Spediz. in abbonamento postale GR. II/70 L. 2.000
(...)

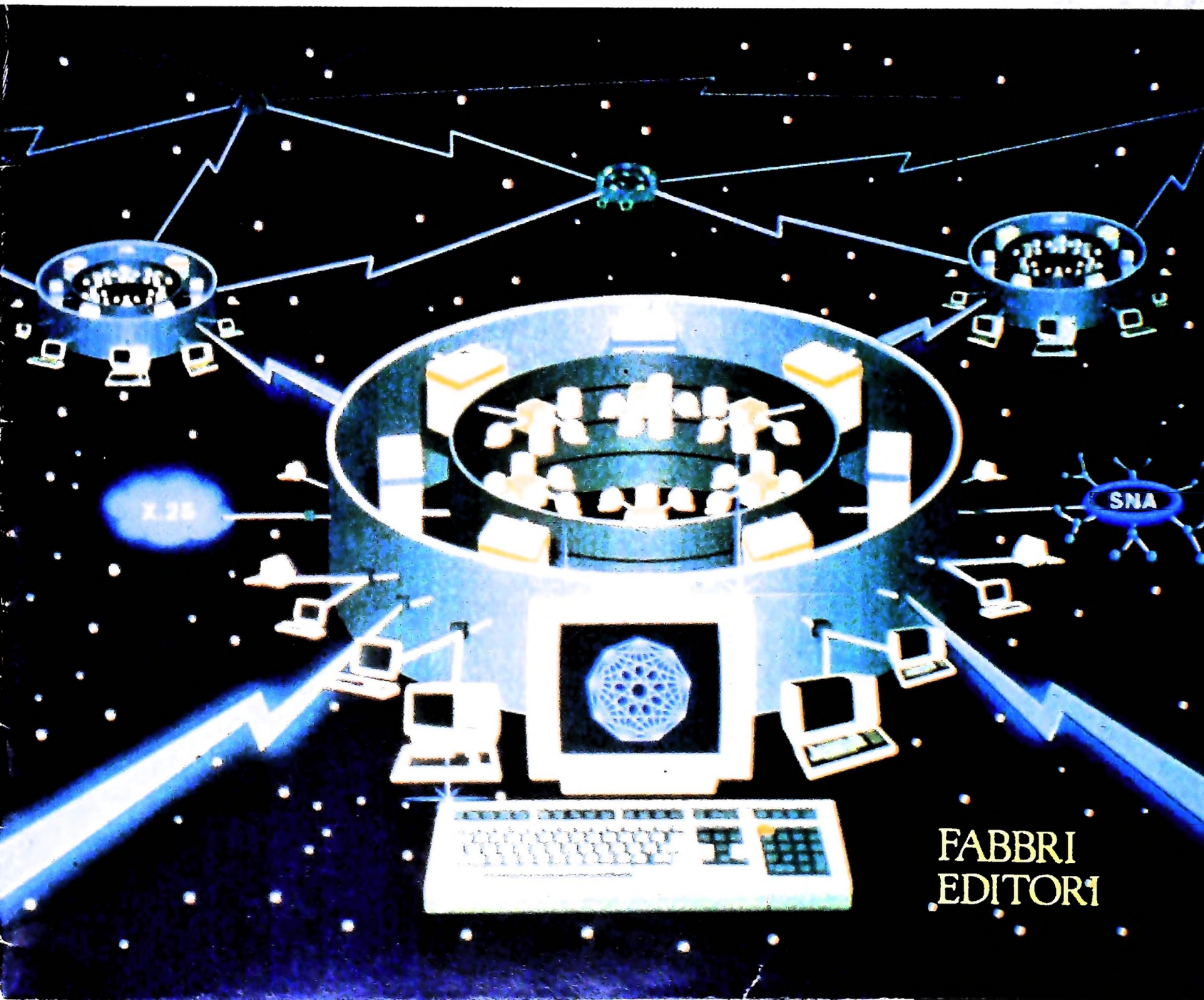
46 CORSO PRATICO COL COMPUTER

421982

è una iniziativa
FABBRI EDITORI
in collaborazione con
BANCO DI ROMA
e **OLIVETTI**

F4 F5 F6 F7 F8 BATTERY LOW

diretto da GIANNI DEGLI ANTONI



FABBRI EDITORI

IL BANCO DI ROMA FINANZIA IL VOSTRO ACQUISTO DI M 10 e M 20

Acquisto per contanti

È la formula di acquisto tradizionale.
Non vi sono particolari commenti da fare, se non sottolineare che troverete ampia disponibilità presso i punti di vendita Olivetti, poiché, grazie al "Corso pratico col computer", godrete di un rapporto di privilegio.

Il servizio di finanziamento bancario

Le seguenti norme descrivono dettagliatamente il servizio di finanziamento offerto dal Banco di Roma e dagli Istituti bancari a esso collegati:

Banca Centro Sud
Banco di Perugia

Le agenzie e/o sportelli di questi istituti sono presenti in 216 località italiane.

Come si accede al credito e come si entra in possesso del computer

1) Il Banco di Roma produce una modulistica che è stata distribuita a tutti i punti di vendita dei computer M 10 e M 20 caratterizzati dalla vetrofania M 10.
2) L'accesso al servizio bancario è limitato solo a coloro che si presenteranno al punto di vendita Olivetti.
3) Il punto di vendita Olivetti provvederà a istruire la pratica con la più vicina agenzia del Banco di Roma, a comunicare al cliente entro pochi gorni l'avvenuta concessione del credito e a consegnare il computer.

I valori del credito

Le convenzioni messe a punto con il Banco di Roma, valide anche per le banche collegate, prevedono:
1) Il credito non ha un limite minimo, purché tra le parti acquistate vi sia l'unità computer base.
2) Il valore massimo unitario per il credito è fissato nei seguenti termini:
   — valore massimo unitario per M 10 = L. 3.000.000
   — valore massino unitario per M 20 = L. 15.000.000
3) Il tasso passivo applicato al cliente è pari al "prime rate ABI (Associazione Bancaria Italiana) + 1,5 punti percentuali".
4) La convenzione prevede anche l'adeguamento del tasso passivo applicato al cliente a ogni variazione del "prime rate ABI"; tale adeguamento avverrà fin dal mese successivo a quello a cui è avvenuta la variazione.
5) La capitalizzazione degli interessi è annuale con rate di rimborso costanti, mensili, posticipate; il periodo del prestito è fissato in 18 mesi.
6) Al cliente è richiesto, a titolo di impegno, un deposito cauzionale pari al 10% del valore del prodotto acquistato, IVA inclusa; di tale 10% L. 50.000 saranno trattenute dal Banco di Roma a titolo di rimborso spese per l'istruttoria, il rimanente valore sarà vincolato come deposito fruttifero a un tasso annuo pari all'11%, per tutta la durata del prestito e verrà utilizzato quale rimborso delle ultime rate.
7) Nel caso in cui il cliente acquisti in un momento successivo altre parti del computer (esempio, stampante) con la formula del finanziamento bancario, tale nuovo prestito attiverà un nuovo contratto con gli stessi termini temporali e finanziari del precedente.

Le diverse forme di pagamento del finanziamento bancario

Il pagamento potrà avvenire:
- ☐ presso l'agenzia del Banco di Roma, o Istituti bancari a esso collegati, più vicina al punto di vendita Olivetti;
- ☐ presso qualsiasi altra agenzia del Banco di Roma, o Istituto a esso collegati;
- ☐ presso qualsiasi sportello di qualsiasi Istituto bancario, tramite ordine di bonifico (che potrà essere fatto una volta e avrà valore per tutte le rate);
- ☐ presso qualsiasi Ufficio Postale, tramite vaglia o conto corrente postale. Il numero di conto corrente postale sul quale effettuare il versamento verrà fornito dall'agenzia del Banco di Roma, o da Istituti a esso collegati.

BANCO DI ROMA
CONOSCIAMOCI MEGLIO.

Direttore dell'opera
GIANNI DEGLI ANTONI

Comitato Scientifico
GIANNI DEGLI ANTONI
Docente di Teoria dell'Informazione, Direttore dell'Istituto di Cibernetica dell'Università degli Studi di Milano

UMBERTO ECO
Ordinario di Semiotica presso l'Università di Bologna

MARIO ITALIANI
Ordinario di Teoria e Applicazione delle Macchine Calcolatrici presso l'Istituto di Cibernetica dell'Università degli Studi di Milano

MARCO MAIOCCHI
Professore Incaricato di Teoria e Applicazione delle Macchine Calcolatrici presso l'Istituto di Cibernetica dell'Università degli Studi di Milano

DANIELE MARINI
Ricercatore universitario presso l'Istituto di Cibernetica dell'Università degli Studi di Milano

Curatori di rubriche
MARCO ANELLI, DIEGO BIASI, ANDREA GRANELLI, ALDO GRASSO, MARCO MAIOCCHI, DANIELE MARINI, GIANCARLO MAURI, CLAUDIO PARMELLI

Testi
DOMENICO CAVALLOTTO, CLAUDIO PARMELLI,
Etnoteam (ADRIANA BICEGO)

Tavole
Logical Studio Communication
Il Corso di Programmazione e BASIC è stato realizzato da Etnoteam S.p.A., Milano
Computergrafica è stato realizzato da Eidos, S.c.r.l., Milano
Usare il Computer è stato realizzato in collaborazione con PARSEC S.N.C. - Milano

