO L'MSX (Parte II[a])

IL PROCESSORE VIDEO

Nell'Msx 1 il VDP (ViDeo Processor), è un Texas Instrument TMS 9129A. Si tratta di un integrato a 40 pin standard "dual in line" (DILIP) che usa 16Kb di RAM dinamica.
Solo il VDP può accedere alla RAM dedicata al video usando un bus dati unidirezionale a 8 bit. Il VDP viene controllato usando tre linee: RAS, CAS e R/W che derivano dal bus indirizzi e dalle linee di controllo del processore. Il VDP è inoltre connesso al bus dati di sistema.
Per effettuare da BASIC un'operazione di letura o di scrittura sulla RAM del video è necessario usare le istruzioni VPOKE e VPEEK oppure le appropriate procedure in ROM a livello di linguaggio Assembler.

LA STRUTTURA VIDEO DEL VDP

In tutti i modi di funzionamento il VDP costruisce l'immagine da visualizzare principalmente a partire da due tabelle nella RAM video: la tabella generatrice dei modelli e la tabella dei nomi e dei modelli.
Nella tabella generatrice dei modelli sono contenute le definizioni di ciascun carattere visualizzabile (cioè la descrizione, in termini di posizione dei pixel dei vari caratteri).
L'indirizzo di partenza della tabella generatrice dei modelli in ognuna della modalità di visualizzazioni è la seguente:

Modalità 0 - TEXT 40 - Indirizzo: 2048

Modalità 1 - TEXT 32 - Indirizzo: 0

Modalità 2 - HRG - Indirizzo: 0

Modalità 3 - MULTICOLORE - Indirizzo: 0

In tutte le modalità, tranne quella multicolore, ciascun carattere è definito nella tabella generatrice dei modelli per mezzo di otto byte con cui viene specificata la geometria dei pixel - tutti i caratteri standard sono inoltre giustificati a sinistra con le due colonne più a destra e la riga di base lasciate vuote.
Nella modalità testo a 32 colonne possono essere usate fino a 256 definizioni di carattere, nel qual caso la tabella risulta lunga 2048 byte.
La modalità HRG consente la definizione di 768 caratteri, producendo in tal modo una tabella lunga 6144 byte. La modalità testo a 40 colonne consente la definizione di 256 caratteri, ognuno di 8 byte, ma non vengono visualizzate le due colonne più a destra di ciascun carattere. In questo modo è possibile ottenere una schermata di 40 colonne con delle matrici di carattere di 8 x 6 pixel e non di 8 x 8 come nella modalità HRG o testo a 32 colonne.
Nella modalità multicolore la tabella generatrice dei modelli contiene 192 entrate, ciascuna di otto byte, formate da quattro coppie di due byte ciascuna. Ogni coppia non definisce il carattere bensì il colore delle celle 4 x 4 nella matrice complessiva di 8 x 8.

ESEMPIO

Nella modalità testo a 32 colonne la tabella generatrice inizia all'indirizzo 0 della VRAM, quindi il carattere con codice 65 - una "A" - viene definito dalle locazioni 65 x 8 = 520 fino alla 527.
La tabella dei nomi dei modelli determina quale carattere compare in ogni posizione sullo schermo (nella modalità multicolore vengono determinati i colori).
Nella modalità testo a 32 colonne ci sono 24 righe di 32 colonne e quindi 32 x 24 = 768 posizioni. Ciascuna di queste può contenere uno dei 256 possibili caratteri e di conseguenza la tabella dei nomi è lunga 768 byte - ognuno dei quali specifica quale carattere verrà messo in corrispondenza con una posizione.
Nella modalità testo a 40 colonne la tavola dei nomi per lo schermo ha 40 x 24 = 960 byte. Analogamente, ciascuno di questi descrive il carattere visualizzato in una determinata posizione dello schermo. Così come nella modalità testa a 32 colonne, la scelta è ristretta ai 256 modelli contenuti nella tabella generatrice dei modelli.

