di Andrea de Prisco

File, liste e chiavi primarie

L'argomento di questo mese riguarda le organizzazioni di strutture dati in memoria. Parleremo di file sequenziali, di file ad accesso diretto, delle non troppo note organizzazioni a lista, e infine delle modalità di accesso alle registrazioni memorizzate con trasformazione della chiave primaria.

Prologo

Qualche numero fa dicemmo che un ruolo molto importante in un sistema di calcolo è svolto dalle periferiche di ingresso/uscita. Fra queste, le memorie di massa (dischi, nastri e tamburi), pur non essendo la loro presenza teoricamente indispensabile (è sempre possibile ipotizzare l'esistenza di calcolatori senza tali dispositivi) permettono, a un costo abbastanza contenuto, di memorizzare grosse quantità di dati. Dicevamo inoltre che il loro maggiore difetto era la scarsa velocità di accesso alla registrazione inferiore di molti ordini di grandezza rispetto alle memorie centrali. Inoltre tale limitatezza era maggiormente avvertibile quanto più la nostra unità era economica. In altre parole le unità a nastri erano sì le più economiche ma anche le più lente alla risposta, contro le unità a tamburo che ad una velocità assai più elevata accompagnano un costo altrettanto alto.

Morale della favola: occorre sfruttare al meglio tali dispositivi, se si vuole contenere il più possibile la loro naturale lentezza. Cominciamo col dare uno sguardo ai due più significativi tipi di organizzazione di registrazioni su memorie di massa.

File sequenziali

File in inglese vuol dire archivio, elenco. Un file memorizzato su memoria di massa è un insieme più o meno grande di registrazioni alle quali possiamo accedere specificando opportuni parametri, in dipendenza del tipo di file stesso che ci accingiamo ad usare. Il primo tipo che vedremo è il file sequenziale, una struttura abbastanza semplice che deve il suo nome al modo in cui le registrazioni vengono memorizzate: una di seguito all'altra. Tale organizzazione è anche la prima in ordine cronologico, infatti le prime memorie di massa permettevano solo architetture di questo tipo essendo loro stesse di natura sequenziale. Ci stiamo riferendo alle unità a nastri magnetici: niente di meglio per loro che disporre le registrazioni sequenzialmente, lungo il verso di scrittura della testina. Per motivi altrettanto storici, i file sequenziali non sono spariti con l'avvento dei dispositivi ad accesso diretto (dischi e tamburi), primo perché molti programmi erano già stati scritti pensando «sequenzialmente» e poi perché in alcuni casi la semplicità di un file sequenziale unita al fatto che alcune volte la possibilità di accedere direttamente a un dato punto di file non serve, fanno sì che questi tipi di file siano ben lungi dall'essere inutili.

In figura 1 è mostrata l'organizzazione di un file sequenziale. Come si può notare c'è ben poco da dire: le varie registrazioni sono messe l'una di seguito all'altra, semplicemente separate da un opportuno carattere separatore. Si vede inoltre come le varie registrazioni possono essere di lunghezza diversa, dato che ognuna di queste è racchiusa tra due separatori e dunque non c'è modo di generare confusione: se si vuole ad esempio la quarta registrazione è sufficiente leggere il file dall'inizio, contare tre separatori, prelevare la nostra registrazione che «durerà» sino al separatore successivo.

Per creare un file sequenziale è necessario semplicemente dare un comando di apertura specificando il nome del file, inserire le varie registrazioni una di seguito all'altra e infine chiuderlo con un opportuno comando all'unità. In generale non sono ammesse modifiche all'interno del file creato proprio in virtù delle registrazioni a lunghezza variabile. Tuttalpiù è possibile appendere nuove registrazioni in coda a quelle già memorizzate in modo da allungare la dimensione del file stesso.

