

Spediz. in abbonamento postale GR II/70 L. 2.200
(...)

52 CORSO PRATICO COL COMPUTER

422048

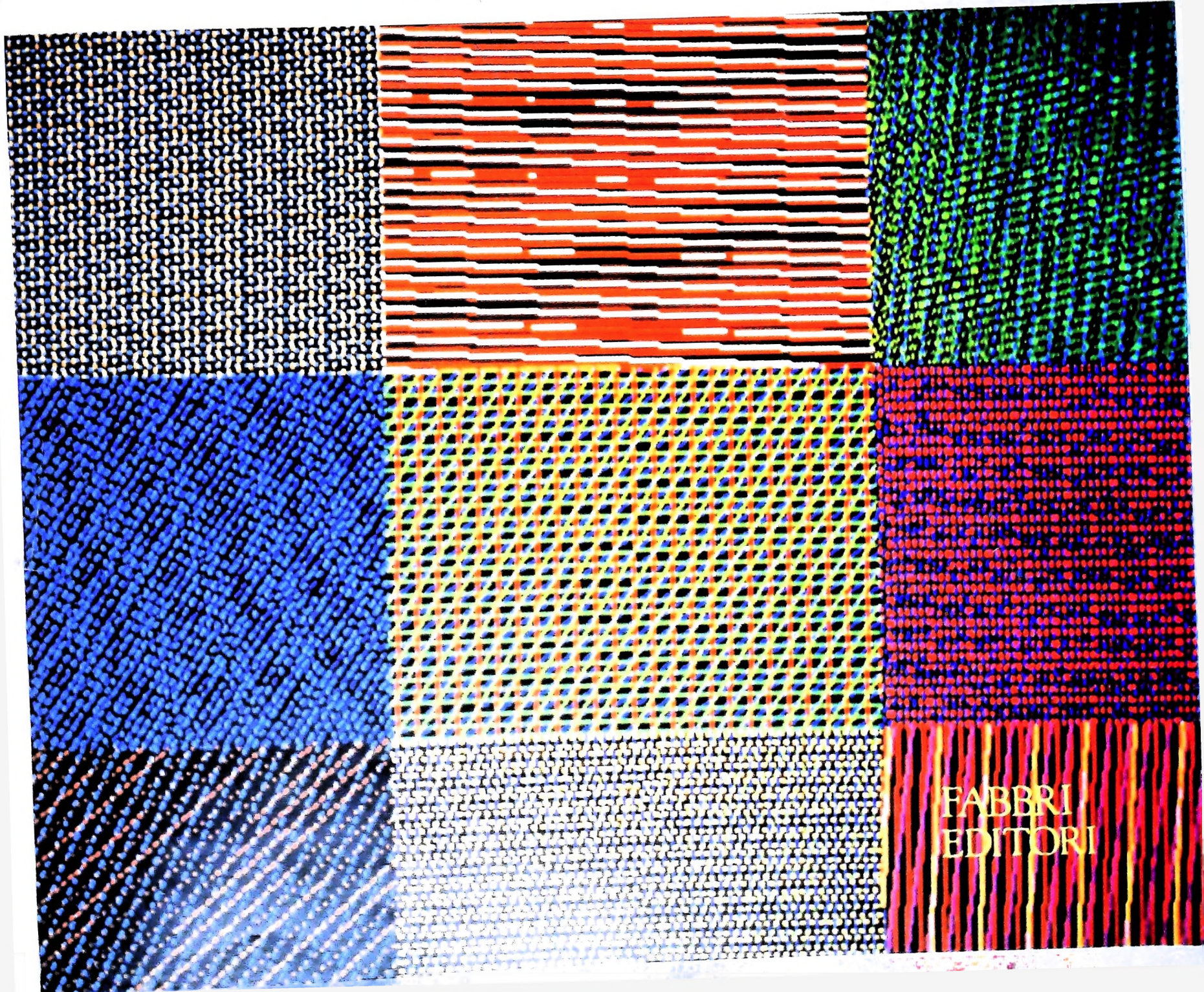
diretto da GIANNI DEGLI ANTONI

è una iniziativa
FABBRI EDITORI

in collaborazione con
BANCO DI ROMA

e **OLIVETTI**

BATTERY LOW



**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 giorni 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 massimo 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 trattene 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
GIANCARLO MAURI, ALDO GRASSO, ANDREA GRANELLI, EIDOS (BRUNO MOTTA)

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 FENGLI

Redazione
CARLA VERGANI
LOGICAL STUDIO COMMUNICATION

Art Director
CESARE BARONI

Impaginazione
BRUNO DE CHECCHI
PAOLA ROZZA

Programmazione Editoriale
ROSANNA ZERBARINI
GIOVANNA BREGGE

Segretarie di Redazione
RENATA FRIGOLI
LUCIA MONTANARI

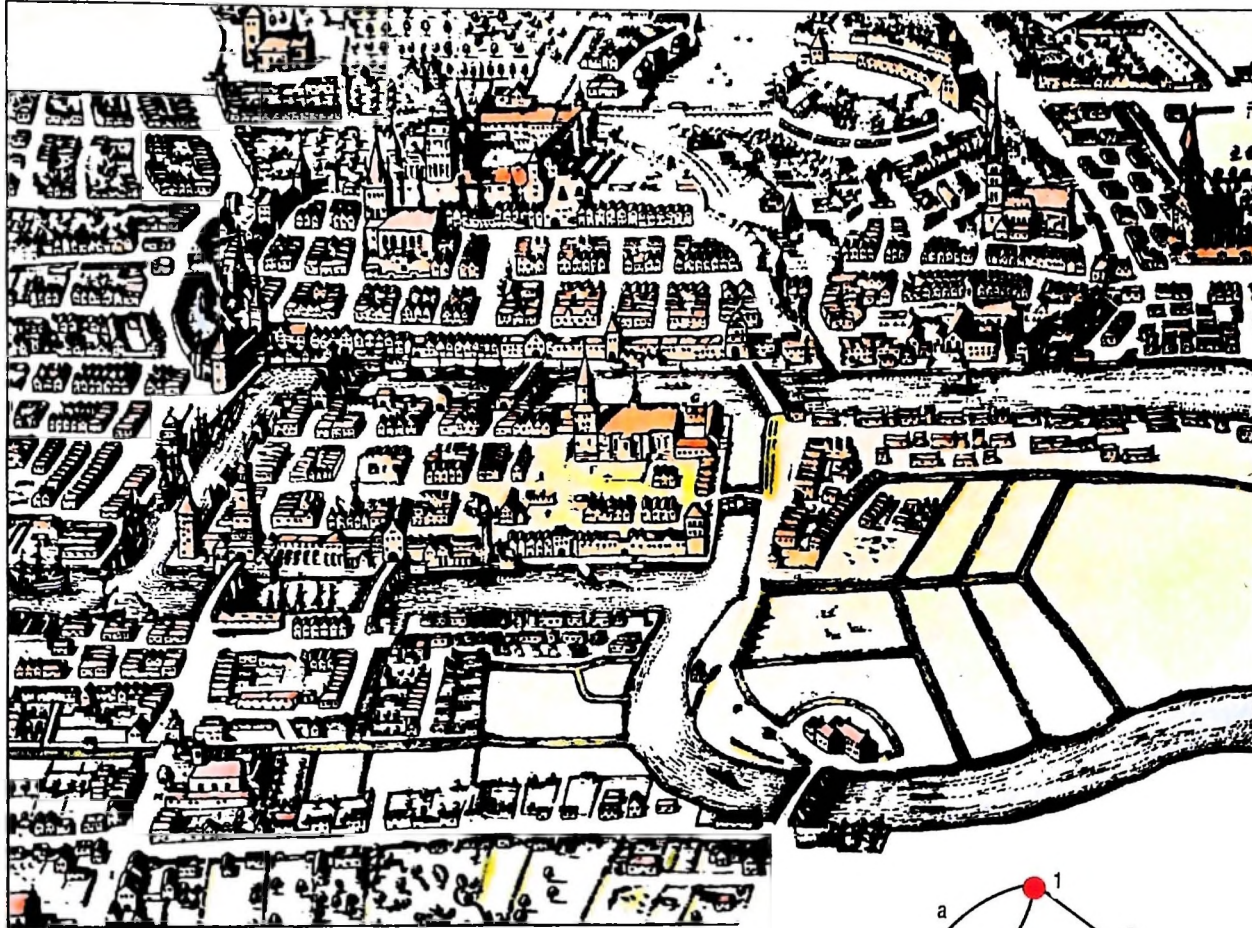
AVVISO AI LETTORI

Questa settimana è in edicola la copertina per rilegare il quarto volume del "Corso pratico col computer".

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. 52 - 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.

LA TEORIA DEI GRAFI

Uno strumento matematico molto importante in diversi settori dell'informatica.



In alto: le quattro zone da cui era costituita la città di Königsberg nel 1736 erano collegate tra loro da sette ponti sul fiume Pregel. A lato: il grafo che rappresenta i collegamenti tra le diver-

se zone di Königsberg. Ogni zona della città è rappresentata da un punto (nodo) e ogni ponte da un segmento (arco) che unisce i nodi corrispondenti alle zone che esso collega.

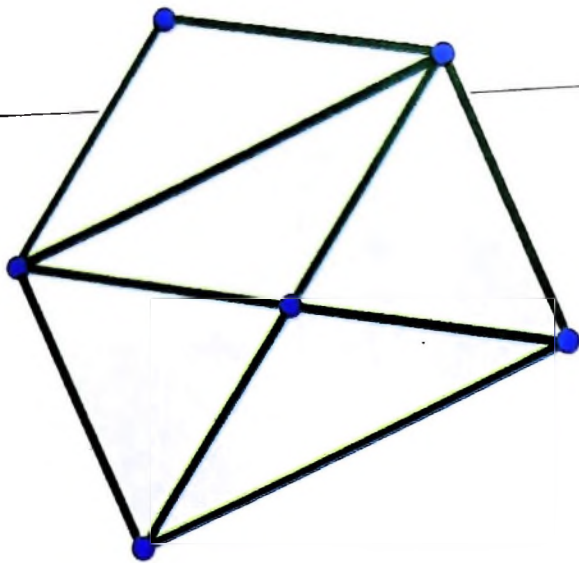
I ponti di Königsberg

Nel 1736, la città di Königsberg era costituita da quattro zone separate dai rami del fiume Pregel, e collegate tra loro da sette ponti disposti come nell'illustrazione riprodotta sopra. Il grande matematico Eulero, che abitava in questa città, si chiese se fosse possibile, partendo da un punto qualsiasi della città, ritornarvi dopo aver attraversato esattamente una volta ognuno dei sette ponti.

Risolvere il problema per tentativi, provando tutte le soluzioni possibili, sarebbe stato un procedimento lungo, noioso

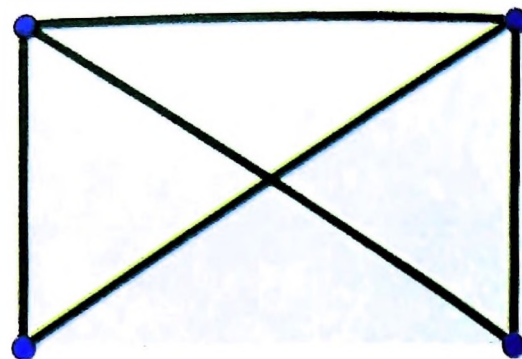
e soggetto a errori, che Eulero evitò grazie a una idea geniale, e soprattutto generalizzabile ad altri problemi simili, che gli consentì di dimostrare in modo semplice e inconfutabile l'impossibilità della cosa.

