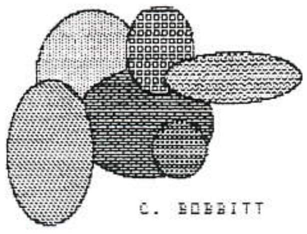
PROGRAM BITEN 93-4



RODGERS AND HAMMERSTEIN'S
SOUTH PACIFIC
Volume I
SELECTIONS FROM THE MUSICAL FOR THE TI-99 W/32K
A FAIRWARE PROGRAM BY KEN GILLILAND

"Some Enchanted Evening"

ISSN 0281-1146

KALLELSE ÅRSMÖTE

Härmed kallas till Årsmöte Lördagen 5 mars 1994 kl.13 hos Kent Edgardh, Albatrossvägen 46, Haninge:

1. Mötet öppnas

2. Val för mötet av ordförande, sekreterare och två justeringsmän

3. Mötets behörighet
   - utlysning
   - röstberättigade närvarande

4. Beslut om dagordning

5. Styrelsens årsredovisning för 1993. Verksamhetsberättelse och kassarapport delas ut på mötet.

6. Revisorernas berättelse för 1993

7. Om ansvarsfrihet för styrelsen för 1993

8. Val av styrelse för 1994. Årsmötet väljer enligt stadgarna en styrelse bestående av ordförande och kassör, vilka samtliga utses till befattning på årsmötet, samt därtill tre och högst sju övriga ledamöter.

9. Val av revisorer för 1994 (två revisorer och en suppleant).

10. Val av valberedning (två ledamöter varav en sammankallande).

11. Om årsavgift 1994.

12. Årsmötets avslutas. ■

VIDEO DIGITIZER

Niklas Johansson, Prästgårdsliden 10 A, 595 42 MJÖLBY frågar: Jag såg en ritning i tidningen Dator Magazin nr 17-93 där det fanns en ritning på en (videodigitizer+) sampler+midi. Ritningen visade dock bara sampler och midi delen. Digitizern och samplern kopplas till parallellporten på Amiga 500-4000. Videodigitizern och samplern använder samma A/D omvandlare och kontakt, medan midi kopplas till RS232-porten. Jag undrar nu om det skulle vara möjligt att använda denna digitizer tillsammans med en TI-99/4A eller Geneve med en adapter. Själv kan jag inte se något hinder, mer än det skulle vara strobe-signalen som skiljer TI och Amigan:s signaler. ■

Redaktör: Jan Alexandersson
Medlemsregister: Claes Schibler
Programbankir: Börje Häll

Föreningens adress:
Föreningen Programbiten
c/o Schibler
Wahlbergsgatan 9 NB
S-121 46 JOHANNESHOV, Sverige

Postgiro 19 83 00-6
Medlemsavgiften för 1992 är 120:-

Datainspektionens licens-nr 82100488

Annonser, insatta av enskild medlem (ej företag), som gäller försäljning av moduler eller andra tillbehör i enstaka exemplar är gratis.

Övriga annonser kostar 200 kr för hel sida. Föreningen förbehåller sig rätten att avböja annonser.

Föreningens tillbehörsförsäljning: Följande tillbehör finns att köpa genom att motsvarande belopp insätts på postgiro 19 83 00-6 (porto ingår)

Användartips med Mini Memory	20:-
Nittinian T-tröja	40:-
99er mag. 12/82, 1-5,7-9/83(st)	40:-
Nittinian årgång 1983	50:-
Programbiten 84-89 (per årgång)	50:-
90-91 (per årgång)	80:-
TI-Forth manual	100:-
Hel diskett ur programbanken(st)	30:-

Enstaka program 5:- st + startkostnad 15 kr per skiva eller kassett (1 program=20kr, 3 program=30 kr). Se listor i PB89-3 och PB90-4.