Per quanto riguarda l'operazione di ricerca, posto che non sappiamo in quale posizione si trova la registrazione che cerchiamo (e quindi non possiamo contare sui separatori), ma semplicemente ad esempio sappiamo che quanto cercato inizia per «Rossi Mario», dobbiamo distinguere due casi. Il primo riguarda registrazioni inserite per così dire alla rinfusa: sembrerà banale, ma l'unico modo per pescare una registrazione è cercarla brutalmente fra le tante, accedendo una dopo l'altra a tutte.

Se invece il nostro file è ordinato, ossia le registrazioni sono ad esempio in ordine alfabetico, sarà necessario continuare a fare accessi al file fino a quando non superiamo nell'ordine la registrazione che cercavamo.

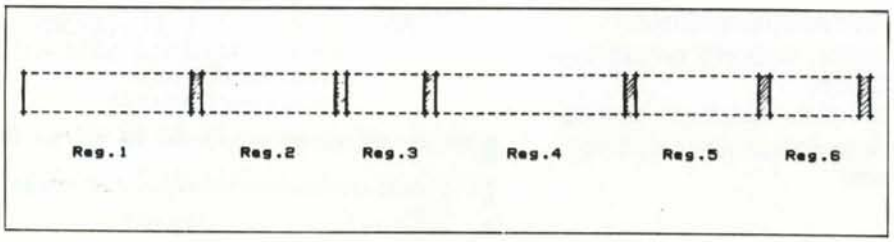


Figura 1 - File sequenziale: si noti come le registrazioni possono avere lunghezze diverse.

File ad accesso diretto

Con la nascita dei dispositivi ad accesso non sequenziale sono nati ovviamente anche i file ad accesso diretto. In tale tipo di organizzazione, è possi-

bile memorizzare più registrazioni anche in ordine non sequenziale: posso cioè inserire la registrazione 5, poi la 26, poi la 3 e così via. Allo stesso modo, in lettura, se voglio la registrazione 8, basterà specificarlo tramite un opportuno comando e l'unità di memoria di massa provvederà a pescarla e a darmela. L'unica limitazione imposta dal tipo di organizzazione è che bisogna specificare a priori la lunghezza massima delle registrazioni in quanto, se ad esempio cominciamo coll'inserire l'ottava, bisognerà che il sistema lasci disponibili davanti a questa altre sette posizioni momentaneamente inutilizzate. Come conseguenza di questo fatto abbiamo che (vedi fig. 2) le differenze di lunghezza delle varie registrazioni si traducono, di fatto, in spazi di memoria inutilizzati.

Poniamoci ora, come prima, nel caso in cui cerchiamo una data registrazione, ma non sappiamo né se c'è, né in quale posizione del nostro file è ubicata. Come prima sappiamo ad esempio che questa inizia per «Rossi Mario». Anche qui due casi: registrazioni inserite alla rinfusa o registrazioni in ordine alfabetico. Nel primo caso niente da fare: occorrerà scandire sequenzialmente tutto il file, con ovvi svantaggi. Se invece le registrazioni sono ordinate, per risparmiare accessi al file possiamo procedere secondo uno di questi due algoritmi: Ricerca Binaria e Ricerca a Salti.

Col primo algoritmo, posto ad esempio di avere un file di 100 posizioni, proviamo a controllare la 50-esima. Se la registrazione prelevata inizia per «Rossi Mario» l'abbiamo trovata, altrimenti se inizia per una coppia Cognome-Nome in ordine alfabetico precedente a «Rossi Mario» andremo a cercare nei secondi 50 elementi, in caso contrario cercheremo nei primi 50. Analogamente possiamo applicare il nostro algoritmo alle 50 registrazioni così selezionate, spaccando nuovamente in mezzo e confrontando sempre con «Rossi Mario». Se ancora non l'abbiamo trovata selezioneremo un altro sottofile, questa volta di 25 elementi, sul quale applicare ancora una volta il nostro algoritmo. A furia di spaccare il nostro file arriveremo certamente a un sottofile di una sola posizione: a questo punto se questa inizia con «Rossi Mario» l'abbiamo trovata, altrimenti vuol dire che non c'è. È facile convincersi che con tale procedimento si risparmiano molti accessi. Infatti nel caso del file sequenziale, nella peggiore delle ipotesi se 100 sono le registrazioni, occorre fare 100 tentativi (se uno è sfortunato e la registrazione che cerca è proprio l'ultima del suo file). Coi file ad accesso diretto e l'algoritmo della ricerca binaria tale numero (sempre nella peggiore delle ipotesi) si riduce a 7 tentativi: un bel colpaccio!

