Ritorniamo sull'archivio indirizzi. Prima di presentare il programma di questo mese ci sentiamo in dovere di tornare nuovamente a parlare del programma "Archivio Indirizzi" presentato su MC n. 19.

In redazione sono giunte numerose lettere e telefonate da parte di lettori che si sono trovati in difficoltà di vario genere; non è stato purtroppo possibile risolvere personalmente ogni problema e pertanto abbiamo deciso di pubblicare alcuni consigli in modo da mettere in grado coloro che fossero nei guai di uscirne da soli. Precisiamo comunque che nel listato non esistono errori che precludano a priori il buon funzionamento del programma.

Caccia all'errore

Cominciamo dalle raccomandazioni più banali ma non per questo meno importanti; molto spesso errori "stupidi" sono i più difficili a trovarsi. Il listato di un programma è come una formula matematica, cambiate anche solo una virgola senza cognizione di causa ed il risultato sarà imprevedibile. Confondere lo 0 con una O o l'1 per una I può sempre capitare...

Vediamo dunque quali possono essere le principali cause di malfunzionamento del programma in questione. Probabilmente ci siamo resi colpevoli di non aver approfondito la descrizione del listato facendo troppo affidamento su una sua interpretazione diretta, resa difficoltosa (in questo caso) da una logica di funzionamento piuttosto complessa. Diamo per scontata la mancanza di errori di trascrizione; questo problema purtroppo dovete sobbarcarvelo da soli, con una buona spunta del vostro listato. Come la maggioranza delle macchine il TI 99 considera il segno della virgola come un carattere speciale e non ne accetta l'inserimento in una stringa tramite una INPUT; chi è normalmente abituato a separare il numero civico della via o della piazza tramite l'uso della virgola suddetta si troverà, suo malgrado, nella necessità di astenersi dal continuare a farlo, pena un temporaneo inchiodamento del programma tramite un INPUT ERROR IN XXX.

Nella procedura di inserimento è possibile modificare quanto scritto nei campi precedenti dello stesso indirizzo introducendo in quello attuale il simbolo dell'uguale (=) fino a raggiungere, a ritroso, quello errato e reinserendo poi tutti i dati dei campi seguenti.

Ad esempio se alla domanda finale "Va bene?" rispondiamo N, il computer ci chiederà di introdurre nuovamente le annotazioni; a questo punto digitando = e premendo ENTER, torneremo ancora indietro ed il computer ci chiederà il telefono, e così via fino a raggiungere il dato errato.

È sicuramente possibile modificare il programma in modo da dover riscrivere l'intero indirizzo nel caso qualche cosa non andasse bene, ma francamente non riusciamo a vedere il motivo per cui dovremmo rendere più lunga una procedura studiata appositamente per abbreviare i tempi di inserimento. Se comunque volete farlo dovete eliminare le linee 700-730-760-790-830-860 e modificare la 910 in IF A < >35 THEN 660.

Per quanto riguarda il campo Provincia questo deve necessariamente contenere un dato di due lettere corrispondente alla sua targa automobilistica; se la cosa non fosse di vostro gradimento dovete agire sulle linee che effettuano il controllo della lunghezza di stringa (800 e 1440).

Se il vostro registratore non è sincronizzato tramite il controllo di Remote con la consolle, non potete usarlo per questo programma, inoltre dovete fare attenzione poiché se in fase di lettura dell'archivio non posizionate la cassetta nel punto esatto in cui si trovava al momento della creazione, otterrete sicuramente un bel FILE ERROR IN 250 o IN 300. Questo perché nel momento in cui la CPU deve leggere il record di testa, contenente il nome dello schedario e gli elementi inseriti, è possibile che invece, a causa di un cattivo posizionamento del nastro, vengano trasferiti i dati di un indirizzo, causando una condizione di errore; assicuratevi pertanto di aver riavvolto completamente la cassetta o segnatevi il valore di partenza del contanastri al momento della creazione dello schedario.

Precauzioni d'uso

Dopo aver visto alcune delle possibili cause (le più comuni) di malfunzionamento, occupiamoci adesso di alcuni inconvenienti che si possono presentare dopo aver inserito una quantità di dati superiore a quella contenibile nella memoria del TI 99.