GRAFICA AD ALTA RISOLUZIONE

Nella modalità HRG ci sono ancora una volta 768 posizoni nella tabella dei nomi, ma poiché in questa modalità sono disponibili 768 definizioni di carattere, come può un byte stabilire la selezione? A tale scopo lo schermo è concettualmente diviso in tre zone e le prime 256 entrate nella tabella dei nomi selezinano i caratteri da 0

a 255, le seconde 256 selezionano i caratteri da 256 a 511 e le ultime 256 i caratteri da 512 a 768.
La tabella dei nomi nella modalità multicolore è lunga 768 bytes.
Il valore contenuto in ciascun byte è un riferimento a un blocco di otto byte nella tabella generatrice dei modelli per il multicolore.
Due bytes di questo blocco indicano i colori dei blocchi 4 x 4 che formano la matrice del carattere. La colonna sullo schermo ove si trova la matrice carattere determina a quale dei quatro blocchi si fa riferimento. Se si trova sulla riga più alta vengono referenziati i primi due bytes, se si trova sulla seconda riga sono interessati i bytes 3 e 4 e così via, con la coppia successiva usata per la riga seguente.
Quindi le righe 0, 4, 8, 12, 16, 20 faranno tutte riferimento ai primi due bytes.

COLORE

Benché il TMS 9129 VDP sia in grado di usare 15 colori più il trasparente (compresi il bianco e il nero), nella modalità testo a 40 colonne possono essere usati solo due colori, e il bordo assume il colore dello sfondo. In ambiente BASIC questi sono impostati (come per le modalità) per mezzo dell'istruzione COLOR, mentre in linguaggio macchina il registro a sola scrittura numero 7 del VDP determina entrambi i suddetti colori.
Per modificare quest'ultimo è preferibile usare la procedura appropriata contenuta nella ROM.
Nella modalità testo a 32 colonne, la tabella generatrice dei modelli, con dimensione 2k, è divisa in più insiemi di otto definizioni di carattere.
Per ciascun insieme, il colore del primo piano e quello dello sfondo; sono determinati da un solo byte in una tabella generatrice di colori. Quindi la tabella dei colori è lunga 32 byte. Il colore del primo piano è contenuto nel semi byte più significativo e viceversa per il colore dello sfondo.
Ne consegue che per visualizzare una lettera due volte con differenti colori, la definizione del carattere deve essere duplicata in un'altra posizione della tabella generatrice dei modelli: a quel punto ad essa può essere assegnato un diverso colore. Nella modalità HRG la combinazione sfondo/primo piano può essere specificata per ciascuna delle otto "linee" di ciascuna matrice carattere. A questo scopo sono necessari 768 x 8 = 6144 byte che vengono organizzati in un formato analogo a quello della tabella generatrice dei modelli. Questo metodo di allocazione dei colori rende problematica un'eventuale procedura per lo scroll del video.

SPRITE

Uno sprite è un blocco costituito da un quadrato il cui lato è di 1, 2 o 4 celle carattere. E' indipendente dal piano di visualizzazione e viene mosso facendo variare le coordinate X e Y che sono in relazione ad esso. Può essere acceso o spento come si desidera; inoltre ogni collisione tra due sprite viene rilevata automaticamente.
Ogni sprite è dotato di una priorità e se una sezione non trasparente di uno sprite a più alta priorità passa davanti ad un'altro, quest'ultimo viene parzialmente nascosto.
Gli sprite consentono di ottenere con facilità effetti piuttosto sofisticati come, ad esempio, simulazioni tridimensionali. Gli sprite non sono disponibili nella modalità testo a 40 colonne.
In tutte le altre possono essere utilizzati sullo schermo fino a 32 sprite, con un massimo di quattro per ogni linea. Se si supera questo limite, solo i quattro sprite a più alta priorità vengon ovisualizzati sulla linea. Inoltre viene sollevata la condizionione di "quinto sprite" ed il suo numero viene posto nel registro di stato del VDP.
Ciascun sprite può essere di un unico colore e la sua priorità non può essere cambiata. Il BASIC mette a disposizione dell;utente due istruzioni per controllare gli sprite. La prima, SPRITE$(N) viene usata per definire un banco di modelli di sprite e la seconda, PUT SPRITE determina la posizione, il colore e il numero di ogni sprite visualizzato.
Questo comandi agiscono su due tabelle nella RAM video che contengono tutti i dati di visualizzazione degli sprite.
La tabella dei modelli di sprite che parte dalla locazione 14336 della VRAM (RAM Video) - indipendentemente dalla modalità di visualizzazione - contiene la definizione della forma per tutti gli sprite. La tabella degli attributi degli sprite contiene le coordinate X e Y, il colore e il modello di ogni sprite. Gli sprite poossono avere dimensioni sia di 8 x 8 pixel, sia di 16 x 16. Inoltre possono essere visualizzati con dimensione doppia di quella di definizione -quindi si arriva ad un massimo di 32 x 32 pixel.
Il loro posizionamento è relativo all'angolo sinistro in alto dello schermo e possono essere mossi di un pixel alla volta.