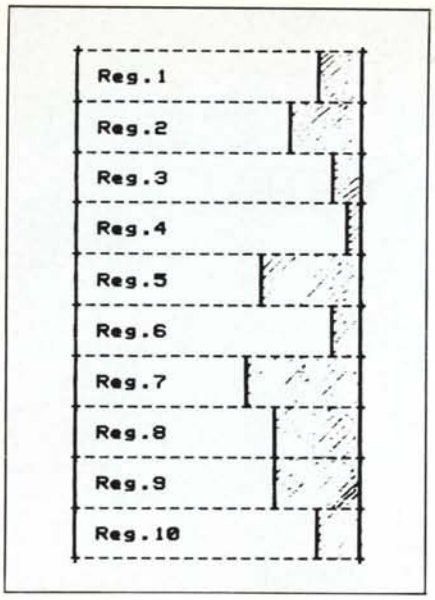


Figura 2 - File ad accesso diretto; qui le singole registrazioni o hanno tutte la medesima lunghezza o si assume come dimensione degli elementi del file la lunghezza della registrazione più lunga (nel nostro caso la 4).

Con l'algoritmo della ricerca a salti si procede così: siamo sempre nell'ipotesi di 100 registrazioni e quella che noi cerchiamo inizia per «Rossi Mario». Si fa il primo tentativo nella posizione 10. Se non l'abbiamo trovata e questa precede in ordine alfabetico quella che cerchiamo, si prova alla posizione 20, poi alla 30 e così via (effettuiamo cioè salti di 10) fino a quando non troviamo una registrazione che supera in ordine alfabetico «Rossi Mario». Se tale registrazione non c'è ossia saltando-saltando arriviamo a fine file, vorrà dire che nemmeno «Rossi Mario» è presente nel file. Appena invece troviamo una registrazione che supera, torniamo indietro (al salto precedente) e scandiamo sequenzialmente l'intervallo così selezionato o applichiamo su di questo nuovamente l'algoritmo a salti, ovviamente compiendo salti più piccoli. Si dimostra come il salto ottimale in termini di minor numero di accessi da compiere, se N sono le registrazioni e si procede prima per salti e poi per scansione lineare, sia pari alla radice quadrata di N (noi avevamo 100 posizioni, saltavamo di 10 in 10).

Liste e puntatori

Chi è abituato a lavorare in Basic, avrà certamente padronanza nel manipolare array di più dimensioni, variabili intere, in virgola mobile... e basta!

Esistono però altre strutture dati altrettanto importanti in informatica che vivono un po' all'oscuro, solo perché... il Basic non le implementa. Così capita che all'esame del primo anno «Teorie ed Applicazioni delle Macchine Calcolatrici» del corso di laurea in Scienze dell'Informazione, gli studenti il più delle volte tremano davanti a esercizi che coinvolgono la manipolazione di tali strutture dati.

Pare infatti che tale linguaggio di programmazione sia stato inventato solo per insegnare le nozioni di base della programmazione a non si sa quale categoria di studenti. Vista poi la facilità con cui questi imparavano a programmare, qualcuno ha pensato di implementarlo realmente su una macchina reale. Come se non fosse bastata la già enorme confusione che aveva provocato il Fortran circa la programmazione di un computer.

Bando comunque alle polemiche, con la promessa di ritornare in questa stessa rubrica sul tema di linguaggi di programmazione un po' più seri, l'argomento di questo paragrafo riguarda, come dice il titolo, le strutture dati a lista o se preferite con puntatori.