Artiklar sändes till redaktören:
Jan Alexandersson
Springarvägen 5, 3tr
142 61 TRÅNGSUND
Tel. 08-771 0569
(Ring eller skriv till mig om du har frågor om program eller hårdvara)

TECKEN 128-254 MED NEC PINWRITER P6

Ç	É	á	░	└	╨	α	≡
ü	æ	í	▒	┴	╤	β	±
é	Æ	ó	▓	┬	╥	Γ	≥
â	ô	ú	│	├	╙	Π	≤
ä	ö	ñ	┤	─	╘	Σ	⌠
à	ò	Ñ	╡	┼	╒	σ	⌡
å	û	ª	╢	╞	╓	µ	÷
ç	ù	º	╖	╟	╫	τ	≈
ê	ÿ	¿	╕	╚	╪	Φ	°
ë	Ö	⌐	╣	╔	┘	Θ	∙
è	Ü	¬	║	╩	┌	Ω	·
ï	¢	½	╗	╦	▒	δ	√
î	£	¼	╝	╠	▄	∞	^{n}
ì	¥	¡	╜	═	▌	φ	^{2}
Ä	₧	«	╛	╬	▐	ε	■
Å	ƒ	»	┐	╧	▀	∩	

SKRIV CHR 128-254

(Skriver ut ovanstående tabell)

```
100 OPEN #1:"DSK2.ASCII"
110 FOR I=128 TO 143
120 FOR J=0 TO 112 STEP 16
130 PRINT #1:CHR$(I+J);"    ";
140 NEXT J
150 PRINT #1: : :
160 NEXT I
170 CLOSE #1
```

LADDA UTAN FW TAB

Tony McGovern tipsar att du kan undvika att ladda TAB-inställningen med en DIS/VAR 80 fil till TI-Writer (eller äldre FW) genom att Load File av 0 DSK1.FILNAMN till en tom skärm. Du lägger således till en hel fil till en tom buffert och behöver inte veta längden på filen.

TECKEN FRÅN OLIKA SKRIVARE

av Jan Alexandersson

SJU OCH ÅTTA BITARS SKRIVARE

En skrivare kan använda 7 eller 8 bitar av varje byte som skickas från RS232/PIO-kortet. En äkta 7-bitars skrivare kommer inte att bry sig om den mest signifikanta biten så att 128 subtraheras från alla tecken större än 127. Både tecken 65 och 193 kommer således att skrivas som stort A med en 7-bitars skrivare. Min egen 8-bitars NEC Pinwriter P6 kan ställas in som 7-bitars skrivare med kontrollkoden FS I (0) och kopplas tillbaka till 8-bitars skrivare med FS I (1). FS I (0) skrivs som CHR$(28);"I";CHR$(0) i Basic.

Det finns även skrivare som har en blandning av 7 och 8 bitar så att tecken 0-31 och tecken 128-159 blir identiska medan tecken större än 159 skiljer sig från motsvarande lägre ASCII-koder. Du kan således sätta eller inte sätta den mest signifikanta biten och ändå få samma resultat. Alla andra tecken skiljer sig åt beroende på den mest signifikanta biten. Jag har sett denna 7/8 bitars blandning hos Epson FX 85/185 (kallas IBM Standard Character Set), Star SG10 (kallas IBM #1) och Gemini 10X (tecken 160-254 är unika och liknar inte IBM Set).

IBM TECKENSET

Det kan finnas vissa skillnader mellan skrivare trots att de använder IBM teckenset. Detta betyder att du kan få problem om du skriver ut texten på din egen skrivare och sedan sänder filen till en annan användare som sedan skriver ut filen på en annan skrivare. I de flesta fall så framgår det av sammanhanget vilket tecken som avsågs men se upp med några av de omtvistade tecknen. Jag har jämfört utskrifter från följande källor:
- NEC Pinwriter P6 (min skrivare)
- Panasonic KX-P1081(N.Johansson)
- Epson FX 85/185 (Charles Good)
- Star SG10 (Charles Good)
- Jet Data-Handbok

TECKEN 0-31 OCH 128-159

Epson FX 85/185 kan välja två olika set: IBM Standard Character Set respektive IBM Alternate Character Set. Den förra skriver inga synliga tecken med 0-31 eller 128-159 så att tecken 128-159 betraktas som samma kontrolltecken som 0-31. Star SG10 kallar detta Standard IBM #1 respektive Alternate IBM #2. Du bör alltid använda Alternate Set eftersom det bäst motsvarar Funnelweb All Char Set.