La tabella dei modelli degli sprite può contenere fino a 256 blocchi di otto bye.
Se gli sprite hanno dimensioni di 8 x 8 pixel (non si possono usare contemporaneamente quelli a 8 e a 16) allora possono essere definiti 256 modelli di sprite diffirenti - il modello 0 dalla locazione 14336 alla 14343, il numero 1 dalla 14344 alla 14351 e così via.
Se invece si scelgono gli sprite a 16 pixel, ciascuno di essi viene definito usando quattro blocchi consecutivi di otto byte.
Ciascun blocco definisce una sezione pari a un quarto dello sprite, nella sequenza:

1 3
2 4

Dopo la definizione della forma di uno sprite, esso viene posto sullo schermo tramite un'entrata della tabella degli attributi degli sprite.
In tutti i modi di visualizzazione che consentono l'uso deglio sprite, la tabella inizia alla medesima locazione della VRAM: 6912 e ha una lunghezza massima di 32 sprite x 4 byte = 128 byte.
Ciascuna entrata serve ad indicare la posizione, il colore e il numero di modello per un singolo sprite. I primi due byte danno le coordinate Y e X mentre il terzo identifica la forma selezionando uno o più blocchi nella tabella dei modelli degli sprite. L'ultimo byte contiene nel semibyte meno significativo il colore di sfondo dello sprite.
In aggiunta il bit più significativo consente di spostare a sinistra, quando è posto a 1, lo sprite di 32 pixel ("clock early bit"). Le collisioni tra sprite vengono gestite in BASIC usando le istruzioni

SPRITE ON e ON SPRITE.

IL GENERATORE PROGRAMMABILE DI SUONI

Per questa funzione viene utilizzato un chip Yamaha AY-3-8910 PSG.
Si tratta di un generatore programmabile di suoni basato su registri che rende disponibili tre voci su un'estensione di otto ottabe. Inoltre l'ingresso da joystick è gestito tramite due porte di I/O indipendenti, situate sulla scheda.
Ogni voce ha un controllo separato di volume e può produrre un suono pulito oppure un rumore. Però la stessa frequenza di rumore e forma d'inviluppo viene usata per tutti i canali - se due canali producono rumore, essi producono lo STESSO rumore; può essere regolato solo il volume.
Il PSG ha 16 registri interni di lettura/scrittura, due dei quali hanno funzioni di registri di memorizzazione dati per le due porte del joystick.
I restanti registri possono essere divisi in 5 gruppi; in base alla loro funzione:

R0 - R5:
controllo della frequenza di canale.

R6:
frequenza del generatore di rumore.

R7:
registro di canale per la scelta tra rumore e suono pulito. I due bit più significativi determinano la direzione del trasferimento dati delle due porte di I/O.