Anzitutto, occorre rappresentare la situazione in una forma sintetica, che evidenzia solo gli elementi essenziali del problema, trascurando tutto il resto. Poiché ciò che è importante evidenziare sono i collegamenti tra le quattro zone della città, possiamo dare una rappresentazione come nella figura piccola in alto, in cui ogni zona è rappresentata da un punto e i collegamenti (i ponti) sono rappresentati da segmenti: il



Un gioco analogo a quello dei ponti di Königsberg consiste nel disegnare una busta aperta o chiusa senza staccare la matita dal foglio e senza passare due

volte sullo stesso tratto. È facile verificare che questo è possibile nel primo caso, quando la busta è aperta, ma non nel secondo, quando è chiusa.



disegno che si ottiene è un esempio di grafo. Supponiamo ora di partire dal punto 1. Poiché non ci siamo ancora mossi, siamo liberi di percorrere uno qualsiasi dei tre ponti (segmenti) che partono da 1, per esempio a. Poiché vogliamo percorrere tutti i ponti, a un certo punto dovremo percorrere il ponte b, ritornando così in 1. A questo punto, possiamo solo percorrere il ponte c, poiché a e b sono già stati percorsi; ma questo significa che non avremo più la possibilità di tornare in 1, senza percorrere un ponte già attraversato.

Problemi analoghi a quello dei ponti di Königsberg consistono nello stabilire se figure del tipo di quelle rappresentate nella figura di questa pagina, in alto, possono essere disegnate senza staccare la matita dal foglio e senza ripassare due volte sullo stesso segmento. Prima di generalizzare il ragionamento di Eulero a questi nuovi problemi, è opportuno definire in modo più preciso cos'è un grafo.

Grafi orientati e non orientati

La definizione formale di grafo può essere facilmente ottenuta astruendo gli elementi essenziali dei diagrammi delle due ultime figure. Ognuno di questi diagrammi consta di un insieme di punti, eventualmente indicati con numeri o lettere, alcuni dei quali sono collegati tra loro da segmenti. La posizione dei punti e la lunghezza dei segmenti non hanno invece alcuna importanza, se non di tipo estetico, poiché l'unico obiettivo è quello di evidenziare i collegamenti. Possiamo allora dare la seguente definizione:

un grafo è una coppia $G=(V,A)$ dove:

- V è un insieme finito di elementi detti *vertici* o *nodi*;
- A è un insieme di coppie di vertici, del tipo (v,z) , non necessariamente distinte, che si dicono *archi* del grafo. Se le coppie sono ordinate, per cui $(v,z) \neq (z,v)$, il grafo si dice *orientato*; altrimenti si dice *non orientato*.

In sostanza, gli archi orientati possono essere pensati come

strade a senso unico, quelli non orientati come strade a doppio senso. I grafi vengono usualmente rappresentati in forma grafica (da cui il nome), rappresentando i nodi con punti sul piano (o nello spazio tridimensionale) e ogni arco del tipo (v,z) con una freccia o con un segmento dal nodo v al nodo z a seconda che il grafo sia orientato o non orientato. Un grafo orientato e uno non orientato sono rappresentati nelle due figure della pagina seguente. Si osservi qui che due nodi possono essere collegati da due (o anche più) archi diversi; se questo non si verifica, il grafo si dice *semplice*. Inoltre, possono esistere archi che collegano un nodo a se stesso (si veda nella prima di queste due figure l'arco che si chiude sul nodo 4); archi di questo tipo si dicono *cappi*.

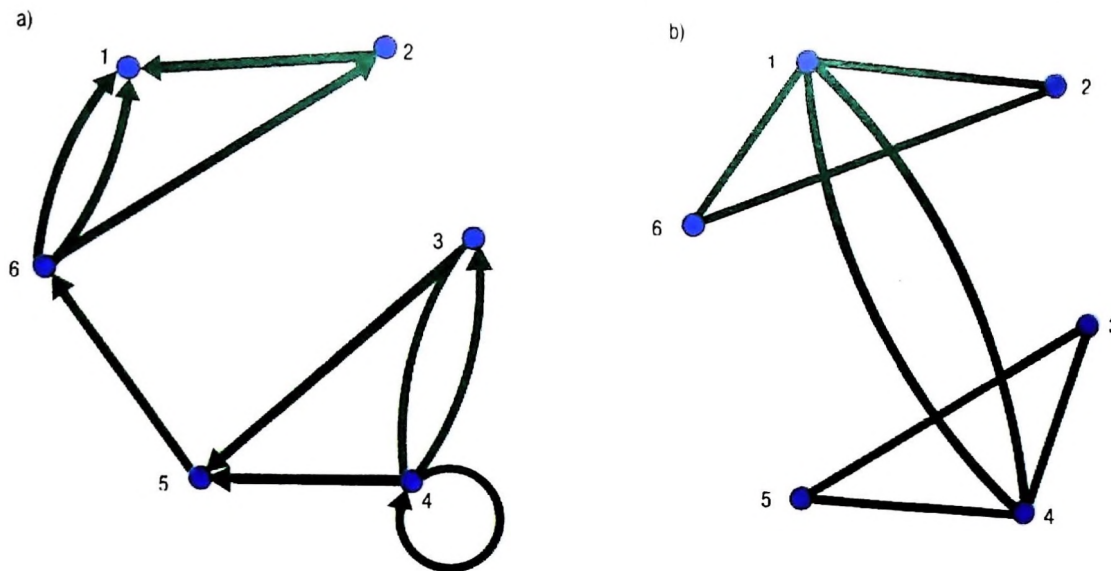
Infine, un'altra nozione importante è quella di *grado* di un nodo, che è definito come il numero di archi che fanno capo al nodo dato.

I grafi sono uno strumento molto utile per rappresentare in forma simbolica sintetica situazioni e problemi diversi, e quindi per semplificare la ricerca della soluzione, in varie discipline. Oltre alle applicazioni nelle diverse branche della matematica, dall'algebra alla geometria e alla topologia, possiamo citare applicazioni alla chimica, alla sociologia, ai trasporti, ai giochi, alla ricerca operativa e così via.

Per quanto riguarda le applicazioni all'informatica, in precedenza abbiamo trovato diversi esempi di grafi: grafi sono infatti i diagrammi di transizione degli automi, gli schemi di programma e i diagrammi a blocchi, gli schemi dei circuiti logici, le reti di Petri. Vediamo ora alcuni dei più interessanti problemi matematici sui grafi, mostrando come corrispondano a problemi reali di notevole importanza.

Problemi su cammini

La prima classe di problemi che trattiamo riguarda i *cammini* sui grafi.



A sinistra, un grafo orientato; a destra uno non orientato.

Dati due nodi v_i e v_F in un grafo G , un *cammino* da v_i a v_F è una successione di nodi v_1, v_2, \dots, v_n tali che v_1 è collegato a v_2 , v_2 è collegato a v_3 , ..., v_{n-1} è collegato a v_n , v_n è collegato a v_F e, per ogni i tra 1 e $n-1$, v_i è collegato a v_{i+1} . In particolare, se il nodo iniziale v_1 coincide con il nodo finale v_n , il cammino si dice *circuito*.

I problemi sui cammini consistono nello stabilire se in un dato grafo G esistono cammini con particolari proprietà. Per esempio, il problema dei ponti di Königsberg può essere ridotto a stabilire se nel grafo della prima pagina vi è almeno un circuito contenente una e una sola volta ogni arco. Cammini e circuiti con questa proprietà vengono detti *euleriani*, in omaggio a Eulero.

Proprio la soluzione del problema dei ponti ci offre la chiave per dimostrare un teorema generale sui circuiti euleriani. Infatti, l'impossibilità di percorrere esattamente una volta tutti i ponti di Königsberg era legata al fatto che nel grafo corrispondente alcuni nodi avevano grado dispari, mentre occorre un numero pari di archi per poter entrare e uscire da un nodo senza utilizzare due volte lo stesso arco. Per risolvere il problema per tutti i grafi possibili, occorre però fare una ulteriore osservazione abbastanza sottile. Se ogni nodo ha grado pari, per ogni arco che porta in un certo nodo v esiste sicuramente un arco diverso che consente di uscirne. Il problema è che non siamo sicuri, partendo da un qualsiasi altro nodo, di riuscire ad arrivare al nodo v . Si prenda il grafo della figura a sinistra della pagina seguente, in alto; esso ha la particolarità che i suoi nodi sono suddivisi in due insiemi tali che non esiste nessun arco che collega nodi del primo insieme a nodi del secondo, il che porta, nella rappresentazione grafica, a vederlo come costituito da due figure distinte, che si dicono *componenti connesse* del grafo. Benché tutti i nodi abbiano grado pari, è evidente che in un grafo del genere non possono esistere circuiti euleriani, poiché da una componente non si può raggiungere alcun nodo dell'altra.

Un grafo in cui, dati due nodi qualsiasi v e z , esiste sempre un cammino da v a z , si dice *connesso*.

La proprietà di connessione è una proprietà molto importante dei grafi, anche in relazione alle loro applicazioni pratiche. Si pensi per esempio a un grafo che rappresenta un sistema di trasporti o di comunicazioni, oppure un circuito elettrico: è evidente l'importanza che grafi di questo tipo siano connessi, in modo che da ogni punto sia possibile raggiungere tutti gli altri. Per questo, sono state studiate tecniche per dimostrare la connessione di un grafo.

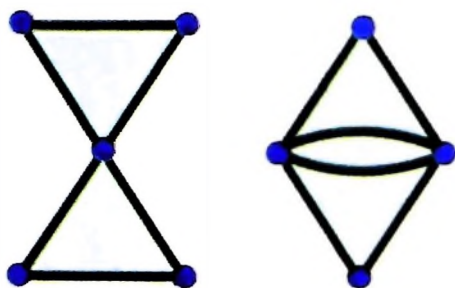
Possiamo a questo punto concludere la nostra discussione affermando che un grafo non orientato possiede un circuito euleriano se e solo se è connesso e non contiene vertici di grado dispari.

Analogamente è dimostrabile che in un grafo esiste un cammino euleriano che non sia un circuito (e quindi il grafo può essere disegnato senza staccare la penna dal foglio e senza passare due volte sullo stesso tratto) se e solo se è connesso e contiene esattamente due nodi di grado dispari, uno dei quali è il nodo di partenza e l'altro il nodo di arrivo (come accade nella prima figura della pagina precedente).

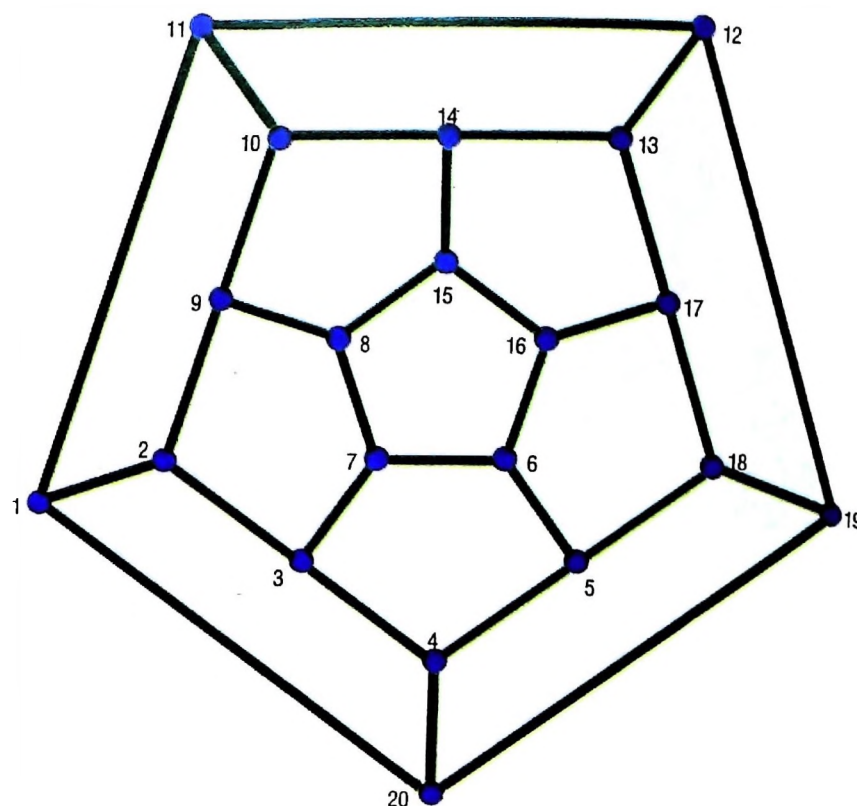
Un problema molto simile a quello di Eulero è stato proposto da un altro matematico famoso, Hamilton, che inventò un gioco in cui i giocatori dovevano "viaggiare" intorno a un "mondo" rappresentato da un dodecaedro regolare i cui venti vertici erano contrassegnati con nomi di città, in modo da tornare al punto di partenza dopo aver toccato esattamente una volta ogni città, seguendo gli spigoli del dodecaedro.