GRAFISKA TECKEN MED DUBBLA LINJER

Många men inte alla grafiska tecken 181-216 ska ha dubbla linjer. Tecken 186 (║) ska vara vertikal dubbel linje och tecken 205 (═) ska vara horisontell dubbel linje. Min egen NEC Pinwriter P6 och Jet Data gör detta riktigt medan andra skrivarna skriver ut enkla linjer: Epson FX 85/185, Star SG10, Panasonic KX-P1081.

TECKEN 159

Epson FX 85/185 lämnar tecken 159 blank. Andra skrivare kommer att skriva ut detta som ett långt lutande f (*ƒ*): NEC Pinwriter P6, Star SG10, Jet Data och Panasonic KX-P1081. Jag är osäker på vad tecken 159 ska användas till. Är det en matematisk funktion eller symbol för någon valuta (florin eller forint)? Kan någon förklara detta för mig?

TECKEN 176-178

Min tolkning är att alla dessa tecken ska representera olika nyanser av grått: 176 (░) ljust, 177 (▒) medium, 178 (▓) mörkt. Ingen av dessa ska vara helt svart. Jet Data har tecken 178 och 219 lika och helt svarta. Det kanske är storleken som skiljer istället för färgen eller att trycket blev för mörkt så att skillnaden inte syns. Min egen NEC

Pinwriter P6 simulerar detta med korta linjer där längden ökar från 176 till 178. Andra skrivare har punkter som kommer närmare varandra från 176 till 178. Jag tror inte att dessa skillnader är något problem vid betraktandet av utskriften. Kom gärna med synpunkter på detta om jag missuppfattat något.

TECKEN 219

Min NEC Pinwriter P6 skriver ut tecken 219 (▒) som skuggad med små punkter. Jag utgår från att detta är fel. Alla andra skrivare har detta fullständigt svart. Tecken 219 kan användas tillsammans med tecken 220 och 223 för att åstadkomma en tjock ramlinje men detta är inte möjligt med min skrivare. Jag var tvungen att istället använda tecken 221 och 222 som ramlinje på framsidan av Programbiten.

TECKEN 227

Tecken 227 ska vara den grekiska bokstaven lilla pi (π) men min NEC Pinwriter P6 skriver ut den som stora pi. Alla andra skrivare gör detta rätt.

TECKEN 231

Tecken 231 ska vara grekiska lilla tau (τ). Detta gäller för NEC Pinwriter P6, Epson FX85/185, Star SG10, Jet Data. Däremot kommer Panasonic KX-P1081 att skriva ut detta som lilla gamma vilket är fel. Några av de andra skrivarna har tau utan böjen nere vid foten så att den blir ganska lik gamma.

TECKEN 237

Tecken 237 ser ut som grekiska lilla phi (ø) men jag är inte säker på detta. De grekiska tecknen finns på 224-235 medan tecken 236-239 verkar vara avsedda för mängdlära. Tecken 237 skulle kunna ha med mängdlära eller diameter att göra. Finns det någon som vet säkert? Utskriften från Star SG10 liknar mer diameter än phi eftersom den har ett stort O med /. Panasonic KX-P1081 är mycket lik phi med en bred låg oval med /. Jet Data har en vertikal linje genom O. Övriga skrivare har något mitt emellan dessa extremfall (små runda o). Nationalencyklopedin vissa två olika versioner av lilla phi men båda har vertikal linje och inte något lutande /.

TECKEN 238

Tecken 238 liknar grekiska lilla epsilon (ε) på min NEC Pinwriter P6 och på Panasonic KX-P1081. Alla andra skrivare (Epson FX85/185, Star SG10, Jet Data) skriver ut detta tecken som tillhörande i mängdlära. Jag tror att mängdlära är den riktiga användningen av tecken 238 men jag har själv större användning för epsilon. Tecken 239 används också för mängdlära: skärningsmängd. Nationalencyklopedin anger två olika versioner av epsilon: en bakvänd 3:a respektive ett C med ett - på mitten.