Direttore Editoriale
ORSOLA FENGHI

Redazione
CARLA VERGANI
LOGICAL STUDIO COMMUNICATION

Art Director
CESARE BARONI

Impaginazione
BRUNO DE CHECCHI
PAOLA ROZZA

Programmazione Editoriale
ROSANNA ZERBARINI
GIOVANNA BREGGÈ

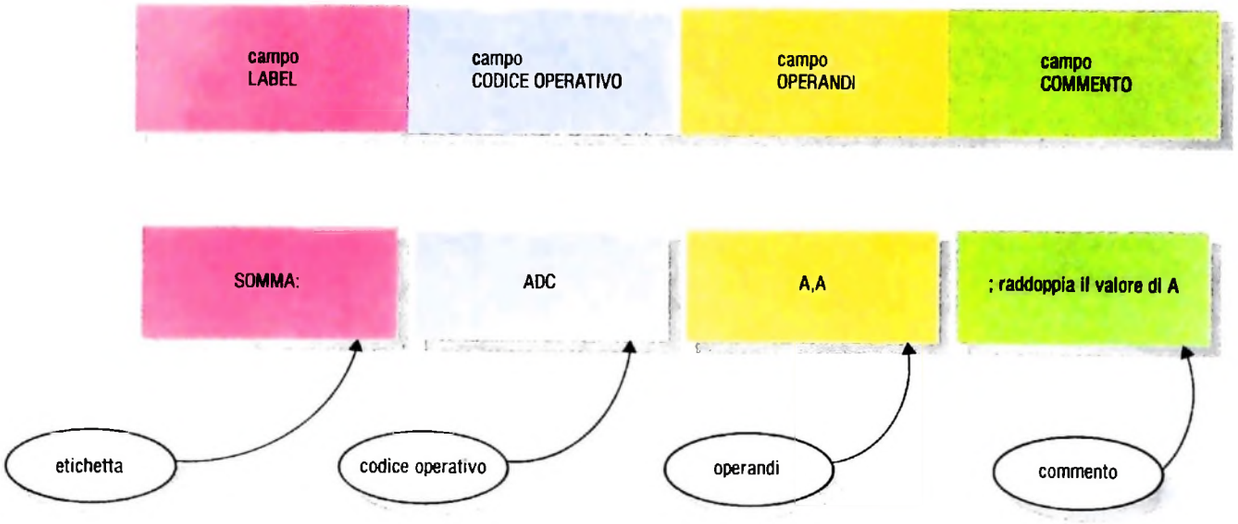
Segretarie di Redazione
RENATA FRIGOLI
LUCIA MONTANARI

Corso Pratico col Computer - Copyright © sul fascicolo 1985 Gruppo Editoriale Fabbri, Bompiani, Sonzogno, Etas S.p.A., Milano - Copyright © sull'opera 1984 Gruppo Editoriale Fabbri, Bompiani, Sonzogno, Etas S.p.A., Milano - Prima Edizione 1984 - Direttore responsabile GIOVANNI GIOVANNINI - Registrazione presso il Tribunale di Milano n. 135 del 10 marzo 1984 - Iscrizione al Registro Nazionale della Stampa n. 00262, vol. 3, Foglio 489 del 20.9.1982 - Stampato presso lo Stabilimento Grafico del Gruppo Editoriale Fabbri S.p.A., Milano - Diffusione - Distribuzione per l'Italia Gruppo Editoriale Fabbri S.p.A., via Mecenate, 91 - Milano - tel. 02/50951 - Pubblicazione periodica settimanale - Anno II - n. 46 - esce il giovedì - Spedizione in abb. postale - Gruppo II/70. L'Editore si riserva la facoltà di modificare il prezzo nel corso della pubblicazione, se costretto da mutate condizioni di mercato

I LINGUAGGI ASSEMBLER (I)

Una serie di sigle mnemoniche che corrispondono in maniera biunivoca alle istruzioni in linguaggio macchina.

Il linguaggio Assembler (Assembly language) è un linguaggio in cui a un'istruzione corrisponde, in genere, un'operazione elementare eseguibile dal calcolatore; ogni istruzione assembler deve essere solo trasformata dalla sua forma mnemonica e utilizzata dal programmatore per scriverne le istruzioni, in sequenze di zeri e uno. Il programma che esegue questa operazione si chiama Assembler o Assemblatore.
Con tale linguaggio si scrivono programmi che devono avere doti di velocità ed efficienza occupando meno memoria possibile, in quanto il programmatore ha il completo controllo delle risorse della macchina e può gestirle nel modo migliore.
Un tale linguaggio è quindi legato all'hardware per cui è realizzato e non è quindi possibile dare una serie di regole descrittive che valgano per qualunque calcolatore.
Lo scopo di questa trattazione non è quindi quello di presentare un linguaggio macchina particolare, ma quello di darne le linee generali, portando esempi di casi concreti.
Si è scelto anche di non parlare di linguaggi macchina di grandi calcolatori ma di fare un riassunto delle istruzioni dei più utilizzati microprocessori nei personal computer, come per esempio Z80, 6502, 8086.
Prima di analizzare alcune possibili istruzioni in linguaggio Assembler vediamo come normalmente esse sono rappresentate. Un'istruzione viene scritta su una riga (linguaggio a formato fisso) e ogni riga è divisa in 4 campi (figura in basso).
Il campo *label* (etichetta): è un identificatore mnemonico dell'istruzione che corrisponde all'indirizzo effettivo dell'istruzione stessa: tale etichetta può essere omessa.
Il campo *codice operativo*: identifica l'operazione che viene eseguita da un'istruzione; è anche chiamato mnemonico in quanto è un condensato del nome dell'operazione.
Il campo *operandi*: esplicita uno o più operandi su cui viene eseguita l'operazione; non tutte le istruzioni prevedono operandi.
Il campo *commento*: molto importante, permette di spiegare in modo comprensibile al lettore umano quello che si è detto al calcolatore nel suo linguaggio.
Prima di passare all'analisi di alcune possibili istruzioni del linguaggio Assembler vediamo come si può presentare un programma scritto in Assembly, proprio con lo scopo di ren-



Sopra: il formato di una istruzione in ASSEMBLER. Sotto: l'esempio mostra un'istruzione che preleva il contenuto del registro di nome A e lo somma al valore contenuto in A stesso, in modo che il contenuto di tale registro risulta raddoppiato.

```
;*************************************
;
;Routine di somma di interi di 6 bytes
;
;INPUT: All'indirizzo OPER1 i 6 bytes del primo addendo
;       All'indirizzo OPER2 l'indirizzo del secondo addendo
;
;OUTPUT: All'indirizzo OPER1 i 6 bytes del risultato
;
;I registri usati sono: A,C,D,E,H,L.
;I flag non sono preservati.
;
;*************************************************************

ADD6:   LD      HL,(OPER2)      ;Prendo l'indirizzo del 2^ addendo
        LD      D,H             ; e lo sposto nei registri D
        LD      E,L             ; ed E
        LD      HL,OPER1        ;Prendo l'indirizzo del 1^ addendo
        OR      A,A             ;Azzero il flag di carry
        LD      C,6             ;Inizializzo il contatore
LOOP:   LD      A,(HL)          ;Prendo il primo addendo
        EX      DE,HL           ;Scambio gli indirizzi
        ADC     A,(HL)          ;Sommo il secondo addendo
        EX      DE,HL           ;Riscambio gli indirizzi
        LD      (HL),A          ;Memorizzo il risultato parziale
        INC     DE              ;Punto al byte successivo
        INC     HL              ; dei due addendi
        DEC     C               ;Decremento il contatore
        JR      Z,FINE          ;Salta se il contatore e' zero
        JP      LOOP            ;Altrimenti continua l'operazione
FINE:   RET

;Fine della routine
```

Esempio di programma scritto in Assembly Language (ASSEMBLER). L'esempio è una routine di somma di numeri interi di 6 byte scritta nel linguaggio del microprocessore Z80. Sono usati registri di un byte (vale a dire di 8 bit) come A e C, oppure registri visibili a coppie di 2 byte, come HL. Nel testo l'esempio riportato in figura è chiarito nei dettagli.

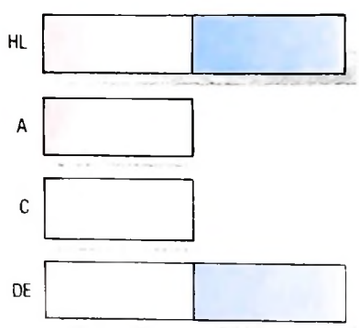
derci conto di quali siano le caratteristiche di un tale linguaggio rispetto a quelli esaminati finora, cioè estrema povertà nell'effettuare operazioni ed estrema capacità nell'agire su risorse hardware della macchina.

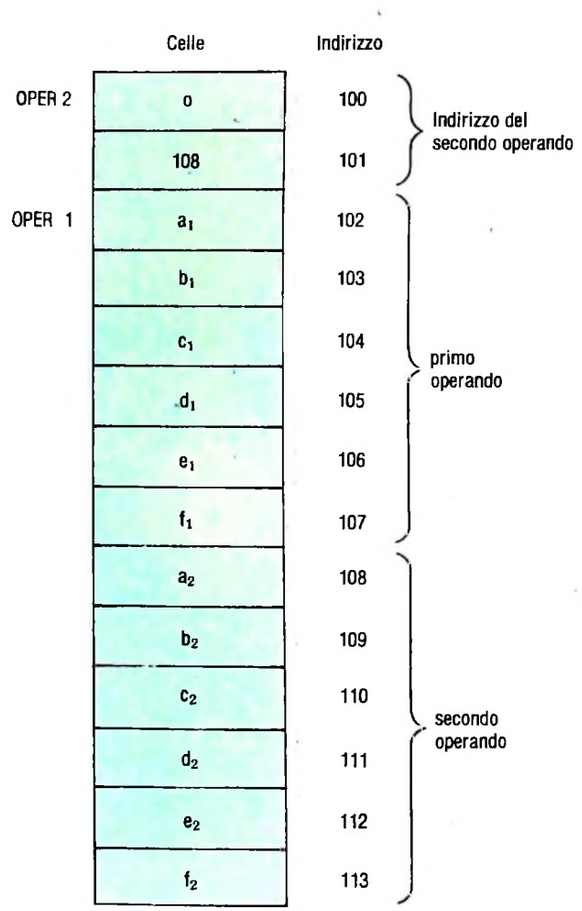
L'esempio qui riportato è senz'altro di impegnativa lettura e, di più, lo vediamo prima ancora di avere esaminato le caratteristiche del linguaggio vero e proprio; tuttavia proprio tale approccio, che abbiamo seguito per gli altri linguaggi di programmazione, avrà un significativo impatto che ci farà intuire i problemi e le difficoltà tecniche di tale programmazione.

L'esempio è una routine di somma di numeri interi di 6 byte scritta nel linguaggio del microprocessore Z80.

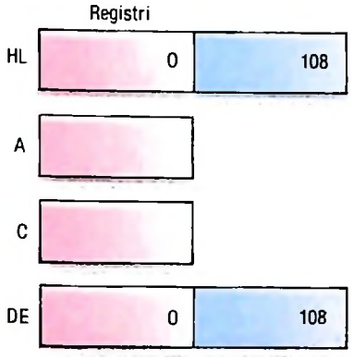
Il microprocessore è in grado di effettuare operazioni di somme di interi solo di 1 byte; quindi sarà necessario ricostruire l'intera operazione a partire da quest'ultima.