Nell'ultima figura a destra della pagina seguente è rappresentato un grafo che modella il problema; si tratta di trovare un circuito che passi esattamente una volta per ogni nodo (circuito *hamiltoniano*).

Nonostante la forte analogia con il problema dei cammini e dei circuiti euleriani, che come abbiamo visto si risolve in modo immediato, semplicemente verificando il grado dei nodi, per il problema dei circuiti e dei cammini hamiltoniani non esistono algoritmi di soluzione altrettanto semplici. Anzi, il problema dei circuiti hamiltoniani è uno dei più noti



In alto, è mostrato un grafo non connesso. Non esiste alcun cammino che consente di passare da un nodo appartenente alla prima componente a uno appartenente alla seconda. A destra, un grafo che rappresenta il gioco del "Giro del mondo" di Hamilton. Attraversando i nodi nell'ordine indicato si ottiene un cammino hamiltoniano.



problemi NP-completi, e quindi (si vedano i capitoli sulla complessità computazionale da considerarsi) molto difficile.

Grafi pesati e problemi associati

In molte situazioni concrete modellabili con grafi la scelta di un cammino piuttosto che un altro per passare da un nodo v a un nodo z può presentare vantaggi o svantaggi che spesso possono essere valutati in termini numerici.

Per esempio, nel caso di grafi che rappresentano sistemi viari due punti possono essere collegati da cammini diversi cui corrispondono nella realtà percorsi di lunghezza diversa (su cui quindi si avranno diversi consumi di benzina), mentre nella forma in cui abbiamo definito i grafi la lunghezza degli archi è del tutto irrilevante.

Per consentire di tener conto anche di questi fattori, si definiscono allora i *grafi pesati*, in cui a ogni arco è associato un valore numerico o simbolico che ne rappresenta il *costo*. Nell'esempio stradale fatto in precedenza, il costo di un arco sarà per esempio la lunghezza del tratto di strada corrispondente, oppure il tempo impiegato a percorrerla da un mezzo pubblico.

Se ora consideriamo un cammino, potremo dire che il suo costo è la somma dei costi degli archi da cui è costituito.

Per esempio, nel grafo della rete ferroviaria italiana vi sono diversi cammini che collegano Milano a Pisa (via Genova, via Firenze, via Fidenza), e la lunghezza complessiva di ognuno si calcola sommando le lunghezze dei singoli tratti.

Il problema più importante che si pone relativamente ai cammini su grafi pesati, soprattutto per le applicazioni ai trasporti, ma anche per il disegno di circuiti, è quello del cammino minimo, in cui si chiede, dati due nodi v e z , di trovare il cammino di costo minimo che li congiunge.

Questo problema può essere risolto in modo relativamente semplice sul piano della complessità computazionale utilizzando un algoritmo proposto da Dijkstra, che richiede un tempo di calcolo proporzionale al quadrato del numero di nodi del grafo, e quindi polinomiale con grado basso (si ricordi che i problemi che ammettono algoritmi di soluzione in tempo polinomiale sono considerati computazionalmente facili; si vedano i capitoli sulla complessità computazionale).

Non ha ovviamente senso in un grafo cercare cammini o circuiti euleriani di costo minimo poiché, dovendo includere tutti gli archi, il loro costo è sempre lo stesso. Invece, il problema di trovare un circuito hamiltoniano di costo minimo è un problema di estrema importanza che abbiamo già incontrato in precedenza nella forma, meno generale, del problema del commesso viaggiatore: date n città con le relative distanze chilometriche tra due qualsiasi di esse, trovare il percorso più breve che le attraversa tutte una ed una sola volta e riporti al punto di partenza.

Anche questo problema è NP-completo, e quindi computazionalmente difficile, per cui ci si accontenta in genere di trovare soluzioni non ottimali (per esempio, con algoritmi che garantiscono di trovare un cammino lungo non più del doppio del cammino più breve, ma che in genere trovano un cammino quasi ottimo).

IL SISTEMA OPERATIVO: UNA PRESENZA DISCRETA

Un insieme di programmi che gestiscono il funzionamento del calcolatore.

OS: tra voi e la macchina

Il sistema operativo è un insieme di programmi che permette, anche agli utenti che non si occupano degli aspetti tecnici, di scrivere, modificare, eseguire i propri programmi; è quindi il manager delle risorse della macchina, che deve gestire nel migliore dei modi rendendo non visibile all'utente questa gestione. Il sistema operativo in generale provvede a:

- definire un "colloquio" con l'utente, cioè definire in che modo è disposto ad accettare i comandi e con quali messaggi risponderà;
- suddividere le risorse del calcolatore tra i vari utenti (è chiaro che nei personal, normalmente, questo problema non si pone, essendo questi monoutente);

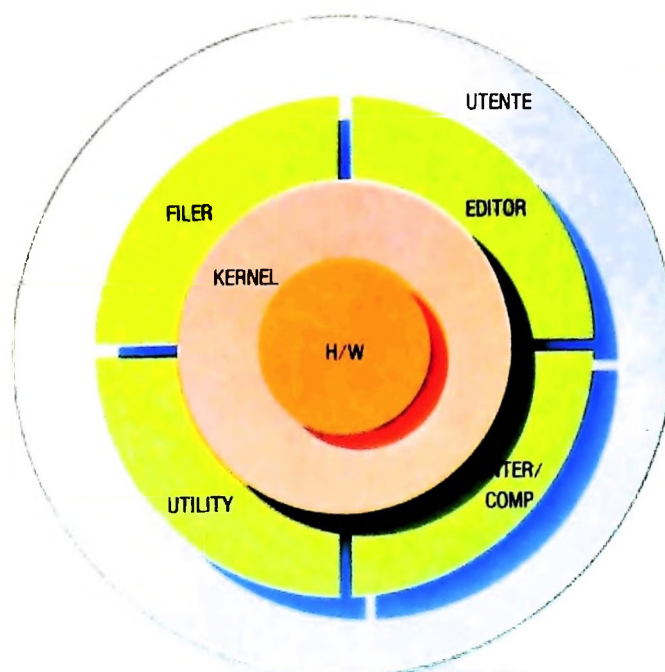
- gestire le operazioni di INPUT/OUTPUT.

Dal punto di vista dell'utente di personal e home computer la cosa più rilevante del sistema operativo con il quale avrà a che fare è il sistema di colloquio, il quale dovrà essere il più possibile semplice e conversazionale.

Come è fatto un sistema operativo

La figura mostra una struttura detta "a cipolla" dove ogni "strato" (di software) poggia su uno strato più interno. Normalmente muovendosi verso l'esterno si va verso un livello di astrazione maggiore, perdendo la visione dei problemi fisici.

L'OS (Operating System) è un insieme di programmi, e risiede permanentemente nel computer. Ha funzioni di supervisionare l'esecuzione dei programmi applicativi e di controllare il funzionamento dei dispositivi di INPUT/OUTPUT. Nella figura abbiamo un esempio di struttura detta "a cipolla", dove ogni strato di software poggia su uno strato più interno.



Nella figura notiamo che sull'hardware poggia uno strato di "procedure", tipicamente residenti su memoria non volatile, che formano il nucleo del sistema, detto "kernel".

Il kernel può essere visto come il supervisore del funzionamento fisico della macchina.

Sul kernel poggiano una serie di programmi che permettono la scrittura, la modifica, la memorizzazione, la correzione e l'esecuzione dei programmi utente.

Più in dettaglio vediamo che tipicamente, quindi, un sistema operativo si compone di:

- un EDITOR per la scrittura e la modifica di file contenenti

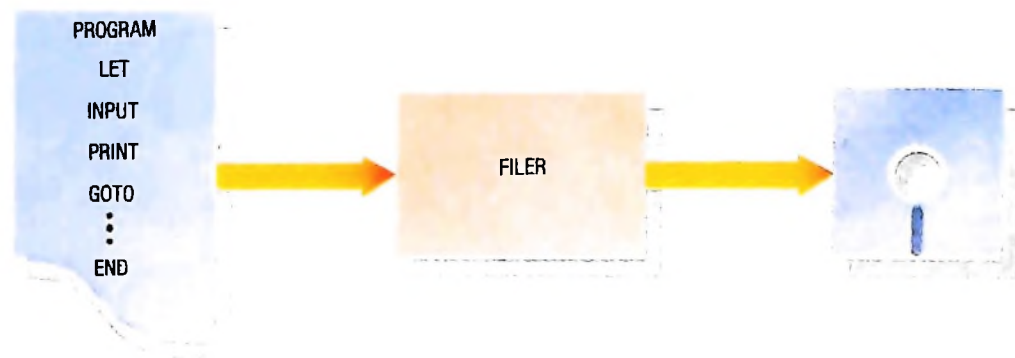
programmi utente e/o testi;

- un FILER, cioè tutta una serie di procedure di utilità per l'archiviazione dei dati, copia da e su memoria di massa (dischetti flessibili, cassette, disco rigido);

- INTERPRETI E COMPILATORI, cioè traduttori di programmi da linguaggi ad alto livello a linguaggio macchina per permetterne l'esecuzione;

- una serie di PROCEDURE DI UTILITÀ.

Il vantaggio della struttura "a cipolla" è principalmente quello di avere un sistema "aperto" a eventuali aggiunte, molte delle quali già sul mercato, altre, al limite, costruite



dall'utente stesso in base alle proprie esigenze.

Vediamo ora, singolarmente, i componenti di un sistema operativo.

Il compito svolto dal filer è illustrato nella figura sopra.

L'utente con un semplice comando (per esempio SAVE) comunica al filer che vuole memorizzare un file, dopo di che è il filer che provvede a gestire fisicamente l'allocazione su floppy o cassetta.

Analogamente per un comando del tipo LOAD, è sempre il filer che legge il file dal supporto e lo carica in opportune

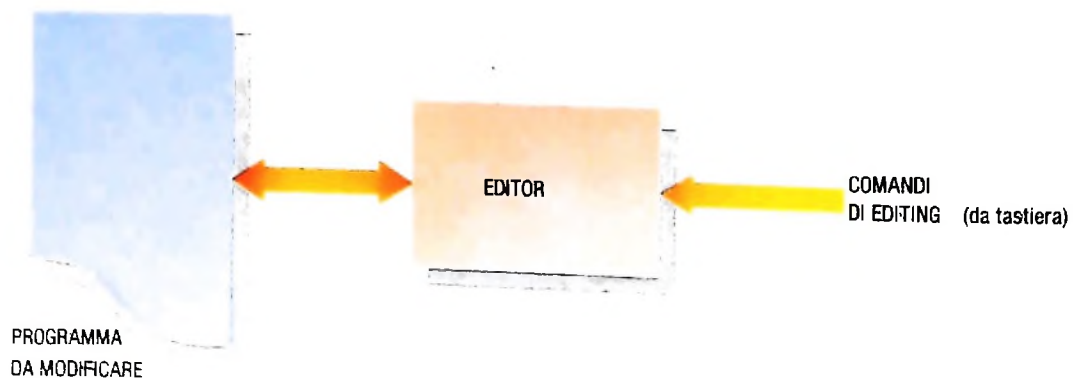
locazioni di memoria.

Altri comandi tipici di un filer sono:

- fornire l'elenco dei file contenuti su memoria di massa
- duplicare un file
- cancellare un file
- cambiare nome a un file.

L'editor può essere considerato una "segretaria" che accoglie e soddisfa le richieste di manipolazione di file di testo.