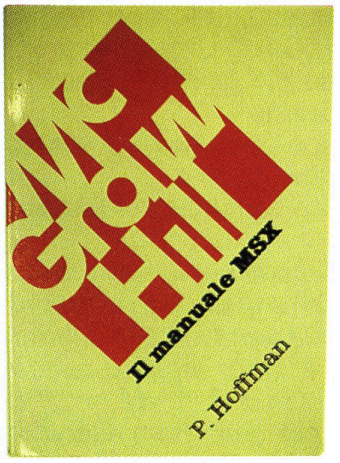
R10 - R12 (Ottale):
registri per la scelta tra apmpiezza del suono controllata dall'inviluppo e ampiezza del suono fissata.

R13 - R15 (ottale):
registri per la selezione della forma dell'inviluppo e del periodo.

Le due porte di ingresso e uscita sono del tutto indipendenti dalle funzioni di generazione del suono del PSG.
Il BASIC MSX comprende un buon numero di istruzioni per "fare musica" per le quali molti parametri sono fissati per default. In combinazione con la capacità di generare interruzioni temporizzate, tutto quanto descritto permette di ottenre effetti musicali complessi con estrema facilità.
Questo è tutto per questa volta.

Nel prossimo numero cominceremo ad analizzare il funzionamento dell'MSX BASIC fino ad utilizzare le funzioni avazate che mette a disposizione, tutto in funzione della programmazione Assembler.

IL LIBRO DEL MESE



IL MANUALE MSX
di P. Hoffman
Edito da: McGraw-Hill
Pagine: 330
Prezzo: 27.000 lire

Questo libro è una completa guida al nostro beneamato standard che ci mette a disposizione un ottimo sistema operativo - l'Msx-Dos derivato direttamente dall'Ms-Dos - e da un potente Basic che non ha nulla da invidiare al famoso Basic Microsoft Standard da cui deriva direttamente. Dopo un'introduzione alle caratteristiche hardware e software dei sistemi MSX, attraverso istruzioni e le funzioni dell'Msx-Basic, questo manuale porta l'utente, anche il più inesperto, a programmare il proprio computer e a sfruttarne le avanzate prestazioni grafiche e sonore.
L'utente più esperto, che si ritrova con un sistema decisamente completo dotato di un ottimo disk drive che serve anche ad usare Msx Disk, troverà una buona presentazione analitica dei comandi dell'Msx-Dos per la gestione dei file su disco e per la creazione di files batch.
Il libro è decisamente di buon livello e completo, e non si dilunga in descrizioni troppo lunghe e complesse che possono annoiare il lettore.
Dal punto di vista tecnico è un supporto valido e può essere di aiuto anche ai programmatori più esperti.
IL MANUALE MSX è diviso in tre parti fondamentali:

La prima - **IL COMPUTER MSX** - introduce al lettore il sistema, introducendo i diversi termini tecnici e le caratteristiche dell'Msx. Parla anche dei diversi programmi esistenti, di Word Processor e Fogli elettronici, e di cosa l'utente deve cercare in un programma. Parla poi dell'espandibilità del sistema e di come farlo, concludendo con la descrizione di come installare il sistema.

La seconda parte - **L'MSX-BASIC** -parte dai più semplici concetti della programmazione fino a portare il lettore principiante a fare un proprio programma. Parla dei diversi comandi Basic dividendoli nei diversi tipi e nelle diverse applicazioni possibili, concludendo con l'introduzione della programmazione in Assembler e dell'interfacciamento di questo col Basic.

La terza parte - **L'MSX-DOS** - è interamente dedicata al sistema operativo e inizia con un'introduzione semplice e chiara per concludere il tutto con la descrizione e l'uso delle potenzialità di questo Dos.

In appendice troviamo una guida rapida all'Msx-Basic e una guida rapida all'Msx-Dos, una descrizione dei messaggi d'errore del Basic, una tabella di caratteri ASCII e una guida alla tastiera.
Il libro, come tutti i prodotti dedicati all'MSX, è abbastanza difficile da reperire e forse potete trovarne qualche copia avanzata nelle migliori librerie specializzate. Se non lo trovate l'unica soluzione è quella di rivolgersi all'Msx Club Italia che mette a disposizione dei soci delle fotocopie di diversi testi, compreso questo.
I prezzi ovviamente non sono quelli di copertina, ma coprono le spese delle fotocopie.
Per maggiori informazioni l'indirizzo è:
MSX CLUB ITALIA - C.P. 34
20075 LODI CENTRO (MI).