Ogni singolo indirizzo può essere composto da un massimo di 128 caratteri; 9 caratteri occorrono per memorizzare il codice e 7 per indicare la lunghezza delle stringhe contenenti i vari campi; ne consegue il fatto che si hanno a disposizione 128 - 9 - 7 = 112 caratteri utili per la memorizzazione dei dati. Se si supera tale limite, in fase di scrittura dell'archivio su nastro si avrà un FILE ERROR.

Sarebbe stato sicuramente possibile inserire un controllo sulla lunghezza dei vari campi, ma questo avrebbe causato una minore flessibilità d'uso in quanto tale lunghezza non sarebbe stata dinamica ma stabilita a priori; ossia se aveste voluto inserire un nome di 40 caratteri ciò non sarebbe stato possibile (posta la lunghezza massima a 30) mentre attualmente potete farlo, purché recuperiate i 10 caratteri eccedenti in un altro campo.

```
864 IF LEN(A$(X)&B$(X)&C$(X)&D$(X)&E$(X)&F$(X)&G$(X)) <=112 THEN 870
865 PRINT :::"INDIRIZZO TROPPO LUNGO":::
866 GOTO 660
```

Figura 1

Se volete che sia il programma a controllare il non superamento di tale limite è possibile aggiungere un controllo di lunghezza sul concatenamento delle stringhe costituenti i singoli campi (figura 1) in modo che al superamento del limite di 112 byte l'indirizzo venga rifiutato. Potete anche, se volete, modificare la specifica della lunghezza del record (linee 130 e 240) in 192; avrete così a disposizione 192 - 9 - 7 = 176 caratteri per ogni indirizzo, ma attenzione perché, a questo punto, bisogna tirare in ballo il dannato problema della quantità di memoria disponibile sul TI 99 per la memorizzazione dei dati.

Un problema problematico

Una delle caratteristiche meno simpatiche del TI 99 è che non esiste il modo di sapere, tramite il TI-Basic, quale sia la quantità di memoria occupata dal programma e quindi, per differenza, quella rimasta libera per i dati. Questo implica la possibilità di trovarsi un bel messaggio di MEMORY FULL sul video proprio nel momento meno opportuno.

Si può aggirare l'ostacolo caricando il programma tramite l'Extended Basic che permette con l'istruzione SIZE di sapere quanta memoria libera sia a disposizione

dell'utente; facendo la differenza tra la Ram libera prima del caricamento e quella disponibile dopo risulta che il programma Archivio Indirizzi occupa 5122 byte, che diventano 7538 dopo aver dato il RUN a causa dello spazio necessario per il dimensionamento delle matrici.

Ad ogni inserimento di un indirizzo al limite delle specifiche (ossia impiegando tutti i caratteri disponibili) vengono usati circa 150 byte; ne consegue il fatto che nel peggiore dei casi sarà possibile registrare all'incirca 45 record, mentre si potrà raggiungere il numero di 100 solo nel caso di uno sfruttamento parziale del limite di 112 caratteri per indirizzo. Tenete presente comunque che, in media, 60 caratteri sono sufficienti per i dati che normalmente vengono inseriti; se poi volete evitare a priori ogni possibile inconveniente, stabilite un numero massimo di 45 indirizzi per ogni schedario.

Il labirinto, il topo ed il formaggio

di Marco Toffolo - Melegnano (MI)

Questo programma prevede la costruzione di un labirinto casuale, con una sola entrata ed una sola uscita, che verrà percorso e risolto da topo (rappresentato con un asterisco), il quale dopo aver trovato l'uscita ed il formaggio (simboleggiato con un quadratino giallo) tornerà al punto di partenza percorrendo questa volta la via più breve, ossia senza imboccare vicoli ciechi.

Il gioco risulta divertente ed interessante soprattutto per chi non ne conosce le regole. L'ignoranza delle regole di risoluzione provoca la stessa curiosità e lo stesso interesse che si prova di fronte ad un qualsiasi trucco visivo, con la differenza che qui l'inganno non è per la vista ma per la ragione. E insomma il gusto per le apparenze, per ciò che sembra ma che in realtà non è.