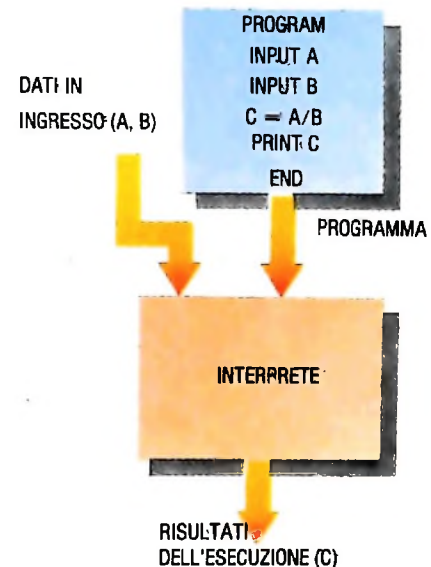
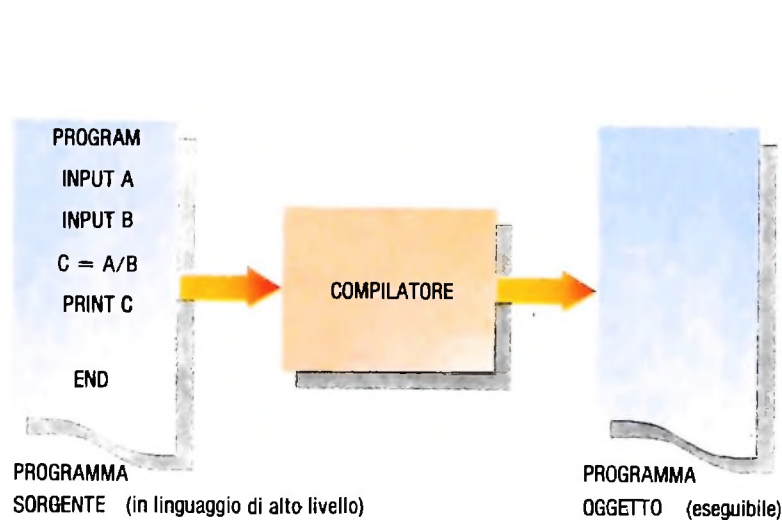
Provvederà, quindi, a eseguire comandi di modifica, cancellazione, inserimento, ricerca, sostituzione di parti in file con-



tenenti programmi utente e/o testi qualunque. La sua funzione è descritta nella figura in basso.

Il compilatore e/o l'interprete viene normalmente fornito con il sistema operativo vero e proprio, questo perché senza di esso la macchina non sarebbe in grado di eseguire i programmi scritti dall'utente.

La loro funzione è visualizzata nella figura di questa pagina. Il compilatore legge il file contenente il programma e lo "traduce" in linguaggio macchina, il solo linguaggio, come abbiamo già detto in articoli precedenti, che il computer è in grado di comprendere e di far girare.



I sistemi operativi per i microelaboratori

Negli ultimi anni sono stati sviluppati numerosi sistemi operativi per personal computer, o comunque, per sistemi basati su microelaboratori.

Tuttavia ancora non si è arrivati a uno standard, anche se alcuni sistemi si sono imposti più di altri e sono disponibili su un gran numero di macchine diverse. Tra i più noti:

- CP/M (e le sue evoluzioni CP/M 86 e concurrent CP/M)
- MS-DOS

- UCSD p-system
- UNIX

Questi quattro sistemi saranno trattati in dettaglio più avanti; vediamone, ora, una breve presentazione.

Il CP/M è stato realizzato dalla Digital Research nella metà degli anni Settanta, per micro elaboratori 8080 e Z80.

È, probabilmente, il sistema operativo di cui esiste il maggior numero di installazioni.

È monoutente e mette a disposizione un editor, un assembler e un debugger; è possibile arricchirlo con compilatori di linguaggi ad alto livello, altri editor, linker e molti altri programmi di utilità.

MS-DOS (MicroSoft-Disk Operating System) è un sistema operativo residente su disco, elaborato dalla nota Microsoft per il personal computer IBM, e ora disponibile pressoché su tutti i personal computer basati sui microelaboratori INTEL 8088 e 8086.

È, oggi, considerato uno "standard" per i personal basati su microprocessori a 16 bit.

È costituito da quattro programmi fondamentali che ne costituiscono il cuore:

- il primo è quello che viene caricato all'accensione e che predispose la macchina all'uso;

— il secondo gestisce l' INPUT/OUTPUT;

— il terzo è un programma di gestione file più una serie di funzioni di servizio utilizzabili dai programmi che vengono eseguiti in ambiente DOS;

— il quarto dialoga con l'utente: ne accetta i comandi ed esegue i programmi corrispondenti.

L'UCSD p-system sviluppato da Kennet Bowles e dai suoi collaboratori presso la University of California, San Diego (UCSD), è ora commercializzato dalla società californiana Softech Micro systems.

Storicamente, tale sistema è nato per supportare il linguaggio UCSD Pascal; oggi sono disponibili, nell'ambito del p-system, anche altri linguaggi, per esempio, il basic, il fortran, l'assembler.

I linguaggi disponibili in ambiente p-system sono tradotti dai compilatori relativi in un linguaggio intermedio indipendente dalla macchina, detto p-code. Il p-code viene quindi interpretato. Ciò rende facilmente portabili da una macchina all'altra i programmi scritti nell'ambito del p-system: l'unico componente del sistema che dipende dalla macchina particolare è, infatti, l'interprete del p-code. In effetti, oggi l'UCSD p-system è disponibile su un numero elevato di macchine, anche basate su microprocessori diversi.

UNIX nasce all'inizio degli anni Settanta nei BELL Labora-

tories, a opera di D.M. Ritchie e K. Tompson, come sistema operativo destinato ad aiutare i progettisti nello sviluppo di software. La sua fortuna è dovuta, tra l'altro, al fatto che ormai esso è entrato prepotentemente nel mondo universitario e della ricerca.

Si devono infatti, per esempio, all'università di Berkley numerosi successivi ampliamenti del sistema.

UNIX è un sistema:

- multiutente (cioè permette a più utenti di lavorare contemporaneamente);
- multiprocessing (cioè più programmi possono essere eseguiti contemporaneamente);
- time-sharing (cioè il sistema fa in modo che nessun utente monopolizzi la macchina, facendo eseguire "a turno" i programmi dei diversi utenti).

Le prime versioni sono apparse su mini-computer Digital e successive versioni sono state prodotte sia per grossi sistemi che per microelaboratori.

Negli ultimi anni con la massiccia diffusione dei personal computer è stata realizzata una versione ridotta che ne permette l'uso anche sui personal, per singolo utente.

Il successo di UNIX è dovuto, soprattutto, alla grande varietà di strumenti software che vengono resi disponibili assieme al sistema operativo vero e proprio: strumenti per la elaborazione testi, per la gestione dei programmi, per lo sviluppo di software, per la comunicazione fra più utenti, e così via.

È da sottolineare che il sistema operativo non è rigidamente collegato alla macchina e viceversa, ma una stessa macchina può ospitare più sistemi diversi.

Per esempio il personal computer IBM offre la possibilità di lavorare con l'UCSD, con MS-DOS e con CP/M 86: sarà l'utente, a seconda delle esigenze, a scegliere di volta in volta il sistema a lui più congeniale.

Stili di colloquio

Abbiamo già detto che per gli utenti di personal computer una delle caratteristiche più importanti è la cosiddetta "interfaccia utente", cioè lo stile di colloquio con il quale si comunica con la macchina.

Gli stili di colloquio normalmente usati dai sistemi operativi sono due:

- colloquio a comandi
- colloquio a menù.

Nel primo caso (MS-DOS, CP/M, UNIX) per fare in modo che il sistema esegua un determinato comando bisogna digitare il nome del comando con gli opportuni parametri.

Per esempio per fare in modo che il sistema memorizzi un file "prova" sul floppy "B" bisognerà digitare:

SAVE B: prova

È, quindi, l'utente che dovrà sempre ricordarsi i nomi dei comandi desiderati.

Nel secondo caso (UCSD) è, invece, il sistema che visualizza un "menù" di comandi che in quel momento può soddisfare.

Per esempio una schermata tipica dell'UCSD è:

```
E<DIT C<OMPILING F<ILER L<IBRARY
X<CHANGE
```

premendo la "E" (per EDIT) il sistema manda in esecuzione l'editor, il quale, a sua volta, visualizzerà il menù dei propri comandi, che sono:

```
I<NSERT D<ELETE F<IND X<CHANGE C<OPY
Z<AP Q<UIT
```

a questo punto, cioè, sul file è possibile inserire nuove parti (Insert), cancellarne delle altre (Delete), ricercare occorrenze (Find), sostituire delle lettere con delle altre (X change). Tutto questo con una guida interattiva e semplice, con dei comandi mnemonici che rendono il lavoro meno pesante.

Glossario

Microprocessore - una unità di elaborazione completa, integrata su un unico chip di silicio, anziché essere ottenuta dal collegamento di più componenti discreti. Il microprocessore ha reso possibile la nascita dei microcomputer, e dei personal computer in particolare: ne costituisce il "cuore", l'unità di elaborazione.

Mouse - un dispositivo di ingresso per una unità di elaborazione, che ha esternamente la forma di una piccola scatola con uno o più pulsanti. Il mouse governa il cursore: spostando il mouse sul piano del tavolo, il cursore si sposta nella stessa direzione, in misura proporzionale. Il metodo è molto utile in abbinamento a programmi che fanno grande uso di menù: con il mouse si porta il cursore sull'opzione desiderata, e con uno dei pulsanti (che svolge una funzione analoga a quella del tasto RETURN) si comunica alla macchina che quella è la scelta effettuata. Il mouse fa parte dei dispositivi studiati per rendere più amichevole l'interfaccia fra l'utente e la macchina, ed è molto usato in calcolatori orientati alla grafica in tutte le loro operazioni, come è ad esempio il Macintosh della Apple.

Porta logica - qualsiasi circuito che realizzi una funzione logica elementare: NOT, AND, OR, NAND, NOR. Non è importante che il circuito sia di tipo elettronico: sono state realizzate, per esempio, an-

che porte logiche di tipo ottico (il cui funzionamento, cioè, è basato sullo sfruttamento di meccanismi luminosi). Nei circuiti elettronici, i valori logici "vero" (1) e "falso" (0) sono rappresentati da segnali, in ingresso e in uscita, caratterizzati da due valori distinti di tensione: il livello di tensione elevato rappresenta normalmente il valore 1, il livello basso rappresenta il valore 0.

PROLOG - acronimo di Programming in Logic, indica un linguaggio di programmazione di alto livello, utilizzato soprattutto per applicazioni di intelligenza artificiale. Formulato agli inizi degli anni sessanta da Alain Colmerauer dell'Università di Marsiglia, è un linguaggio dichiarativo, anziché imperativo come i linguaggi più diffusi (BASIC, Pascal, FORTRAN, COBOL e simili). Le fasi del linguaggio, cioè, non hanno la forma di istruzioni che il computer deve eseguire, ma di dichiarazioni di fatti che la macchina deve memorizzare, e su cui poi potrà fornire informazioni, se opportunamente interrogata secondo le modalità previste dal linguaggio stesso. Linguaggio tipicamente interpretato e per sua stessa natura particolarmente adatto per la costruzione e l'interrogazione di grandi basi di dati "intelligenti" e la costruzione di sistemi esperti, il PROLOG è stato scelto come linguaggio fondamentale nel progetto giapponese di elaboratori della quinta generazione.