TECKEN 240

Tecken 240 är det matematiska tecknet för identisk (≡). Alla skrivare gör detta riktigt. Min NEC Pinwriter P6 gör detta riktigt i alla normala moder men kommer att skriva grekiska stora xi med superscript/subscript. Xi har det mellersta strecket kortare än de två andra linjerna. Detta är inte riktigt.

TECKEN 252

Tecken 252 är grekiska lilla eta (η) på min NEC Pinwriter P6 och Jet Data. Alla andra skrivare (Epson FX 85/185, Star SG10, Panasonic KX-P1081) har lilla n som exponent. Tecken 251-253 är grupperade tillsammans och bör således ha något gemensamt: roten ur och exponent. Jag tror att n som exponent är den riktiga användningen men jag har större nytta av den grekiska eta eftersom exponent kan ordnas med superscript.

NATIONELLA TECKENSET

Alla nationella teckenset kan väljas med ESC R (n) på min NEC Pinwriter P6 där n bestämmer språket (CHR$(27);"R";CHR$(n) i Basic):

n	CTRL-U	SPRÅK
0	SHIFT-@	USA
1	SHIFT-A	Frankrike
2	SHIFT-B	Tyskland
3	SHIFT-C	England
4	SHIFT-D	Danmark I
5	SHIFT-E	Sverige
6	SHIFT-F	Italien
7	SHIFT-G	Spanien
8	SHIFT-H	Japan
9	SHIFT-I	Norge
10	SHIFT-J	Danmark II
11	SHIFT-K	Nederländerna

Endast vissa av dessa finns tillgängliga på skrivarna:

NEC Pinwriter P6	n = 0-11
Epson FX 80	n = 0- 8
Epson FX 85/185	n = 0-10
Star SG10	n = 0- 7
Star NL-10	n = 0-10
Gemini 10X	n = 0- 7
(men i annan ordning)	
Seikosha GP-550A	n = 0- 7
(men i annan ordning)	
Panasonic KX-P1081	n = 0-10

NEC PINWRITER P6

Min skrivare fungerar inte som det står i manualen!!! Danmark I och Norge skrivs ut som USA medan Danmark II skrivs ut som Danmark I. Italien tecken 126 skrivs ut som upp och nedvänt ! istället för i med grav accent. Jag har inte sett Nederländerna på någon annan skrivare så jag är osäker på tecken 35, ska det vara pund eller något annat. Den holländska valutan NLG kallas florin på svenska. Kan detta vara samma tecken som IBM-tecken 159?

STAR SG10

Jag har endast en typografiskt satt kopia av manualen så den verkliga utskriften kan skilja sig från detta och mina synpunkter kanske inte är relevanta. Frankrike 35 är pund istället för #. Danmark 92 är grekiska stora phi istället för danskt Ø med O och /. Sverige 36 fattas men jag gillar ändå inte detta svenska tecken (vem har hittat på något så dumt). Italien 64 är paragraf istället för @ och Italien 92 är c med , istället för bakåtlutande snedstreck. Spanien 35 är # istället för peseta. Däremot skriver Star NL-10 ut allt på ett riktigt sätt.

GEMINI 10X

Jag har endast en typografiskt satt kopia av manualen så den verkliga utskriften kan skilja sig från detta. Frankrike 35 är pund istället för #. Danmark 92 är grekiska stora phi istället för danskt Ø med O och /. Danmark 124 är grekiska lilla phi istället för lilla danska ö med o och /. Sverige 36 saknas och Sverige 123 måste vara ett skrivfel eftersom två punkter saknas över a. Italien 64 är paragraf istället för @ och Italien 92 är c med , istället för bakåtlutande snedstreck. Spanien 35 är # istället för peseta. Star SG10 och Gemini 10X ser ut att vara ganska lika. ■

SÄLJES TI-99/4A

SÄLJES TI 99/4A med "allt". Texasdator, speech, exp box innehållande: RS232, 32k minne, p-kods kort, Corcomp 512 KB RAM-disk, Corcomp disk controller och 2 st DS/DD halvhöjds diskettenheter. Dessutom: RGB-modulator, joysticks, X-Basic, 2 st externa DS/DD diskettenheter (med datakabel till box, men utan strömförsörjning), skrivare Star SG-10, Maximem med drygt 50 moduler på disk (Multiplan, TI-Writer m.fl.). Massor med manualer och cirka 100 disketter med program och dokumentation (PB-Forth, TI-Forth, UCSD-Pascal, Infocom äventyrsspel m.fl.)