Il calcolatore ha registri di 1 byte (cioè 8 bit) come A, C, oppure visibili a coppie di 2 byte, come HL. Le celle sono tutte di 1 byte. Supponiamo che la memoria sia inizialmente nella seguente situazione (in cui riportiamo i valori numerici espressi in decimale anziché in binario):

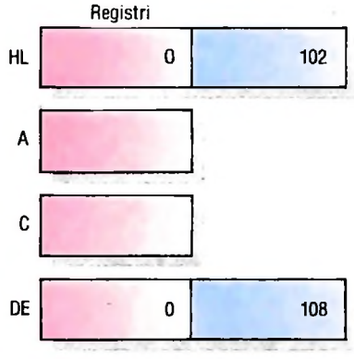




Le prime tre istruzioni caricano in HL il contenuto delle celle OPER 2 e successiva, quindi ricopiano H in D e L in E.

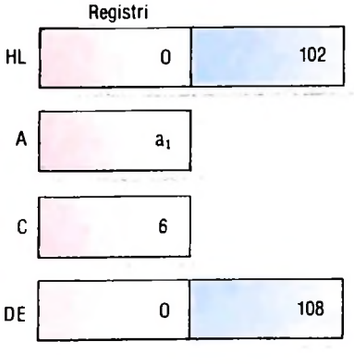


La quarta istruzione carica in H e L l'indirizzo del primo addendo (si noti che esso non è espresso tra parentesi); con ciò si intende richiedere l'indirizzo:

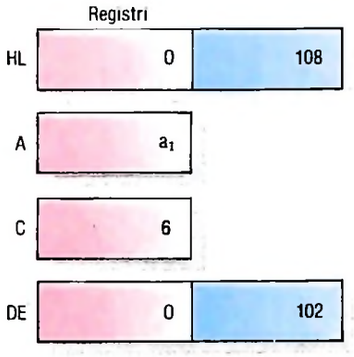


L'istruzione successiva (OR) ha solo lo scopo di eliminare la presenza di un eventuale "riporto" che viene registrato in un opportuno bit detto "flag di carry"; il fatto di compiere una operazione OR del registro A su se stesso metterà a zero tale flag, in quanto tale operazione non cambia il contenuto del registro A e quindi sicuramente non crea un "riporto". Quindi si carica sul registro C il valore 6, che deve contare le operazioni di somme che si dovranno fare.
A partire dall'etichetta LOOP:

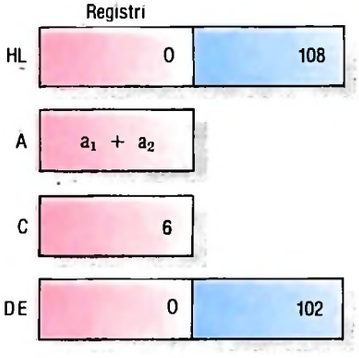
- si carica il registro A con il valore contenuto nella cella il cui indirizzo è in HL, cioè il contenuto della cella di indirizzo 102:



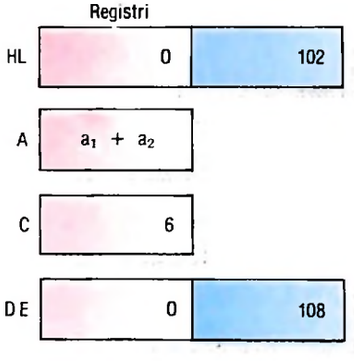
- si scambiano i contenuti della cella i cui indirizzi sono in HL e in DE:

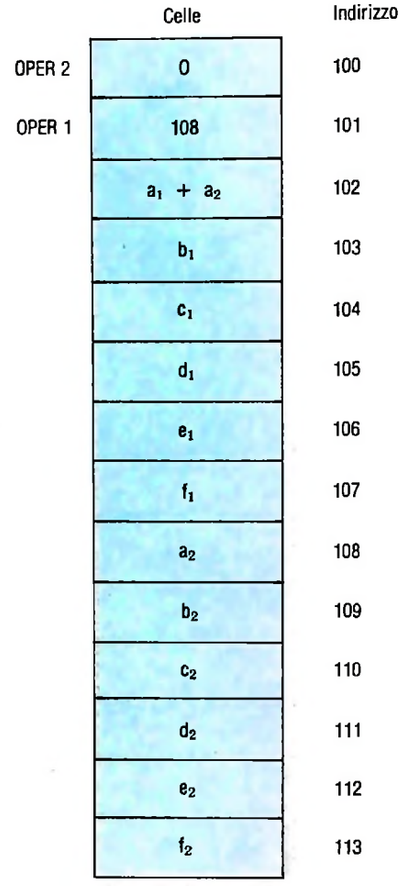


- si somma ad A il contenuto della cella il cui indirizzo è contenuto in HL, cioè della cella 108:

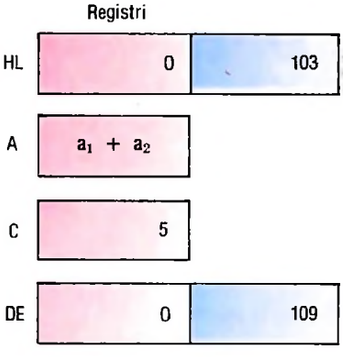


- quindi si riscambiano i contenuti della cella i cui indirizzi sono in HL e in DE e si salva il risultato parziale della somma di $a_1 + a_2$ nella cella il cui indirizzo è contenuto in HL:



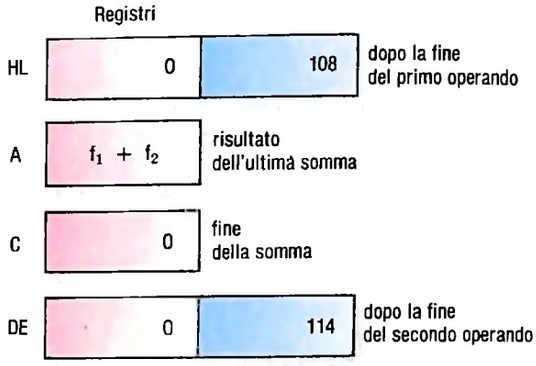


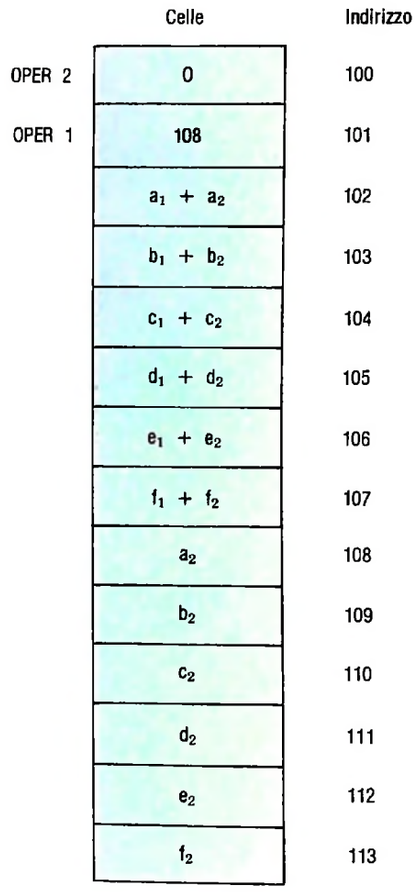
- si incrementano quindi DE e HL per indirizzare i due byte successivi e si decrementa C per ricordare che una somma delle sei è stata fatta:



- si controlla quindi che C non abbia raggiunto il valore zero, nel caso si salta all'etichetta FINE: altrimenti, come è il nostro caso, si torna alla etichetta LOOP, ove si rieffettuano le stesse operazioni sulle nuove celle.

Se immaginiamo che i numeri su 6 byte contenuti nei due operandi abbiano le parti più significative negli indirizzi più alti, l'operazione effettuata corrisponde ad avere sommato le due parti meno significative, salvando il risultato nella parte meno significativa del primo addendo; l'eventuale riporto è automaticamente gestito dall'istruzione ADC.
Alla fine avremo:





Vedremo in seguito i dettagli delle istruzioni assembler in modo più analitico.

I DATA-BASE RELAZIONALI: QUALCHE ESEMPIO

Una rapida panoramica dei sistemi più interessanti.

In questo terzo e ultimo articolo sui data-base, dopo aver esaminato i problemi della corretta organizzazione dei dati, daremo una rapida panoramica alle interfacce utente messe a disposizione da alcuni dei più interessanti data-base relazionali presenti sul mercato. Come più volte accennato, il successo dei data-base relazionali è in gran parte dovuto al tipo di interfaccia che propongono all'utente finale. Nei sistemi che analizzeremo l'utente finale interagisce con il DBMS (Data Base Management System) o attraverso un interprete in grado di riconoscere richieste espresse in un linguaggio di tipo pseudo-naturale (per le persone di madrelingua inglese), oppure esemplificando le proprie richieste direttamente sulle tabelle che rappresentano le relazioni, come nel caso del sistema QBE.

Ingres

Il sistema INGRES, che per primo analizziamo, è stato originariamente sviluppato nei laboratori universitari di Berkeley ed è in seguito diventato un prodotto commerciale, come spesso accade negli Stati Uniti.
INGRES (INteractive GRaphics and REtrieval System) è attualmente operante su una larga fascia di elaboratori, che vanno dai microcomputer 16 bit ai supermini della serie VAX e offre un ambiente relazionale in linea con quanto abbiamo visto negli articoli precedenti. Il data-base viene quindi visto dall'utente come un insieme di tabelle, sulle quali è possibile effettuare operazioni di inserimento, aggiornamento, cancellazione e selezione in maniera "vicina" alla mentalità dell'utente, alle prese nel mondo reale con strutture dati di tipo tabellare.
Oltre al modello relazionale di base, INGRES offre un certo insieme di strumenti di interazione con l'utente, tra i quali spicca il linguaggio di interrogazione QUEL. Questo linguaggio mette a disposizione una serie di comandi con una sintassi abbastanza semplice, come per esempio:
APPEND, COPY, CREATE, DEFINE, DELETE, DESTROY, HELP... che possono essere composti per ottenere comandi più potenti che rispondono a esigenze più complesse. QUEL è utilizzabile sia in modalità interattiva che batch. Nel primo caso INGRES mette a disposizione un interprete in grado di tradurre comandi espressi secondo la sintassi di QUEL nelle azioni corrispondenti, mentre nel secondo le istruzioni QUEL possono essere utilizzate all'interno di un normale programma, scritto per esempio in linguaggio C di cui sfrutta tutte le strutture di controllo e l'ambiente di programmazione.
In QUEL una relazione può essere creata ex novo con il comando CREATE oppure può essere formata a partire da altre preesistenti. Nel primo caso il comando:

```
CREATE inventario(
        oggetto = c20,
        fornitore = c20,
        quantita' = i2)
.
```