Le regole per la risoluzione dell'andata e del ritorno in un labirinto di questo genere sono molto semplici: per uscire è sufficiente immaginare di appoggiare una mano sulla parete di destra (o di sinistra) e di avanzare senza mai perdere il contatto con essa, mentre per tornare per la via più breve è necessario usare il "filo di Arianna" che, nel percorso di andata, viene recuperato nei tratti che si è costretti a ripercorrere dopo aver imboccato un vicolo cieco.

Descrizione del programma

Il programma è diviso in cinque blocchi: il primo è relativo alla titolazione e alla stampa delle poche istruzioni necessarie; il secondo riguarda la grafica preparatoria per il campo; il terzo genera il labirinto casuale; il quarto controlla il movimento del topo e la ricerca della via d'uscita; il quinto infine determina il ritorno per la via più breve.

Per generare il labirinto è stato usato un algoritmo già noto, che opera su un campo base che consiste di un numero dispari di righe e di colonne.

La partenza avviene da una cella qualsiasi di coordinate dispari (nel nostro caso dalle coordinate X = 1, Y = 13 del labirinto, corrispondenti alla colonna 4, riga 15 dello schermo), quindi viene scelta una direzione a caso e si ispeziona con una CALL GCHAR una corrispondente cella che dista esattamente due celle da quella attuale; se questa fa ancora parte del campo libero viene stampato il carattere 32 in questa nuova coordinata e nella barriera che separa quest'ultima dalla precedente. Se invece la cella ispezionata fa già parte del percorso si prende comunque questa come punto di partenza per un nuovo calcolo ma in tal caso ovviamente non si elimina la barriera.

Il ciclo si ripeterà fino al completamento del labirinto.

La costruzione del labirinto, che in ogni caso avrà una sola via di uscita possibile, può comunque essere arrestata premendo un qualsiasi tasto dal momento in cui un braccio avrà toccato la cornice di destra; non prima pena la segnalazione di errore e il conseguente resettamento del programma.

Il movimento del topo per la ricerca dell'uscita è stato realizzato tramite l'uso della matrice A(4,6) che contiene nelle prime due colonne di ogni riga i diversi incrementi di Y e X per le quattro possibili direzioni, e nelle restanti quattro le priorità di scelta delle direzioni da ispezionare (destra, avanti, sinistra, indietro) in rapporto alla direzione di provenienza.

Il codice ASCII 42, rappresentante il topo, dopo essere stato visualizzato nella nuova posizione viene cancellato dal carattere 80 (che appare sul video come il 32 dello spazio) nel caso in cui quella via non sia ancora stata percorsa, e con il carattere 32 quando, dopo aver imboccato un vicolo cieco, il topo è costretto a tornare sui suoi passi.