POSTA

Spett. Gruppo Editoriale... Vorrei sapere se il TotoMSX su dico (i programmi per fare 13 al Totocalcio) esiste anche su cassetta, perché ho un computer che funziona con quest'ultima. Se affermativo, inviati i programmi su cassetta (in contrassegno) al seguente indirizzo...

Carlo ROSTAGNO - Torralba (SS)

Non è la prima volta che ci sentiamo rivolgere questa domanda, ma la risposta è no.
Purtroppo non abbiamo disponibile la versione su cassetta di TotoMsx, ma potete trovare qualcosa all'Msx Club Italia.
L'indirizzo è ormai consueto:

MSX CLUB ITALIA - C.P. 34
20075 LODI CENTRO (MI).

Spett.le redazione di Msx Disk, sono un ragazzo di 12 anni, appassionato di informatica, felice possessore di un Msx 2 Philips e vostro assiduo lettore.
Sono stato molto felice delle novità su Msx Disk e soprattutto della adventure VALLEY. Lanciato il gioco (VALLEY) mi si pone davanti il messaggio "Syntax Error in 332", allorché ho listato la linea 332 ed ho scoperto che era di data (scritti correttamente). Ho visualizzato tutte le linee DATA ed ho visto che alla fine di ogni blocco contenente delle informazioni differenti c'erano al alcune data contenenti dei codici FM, FA, FO che erano anche le variabili che indicavano il numero di indice del vettore in cui il dato doveva risiedere.
Ho provato con un RENUM 333,332,1 ed ho creato spazio per un'altra linea quale 332 DATA "FD" (FD: variabile di incremento vettore contenente i vocaboli) ed il gioco gira, cosi', correttamente.
Ad un certo punto (non so come) sono riuscito ad aprire una buca nel terreno nella quale non mi sono potuto calare neanche provando tutti i vocaboli a conoscenza del programma. Sono forse questi i motivi di incompatibilità di cui si parla nell'editoriale del numero 17?
Anche l'utility DEBUG mi crea dei problemi dovuti forse ad un suo cattivo uso. Durante il caricamento di suddette utility si spegne la spia LED del drive nonostante questo continui a funzionare.
Scegliendo [F1] e quindi Read Master Test Disk, mi si pone una scritta contenente codici incomprensibili (es. ERR 107-C9). Altre volte con le opzioni [F3] ed [F4] mi appare la scritta ERR 206; è un codice di errore? Forse perché tengo il disco protetto (eppure Disk Write Protected è 68)? Potreste spiegarmi meglio il funzionamento di questo programma?
Altra cosa che mi assilla è Music Maker (Msx Disk 11). Non riesco a comporre i legamenti fra 2 note (anche se nella demo FUGA II di BACH appaiono). Come si fa?
Altre domande: sul catalogo Philips, ma anche Toshiba (pubblicato in Guida Computer di MCmicrocomputer) non vedo più gli MSX. Perche'? Sono forse usciti di produzione?
Scusandomi dello spazio rubatovi, vi saluto e vi ringrazio di un'eventuale risposta.

Giacinto TENAGLIA - Orsogna (CH)