crea una relazione dal nome "inventario" i cui attributi "oggetto" e "fornitore" saranno stringhe di 20 caratteri e "quantità" sarà un numero intero tra -32768 e 32767 (i2 sta per intero a 2 byte). In questo modo abbiamo però creato solamente la struttura della relazione; per poter inserire delle ennuple in essa abbiamo a disposizione il comando APPEND. Per esempio:

```
APPEND TO inventario(oggetto = "matite",
                     fornitore = "Rossi",
                     quantita' = 100)
```

aggiungerà alla relazione la ennupla (matite, Rossi, 100).
In tutti i comandi è possibile utilizzare delle variabili al posto dei nomi delle relazioni per riferirsi agli attributi di queste; per esempio un risultato simile a quello del comando precedente potrebbe essere ottenuto con:

```
RANGE OF i IS inventario1
APPEND TO inventario2 ( i.oggetto,
                        i.fornitore,
                        i.quantita')
```

con cui abbiamo copiato i campi della relazione "inventario 1" in quelli della relazione "inventario 2" sfruttando la variabile i.
Una volta create le relazioni è possibile recuperare selettivamente l'informazione in esse contenuta.
Supponendo di aver definito una relazione impiegati con attributi nome, salario e manager, sarà possibile per esempio ottenere tutti i nomi degli impiegati che hanno un salario maggiore del loro manager, mediante l'uso del comando RETRIEVE nella forma:

```
RANGE OF i IS impiegati
RANGE OF m IS impiegati
RETRIEVE (i.nome) WHERE i.manager = m.nome
                  AND i.salario > m.salario
```

In questo caso la stessa relazione impiegato viene utilizzata come se fossero due distinte relazioni, cui ci si riferisce con le variabili i ed m; il comando funziona in quanto si suppone che anche i manager siano classificati nella relazione impiegati. Vengono ricercati quei nomi (i.nome) cui corrisponde un manager (i.manager), classificato nella relazione m (m.nome) e il cui salario (i.salario) sia maggiore di quello del manager corrispondente (m.salario).
Come abbiamo visto in questo esempio, è possibile utilizzare alcuni tipici operatori relazionali; per selezionare le ennuple delle relazioni; fra questi sono disponibili gli operatori $<$, $<=$, $>$, $>=$, $=$, $!=$ e anche gli operatori logici NOT, AND, OR. Associati ad INGRES esistono alcuni strumenti di supporto come il *query by forms*, che permette di interagire con il sistema in maniera grafica, evitando così all'utente l'apprendimento del linguaggio QUEL. Esistono inoltre due programmi di utilità appositamente studiati per stilare dei "report": fra questi il GRAPH BY FORMS permette di ottenere grafici a partire dai dati contenuti nelle relazioni sia su video che su supporto cartaceo.

Dbase II

Nel mondo dei microcomputer, quello dominato dai sistemi operativi CP/M e MS/DOS esiste un data-base relazionale che può essere considerato il progenitore della recente esplosione di questa gamma di prodotti sul mercato del software. Si tratta del Dbase II della Ashton-Tate. Il successo di questo prodotto è stato tale che un folto gruppo di società è nato e prospera attorno a questo prodotto fornendo corsi di introduzione, scrivendo applicazioni che si appoggiano su di esso e pubblicando libri che spiegano il suo uso ottimale. Anche se questo prodotto ha delle limitazioni, queste derivano più dalle capacità di memoria disponibili che dalla mancanza di funzionalità intrinseche. Dbase II rappresenta un potente strumento sia per scrivere semplici applicazioni, come un indirizzario, sia per applicazioni più complesse, come gestione di piccole e medie aziende.

L'interfaccia che Dbase II presenta all'utente è un interprete di comandi molto potente attraverso il quale è possibile creare, modificare e ispezionare in modo semplice le relazioni che formano le basi di dati.
L'interprete è abbastanza colloquiale, nel senso che in risposta ad un comando dell'utente il sistema propone delle domande che risolvono il problema di dover imparare una sintassi molto precisa con opzioni predefinite. Per esempio per creare una relazione si utilizza il comando CREATE. A questo punto viene richiesto il nome della relazione da definire e in seguito viene presentata la frase:

```
ENTER RECORD STRUCTURE AS FOLLOWS:
FIELD NAME, TYPE, WIDTH, DECIMAL PLACES
```

con cui il sistema richiede l'inserimento della struttura della relazione con la specifica del nome, del tipo (caratteri, numeri o espressioni logiche) e delle dimensioni degli attributi.
Dbase II permette l'utilizzo di una sola relazione alla volta, quindi è necessario specificare quella in uso mediante il comando USE seguito dal nome della relazione.
Per inserire le ennuple nella relazione selezionata è sufficiente digitare il comando APPEND. Questo comando farà apparire una maschera in cui l'utente potrà inserire i dati in ogni campo della relazione con la possibilità di correggere quanto scritto e di spostarsi da un campo all'altro con i comandi tipici di un word-processor.
Una volta inseriti i dati nella relazione è possibile ottenere un listato selettivo delle ennuple mediante i comandi LIST e DISPLAY. È possibile selezionare sia i campi che s'intendono visualizzare sia le caratteristiche delle ennuple cui si è interessati. Per esempio, avendo definito una relazione impiegati come quella precedentemente utilizzata, il comando:

```
USE impiegati
LIST nome, salario
```

produrrà una lista dei nomi e dei salari degli impiegati presenti nella relazione, mentre:

```
USE impiegati
LIST nome, salario FOR salario > 100000000
```

elencherà nome e salario di quei ricchi impiegati che guadagnano più di cento milioni all'anno.
Una caratteristica particolare di Dbase II è quella di poter essere utilizzato come strumento di programmazione con cui creare quelli che vengono definiti file di comandi, in cui si possono raggruppare sequenze di comandi semplici e quindi eseguirli in blocco con il comando DO seguito dal nome del file di comandi.

Dbase II come linguaggio di programmazione mette a disposizione le strutture di controllo tipiche della programmazione strutturata, quali IF... THEN... ELSE..., i cicli DO WHILE... ENDDO e la struttura di CASE.
Oltre a queste strutture sono disponibili potenti strumenti di debugging quali SET STEP ON o SET ECHO ON, con cui è possibile seguire passo passo l'esecuzione di un file di comandi.
Per modificare i dati contenuti nelle relazioni si possono usare i comandi EDIT (che permette di modificare una ennupla alla volta sempre utilizzando i semplici comandi di editor per spostarsi, inserire e correggere) e BROWSE (che presenta una tabella di tutte le ennuple in cui ci si sposta per le modifiche usando gli stessi comandi di editor).
Come per INGRES, sono stati creati una serie di strumenti accessori che servono per la generazione di report, per la creazione di interfacce a menù, per la definizione di maschere di inserimento e, inoltre, per ottenere una rappresentazione grafica dei dati.
In definitiva queste caratteristiche, e molte altre non citate, fanno sì che i data-base relazionali Dbase II, e il più recente Dbase III, rappresentino due strumenti di lavoro quasi indispensabili per chiunque debba trattare basi di dati su personal computer.

Query By Example

QBE (Query By Example) si pone in un ambiente totalmente differente da Dbase II: esso infatti è disponibile esclusivamente su grandi calcolatori o mainframe, in particolare su quelli IBM.
QBE è stato sviluppato presso i laboratori di ricerca IBM di San José e la sua filosofia seppur appartenente al filone relazionale si distacca in maniera netta dagli esempi visti in precedenza.
Esso basa la propria filosofia di utilizzo sul fatto che l'utente vede la creazione e la manipolazione delle informazioni come se queste fossero organizzate in modo simile alle tabelle bidimensionali.
La ricerca delle informazioni in queste tabelle avviene proponendo degli esempi con cui si rappresentano le risposte caratteristiche che si vorrebbero avere dal sistema.
Supponiamo di aver definito la relazione impiegati già vista in precedenza.
Operativamente l'utente richiama sul proprio terminale lo scheletro (privo di contenuti) della relazione semplicemente digitando il nome della stessa. Digitando impiegati apparirà la seguente struttura grafica tabellare:

Nome	Salario	Manager

Arrivati a questo punto è possibile avere una lista di tutti i nomi degli impiegati semplicemente inserendo nella tabella i seguenti dati:

Nome	Salario	Manager
P. Bianchi		

Il nome Bianchi è usato a titolo esemplificativo e si sarebbe potuta usare una qualsiasi altra stringa di caratteri, come X o XYZ; infatti le parole sottolineate sono da considerare variabili qualsiasi, anche senza alcuna connessione con il contenuto del data-base.
La lettera P. (P maiuscola seguita da punto) sta per l'istruzione abbreviata PRINT e indica che si vuole stampare la colonna in cui compare.
Allo stesso modo è possibile ottenere l'elenco di tutti gli impiegati il cui salario è superiore ai dieci milioni.
Sarà sufficiente indicare questa condizione nella colonna corrispondente:

Nome	Salario	Manager
P. Bianchi	>10000000	

Allo stesso modo si possono ricercare, come nell'esempio che abbiamo fatto per INGRES, tutti i nomi degli impiegati che percepiscono un salario superiore a quello del loro manager:

Nome	Salario	Manager
Rossi	X	
P. Bianchi	>X	Rossi

In questo caso il fatto che il nome Rossi e la lettera X compaiano in punti distinti è significativo, infatti indica che l'impiegato che guadagna X deve essere il manager di un altro che guadagna più di X.
Nel caso si vogliano avere delle corrispondenze precise sarà sufficiente indicarle inserendo nelle colonne opportune le parole non sottolineate.
Se per esempio si volessero conoscere tutti gli impiegati che hanno come manager proprio il signor Bianchi si dovranno inserire i seguenti dati:

Nome	Salario	Manager
P. XX		Bianchi

Tutte le operazioni che abbiamo visto sono possibili anche utilizzando più di una relazione alla volta; non solo. Si possono richiedere corrispondenze incrociate fra queste, cioè che attributi di una relazione siano posti in una certa relazione con attributi di un'altra relazione.
Le operazioni logiche di AND e OR sono facilmente realizzabili in QBE, sempre utilizzando il meccanismo delle "parole sottolineate". Per esempio per conoscere il nome del manager che dirige sia il signor Bianchi che il signor Rossi, si dovrebbe utilizzare la seguente forma:

Nome	Salario	Manager
P. X		
Bianchi		X
Rossi		X

mentre per sapere il nome di tutti gli impiegati che si trovano alle dipendenze di Bianchi o Rossi bisognerebbe usare quest'altra forma:

Nome	Salario	Manager
P. X		Bianchi
P. X		Rossi

Un operatore particolarmente importante è SUM, che permette di effettuare totalizzazioni.
Per conoscere la somma degli stipendi degli impiegati alle dipendenze di Rossi si userà:

Nome	Salario	Manager
	P. SUM ALL X	Rossi

Altrettanto importante è l'operatore COUNT, che svolge le funzioni di contatore.
Una domanda come "quante persone ha Rossi alle proprie dipendenze?" in QBE si porrà nel modo seguente:

Nome	Salario	Manager
P. COUNT ALL X		Rossi

Per concludere questa breve panoramica su QBE si deve ricordare che questo sistema ha il notevole vantaggio di realizzare il calcolo dei predicati a partire da una descrizione esemplificativa.
Ciò significa che è logicamente svincolato da una lingua particolare (come l'inglese) tranne che per i nomi di pochi operatori (SUM, COUNT, ALL ecc.).
Questo fattore distingue nettamente QBE dagli altri sistemi che sono generalmente basati su un linguaggio di interazione di tipo pseudo-naturale e quindi legati ad uno specifico insieme di vocaboli e di regole sintattiche che presumono la conoscenza della lingua inglese.