Cominciamo a dare uno sguardo alla figura 3: quello sgorbio rappresenta una lista di elementi. Teniamo a precisare che tale organizzazione possiamo trovarla sia nella memoria centrale di un calcolatore quanto per organizzare registrazioni (o meglio, blocchi di registrazioni) su disco. Possiamo notare, sempre dalla figura 3, che gli elementi di una lista sono in qualche modo collegati tra di loro (tramite i puntatori, le frecce) e che ognuno di loro è formato da due campi: un primo campo informazione e un secondo campo puntatore. In questo modo la lista si comporta

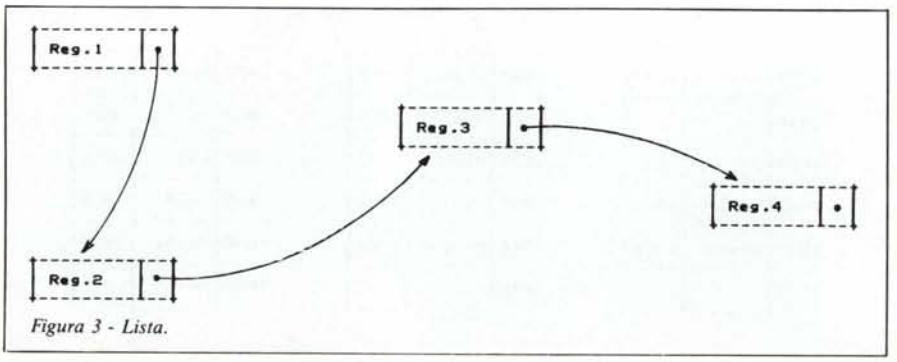


Figura 3 - Lista.

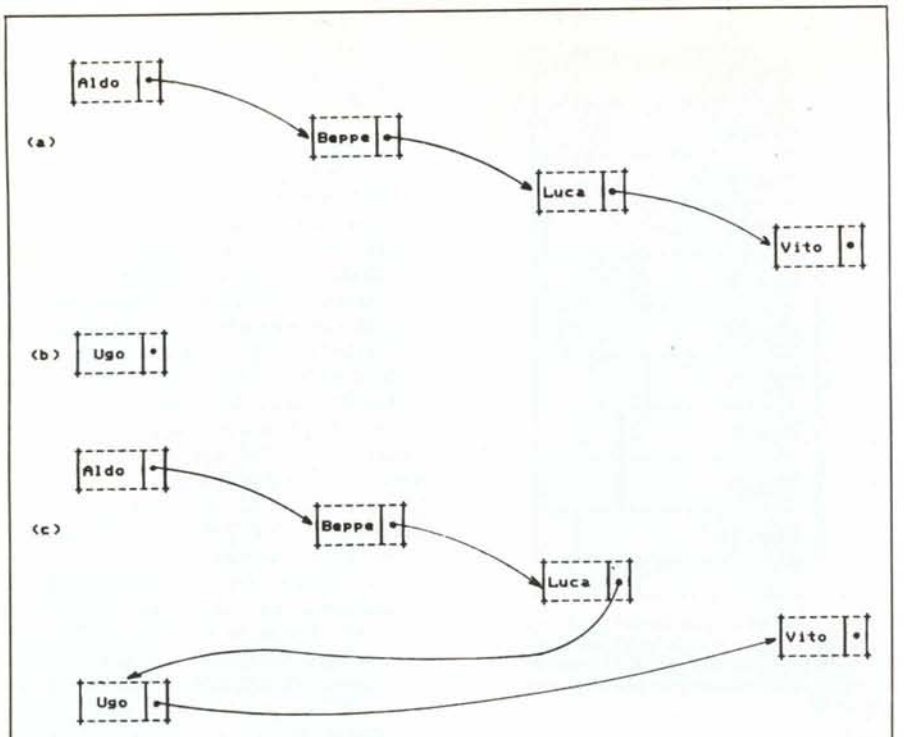


Figura 4 - Inserimento in lista

come una struttura sequenziale in quanto per raggiungere un determinato elemento bisogna scorrerla, come si farebbe con un file sequenziale. Ciò significa che per raggiungere il terzo elemento, prendiamo il primo, tramite il suo puntatore ci spostiamo sul secondo e da questo, semplicemente interrogando il suo puntatore, possiamo passare al terzo elemento, quello cercato.