Innanzitutto, complimenti per il livello di conoscenza della tua macchina e la tua ottima preparazione. Davvero notevole. Ma veniamo alle domande.
L'avventura Valley è molto lunga e occupa molta memoria nei 32Kilobyte di Ram disponibili per il Basic, così arriva al limite della Ram riservata al Disk Basic e alle variabili di sistema. Questo significa che quando si usa l'avventura caricandola da disco si occorre spesso e volentieri in strani errori o, più generalmente, in un errore "Out of Memory".
Così può capitare che il programma funzioni una volta sì e una no, oppure che non funzioni proprio, ma funziona sempre quando si carica il programma da cassetta dopo aver staccato il drive oppure dopo aver acceso il sistema premendo lo Shift. Infatti, quest'ultima operazione disattiva il drive in un Msx con il drive interno.
Per questo problema abbiamo preparato una nuova versione del disco con Valley in cui abbiamo inserito una nuova avventura funzionante perfettamente. Chiunque lo desideri, può inviarci l'Msx Disk con Valley per la sostituzione gratuita.
Sinceramente non sappiamo perché la luce del drive si spenga durante il caricamento di Debug, ma cercheremo di contattare gli autori per saperne di piu'. Per quanto riguarda gli errori, non crediamo che siano codici standard del'Msx ma piuttosto codici errore interni del programma.
Comunque, può darsi che faccia difetto il tuo dischetto perché non ci è capitato altre volte un problema del genere. L'unica soluzione è inviarci il disco per un esame diretto. Il disco verrà comunque sostituito. Per quanto riguarda la protezione da scrittura è sempre meglio usare una copia del disco di Msx Disk senza protezione da scrittura.
Music Maker è un programma che non conosciamo bene, soprattutto perché siamo digiuni di nozioni musicali, ma cercheremo di farti sapere di più in uno dei prossimi numeri.
Nel penultimo catalogo generale della Philips, nella parte dedicata ai computer c'era ancora uno spazio dedicato agli Msx e ai suoi programmi. Nell'ultima edizione di questo catalogo non si trova più nulla anche se nel listino prezzi troviamo diversi accessori e l'NMS 8280 ad un prezzo dimezzato rispetto al catalogo precedente.
Questo vuol dire che la Philips ora vende solo ciò

che le è rimasto in magazzino. Purtroppo, la produzione di Msx da parte di quasi tutte le case è cessata da molto tempo anche se in Europa del software viene ancora prodotto per la grande quantita di pezzi venduti in alcuni paesi: è il caso di Olanda, Belgio e Spagna. In Spagna, addirittura, vengono prodotte le conversione di molti dei giochi più recenti che escono su Commodore e Pc compatibili. Dobbiamo dire che fin dalla sua comparsa il sistema Msx è sempre stato boicottato, soprattutto in Inghilterra, dai produttori di altre marche e dalle Software House, forse perché uno standard Giapponese.
Ma in estremo oriente la situazione è diversa. Come avrai potuto leggere anche negli scorsi numeri, nel paese del Sollevante vengono ancora prodotti gli Msx dalla Sony e dalla Sanyo, e attualmente si è arrivati alla versione Msx 2 Plus. In Giappone il mercato è vastissimo e nell'estate del 1991 dovrebbe uscire, prodotto dalla ASCII, l'Msx 3.
Tra l'altro, come non tutti sanno, proprio in Giappone sono nate le console più diffuse come Sega e Nintendo che, guardacaso, sono state generate dai computer più diffusi in quel paese: gli Msx. Infatti, come è possibile constatare anche usando gli stessi Joystick, il Sega è un Msx 2 a tutti gli effetti, mentre il Nintendo è un Msx 2+ con alcune piccole differenze. In Europa è possibile trovare gli ultimi modelli Msx un pò dovunque, soprattutto il Philips Telematico che è un Msx a tutti gli effetti e che in Italia viene distribuito per posta tramite il Postal Market, ma soprattutto i Spagna e Olanda è facile rifornirsi di Hardware grazie a importatori paralleli.

Spettabile Msx Disk, sono un grande affezionato alla vostra mega rivista, e vi chiedo se nel prossimo numero di MSX DISK potreste mettere i seguenti giochi: Berretti Berdi, Carrier Command, Thundercats, Yes, prime minister for you, MASK, Xybots e infine un programma per trasportare i giochi da cassetta a disco.
Per migliorare la rivista vi consiglio di mettere la rubrica il MERCATO DI HARDWARE E SOFTWARE; mettere più pagine di recensione giochi rappresentando la schermata del gioco, mettere anche delle pagine dove vengono pubblicati listati di giochi o programmi vari. Infine vi chiedo di stampare nella copertina le schermate dei giochi spiegando in modo sintetico di che cosa "tratta". Continuate così che siete forti!!