Conclusioni

Abbiamo visto in questa serie di articoli alcune delle caratteristiche più notevoli dei data-base e più in particolare abbiamo esaminato le caratteristiche di quelli relazionali. Di questi ci siamo occupati in quanto ci sembra rappresentino la strada più semplice per il trattamento dell'informazione e la manutenzione di consistenti banche di dati.
Le caratteristiche di facilità d'uso e comprensibilità di questi sistemi li rendono adatti anche agli utenti che non siano particolarmente preparati nel campo dell'informatica. I prodotti descritti in quest'ultimo articolo, sebbene mostrino già una notevole facilità d'uso, non rappresentano sicuramente le punte avanzate della ricerca in questo campo, ma sono prodotti già da tempo disponibili sul mercato e di sicuro affidamento.
Nello sviluppo dei data-base attualmente si tende a far diminuire il costo e le dimensioni dei sistemi su cui è possibile operare e si cerca di rendere ancora più semplice il "linguaggio" di interfaccia.
In quest'ultimo campo sono particolarmente interessanti alcuni prototipi di sistemi, in grado di operare su personal computer, che permettono la definizione del linguaggio di interrogazione da parte dell'utente, oppure i sistemi che sfruttano il "linguaggio" iconico (cioè quello strutturato per immagini), di cui si è già parlato in un articolo di questa stessa rubrica, per la definizione e l'interrogazione del data-base.

Lezione 45

Disegni colorati

Finora abbiamo usato il microplotter per scrivere caratteri e sequenze di caratteri, che, pur con le possibilità di variazione di dimensione e di colore, sfruttano ben poco le capacità della grafica di un plotter.
In questa lezione inizieremo a costruire veri e propri disegni.
I comandi che useremo allo scopo sono:

D per tracciare linee da un punto a un altro specificando in modo ASSOLUTO (specificando cioè le coordinate del punto di arrivo rispetto all'"origine" righe-colonne);

J per tracciare linee da un punto a un altro, come il comando precedente, ma con riferimento RELATIVO (cioè indicando il numero di righe e di colonne di cui spostarsi rispetto alla posizione del pennino in quell'istante).

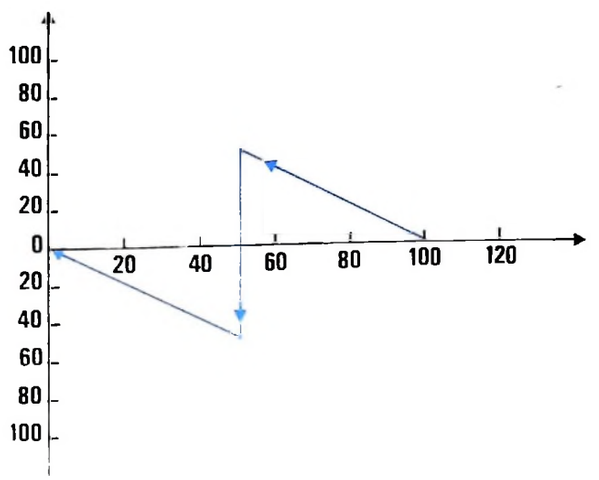
Ambedue i comandi devono essere seguiti da un insieme di coppie di numeri che specificano i vari punti da percorrere; ambedue devono essere inviati al microplotter con la solita istruzione LPRINT; ambedue hanno effetto solo quando si opera in modo grafico.
Così, immaginando di essere posizionati col pennino all'origine, cioè in un punto sul margine sinistro della carta che identifichiamo con la coppia (0,0) (cioè colonna numero 0, riga numero 0), l'istruzione

```
10 LPRINT "D100,0,50,50,50,-50,0,0"
```

traccerà:

- una linea dall'origine al punto di coordinate (100,0) (cioè, avanzerà fino alla colonna 100, mantenendosi sulla riga 0);
- quindi, una linea dal punto in cui si trova al punto di coordinate (50,50) (cioè alla colonna numero 50, tornando indietro, salendo in alto fino alla riga 50);
- quindi una linea dal punto in cui si trova al punto di coordinate (50,–50) (cioè, mantenendosi sulla stessa colonna scende alla riga –50, in verticale);
- infine, una linea che riporta il pennino al punto di partenza, di coordinate (0,0).

Il movimento e il disegno risultante sono illustrati nella seguente figura:



Per ottenere lo stesso risultato con il comando J, avremmo dovuto usare la seguente istruzione:

```
10 LPRINT "J100,0,-50,50,0,-100,-50,50"
```

che potremmo parafrasare come segue:

- "traccia una linea dal punto in cui ti trovi al punto che si trova 100 colonne più a destra, sulla stessa linea;
- quindi traccia una linea dal punto in cui ti trovi, spostandoti indietro di 50 colonne e in alto di 50 righe;
- quindi traccia una linea dal punto in cui ti trovi spostandoti, sulla stessa colonna, di 100 righe in basso;
- infine traccia una linea spostandoti indietro di 50 colonne e in alto di 50 righe".

Alla fine ci ritroveremo ancora nell'origine.
Usiamo ora questa istruzione per costruire un disegno colorato: disegneremo sul nostro microplotter la bandiera italiana.
Il programma sarà strutturato come segue:

- passaggio in modo grafico;
- posizionamento del pennino nell'origine (per evitare il rischio che, a seguito di esecuzioni precedenti, il pennino si trovi in una posizione non nota);
- tracciamento del contorno della bandiera;
- tracciamento delle suddivisioni verticali della bandiera;
- colorazione della banda verde;
- colorazione della banda rossa.

Il programma risultante è il seguente:

```
10 LPRINT CHR$(18) 'Graphic mode
20 REM Bandiera
25 LPRINT "H"  'Posizione iniziale
30 REM Contorno
40 LPRINT "D120,0,120,80,0,80,0,0"
50 LPRINT "M40,0" 'Prima suddivisione
60 LPRINT "D40,80"
70 LPRINT "M80,80" 'Seconda suddivisione
80 LPRINT "D80,0"
90 REM Colorazione
100 LPRINT "M1,1" 'Sopra l'origine
110 LPRINT "C2"  'Verde
120 FOR I=1 TO 79
130 LPRINT "J39,0" 'Linea orizzontale
140 LPRINT "R-39,1" 'Sposto in su
```

```
150 NEXT I
160 LPRINT "M81,1" 'Terzo settore
170 LPRINT "C3"    'Rosso
180 FOR I=1 TO 79
190 LPRINT "J39,0"
200 LPRINT "R-39,1"
210 NEXT I
```

in cui osserviamo:

- il passaggio in modo grafico alla linea 10;
- il posizionamento del pennino all'origine alla linea 25. Abbiamo usato qui un nuovo comando, H, che ha lo stesso effetto di un comando "M0,0" di posizionamento assoluto del pennino al punto di coordinate (0,0), senza tracciare alcuna linea;
- il tracciamento del contorno della bandiera alla linea 40: si tratta di un rettangolo di base 120 colonne e di altezza 80 righe;
- il posizionamento del pennino alla base della bandiera, a un terzo dal suo margine sinistro, alla linea 50, e il tracciamento della suddivisione verticale alla linea 60, risalendo fino al bordo superiore della bandiera;
- un'analoga operazione per la seconda suddivisione alle linee 70 e 80. Si noti che, essendo già precedentemente posizionati sul bordo superiore della bandiera, abbiamo preferito qui, per minimizzare gli spostamenti del pennino, "correre" sul suo bordo superiore e quindi tracciare la suddivisione dall'alto verso il basso;
- il posizionamento all'estremo inferiore sinistro della bandiera (non all'origine, perché dobbiamo colorare la parte interna, senza intaccare il bordo che abbiamo costruito), alla linea 100;
- la selezione del pennino verde alla linea 110;
- la ripetizione delle seguenti operazioni:
  - tracciamento di una linea verde orizzontale della lunghezza di 39 colonne, alla linea 130
  - ritorno indietro di 39 colonne e risalita di una riga all'istruzione 140 in modo tale che, eseguendo il ciclo FOR per tutte le 79 linee che costituiscono la bandiera, questa risulti con la prima banda completamente colorata in verde;
- spostamento alla base dell'ultima banda all'istruzione 160;
- selezione del pennino rosso all'istruzione 170;
- colorazione della banda nelle istruzioni successive.

Il risultato che otteniamo è il seguente:

Si osservi che le linee orizzontali del plotter sono così fitte, che la colorazione risulta piena, e non vi è più traccia del fatto che sia stata ottenuta riga per riga.

Possiamo ora sbizzarrirci a costruire le bandiere più sofisticate: un esercizio interessante potrebbe essere quello di tracciare la bandiera della Svizzera: una croce bianca su fondo rosso. Qui di seguito presentiamo il listato di un altro programma che disegna una bandiera con due "campi" tagliati in diagonale: uno di questi è colorato in rosso, mentre l'altro è lasciato in bianco.

```
10 LPRINT CHR$(18) 'Graphic mode
20 REM Bandiera diagonale
25 LPRINT "H" 'Posizione iniziale
30 REM contorno
40 LPRINT "D120,0,120,80,0,80,0,0,120,80"
50 REM Colorazione
60 LPRINT "M1,1" 'Sopra l'origine
70 LPRINT "C3" 'Rosso
80 FOR I=1 TO 79
90 LET A$="J"+STR$(I*120/80)+",0"
100 LPRINT A$
110 LET A$="R-"+STR$(I*120/80)+",1"
120 LPRINT A$
130 NEXT I
```

Si osservino le seguenti istruzioni:

- la linea 40 traccia non solo il contorno, ma anche la divisione diagonale tra i due campi
- le linee da 90 a 120 si occupano di colorare un campo, e allo scopo è necessario calcolare di volta in volta la lunghezza della traccia del colore; poiché partiamo dalla posizione (0,0) e dobbiamo raggiungere, in 80 righe, la posizione (120, 80) è necessario allungare la riga tracciata ogni volta di 120/80, cosa che viene fatta con la composizione di A$ in 90 e 110.