A questo punto tutti si saranno chiesti dove sta la differenza con un file sequenziale: semplice, le liste possono essere manipolate a piacimento, si possono fare tutte le inserzioni che si vogliono, tutte le cancellazioni, possiamo scambiare di posto due o più elementi, possiamo in qualsiasi momento ordinare gli elementi e, dulcis in fundo, inserire un elemento in una lista ordinata, lasciando la stessa in ordine, ossia mettendo il nuovo elemento al posto giusto. Scusate se è poco. Ah! dimenticavamo: tutto questo senza mai spostare un solo elemento da dove, fisicamente, si trova.

Facciamo un esempio: in figura 4a è mostrata una lista di nomi. Notiamo subito che tale lista è ordinata alfabeticamente. In figura 4b è raffigurato l'elemento che vogliamo inserire nella lista, rispettando l'ordine alfabetico. Ugo viene dopo di Lucia e prima di Vito: bene, prendiamo la nostra lista e cominciamo a scorrerla. Il primo elemento è Aldo, il suo puntatore punta a Beppe; da questo possiamo passare a Luca e, attenzione, è arrivato il momento di fare l'inserimento. Rullino i tamburi: il puntatore di Luca lo mettiamo a Ugo e quest'ultimo lo facciamo puntare da Luca, ottenendo quanto raffigurato in figura 4c. Potete verificare facilmente che la lista così modificata è ancora ordinata e non abbiamo spostato nessun elemento dal suo posto: il tutto è avvenuto modificando puntatori.

All'inizio può sembrare non troppo chiaro: garantiamo che dipende dal fatto che non siamo troppo abituati a vedere la memoria di un calcolatore come un grosso foglio bianco, su cui gomma e matita alla mano siamo in grado di muovere frecce come appena visto. Nulla di più sacrosanto, quello che abbiamo visto è solo un esempio grafico fatto per far vedere solo un po' qual era l'andazzo della situazione. Tranquilli, le memorie restano sempre insieme di celle numerate, niente frecce, gomme e matite. Mostriamo cosa effettivamente avviene in memoria quando costruiamo una lista.

Diciamo di riservare 6 caratteri per i nostri nomi (come notate sono stati appositamente scelti corti per non sprecare risorse) e 4 caratteri per il puntatore. La nostra lista ha inizio all'indirizzo di memoria 1000 (vedi fig. 5a), dove troviamo il primo elemento, Aldo. Gli altri elementi stanno agli indirizzi mostrati sempre in figura 5a: notiamo subito che sono tutt'altro che in ordine, per di più abbiamo anche un bel buco in mezzo ossia una manciata di celle inutilizzate: non ha importanza! Dalla figura 4a vediamo che Aldo punta a Beppe: metteremo l'indirizzo di questo nel puntatore del primo, così il 1040 affianco ad Aldo rappresenta il primo puntatore. Dopo Beppe c'è Luca, che sta all'indirizzo 1030: metteremo tale valore nel campo puntatore di Beppe. Infine l'indirizzo di Vito (1020) sarà posto nel campo puntatore di Luca e la nostra lista è completa. Si noti il campo puntatore di Vito non utilizzato in quanto non vi sono altri elementi: è ovvio che se dovessimo aggiungere altri dopo Vito non esiteremmo a inizializzarlo. La lista è ordinata: a partire da Aldo, seguendo i vari puntatori, possiamo elencare nell'ordine tutti e quattro i nomi. Vogliamo puntualizzare il fatto che non conta la posizione fisica in memoria in quanto l'ordine vero è dato sempre e solo dai puntatori.