Andrea MANUZZATO - Breganze (VI)

Grazie per i tuoi consigli che cercheremo di seguire nei limite del possibile. Per quanto riguarda i giochi non è facile trovare la versione Msx ma, se esistono, le troveremo. Ti ringraziamo molto per gli stupendi disegni che ci hai mandato.

Cara redazione di Msx Disk sono un ragazzo di 13 anni, molto affezionato a voi e devo farmi i miei più vivi complimenti per la vostra rivista che naviga a gonfie vele. Vivo in una città di nome Romano d'Ezzelino, ma siccome ho la città di Bassano del Grappa appiccicata alla mia, vado sempre a comperare da un mio amico edicolante i vostri dischi. A Bassano ci sono parecchi negozi di computers e Videogame, ma purtroppo pochi, o che sappia io nessuno, hanno quelle "COSE" che vanno bene al mio MSX2 Philips. Come posso risolvere questo problema? Spero che leggiate la mia lettera e che mi rispondiate al più presto. Grazie.

Alessandro MILANI - Romano d'Ezzelino (VI)

Una soluzione al tuo problema può essere, innanzitutto, quella di entrare in contatto con altri utenti Msx con cui scambiare programmi, informazioni e magari anche Hardware. In alternativa puoi rivolgerti ad un club specializzato come l'Msx Club Italia. L'indirizzo è sempre lo stesso, C.P. 34 -20075 LODI CENTRO (MI). Ti ricordo che inviando due francobolli da 500 lire riceverai il catalogo completo del software disponibile. Tra le altre cose voglio informare chi ha già ricevuto il catalogo che tra poco riceveranno anche l'aggiornamento con gli ultimi arrivi: 150 dischi da 720k pieni di novità!

Alla cortese attenzione della redazione di Msx Disk. Sono proprietario di un Msx2 Philips NMS 8280 e vivo in una cittadina dove procurarsi dei programmi per lo standard Msx è molto difficile. Cosi', dopo aver letto la "POSTA" di Msx Disk numero 16, ho pensato di scrivervi, in quanto avete risposto al sig. Bruno Negri che "per procurarso software deve rivolgersi... a noi, grazie ad un nuovo servizio di vendita per corrispondenza di cui - se i tempi produttivi sono giusti - può trovare la pubblicità tra queste pagine". Purtroppo i tempi produttivi non erano giusti, così vi chiedo se avete un catalogo di giochi e di utilità per lo standard MSX da inviarmi. Vi ringrazio gentilmente.

Mario MARIOLA - Nettuno (RM)

I tempi produttivi, come ha visto, non erano giusti così siamo in ritardo, ma tra breve riceverà a casa un catalogo completo dei prodotti offerti dal Gruppo Editoriale International Education.

SPLENDIDI INEDITI DEI MOSTRI SACRI DEL JAZZ

- ○ CHARLIE PARKER CDJJ 610
- ○ BENNY GOODMAN CDJJ 609
- ○ COUNT BASIE CDJJ 604
- ○ SIDNEY BECHET CDJJ 603
- ○ DIZZY GILLESPIE CDJJ 606
- ○ DUKE ELLINGTON CDJJ 602
- ○ LIONEL HAMPTON CDJJ 605

7 COMPACT DISC AL PREZZO DI L. 84.000

Desidero ricevere l'offerta "JAZZ" **codice CD7**
Allego assegno ☐ ricevuta versamento ☐
+ L. 2.500 quale contributo spese postali

NOME COGNOME

VIA N.

C.A.P. CITTÀ

Firma

Compilare il coupon allegando ricevuta (o fotocopia) del versamento effettuato sul C/C n. 11319209 intestato a **Gruppo Editoriale International Education srl** oppure assegno non trasferibile e spedire a:

Gruppo Editoriale International Education srl
viale Famagosta 75
20142 Milano

JAZZ