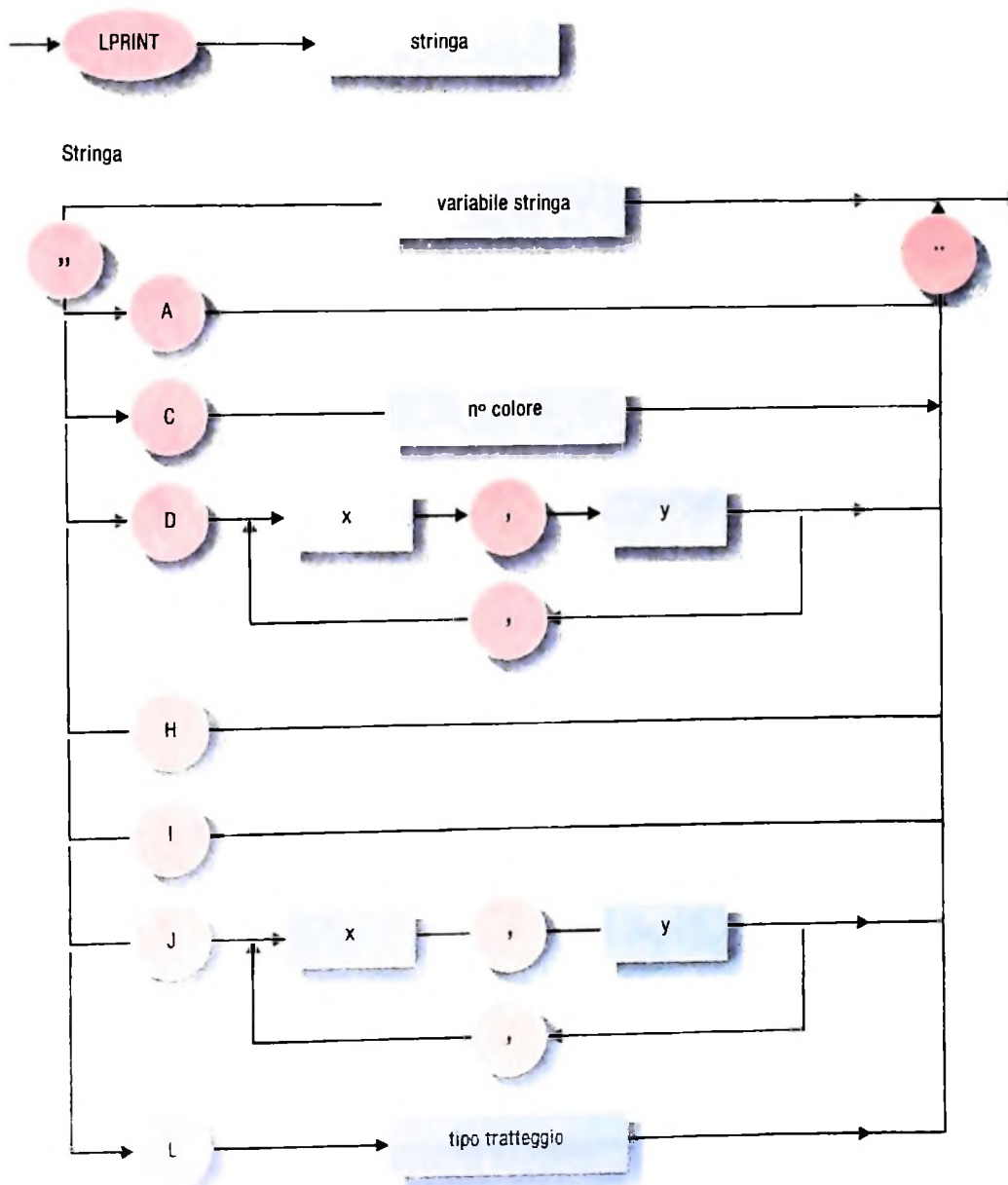
Lezione 51

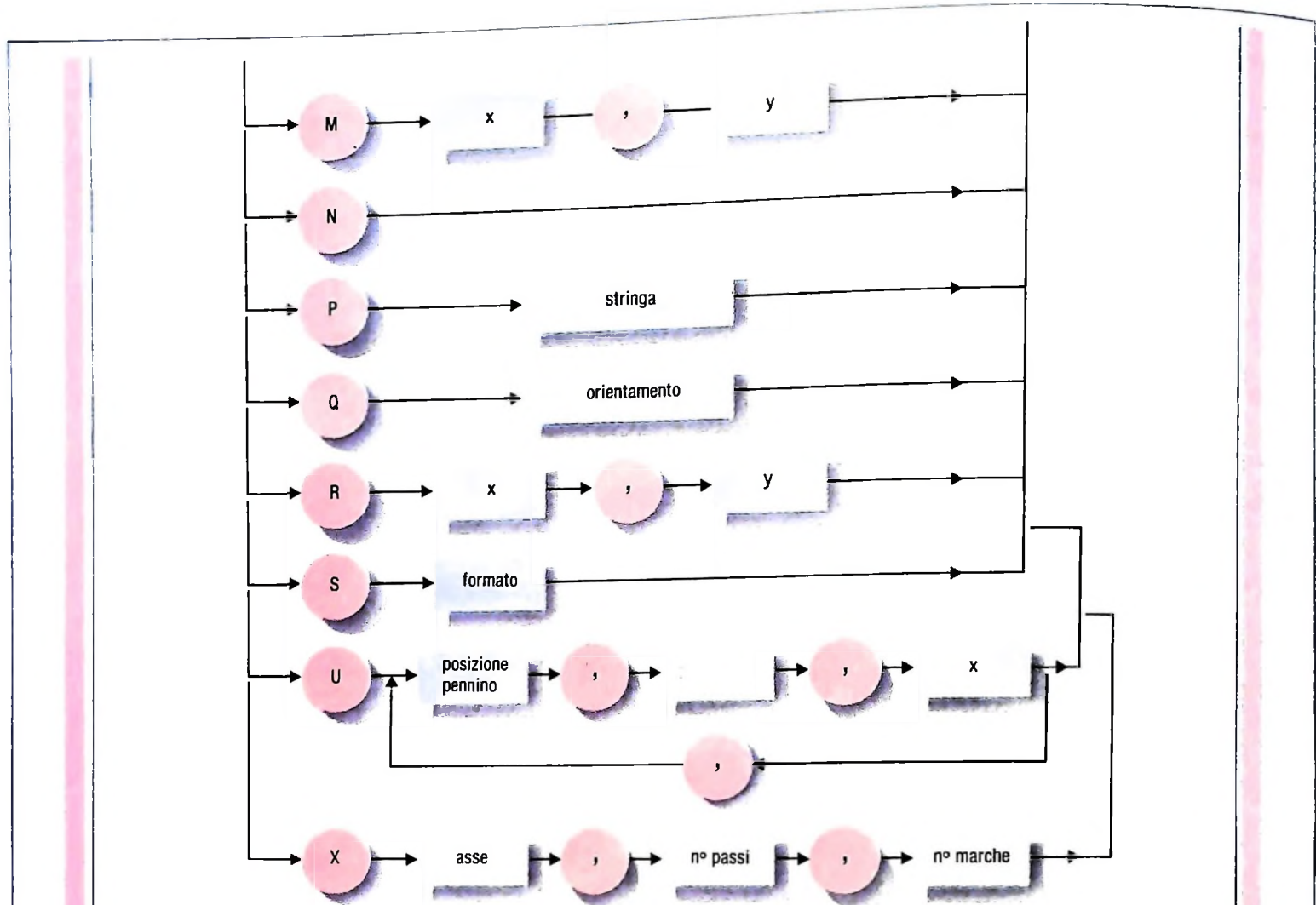
Tutti i comandi del microplotter

Il microplotter, come abbiamo visto, è una periferica molto potente e versatile, per cui è opportuno avere a disposizione la sintassi di tutti i comandi finora visti.

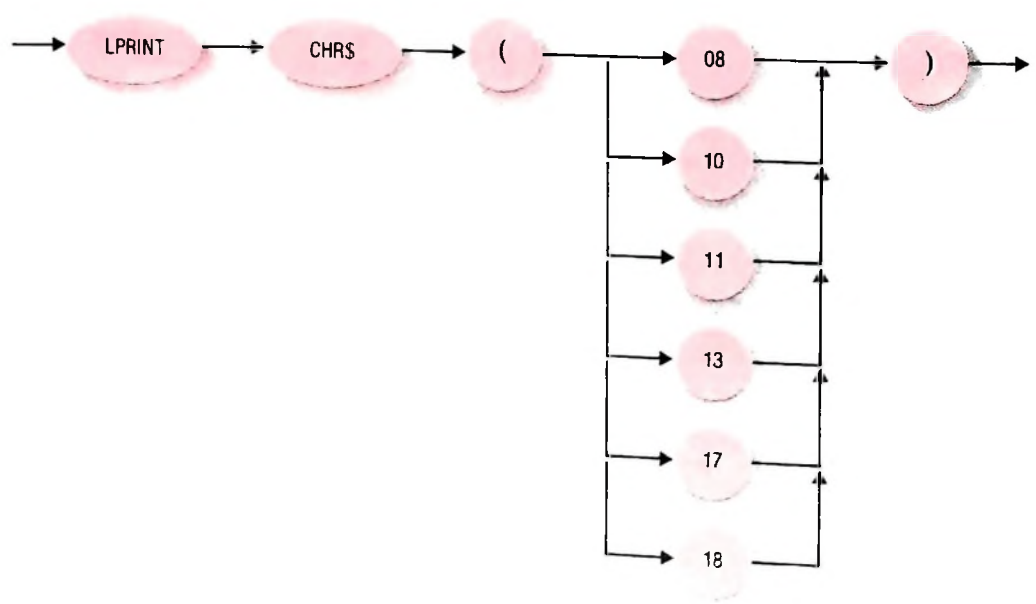
Tra i comandi riportati in questa lezione è presente anche il comando grafico I che non è stato usato negli esempi precedenti. Esso permette di definire come posizione di riferimento quella in cui si trova il pennino nel momento in cui il comando viene eseguito; da quel momento in poi tutti i comandi in assoluto faranno riferimento a quella specifica posizione.

Comandi di microplotter in "graphic mode"





Codici di controllo in "text" mode



ove i vari comandi e parametri hanno il seguente significato:

- A fa tornare in modo testo
- C permette di cambiare colore
- D traccia righe, facendo riferimento a coordinate assolute rispetto alla posizione di riposo del pennino
- H sposta il pennino nella posizione di riposo
- I definisce la posizione attuale del pennino come sua posizione "di riposo" a cui fare riferimento con i comandi H e assoluti
- J traccia linee facendo riferimento a coordinate misurate rispetto alla posizione attuale dello stesso, cioè spostandosi "in relativo"
- L definisce il fatto che le linee tracciate siano continue o tratteggiate
- M sposta il pennino in una posizione, con riferimenti assoluti, senza tracciare alcun segno;
- N traccia nella posizione corrente il carattere speciale definito mediante l'ultimo comando U eseguito
- P stampa la stringa che segue
- Q definisce l'orientamento della linea di stampa
- R effettua uno spostamento del pennino senza lasciare la traccia, in modo relativo, rispetto alla posizione attuale del pennino stesso
- S definisce le dimensioni dei caratteri
- U definisce un "carattere" personalizzato, da stampare con il comando N
- X traccia gli assi

x,y sono coordinate di punti; possono essere definite in relativo o in assoluto; i loro valori sono compresi tra -999 e 999; si ricordi che la lunghezza di una riga è di 480 posizioni

n. colore:	0
	1 blu
	2 verde
	3 rosso
tipo tratteggio:	0 linea continua
	1 tratteggio più piccolo
	15 tratteggio più grande
orientamento:	0 da sinistra a destra
	1 dal basso in alto
	2 da destra a sinistra
	3 dall'alto in basso
formato:	0 80 caratteri per riga
	63 1 carattere per riga

il valore di difetto è 1 pari a 40 crt per riga

posiz. penn:	0 pennino alzato; non scrive
	1 pennino abbassato; scrive
asse:	0 asse verticale delle y
	1 asse orizzontale delle x
n. passi:	numero di passi (righe o colonne) che deve trovarsi tra due marche consecutive sull'asse che si sta tracciando
n. marche:	numero di suddivisioni della dimensione indicata dal parametro precedente, che devono trovarsi sull'asse che si sta tracciando; definisce la lunghezza dell'asse
codici CHR\$:	08 backspace; retrocede di una posizione
	10 line feed; avanza di una linea
	11 line feed inverso; retrocede di una linea
	13 ritorno carrello
	17 passaggio a modo testo
	18 passaggio a modo grafico.