L'aggiunta di un elemento Ugo nella lista, mostrato in fig. 5b sconvolge di fatto i soli puntatori. Ugo «fisicamente» possiamo piazzarlo in qualsiasi posto della memoria; aggiustando i puntatori manteniamo l'ordine. Infine in figura 5c inseriamo l'elemento Zeno che come visto modifica solo il puntatore di Vito.

Figura 5

(a)

1000	Aldo	1040
1010		
1020	Vito	—
1030	Luca	1020
1040	Beppe	1030
1050		

(b)

1000	Aldo	1040
1010	Ugo	1020
1020	Vito	—
1030	Luca	1010
1040	Beppe	1030
1050		

(c)

1000	Aldo	1040
1010	Ugo	1020
1020	Vito	1050
1030	Luca	1010
1040	Beppe	1030
1050	Zeno	—

(a)	(b)
Rossi Mario	10
Bianchi Gino	1
Celi Vito	9
Laini Adolfo	8
Trosi Franco	6
Gabbi Marco	1
Massa Luca	12
Zonin Dario	13
Macchi Paolo	6
Brizzi Mario	0
Galli Beppe	1
Monti Ugo	0
Valli Aldo	10
Simi Fabio	5

Figura 6 - Chiavi primarie delle registrazioni (a) da introdurre nel file e posizioni (b) ottenute dalla trasformazione.

Figura 7

	Cognome Nome	Recapito	Telefono	
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				

Figura 8

	Cognome Nome	Recapito	Telefono	
0	Brizzi Mario			4
1	Bianchi Gino			2
2	Gabbi Marco			3
3	Galli Beppe			
4	Monti Ugo			
5	Simi Fabio			
6	Trosi Franco			7
7	Macchi Paolo			
8	Laini Adolfo			
9	Celi Vito			
10	Rossi Mario			11
11	Valli Aldo			
12	Massa Luca			
13	Zonin Dario			

Accesso per chiave primaria

Giunti a questo punto possiamo mostrare un tipo di organizzazione di registrazioni che combina la praticità dei file ad accesso diretto con la flessibilità delle organizzazioni a lista. Immaginiamo di dover organizzare in un file alcune notizie riguardanti dei nostri conoscenti, ad esempio: Nome, Recapito e Telefono. Poniamoci inoltre nell'ipotesi che conosciamo a priori soltanto grosso modo quante registrazioni dovremo inserire (è importante), e che per qualche motivo i nostri inserimenti saranno tutt'altro che in ordine alfabetico.

Diciamo inoltre che non vi siano conoscenti omonimi in modo da poter individuare col solo nome e cognome una ben precisa registrazione. In figura 6a sono mostrati i nomi delle persone che vogliamo introdurre. In figura 7 il nostro file ancora vuoto: notiamo un campo «Cognome Nome», un campo «Recapito», uno «Telefono» e un campo aggiuntivo che ci permetterà di collegare a lista alcuni insiemi di registrazioni. Tale organizzazione è detta «accesso per chiave primaria con gestione delle collisioni con liste confluenti». Calma: procediamo con ordine. «Accesso per chiave primaria» vuol dire che per trovare una data registrazione non ci baseremo sulla sua posizione nel file (che daltronde non conosciamo) ma semplicemente dalla sua chiave. Per chiave si intende quella parte di registrazione che identifica univocamente la registrazione stessa: nel nostro caso la chiave è il campo «Cognome Nome». Le «collisioni» le vedremo presto.

Cominciamo ora a inserire le nostre registrazioni. Iniziamo da «Rossi Mario»: dove lo mettiamo, in modo da poterlo ripescare presto quando ci servirà? Semplice: ci serviremo di una qualsiasi funzione che presa la chiave restituisce un numero tra 0 e 13, quante sono le posizioni del nostro file. Ad esempio possiamo sommare tra di loro il codice Ascii dei caratteri costituenti la chiave, e del numero ottenuto calcolare il resto della divisione con 14 che dà appunto un numero compreso tra 0 e 13. In fig. 6b sono mostrate le posizioni ottenute con la trasformazione appena mostrata: per ogni chiave «Cognome Nome» è stata calcolata la somma dei corrispondenti codici Ascii dei caratteri, e il numero così ottenuto è stato diviso per 14 prelevando infine il resto di tale divisione.