Cosa abbiamo imparato

In questa lezione abbiamo visto:

- il comando D del microplotter, per tracciare linee da un punto a un altro, con riferimento ASSOLUTO;
- il comando J del microplotter, per tracciare linee da un punto a un altro, con riferimento RELATIVO alla posizione corrente del pennino;
- il comando H del microplotter, per posizionare il pennino all'origine.

ALBUM ELETTRONICO (ALBICO)

Un programma che trasforma l'elaboratore in un album da disegno elettronico.

ALBICO è un programma BASIC per M10 con 32K RAM. È in grado di creare schemi e/o bozzetti sul display per poterli poi stampare (su stampante grafica tipo Radio Shack Line Printer VII e Epson) o registrare in memoria RAM o cassetta magnetica.
Il programma utilizza numerose subroutine in linguaggio macchina per permettere una maggior velocità di risposta nel caso di funzioni grafiche.
Il cuore del sistema è una bit-map area di 1920 byte utilizzati per registrare gli stati di on/off di ciascuno dei 15360 dot (punti) del display. Ciascun bit di questa area corrisponde a un punto dello schermo. Tramite il controllo dello stato di ciascun dot, la bit map rende possibile il trasferimento bit per bit dello schermo su stampante.

Modalità di funzionamento

Eseguendo il programma, per prima cosa apparirà il titolo in negativo al centro del display; dopo alcuni istanti lo stesso scomparirà lasciando il posto a un dot pulsante, posto proprio al centro dello schermo: a questo punto, utilizzando ciascuna delle quattro frecce a disposizione, potremo far muovere il dot a nostro piacimento lungo tutto il video. Per porre in esecuzione le funzioni del programma basterà digitare l'appropriata lettera e rispondere correttamente alle domande che il programma porrà: sullo schermo apparirà ciò che è stato richiesto.

Funzioni

(A)ssi - La funzione disegna gli assi cartesiani con origine nel centro dello schermo; tali assi sono suddivisi in unità di 5 e 10 dot.
(B)ox - La funzione permette di disegnare automaticamente una figura chiusa rettangolare o quadrata. Per eseguire tale funzione, bisogna per prima cosa porre il cursore nel punto dello schermo ove si vuole un angolo della figura, poi premere il tasto (P) e spostare il cursore nell'angolo diametralmente opposto (questo permetterà il dimensionamento della figura da disegnare).
A questo punto sarà sufficiente premere il tasto (B) per ottenere il corrispondente rettangolo o quadrato.
(C)erchio - La funzione consente di ottenere la raffigurazione di un cerchio con centro nel punto colorato. Premendo (P) sposteremo il cursore di un numero di dot equivalente al raggio del cerchio da disegnare. A questo punto sarà sufficiente digitare (C) e il cerchio verrà composto per coppie di punti secondo la regola della simmetria.
(D)isegna - La funzione (una volta premuto il corrispondente tasto) permette di spostare il punto sullo schermo lasciando il segno proprio come se si stesse usando una matita. Da ricordare che il movimento diagonale dovrà essere eseguito alternando i movimenti longitudinali e trasversali; questo perché M10 non riconosce la contemporanea pressione di due tasti.
(F)colora - La funzione, da utilizzare subito dopo un box, permette di colorare la figura creata. È poi possibile intervenire all'interno della stessa figura per disegnare con l'opposto colore.
(O)utput - Consente la memorizzazione dello schema creato su cassetta, printer o in RAM. La funzione richiamata chiederà il nome del file nel quale registrare oppure si assicurerà che il dispositivo stampante sia collegato.
Per la memorizzazione sarà necessario un tempo di circa 3 minuti (deve memorizzare tutta la bit-map).
(J)salta - La funzione controlla il numero di dot da "saltare" ogni volta si usi in contemporanea i tasti SHIFT + freccia. Verrà quindi richiesto il numero di dot da saltare lungo l'asse X e quelli lungo l'asse Y.
Impostati i valori, ogni qualvolta si premeranno i tasti SHIFT + freccia, il cursore si sposterà di tanti dot quanti definiti.
In caso di nessuna definizione, l'uso contemporaneo dei tasti SHIFT + freccia causerà lo spostamento del cursore agli estremi (sinistro-destro o alto-basso) dello schermo.
(L)inea - Causa il congiungimento di due punti (di cui uno spostato con la funzione P) per mezzo di una linea.
L'uso immediato della funzione traccerà una retta tra l'estremo alto a sinistra e il punto corrente.
(M)uovi - Permette il movimento del punto lungo lo schermo senza lasciare alcuna traccia, proprio come quando si alza la punta della matita dal foglio di carta.
(P)untatore - La funzione consente di spostare il punto che rappresenterà il riferimento per l'esecuzione di funzioni grafiche come il cerchio, il box o la linea. Per rendere attiva l'operazione, è sufficiente premere il tasto (P) e quindi spostare il cursore dove si vuole.
(Q)uit - Causa l'uscita dal programma (dopo che è stata confermata l'intenzione rispondendo sì alla sua richiesta).
(S)ripeti - Premendo il tasto S si causa la ripetizione della

figura creata prendendo come riferimento il vertice più a sinistra.
(W)dove - Ritorna i valori dell'ascissa e dell'ordinata del punto in cui ci si trova. Premendo poi un tasto qualsiasi l'informazione sparisce lasciando posto a una croce centrata sul punto in questione.
(X)Y-set - La funzione permette di definire l'ascissa e l'ordinata del punto dove ci si vuole spostare. Dopo la dichiarazione delle coordinate, il cursore si posizionerà automaticamente ove richiesto.
(1)nero - Permette di disegnare in nero su bianco (colore di inizializzazione).
(0)bianco - Esegue il contrario. Tale funzione è utilizzata come "somma" per cancellare, basterà infatti eseguire le funzioni (O) e (D) per disegnare in bianco su bianco. Se si ripercorre, in questa situazione, un tratto disegnato, lo stesso verrà "cancellato" dot per dot.
Alcune funzioni, pur essendo presente la routine, non sono state ancora implementate (in caso di richiamo delle stesse il programma le interpreterà come QUIT).

Installazione dei programmi

Per installare i programmi è necessario svolgere le seguenti operazioni:

- trasferire per prima cosa programmi e file del proprio M10 su cassetta. Questa operazione è necessaria per permettere alla macchina di possedere più memoria.
- utilizzando il programma TEXT, si dovrà in secondo luogo digitare nel file HEX.DO il contenuto del listato n. 2 (senza l'intestazione), in modo continuo senza utilizzare spazi né ritorno carrello;
- terminata la digitazione premere il tasto funzione F8. Caricare a questo punto il listato n. 3 (LOAD.BA); questo programma leggerà il contenuto del file Hex, calcolerà i corretti indirizzamenti e caricherà le istruzioni in linguaggio macchina in memoria.
  Il programma esegue anche un accurato controllo, detto CHECKSUM, sul contenuto del file Hex.
  Nel caso si verificassero errori di digitazione nello stesso file, il programma LOAD avvertirà l'utente e interromperà il caricamento. In questo caso si consiglia di ricontrollare il contenuto del file Hex.
  In caso di caricamento completato correttamente, le istruzioni in linguaggio macchina verranno memorizzate nel file SKSUBR.CO. Tale file è protetto e quindi ogni istruzione di RETURN non ne visualizzerà il contenuto, ma riporterà il sistema al menù principale.
- a questo punto memorizzare su nastro sia HEX.DO che LOAD.BA, cancellarli dalla memoria del computer e digitare il listato n. 1.

```
*****************************************************************************
*                              Listato nr.1                                 *
*                              ------------                                 *
*                                by C.V.P.                                  *
*                                                                           *
*****************************************************************************
```