Ricordiamo inoltre i seguenti aspetti:

- le dimensioni dei caratteri con il comando S hanno le seguenti caratteristiche, in funzione del parametro "formato" passatogli:

larghezza del carattere: $4 * (\text{formato} + 1)$ punti di stampa
altezza del carattere: $6 * (\text{formato} + 1)$ punti di stampa
n° di caratteri per riga: $80 / (\text{formato} + 1)$ caratteri
interlinea: pari all'altezza del carattere

- i caratteri definiti dall'utente mediante il comando U hanno le seguenti caratteristiche:

dimensioni del quadrato in cui possono essere definiti: 7×7 punti
variabilità delle coordinate x, y del comando: da -7 a +7
numero massimo di spostamenti per comando: 15
coordinate x, y: definite in relativo rispetto alla posizione in cui si trova il pennino

- la definizione dei parametri per il tracciamento degli assi con il comando X deve sottostare alle seguenti regole:

il numero di passi definito tra una marca e la successiva deve essere compreso tra -999 e 999

il numero di marche di graduazione dell'asse deve essere un intero compreso tra 1 e 255

nel caso dell'asse orizzontale (parametro "asse" uguale a 1), il numero di passi e quello di marche deve sottostare alla condizione
 $n^{\circ} \text{passi} * n^{\circ} \text{marche} < 480$

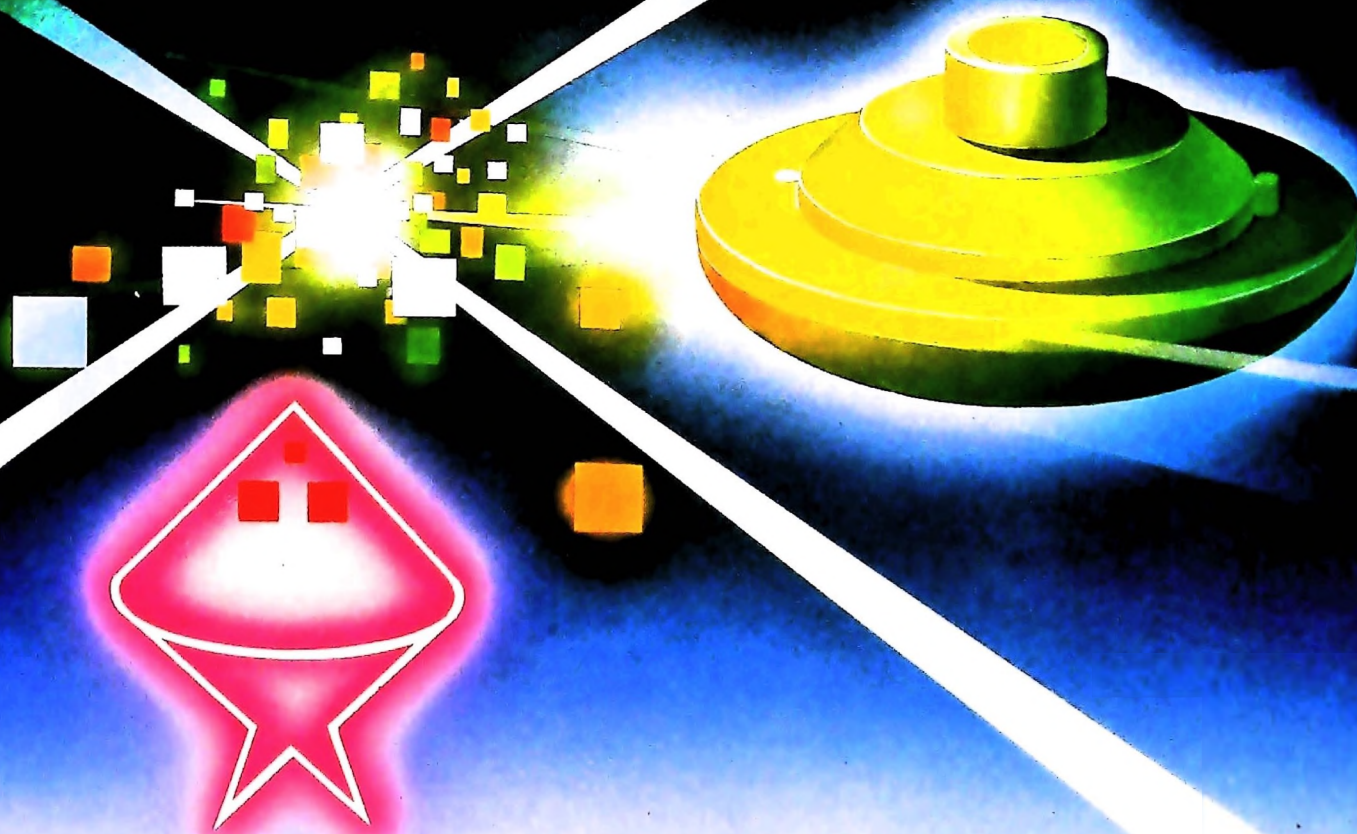
per poter essere contenuto in una riga del microplotter un valore negativo del numero di passi specifica che l'asse deve essere tracciato nella direzione del suo sviluppo negativo (da destra verso sinistra per l'asse orizzontale delle x, dall'alto verso il basso per l'asse delle y).

Cosa abbiamo imparato

In questa lezione non abbiamo imparato nuovi comandi o concetti, ma abbiamo visto un elenco completo ed esaustivo di tutti i comandi fornibili al microplotter, sia per operare in "graphic mode", sia per operare in "text mode".

LE GUERRE STELLARI

Shoot-'em-up: corpi alieni attaccano con ritmo frenetico per sopravvivere però basta avere un braccio veloce



Il primo videogame ispirato a una serie televisiva è stato "Star Trek", non quello prodotto dalla Vetrex, bensì quello del MIT (Massachusetts Institute of Technology). Il MIT è molto conosciuto per il fatto che all'interno "si gioca", anche se la ben meritata reputazione di quella scuola è legata alla produzione di scienziati e non di fantascienziati.

Fatto sta che nel 1962 un giovane laureato, Steve Russel, decise di applicarsi a uno strumento relativamente nuovo, chiamato computer, disegnando missili, razzi e soli iperspaziali su di uno schermo in bianco e nero. Nel ventre misterioso di un calcolatore da mezzo milione di dollari, nasceva il primo videogioco a soggetto. Passarono gli anni, nuove generazioni di studenti del MIT tramandarono questo gioco ai loro successori. Alla fine fu ribattezzato "Spacewar" (Guerra spaziale). Nel 1970, quando la serie di telefilm "Star Trek" era all'apice del successo, qualche ispirato programmatore decise di dare a "Spacewar" lo scenario visibile dalla navicella Enterprise. Fu cambiato il gioco in modo tale che il giocatore potesse vedere Klingons e Romulani attraverso gli occhi del capitano Kirk, come se guardasse direttamente fuori dalla grande vetrata della navicella spaziale.

Nel 1977 esce il film "Star Wars" (Guerre stellari) di Spielberg-Lucas: gli strumenti della più avanzata tecnologia elettronica vengono applicati per raccontare una favola galattica. È un successo strepitoso, è la formidabile spinta che i videogame aspettavano. Installando centomila "Space Invaders" in Giappone, la società Taito Ltd rivela al mondo intero che una nuova forma di gioco elettronico è nata. Il re dei flipper, Bally, compra immediatamente i diritti del gioco per gli Stati Uniti, relativamente alle sale pubbliche, e Atari lancia il gioco nella versione casalinga. Lo schema di gioco è relativamente semplice: il giocatore manovra un missile laser mentre dall'alto cadono corpi alieni a un ritmo via via più frenetico. Eppure con questa battaglia spaziale esplose in tutto il mondo la mania dei giochi elettronici.

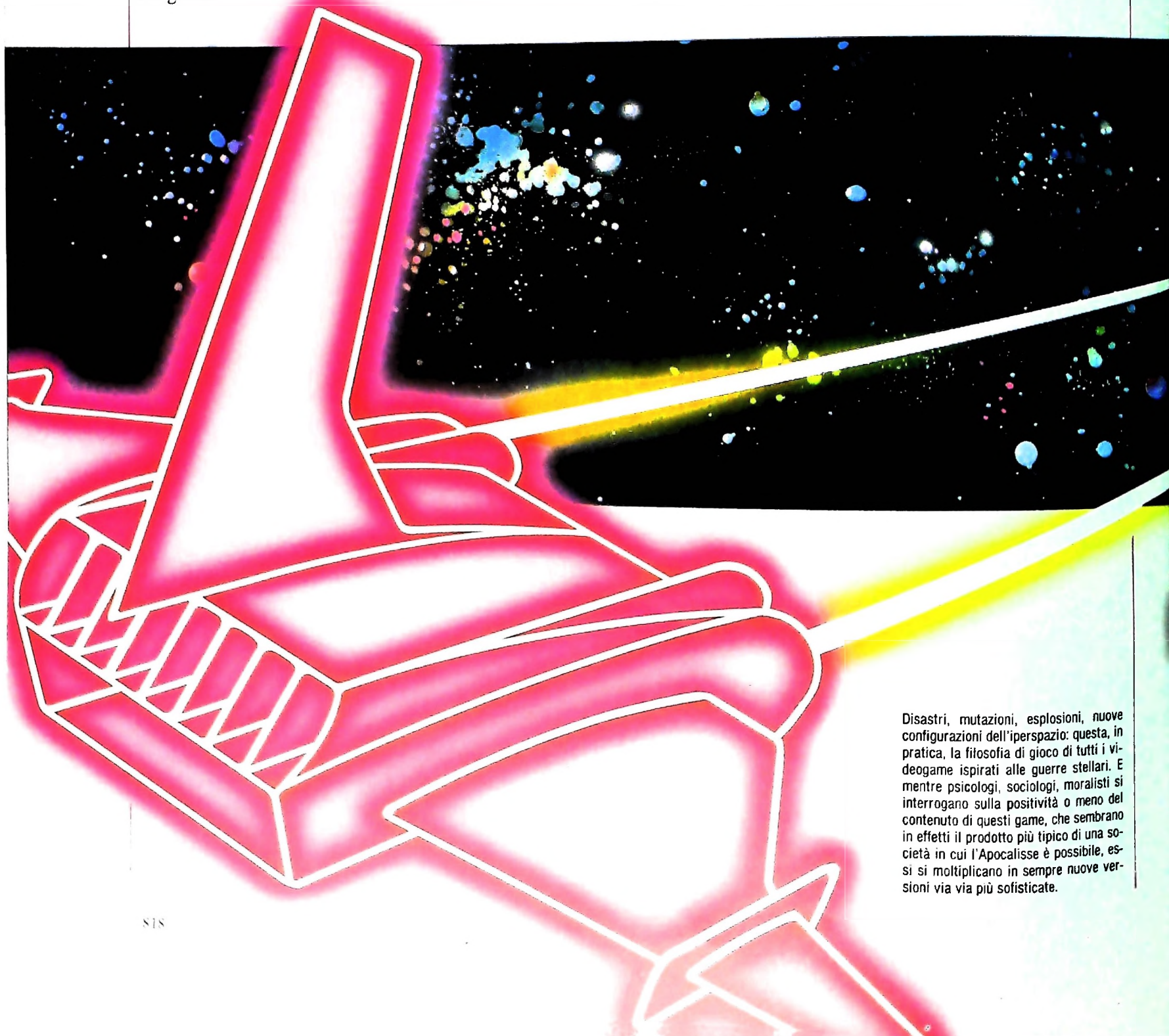
A "Space Invaders" seguono "Defender", "Asteroids", "Mégamania", "Astromash", "Space Cavern", "Phoenix", "Mine Storm", "Nexar", "Vanguard", "Space Spartans", "Space Battle" e tanti altri che segnano chiaramente il passaggio dall'epoca del flipper a quella del videogame. Lo scenario si complica, la grafica diventa più seducente, le storie promettono catastrofi sempre più grandi: questo genere di giochi

viene immediatamente battezzato "shoot-'em-up" a indicare l'essenzialità dell'azione: bisogna colpire la navicella e basta. Puntuale la stampa si interroga sul contenuto pedagogico di questi giochi: "La tecnologia informatica che ha creato i missili reali — si scrive — è la stessa che ha generato questi pseudo missili scintillanti psichedelicamente sugli schermi, e infatti il Pentagono si serve di queste macchine per simulare le sue guerre. Anche se le scene che compaiono sugli schermi si svolgono in lontane costellazioni, dove gli avversari sono indefinibili, la loro funzione è in realtà di far vivere a un altro livello la minaccia nucleare". Francamente, se rapporto c'è fra missili veri e missili schermici, è un rapporto ben più complesso di quello tratteggiato da buona parte della stampa; se mai, si può affermare che le storie che i videogame stellari raccontano sono storie essenzialmente pessimistiche, catastrofiche, dove la fine del combattimento (la distruzione totale) coincide sempre ed ineluttabilmente con la fine stessa del gioco.