Dunque, «Rossi Mario» lo mettiamo nella posizione 10; «Bianchi Gino», nella posizione 1; «Celi Vito» nella posizione 9; «Laini Adolfo» nella posizione 8; «Trosi Franco» nella posizione 6; «Gabbi Marco» nella posizione 1... ehi! un momento: la posizione 1 è già occupata da Bianchi!

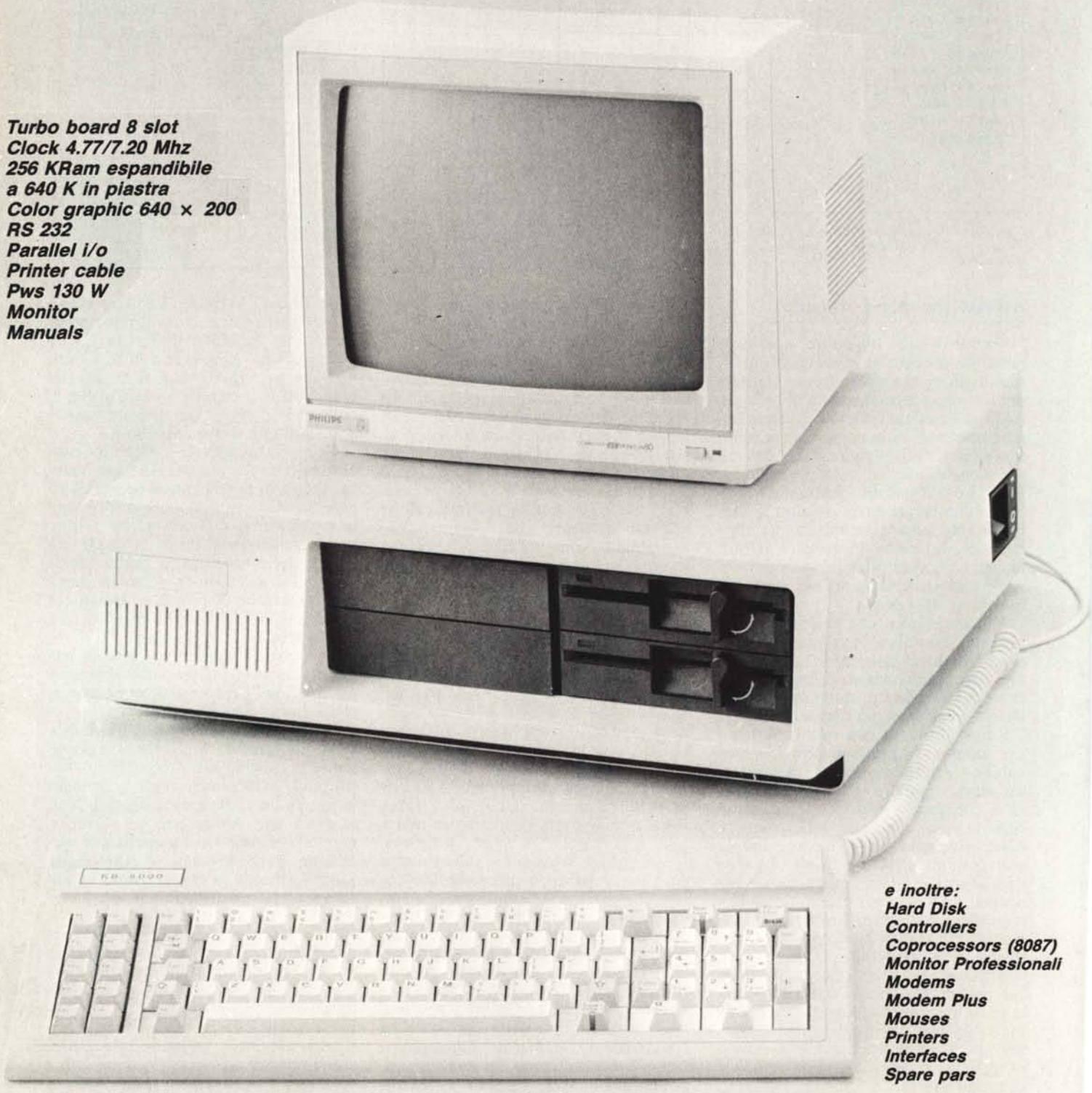
Niente paura: non pretendevate che la nostra funzione Chiave-Posizione fosse così perfetta da mappare una e una sola chiave in ogni posizione. Quello che è successo è soltanto una collisione, peraltro prevedibilissima. Ciò che faremo è semplicemente trovare una posizione libera dopo «Bianchi» che per l'appunto è già occupata, e col meccanismo dei puntatori la collegheremo a lista con questa. Al momento attuale, la posizione libera è la 2, dove metteremo «Gabbi Marco», e nell'ultimo campo della registrazione 1 metteremo il puntatore alla registrazione 2. Continuiamo a introdurre registrazioni: «Massa Luca» e «Zonin Dario» li metteremo, sempre per la trasformazione (vedi fig. 6b), senza problemi nelle posizioni 12 e 13 che sono libere. «Macchi Paolo» invece collide con Trosi e, come prima, troveremo una posizione libera, la 7, che collegheremo a lista con la 6. Continuando così inseriremo tutti gli elementi nel file, come è visibile in fig. 8, collegando a lista come sempre tutte le collisioni che si presenteranno.