Volendo mettere in evidenza l'uso dei caratteri e verificare lo stratagemma del "Filo di Arianna", sarà sufficiente variare il colore del codice 80 (linea n. 270) assegnandogli ad esempio quello rosso tramite una CALL COLOR (7,9,9). MC

```
10 REM   *** IL LABIRINTO DI MARCO TOFFOLO ***
20 CALL CLEAR
30 CALL SCREEN(8)
40 PRINT "IL LABIRINTO, IL TOPO":"ED IL FORMAGGIO"
50 PRINT "-----------------------------":::
60 PRINT "Il programma prevede la"
70 PRINT "generazione di un labirinto"
80 PRINT "casuale, la cui costruzione"
90 PRINT "puo' essere interrotta"
100 PRINT "premendo un tasto qualsiasi"
110 PRINT "quando viene raggiunta la"
120 PRINT "cornice di destra.":::
130 PRINT "Un topo cerchera' l'uscita,"
140 PRINT "prendera' il formaggio e"
150 PRINT "tornera' indietro percorren-"
160 PRINT "la via piu' breve."
170 PRINT :::"*PREMI UN TASTO PER INIZIARE"
180 CALL KEY(0,KEY,STATUS)
190 IF STATUS=0 THEN 180
200 CALL CLEAR
210 CALL SCREEN(5)
220 REM  *** GRAFICA CAMPO ***
230 CALL COLOR(1,8,8)
240 CALL COLOR(15,2,13)
250 CALL COLOR(4,12,12)
260 CALL COLOR(8,2,2)
270 CALL COLOR(7,8,8)
280 CALL COLOR(2,2,8)
290 CALL COLOR(3,5,5)
300 CALL CHAR(150,"FF8080808080808FF")
310 CALL HCHAR(1,1,150,756)
320 CALL HCHAR(1,1,90,64)
330 CALL VCHAR(1,31,90,120)
340 CALL HCHAR(23,1,90,64)
350 CALL VCHAR(1,1,48,24)
360 REM  *** GENERAZIONE LABIRINTO ***
370 Y=15
380 X=4
390 YV=15
400 XV=4
410 DIM A(4,6)
420 FOR R=1 TO 4
430 FOR C=1 TO 2
440 READ A(R,C)
450 NEXT C
460 NEXT R
470 DATA 2,0,-2,0,0,2,0,-2
480 RANDOMIZE
490 B=INT(4*RND)+1
500 CALL KEY(0,KEY,STATUS)
510 IF STATUS<>0 THEN 660
520 CALL GCHAR(Y+A(B,1),X+A(B,2),K)
530 IF K=90 THEN 490
540 Y=Y+A(B,1)
550 X=X+A(B,2)
560 IF K<>32 THEN 600
570 YV=Y
580 XV=X
590 GOTO 490
600 CALL HCHAR((Y+YV)/2,(X+XV)/2,32)
610 CALL HCHAR(Y,X,32)
620 YV=Y
630 XV=X
640 CLL=CLL+2
650 IF CLL<>460 THEN 490
660 CALL HCHAR(15,3,32)
670 X=30
680 Y=24
690 CALL GCHAR(Y,X,K)
700 IF K=32 THEN 730
710 Y=Y-1
720 GOTO 690
730 CALL HCHAR(Y,32,56)
740 CALL HCHAR(Y,31,32)
750 REM  *** TOPO ***
760 FOR R=1 TO 4
770 FOR C=1 TO 6
780 READ A(R,C)
790 NEXT C
800 NEXT R
810 DATA 1,0,4,1,2,3
820 DATA 0,1,1,2,3,4
830 DATA -1,0,2,3,4,1
840 DATA 0,-1,3,4,1,2
850 Y=15
860 X=3
870 CALL HCHAR(Y,X,42)
880 D=2
890 CALL GCHAR(Y+A(D,1),X+A(D,2),K)
900 IF K=32 THEN 950
910 IF K=80 THEN 970
920 C=C+1
930 D=A(B,C)
940 GOTO 890
950 N=1
960 GOTO 980
970 N=2
980 CALL SOUND(-50,700,0)
990 CALL HCHAR(Y+A(D,1),X+A(D,2),42)
1000 ON N GOTO 1010,1030
1010 CALL HCHAR(Y,X,80)
1020 GOTO 1040
1030 CALL HCHAR(Y,X,32)
1040 Y=Y+A(D,1)
1050 X=X+A(D,2)
1060 IF X=31 THEN 1110
1070 C=3
1080 B=D
1090 D=A(B,C)
1100 GOTO 890
1110 REM  *** RITORNO ***
1120 C=2
1130 CALL HCHAR(Y,32,90)
1140 CALL COLOR(2,2,12)
1150 CALL GCHAR(Y+A(D,1),X+A(D,2),K)
1160 IF K=80 THEN 1200
1170 C=C+1
1180 D=A(B,C)
1190 GOTO 1150
1200 CALL SOUND(-50,740,0)
1210 CALL HCHAR(Y+A(D,1),X+A(D,2),42)
1220 CALL HCHAR(Y,X,32)
1230 Y=Y+A(D,1)
1240 X=X+A(D,2)
1250 IF X=3 THEN 1300
1260 C=3
1270 B=D
1280 D=A(B,C)
1290 GOTO 1150
1300 GOTO 1300
```