Il gioco, tutti i giochi insegnano anche questo: è probabile che delle catastrofi accadano, sembra meno probabile che accadano catastrofi già minuziosamente previste, descritte, "giocate". La rappresentazione dei disastri non serve soltanto a stimolare il senso estetico, ma anche a limitare le probabilità che i disastri hanno di realizzarsi.

Il programma

Questo breve programma illustra come implementare le procedure di base per costruire un gioco del genere "Space Invaders". Queste procedure permettono infatti le seguenti funzionalità: innanzitutto fanno muovere uno o più alieni in maniera del tutto indipendente dalle azioni del giocatore, facendoli scendere verso la terra; in secondo luogo fanno muovere il cannone (tasti "A" e "D") e lo fanno sparare (tasto "S") controllando se l'alieno è stato colpito.



Disastri, mutazioni, esplosioni, nuove configurazioni dell'iperspazio: questa, in pratica, la filosofia di gioco di tutti i videogame ispirati alle guerre stellari. E mentre psicologi, sociologi, moralisti si interrogano sulla positività o meno del contenuto di questi game, che sembrano in effetti il prodotto più tipico di una società in cui l'Apocalisse è possibile, essi si moltiplicano in sempre nuove versioni via via più sofisticate.


```

1  REM
2  REM  Esempio base di SPACE INVADERS
3  REM
10 REM
11 REM  inizializza le variabili del programma
12 REM
20 ESC$ = CHR$(27) : te = 0
30 ax = 0 : ay = 0 : cx = 20 : cy = 7
40 px = cx : py = 6
50 ach$ = "A" : cch$ = "!" : pch$ = "."
60 sp% = 0 : REM non ha sparato
70 ch$ = cch$
80 x = cx : y = cy : GOSUB 8000
90 ch$ = ach$
100 x = ax : y = ay : GOSUB 8000
900 GOTO 9000 : REM inizia il programma
2000 REM
2001 REM esegue le azioni corrispondenti al tasto premuto

```



```

2002 REM
2010 IF ky$ = "A" THEN GOTO 2100 : REM a sinistra
2020 IF ky$ = "S" THEN GOTO 2200 : REM spara
2030 IF ky$ = "D" THEN GOTO 2300 : REM a destra
2090 GOTO 2999
2100 REM premuto tasto "A"
2105 ch$ = " "
2107 x = cx : y = cy : GOSUB 8000
2110 cx = cx - 1 : IF cx < 0 THEN cx = 0
2120 ch$ = cch$
2130 x = cx : y = cy : GOSUB 8000
2140 IF sp% + 0 THEN px = cx
2199 GOTO 2999
2200 REM premuto tasto "S"
2210 IF px=ax AND py=ay THEN GOTO 9800

```

```

2220 ch$ = pch$
2225 x=px : y=py : GOSUB 8000
2230 sp% = 1 : REM ha sparato
2299 GOTO 2999
2300 REM premuto tasto "D"
2305 ch$ = " "
2307 x = cx : y = cy : GOSUB 8000
2310 cx = cx + 1 : IF cx > 39 THEN cx = 39
2320 ch$ = cch$
2330 x = cx : y = cy : GOSUB 8000
2340 IF sp% = 0 THEN px = cx
2999 RETURN
3000 REM
3001 REM non e' stato premuto nessun tasto
3002 REM
3003 te = te + 1
3004 IF te MOD 10 <> 0 THEN GOTO 3500
3005 ch$ = " "
3006 x = ax : y = ay : GOSUB 8000
3010 ax = ax + 1
3020 IF ax < 39 THEN GOTO 3400
3030 ax = 0 : ay = ay + 1
3040 IF ax = cx AND ay = cy THEN GOTO 9800
3050 IF ay < 0 THEN GOTO 9999
3400 ch$ = ach$
3410 x = ax : y = ay : GOSUB 8000
3500 RETURN
8000 REM scrivo "ch$" in (x,y)
8010 IF x < 0 OR x > 39 THEN GOTO 8100
8020 IF y < 0 OR y > 7 THEN GOTO 8100
8030 PRINT ESC$+"Y"+CHR$(y+32)+CHR$(x+32);
8040 PRINT ch$;
8100 RETURN
9000 REM
9001 REM programma principale
9002 REM
9030 kys = INKEYS
9040 IF kys = "" THEN GOSUB 3000 ELSE GOSUB 2000
9050 IF sp% = 0 THEN GOTO 9070
9055 ch$ = " "
9057 x=px : y=py : GOSUB 8000
9060 py=py-1
9062 IF px=ax AND py=ay THEN GOTO 9800
9064 ch$ = pch$ : y = py : GOSUB 8000
9065 IF py >= 0 THEN GOTO 9070
9066 py = 6 : px = cx : sp% = 0
9070 GOTO 9030
9800 REM boom
9805 BEEP
9810 ch$ = "*"
9820 x = ax : y = ay-1
9824 IF y < 0 THEN y = 0 : GOSUB 8000
9830 x = x+1 : GOSUB 8000
9840 y = y-1 : GOSUB 8000
9850 x = x-1 : GOSUB 8000
9999 END

```

IMPIEGO INTERATTIVO DEL PLOTTER

Un programma che permette di creare in modo interattivo oggetti a due dimensioni.

In una precedente lezione parlando del plotter abbiamo visto come si possono costruire stringhe di comandi grafici per il microplotter mediante le seguenti istruzioni BASIC:

STR\$ = converte in stringa valori numerici in modo tale che per esempio il numero 123 diventa la stringa di caratteri "123", permettendo così la somma "123" + "34" = "12334" apparentemente molto strana, ma sicuramente utile nel caso in cui si debbano costruire comandi come quelli necessari al funzionamento del plotter;

LPRINT = scrive una stringa o una variabile sul microplotter.

Utilizzando questi comandi e i caratteri di controllo del microplotter si possono costruire istruzioni anche molto complesse. In questa lezione si vuole dare un completamento al pacchetto grafico a due dimensioni fornendo la possibilità di correggere interattivamente eventuali errori commessi durante la realizzazione del disegno, rendendo inoltre interattiva la scelta dei colori di plottaggio.

Un esempio di realismo col computer. Mediante l'utilizzo di tavolozze di 'colori elettronici', e applicando le leggi dell'ottica per simulare l'illuminazione, si possono ottenere effetti di incredibile realismo dell'immagine.



ACM-ARCHIVIO EIDOS

Il programma

La presente realizzazione è una ulteriore estensione del programma PEN e fornisce un semplice strumento per la creazione in modo interattivo di oggetti a due dimensioni.

Si ricorda che l'interattività del programma si basa, in pratica, sulla possibilità offerta dal BASIC di leggere i caratteri in ingresso da tastiera (i tasti premuti) senza ricorrere alla pressione del tasto "ENTER", con l'ovvio vantaggio di una immediata risposta alla pressione del tasto relativo al comando. Tale metodo di lettura è stato utilizzato in modo particolare per riconoscere la pressione dei tasti per il movimento della matita.

Questa versione si basa principalmente sulla possibilità di correggere, sempre in maniera interattiva, parti di disegno.

Il metodo di correzione adottato è del tipo "all'indietro" poiché è possibile cancellare i segmenti a partire dall'ultimo introdotto. Questa opzione è stata strutturata sotto forma di procedura (linee 700-790) e ha richiesto un completo rinnovamento della struttura dati sulla quale si basa tutto il programma. Le precedenti versioni del programma utilizzavano una struttura dati molto semplice, poiché la memorizzazione delle coordinate su data file avveniva in maniera diretta al momento della creazione del vertice relativo non essendo necessaria alcuna successiva manipolazione dei valori numerici ottenuti.

Unica struttura utilizzata era dunque quella relativa alla creazione di una stringa di comandi per il plotter; il dimensionamento compare anche in questa versione alla linea 4.

Nella versione PEN4 si è invece utilizzata come struttura dati base l'array di cui si nota il duplice dimensionamento (CX per coordinate X e CY per coordinate Y) alla linea 5. Tale array viene utilizzato per creare una struttura "ultimo entrato, primo a uscire" in maniera tale che le coordinate non sono memorizzate direttamente nel data file ma vengono inserite negli array CX e CY dove restano disponibili fino alla fine del disegno.

Se durante l'esecuzione di una figura si rende necessaria la correzione di un certo numero di segmenti si preme il tasto "r" minuscolo e si entra in un sottomenù di "rimozione segmenti" dove a ogni successiva pressione del medesimo tasto si ottiene la rimozione fisica (sullo schermo) e logica (nell'array) dell'ultimo segmento visibile. Per fare questo si analizza la procedura di cui alle linee 700-790:

710 se il vertice è uno non è necessaria alcuna rimozione;

715 presentazione del menù di correzione;

718 analisi del comando premuto;


720-730-740-750 rimozione visiva delle linee e decrementazione dell'indice dell'array per cancellare anche logicamente le coordinate;

780 aggiornamento delle coordinate per sapere dopo la cancellazione il nuovo punto di partenza del disegno;

785 chiamata al menù principale;

790 rientro in ambiente operativo primario

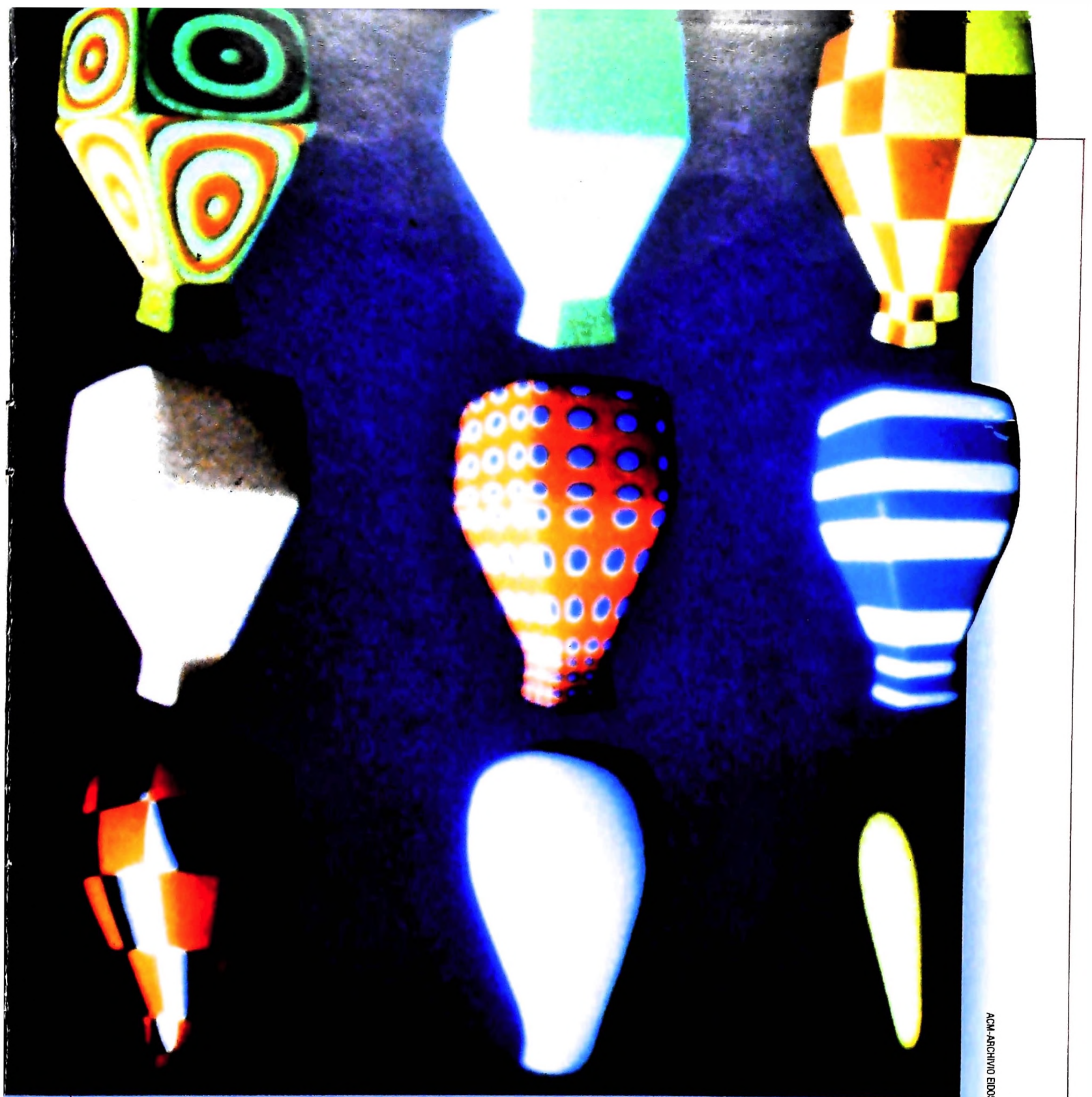
Al momento della pressione del comando "s" in fase di Memo dati, si ottiene la chiusura dell'array relativo ai vertici: la memorizzazione avviene a questo punto mediante il riversa-



Il computer consente di modificare in breve tempo la colorazione e il disegno di un oggetto definito geometricamente e di visualizzarne, contemporaneamente, più soluzioni.

mento totale di tutte le coordinate su di un data file il cui nome era stato scelto all'inizio del programma. A questo punto non è più possibile alcuna correzione e le procedure di visualizzazione e plottaggio operano come nelle precedenti versioni.