Dopo tutta questa fatica, avremo costruito un file ad accesso super veloce: infatti in media con poco più di un accesso riusciamo sempre a ritrovare la registrazione che cerchiamo. L'operazione di lettura, infatti, è assai più veloce della scrittura: a partire dalla nostra chiave, applicheremo la stessa trasformazione di prima, e ottenuta la posizione, non ci resterà altro da fare che prelevare la registrazione se l'abbiamo trovata, oppure scorrere la lista delle collisioni (al massimo due altri tentativi, nel caso nostro) se ne avevamo avute in fase di scrittura.

Facciamo un esempio: il file è mostrato in figura 8; vogliamo la registrazione la cui chiave è Zonin Dario. Applicando la trasformazione otteniamo la posizione 13 (come da fig. 6b). Non ci resta che controllare se effettivamente la posizione 13 è quella che cerchiamo: nel nostro caso sì. Altro esempio: cerchiamo la registrazione «Galli Beppe». La trasformazione dà 1; vediamo cosa contiene la posizione 1: «Bianchi Gino», non va bene: scorriamo la lista corrispondente. Preleviamo il numerino dell'ultimo campo della registrazione 1 (che è un 2) e controlliamo nella nuova posizione così ottenuta. Anche questa non va bene, infatti la posizione 2 contiene la registrazione relativa a Gabbi. Continuiamo a scorrere, come prima, prelevando il puntatore della posizione 2, che nel caso nostro è 3. L'abbiamo trovata, Galli è nelle nostre mani... MC

IL COMPATIBILE È APPARSO!

Turbo board 8 slot
Clock 4.77/7.20 Mhz
256 KRam espandibile
a 640 K in piastra
Color graphic 640 × 200
RS 232
Parallel i/o
Printer cable
Pws 130 W
Monitor
Manuals



e inoltre:
Hard Disk
Controllers
Coprocessors (8087)
Monitor Professionali
Modems
Modem Plus
Mouses
Printers
Interfaces
Spare pars

Pubblicheremo prossimamente l'elenco dei distributori acquisiti

Cerchiamo distributori per zone ancora libere

QUASAR S.r.l. - 13050 Pratrivero (VC) - Tel. 015/778804/377 - Tx 211401 MILFIL