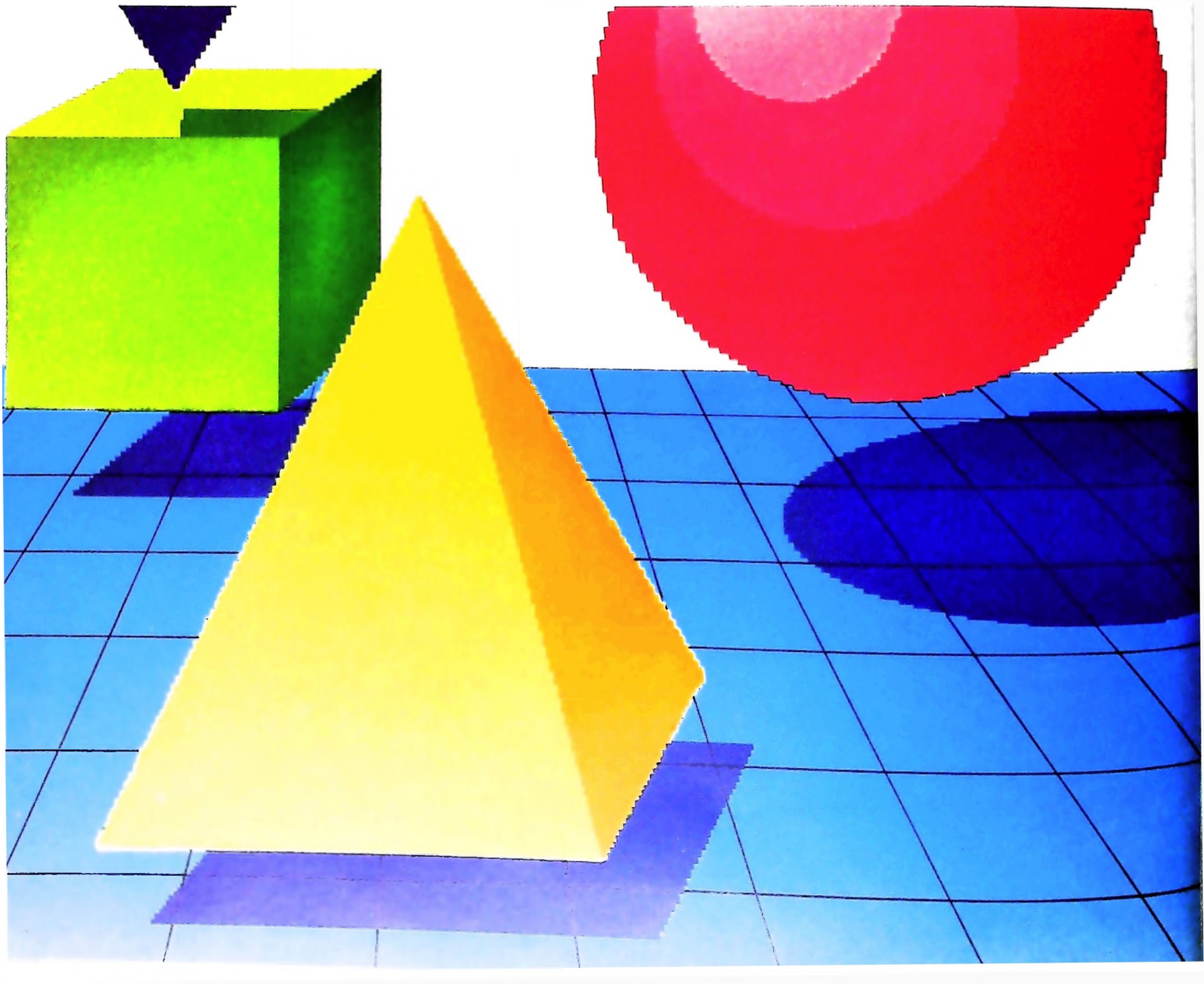
```
1 CLEAR100,MAXRAM-2961
 :LOADM"sksubr"
 :GOSUB3000
 :GOSUB6000
 :GOTO20
2 SC=CH
 :IFCH>DETHEN9ELSEIFCH=DETHENCH=N4
 :GOTO8
3 IFCH>CZTHEN9ELSEIFCH=CZTHENCH=N3
 :GOTO8
4 IFCH>CRTHEN9ELSEIFCH=CRTHENCH=N5
 :GOTO8
5 IFCH>N2THEN9ELSEIFCH=NOTHENCH=N1
 :GOTO9
8 PRINT#1,C2$;
9 PRINT#1,CHR$(CH);
 :CH=SC
 :RETURN
20 GOSUB50
30 K$=""
 :IFK>96THENK=K-32
32 K=K-64
 :IFK<>6THENB=0
 :C=0
33 IFK=-15THENK=25
34 IFK=-16THENK=26
35 IFK<1THEN20
40 ONKGOSUB 1900,200,300,400,94,600,2600,
5000,700,1000,1400,1200,94,1800,100,1600,
1700,2400,900,2000,94,2200,2300,2100,500,
1100:GOTO20
50 IFK$<>""THENRETURNELSEPSET(X,Y)
52 FORI=1TO45
 :K$=INKEY$
 :IFK$<>""THENI=45
53 NEXTI
54 PRESET(X,Y)
 :IFK$<>""THEN60
56 FORI=1TO45
 :K$=INKEY$
 :IFK$<>""THENI=45
57 NEXTI
58 IFK$=""THEN50
60 K=ASC(K$)
 :GOSUB99
 :IFPEEK(S!+HL)ANDATHENPSET(X,Y)
62 IFK=1THENX=X-JX
 :GOTO90
64 IFK=2THENX=X+JX
 :GOTO90
66 IFK=6THENX=X+JX
 :GOTO90
68 IFK=17THENX=0
 :GOTO90
70 IFK=18THENX=239
 :GOTO90
```

```
72 IFK=20THENY=Y-JY
  :GOTO90
74 IFK=23THENY=0
  :GOTO90
76 IFK=26THENY=63
  :GOTO90
78 IFK=28THENX=X+1
  :GOTO90
80 IFK=29THENX=X-1
  :GOTO90
82 IFK=30THENY=Y-1
  :GOTO90
84 IFK=31THENY=Y+1
90 IFX<0THENX=0ELSEIFX>239THENX=239
92 IFY<0THENY=0ELSEIFY>63THENY=63
94 RETURN
95 K$=INKEY$
  :IFK$=""THEN95ELSEK=ASC(K$)
  :RETURN
99 A=X*64+Y
  :HL=A/8
  :A=2^(AMOD8)
  :RETURN
100 CLS
   :PRINT
   :PRINT"Uscita su (R)am, (C)as,
(P)rinter,"
   :PRINTTAB(7)"oppure (N)essuna uscita?"
102 GOSUB95
104  IFK<>78ANDK<>110THEN108
106 K$=""
   :GOSUB2500
   :RETURN
108  IFK<>82ANDK<>114THEN150
110 GOSUB180
   :GOSUB112
   :GOTO118
112 OPENFI$FOROUTPUTAS1
   :PRINT
   :PRINT"schema in uscita,attendere..."
114 M!=S!
   :L1!=V!-N1!
   :RETURN
118 IFM!>L1!THEN148ELSECH=PEEK(M!)
   :CN=N1
120 M!=M!+N1!
   :IFPEEK(M!)=CHANDM!<V!THENCN=CN+N1
   :GOTO120
122 IFCN>N5THEN130
124 FORI=N1TOCN
126 GOSUB2
   :NEXTI
128 GOTO118
130 IFCN>FETHENPRINT#1,C2$;CF$;
   :GOSUB2
   :CN=CN-FF
   :GOTO130
132 IFCN>DETHENPRINT#1,C2$;CHR$(CN);
   :GOSUB2
   :GOTO118
134 IFCN=DETHENGOSUB2
   :PRINT#1,C2$;CD$;
   :GOSUB2
   :GOTO118
135 IFCN=CZTHENGOSUB2
   :PRINT#1,C2$;CZ$;
   :GOSUB2
   :GOTO118
136 IFCN=CRTHENGOSUB2
   :PRINT#1,C2$;CR$;
   :GOSUB2
   :GOTO118
137 IFCN>N5THENPRINT#1,C2$;CHR$(CN);
   :GOSUB2
   :GOTO118ELSEIFCN=NOTHEN118ELSE124
148 CLOSE
   :CALLV!
   :K$=""
   :RETURN
150  IFK<>67ANDK<>99THEN170
152 GOSUB180
   :GOSUB154
   :GOTO158
154 FI$="cas:"+FI$
   :PRINT
   :PRINT"Premi ENTER quando  registr.
pronto"
156 GOSUB95
   :RETURN
158 GOSUB112
   :FORM!=S!TOL1!
160 PRINT#1,PEEK(M!)
   :NEXTM!
162 GOTO148
170  IFK<>80ANDK<>112THEN102ELSE800
180 PRINT
   :LINEINPUT"Nome file: ";FI$
182 RETURN
200 IFX=PXANDY=PYTHENRETURN
202 LINE(PX,PY)-(X,Y),C0,B
204 HL!=PX*256+PY
206 POKEX2!,PX
   :POKEY2!,Y
   :CALLL!,C0,HL!
210 POKEX2!,X
   :POKEY2!,PY
   :CALLL!,C0,HL!
212 HL!=X*256+Y
214 POKEX2!,PX
   :POKEY2!,Y
   :CALLL!,C0,HL!
218 POKEX2!,X
   :POKEY2!,PY
   :CALLL!,C0,HL!
220 B=1
   :RETURN
300 R2!=(PX-X)^2+(PY-Y)^2
   :LC=FIX(SQR(R2!/2)+.5)
   :XX=X
   :YY=Y
302 FORCY=PYTOPY-LCSTEP-1
304 CX!=PX-SQR(R2!-(PY-CY)^2)
   :CX=FIX(CX!+.5*SGN(CX!))
306 GOSUB350
308 NEXTCY
310 FORCX=CXTOPX
312 CY!=PY-SQR(R2!-(PX-CX)^2)
   :CY=FIX(CY!+.5*SGN(CY!))
314 GOSUB350
316 NEXTCX
318 X=XX
   :Y=YY
   :IFP=1THENP=0
   :GOSUB1650
320 C=1
   :RETURN
```

```
350 IFCY<0AND(PY+PY-CY))63THENCX=PX
    :CY=PY-LC
    :RETURN
351 X=CX
    :Y=CY
    :IFX)=0ANDX<240ANDY)=0ANDY<64THENIFF=
0THENGOSUB380ELSEGOSUB390ELSEIFF=
1THENGOSUB390
352 LY=PY+PY-CY
    :Y=LY
    :X=CX
    :IFY<64ANDX)=0THENIFF=0THENGOSUB380ELS
EGOSUB390ELSEIFF=1THENGOSUB390
354 RX=PX+PX-CX
    :X=RX
    :Y=CY
    :IFX<240ANDY)=0THENIFF=0THENGOSUB380EL
SEGOSUB390ELSEIFF=1THENGOSUB390
355 Y=LY
    :X=RX
    :IFY<64ANDY<240THENIFF=0THENGOSUB380EL
SEGOSUB390ELSEIFF=1THENGOSUB390
356 RETURN
380 GOSUB99
    :IFC0=1THENPSET(X,Y)
    :CALL0!,A,HLELSEPRESET(X,Y)
    :CALLF!,A,HL
382 RETURN
390 IFY<00RY)63THENRETURN
391 IFX<0THENX=0ELSEIFX)239THENX=239
392 LINE(X,Y)-(PX,PY),C0
393 HL!=X*256+Y
394 POKEX2!,PX
    :POKEY2!,Y
    :CALLL!,C0,HL!
396 RETURN
400 0X=X
    :0Y=Y
402 K$=""
    :GOSUB50
404 IFK)31THENRETURN
406 IFX=0XANDY=0YTHEN402
408 XX=X
    :YY=Y
410 IF0X=XTHEN412ELSEIFX)0XTHENX=X-1ELSEX
=X+1
412 IF0Y=YTHEN414ELSEIFY)0YTHENY=Y-1ELSEY
=Y+1
414 LINE(0X,0Y)-(X,Y),C0
416 XL!=0X*256+0Y
418 POKEX2!,X
    :POKEY2!,Y
    :CALLL!,C0,HL!
420 X=XX
    :Y=YY
    :GOTO400
500 C0=1
    :RETURN
600 IFB=0THEN620ELSEXX=X
    :YY=Y
602 IFABS(PX-X)=10RABS(PY-Y)=1THENB=0
    :X=XX
    :Y=YY
    :RETURN
604 IFABS(PX-X))2THENIFPX)XTHENX=X+1ELSEX
=X-1
606 IFPX)XTHENPX=PX-
1ELSEPX=PX+1
608 IFABS(PY-Y))2THEN1FPY)YTHENY=Y+1ELSEY
=Y-1
610 IFPY)YTHENPY=PY-1ELSEPY=PY+1
612 GOSUB202
    :GOTO602
620 IFC=0THENRETURNELSEF=1
622 GOSUB300
    :F=0
    :C=0
    :RETURN
700 CLS
    :PRINT
    :PRINT"Input da (R)am, (C)as,
(N)essuno?"
702 GOSUB95
704 IFK$()"N"ANDK$()"n"THEN708
706 K$=""
    :GOSUB2500
    :RETURN
708 IFK$()"R"ANDK$()"r"THEN760
710 GOSUB180
    :GOSUB712
    :GOTO716
712 OPENFI$FORINPUTAS1
    :PRINT
    :PRINT"Carico schema, attendere..."
    :M!=S!
    :RETURN
714 IFEOF(1)THEN715ELSE716
715 CLOSE
    :CALLV!
    :K$=""
    :RETURN
716 CH=ASC(INPUT$(1,1))
718 IFCH=N2THEN730
720 IFCH=N1THENCH=N0
722 POKEM!,CH
    :M!=M!+N1!
    :IFM!=V!THEN715ELSE714
730 CH=ASC(INPUT$(1,1))
732 IFCH<N6THEN750
734 CN=CH
    :CH=ASC(INPUT$(1,1))
736 IFCH=N1THENCH=N0
    :GOTO744
738 IFCH=N2THENCH=ASC(INPUT$
(1,1))ELSE744
740 IFCH=N10RCH=N2THEN744
742 IFCH=N3THENCH=CZELSEIFCH=N5THENCH=
CRELSECH=DE
744 FORI=N1TOCN
    :POKEM!,CH
    :M!=M!+N1!
    :IFM!=V!THENI=CN
746 NEXT
    :IFM!=V!THEN715ELSE714
750 IFCH=N10RCH=N2THEN722
752 IFCH=N3THENCH=CZ
    :GOTO722
754 IFCH=N5THENCH=CR
    :GOTO722
755 CH=DE
    :GOTO722
760 IFK$()"C"ANDK$()"c"THEN702
762 GOSUB180
    :GOSUB154
    :GOSUB712
768 FORM!=S!TOV!-1
```

```
770 INPUT#1,CH
   :POKEM!,CH
   :NEXTM!
772 GOTO715
800 CLS
   :PRINT
   :PRINT"Inserire margini sinistri come"
   :PRINT"numero tra 0 e 39,"
   :PRINT"N per uscire senza stampare."
802 PRINT
   :LINEINPUT"margini:";M$
804 IFM$()"N"ANDM$()"n"THEN808
806 GOSUB2500
   :K$=""
   :RETURN
808 M=VAL(M$)
   :IFM(0ORM)39THENB00
810 PRINT
   :PRINT"Premi un tasto quando stamp
.pronta"
812 GOSUB95
   :GOSUB850
   :GOTO806
850 LPRINTCHR$(18);
   :FORH=0TO7
   :GOSUB880
852 FORM!=S!+HTOS!+1912+HSTEP8
854 CL=(PEEK(M!)AND(2^(7-H)-1))*2^H
856 CH=PEEK(M!-1)\2^(8-H)
858 LPRINTCHR$((CHORCL)OR128);
860 NEXTM!
   :LPRINTCHR$(13);
   :NEXTH
862 GOSUB880
   :FORM!=S!+7TOS!+1919STEP8
864 LPRINTCHR$(PEEK(M!)OR128);
866 NEXTM!
   :LPRINTCHR$(13);
868 GOSUB880
   :FORM!=S!+7TOS!+1919STEP8
870 LPRINTCHR$((PEEK(M!)\128)OR128);
872 NEXTM!
   :LPRINTCHR$(13);
874 LPRINTCHR$(30)
   :RETURN
880 IFM()0THENLPRINTCHR$(28);CHR$(M*6)
;CHR$(128);
   :RETURNELSERETURN
900 IFPX=XANDPY=PTHENRETURN
902 IFPX<XTHENIX=PX
   :XI=XELSEIX=X
   :XI=PX
904 IFPY<YTHENIY=PY
   :YI=YELSEIY=Y
   :YI=PY
906 IFP=1THENP=0
   :GOSUB1650
908 LINE(IX,IY)-(XI,YI),C0,B
910 IB=1
   :RETURN
1000 CLS
     :PRINTCHR$(155)CHR$(154);
     :INPUT"salta:quanti punti (1-239)";JX!
     :IFJX!<1ORJX!>239THEN1000
1002 PRINTCHR$(152)CHR$(153);
     :INPUT"salta:quanti punti
(1-63)";JY!
     :IFJY!<1ORJY!>63THEN1002
1004 GOSUB2500
     :JX=JX!
     :JY=JY!
     :RETURN
1100 C0=0
     :RETURN
1200 IFX=PXANDDY=PYTHEN RETURN
1202 LINE(PX,PY)-(X,Y),C0
1204 IFPY>YTHEN1210
1206 HL!=PX*256+PY
1208 POKEX2!,X
     :POKEY2!,Y
     :CALLL!,C0,HL!
     :RETURN
1210 HL!=X*256+Y
1212 POKEX2!,PX
     :POKEY2!,PY
     :CALLL!,C0,HL!
     :RETURN
1400 PRINT@280,"Cancello l'ultimo
schema(s/n)";
1401 GOSUB95
1402 IFK$="Y"ORK$="y"THEN1404
1403 GOSUB2500
     :K$=""
     :RETURN
1404 CLS
     :X=120
     :Y=32
     :CALLA!,0
     :AX=0
     :G=0
     :IB=0
     :XI=0
     :IX=0
     :IY=0
     :YI=0
     :P=0
     :C0=1
     :R=0
     :GOTO1403
1600 PX=X
     :PY=Y
     :P=1
1602 GOSUB99
     :IF(PEEK(S!+HL)ANDA)<>0THENPRESET(X,Y)
     :CALLF!,A,HLELSEPSET(X,Y)
     :CALLO!,A,HL
1604 RETURN
1650 XX=X
     :YY=Y
     :X=PX
     :Y=PY
     :GOSUB1602
1652 X=XX
     :Y=YY
     :RETURN
1700 PRINT@280,"Rilascio schema (s/n)";
1701 GOSUB95
     :IFK$="S"ORK$="s"THENMENUELSEGOSUB2500
     :K$=""
     :RETURN
1800 R=NOTR
     :CALLN! :CLS
     :GOSUB2500
     :RETURN
1900 IFR=0THENAC=
1ELSEAC=0
```

```
20 OPEN"HEX"FORINPUTAS1
30 PRINT
   :PRINT"CARICAMENTO 'HEX.DO' IN
MEMORIA..'
40 HX$=INPUT$(2,1)
50 D$=LEFT$(HX$,1)
55 IFD$="*"THEN300
60 GOSUB500
65 IFD=-1THEN200
70 DC=D*16
80 D$=RIGHT$(HX$,1)
90 GOSUB500
95 IFD=-1THEN200
100 DC=DC+D
    :CS#=CS#+DC
110 POKE(M!+I),DC
    :I=I+1
120 IFEOF(1)THEN130ELSE40
130 CLOSE
    :IFCS#=77855ANDI=1030THEN140
132 PRINT
    :PRINT"ERRORE DI CHECKSUM. CONTROLLARE
CONTENUTO"
134 PRINT"DI 'HEX.DO' E RERUN IL PROGRAMMA
CON IL FILE CORRETTO"
138 END
140 PRINT
    :PRINT"CARICAMENTO ESEGUITO. 'SKSUBR.CO'
E' STATO CREATO."
150 SAVEM"sksubr",M!,M!+1029,M!+155
160 END
200 CLOSE
    :PRINT
    :PRINT"CARATTERE NON HEX IN 'HEX.DO':"
210 PRINT
    :PRINT"COPPIE #";I+1;"LETTE:";HX$
220 PRINT
    :PRINT"PREGO CORREGGERE E RERUN"
    :END
300 D$=RIGHT$(HX$,1)
310 GOSUB500
320 IFD=-1THEN200
330 CS#=CS#+D
340 IFD=10THENDC=(AD-1)*16+DELSEDC=AD*16+D
350 GOTO110
500 D=ASC(D$)
502 IFD>47ANDD<58THEND=D-48
    :RETURN
504 IFD>64ANDD<71THEND=D-55
    :RETURN
506 D=-1
    :RETURN
```