Una ulteriore espansione è data dalla possibilità di poter scegliere il colore del disegno che verrà riprodotto su plotter. La semplice procedura relativa è indicata alle linee 3000-3050 e si basa in pratica sulla scelta di un codice-colore che costituisce l'elemento numerico per la creazione di un comando del tipo "Cn" dove n varia tra 0 e 3 e rappresenta i colori disponibili sul plotter di M10.



ACM-ARCHIVIO EIDOS

Espansioni

Anche in questo caso si propongono alcune utili espansioni che a questo punto dovrebbero risultare facilmente attuabili:

- il programma non prevede un controllo sulla presenza di un data file con lo stesso nome di uno che si vuole creare (in pratica se si esegue Memo di un file "A" ed esiste già un file "A" esso viene cancellato e sostituito dal nuovo); potrebbe essere molto utile tale controllo da inserire all'interno della procedura di cui alle linee 1270-1510.

- volendo ampliare il programma è utile prevedere di poter effettuare aggiunte a disegni già esistenti con la "riapertura

dei segmenti chiusi"; si propone il seguente schema:

- lettura del data file indicato e caricamento in un doppio array (CX e CY);
- visualizzazione su schermo del disegno;
- aggiornamento delle coordinate di partenza;
- possibilità di disegnare con memorizzazione e correzione;
- nuova chiusura del segmento.

Molte altre aggiunte sono anche possibili (inserimento di primitive da menù, modifica del tratto di linea ecc...) in modo tale da ottenere un prodotto sempre più completo anche se le strutture dati e di controllo difficilmente dovranno essere sostituite per ottenere gli ampliamenti proposti.

```

1 REM**INIZIALIZZAZIONE E MENU**
4 CLEAR 1500 'MAX SPAZIO PER VERTICI**
5 DIM CX(100),CY(100)
8 CLS
10 PRINT@0,"*PEN4* C>LS,F>ILE,Q>UIT,+,-.MOV.
11 MM=0:X=2:Y=10
12 CL=1:LL$=A$:A$=""
13 A=X B=Y
14 REM**LETTURE DA TASTIERA
15 A$=INKEY$:IF A$=""THEN 15
16 REM
17 IFASC(A$)=28THENX=X+1:GOTO 28
18 IFASC(A$)=29THENX=X-1 GOTO 28
19 IFASC(A$)=30THENY=Y-1 GOTO 28
20 IFASC(A$)=31THENY=Y+1:GOTO 28
21 IFA$="F"THEN CL=0 GOTO 28
22 IFA$="S"THEN GOSUB 1000:GOTO 10
23 IFA$="C"THEN GOSUB 800 GOTO 10
24 IFA$="Q"THEN CLS:END
25 IFA$="R"THEN 700
27 GOTO 12
28 '*****
29 '***CONTROLLO SULLE COORDINATE*****
30 LX=3:HX=236
35 LY=10:HY=60
38 IF X<LX THEN X=LX
40 IF X>HX THEN X=HX
42 IF Y<LY THEN Y=LY
44 IF Y>HY THEN Y=HY
45 IF CL=0THEN 60
46 '*****
48 '***DISEGNO E MOVIMENTO MATITA*****
50 LINE(A-1,B-1)-(A+1,B+1),0,B F
52 LINE(X-2,Y-2)-(X+1,Y+1),1
53 IF MM=1 THEN GOSUB 300
54 PSET(X,Y,1)
55 GOTO 12
58 '*****
60 '***GESTIONE FILE SUPPORTO*****
100 PRINT @ 0,"GESTIONE FILE M>EMO,D>ISEGNA,P
>LOTTER "
110 RP$=INKEY$:IF RP$=""THEN 110
115 PRINT@40,"NOME DEL FILE ":INPUT FL$:GOSUB
1270 IF WF=1 AND RP$<>"M"THEN 10
120 IF RP$="M" THEN 250 ELSE IF RP$="P" THEN 500 ELSE
IF RP$<>"D" THEN 110
122 '*****
125 '***DISEGNA IL FILE CREATO*****
130 CLS
150 INPUT# 1,X1,Y1
160 FOR CI=1 TO 2000
170 IF EOF(1) THEN CLOSE 1:GOTO 10
180 INPUT#1,X2,Y2
190 LINE(X1,Y1)-(X2,Y2)
200 X1=X2:Y1=Y2
210 NEXT CI
220 CLOSE 1
230 '*****
250 '***GESTIONE MEMO DEI DATI*****
255 CLS
260 GOSUB 2500
270 MM=1.CX(0)=0.N=1
275 CX(N)=2 CY(N)=9
280 GOTO 48
300 IF ASC(A$)>27 AND ASC(A$)<32 THEN 330 ELSE
RETURN
320 '*****
330 '*****MEMORIZZA I DATI*****
340 IF A$<>LL$ THEN CX(N)=A:CY(N)=B:N=N+1
350 RETURN
360 '*****

```

```

500 '***DISEGNO SU PLOTTER*****
525 LPRINT CHR$(18) '***PLOTTER GRAFICO*
526 GOSUB 3000
527 LPRINT"H"
528 LPRINT"M 8,106"
530 P$="D"
540 INPUT #1,PX,PY
550 P$=P$+STR$(PX*2)+","+STR$(126-PY*2)
560 FOR CT=1 TO 2000
570 IF EOF(1) THEN CLOSE 1:GOTO 610
580 INPUT #1,PX,PY
590 P$=P$+","+STR$(PX*2)+","+STR$(126-PY*2)
595 IF LEN(P$)>240 THEN LPRINT P$:GOTO 530
600 NEXT CT
610 LPRINT P$
615 LPRINT CHR$(17)
620 GOTO 10
630 '*****
700 '***RIMOZIONE DI SEGMENTI DISEGNO**
710 IF N<=2 THEN 780
715 PRINT@0,"RIMOZIONE SEGMENTI, R>IM.,Q>UIT "
718 RR$="":RR$=INKEY$:IF RR$="R" THEN 720 ELSE IF RR$=
"Q" THEN 780 ELSE 718
719 IF N<=2 THEN 780
720 IF FT=0 THEN FT=1:LINE(CX(N-1)-2,CY(N-1)-2)-(X-2,Y
-2),0 GOTO 718
730 LINE(CX(N-2)-2,CY(N-2)-2)-(CX(N-1)-2,CY(N-1)-2),0
740 N=N-1
750 GOTO 718
780 X=CX(N-1):Y=CY(N-1)
785 GOSUB 2500
790 GOTO 29
795 '*****
800 '***CANCELLA IL FILE*****
805 IF MM<>1 THEN CLS ELSE CLOSE 1:CLS:KILL FL$+".DO"
810 RETURN
820 '*****
1000 '***FINE DEL PROGRAMMA*****
1010 IF MM<>1 THEN 1025
1015 CX(N)=X:CY(N)=Y:CY(0)=N
1018 OPEN FL$ FOR OUTPUT AS 1
1020 FOR Z=1 TO CX(0):PRINT #1,CX(Z):CY(Z):NEXT Z
1025 CLOSE 1
1030 RETURN
1040 '*****
1270 'CONTROLLO SE ESISTE IL FILE*****
1280 WF=0
1300 OPEN FL$ FOR APPEND AS 1
1420 CLOSE 1
1460 OPEN FL$ FOR INPUT AS 1
1480 IF EOF(1)AND RP$<>"M" THEN PRINT
@200,"ATTENZIONE,FILE INESISTENTE
":WF=1:CLOSE 1:KILL FL$+".DO"
1490 IF RP$="M" THEN CLOSE 1
1510 RETURN
1520 REM*****
2500 '***MENU' MEMORIZZAZIONE*****
2600 PRINT@0,"MEMO**C>ANCELLA,R>IMOZIONE LINEE.S
>TOP"
2610 RETURN
2620 '*****
3000 REM**SCELTA COLORE DISEGNO PLOTTER
3010 CLS
3015 PRINT"COLORE DISEGNO"
3020 PRINT:PRINT"0)NERO,1)BLU,2)VERDE,3)ROSSO"
3025 PRINT:INPUT"COLORE ":CL$
3030 IF VAL(CL$)>3 OR VAL(CL$)<0 THEN 3000
3040 LPRINT "C"+CL$
3045 CLS
3050 RETURN
3060 '*****

```

— UN NUOVO MODO DI USARE LA BANCA. —

Conto corrente
più

TANTI PENSIERI
IN MENO CON IL CONTO
CORRENTE "PIÙ"
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.

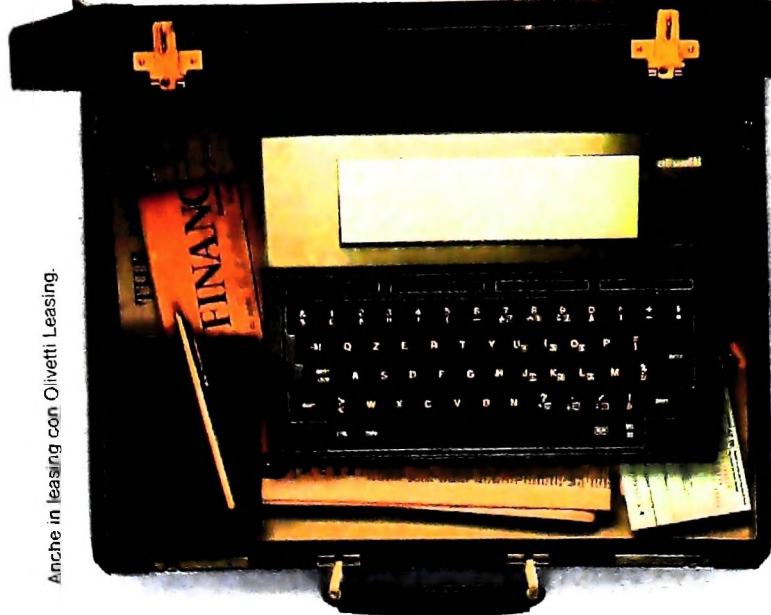
Veniteci a trovare, ci conosceremo meglio.

 **BANCO DI ROMA**
CONSCIAMOCI 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 ventiquattre. 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 contrassegati da Olivetti M10 Punto di Vendita o inviare il coupon a Olivetti, Divisione Personal Computer, Via Meravigli 12, 20123 Milano.

NO ME-COGNOME

VIA/N

CAP/CITTA

TELEFONO