UN NUOVO MODO DI USARE LA BANCA.

Conto corrente più

TANTI PENSIERI IN MENO CON IL CONTO CORRENTE "PIU'" DEL BANCO DI ROMA.

Essere cliente del Banco di Roma vuol dire anche essere titolari del conto corrente "più". Un conto corrente più rapido: perché già nella maggior parte delle nostre filiali trovate gli operatori di sportello che vi evitano le doppie file.

Più comodo, perché potete delegare a noi tutti i vostri pagamenti ricorrenti: dai mutui all'affitto, dalle utenze alle imposte.

Più pratico, perché consente l'utilizzo del sistema di prelievo automatico Bancomat e l'ottenimento della carta di credito.

Inoltre un servizio utilissimo, soprattutto per imprenditori e commercianti denominato "esito incassi", consente di avere comunicazione dell'eventuale insolvenza entro solo cinque giorni dalla scadenza. Una opportunità veramente speciale.

Più sicuro, perché con una minima spesa potrete assicurarvi contro furti e scippi mentre vi recate in banca o ne uscite.

Veniteci a trovare, ci conosceremo meglio.

BANCO DI ROMA
CONOSCIAMOCI MEGLIO.

Anche in leasing con Olivetti Leasing.

PERSONAL COMPUTER OLIVETTI M10

L'UFFICIO DA VIAGGIO

Olivetti M10 vuol dire disporre del proprio ufficio in una ventiquattrore. Perché M10 non solo produce, elabora, stampa e memorizza dati, testi e disegni, ma è anche capace di collegarsi via telefono per spedire o ricevere informazioni.

Qualunque professione sia la vostra, M10 è in grado, dovunque vi troviate, di offrirvi delle capacità di soluzione davvero molto grandi. M10: il più piccolo di una grande famiglia di personal.

olivetti

Per informazioni rivolgersi ai negozi contrassegnati da "Olivetti M10 Punto di Vendita" o inviate in coupon a: Olivetti, Divisione Personal Computers, Via Meravigli 12, 20123 Milano

NOME/COGNOME

VIA/N.

CAP/CITTÀ

TELEFONO