Säljes endast komplett. Hämtpris minst 3000 kr eller högstbjudande. Lennart Thelander, Veckogatan 114, 256 64 HELSINGBORG. Telefon kvällar och helger 042-20 07 21. ■

Du kan fortfarande beställa äldre årgångar av Programbiten.

SWEDLOW TI BITS * 30-31 *

by Jim Swedlow, USA

(This article originally appeared in the User Group of Orange County, California ROM)

WORLD WIDE TOP TEN

Personal Computing recently listed the top ten computers in world wide sales from 1978 to 1988. Guess what?

Commodore 64	7,280,000
IBM PC/XT	4,577,000
Apple II family	4,487,000
Sharp 12/13/15/16 series	4,055,000
Commodore C128	4,003,000
Commodore Vic 20	2,246,000
Apple Mac Family	2,063,000
TI 99 4A	2,053,000
Sinclair ZX 80/81	1,790,000
Tandy TRS 80	1,754,000

BTW, the top five dot matrix printers (1988 sales) were:

Apple ImageWriter II	97,300
Epson LX 800	48,650
Panasonic 1080i	48,650
Star NX 1000	41,700
Panasonic 1090i	41,700

TURBO COPY

If you have a TI disk controller, this is a must have program. It is the fastest track copier available. It does not, however, work with Myarc or CorComp disk controller cards.

Turbo Copy has a number of strengths. It formats as it copies, cutting total time drastically. Also, it can make two copies from one original (if you have a three drive system). Here's how you do it:

- Put your master in drive 1 and blank disks in drives 2 and 3.

- From the main Turbo menu, press 1 for Copies and then press the space bar.

- Press ENTER to accept 1 as the MASTER DRIVE.

- Change the COPY DRIVE to t (lower or upper case is OK).

- Next press ENTER twice more. Do not worry that the FORMAT drive is only 2, Turbo Copy will format both target disks. Also, leave FORMAT as Y (for yes).

You can now start copying by pressing FCTN 5 (or BEGIN). If you just want to copy from drive one to drive two, press 1 for copies and then FCTN 5.

It is always a good idea to write protect your source or master disk before copying. Nothing should happen but it is inexpensive insurance (for example, some people we won't name have put the source disk in the target drive and destroyed it -- not me, of course!!).

The only known bug in Turbo copy is that it incorrectly formats double sided disks. It sets a bite on sector zero incorrectly so that the disk looks like it is single sided when DM1000 or other programs copy it.

Don't use Turbo Copy to format disks, but DO use it to copy them.

TI LIVES

Well here we are on the verge of a the 90's. Not a new decade - that does not start until 1991 just as the twenty first century does not start until 2001 (and you wondered why the Arthur C. Clark used that year for his classic movie).

Anyway, who would of thought? TI abandoned us years ago and we are still viable. New and better software and hardware continues to be available. No one is getting rich supporting the 4A and we loose vendors from time to time, but the core is still there. Amazing.

Here are my suggestions for keeping the 4A alive:

- Support your user group. If you are a member, come to meetings and participate. If you are not, find one and join, even if only by mail. Perhaps you could do a demo at a meeting or write something for the newsletter. Do something because the user groups are the life support system of any computer.

- Support 4A vendors. If you want to make sure that someone provides software and hardware for the 4A, buy from them.

- Support freeware authors. They are still producing some of the best software out there (BOOT, Funnelweb, etc.). But unless you let them know (in words and dollars) that you appreciate their work, they stop writing new stuff.

- Help someone who has a 4A. There are tens of thousands (hundred of thousands?) of 4A's in closets and garages. Helping someone use their computer helps the 4A community survive.

FUNNELWEB 4.30 RELEASED

Funnelweb version 4.30 is out. The support for 80 column display is vastly improved. For the rest of us, the changes are minor.

First, you can configure Funnelweb to have one of three functions appear first when you load with Editor Assembler: Funnelweb, your User List or Disk Review.

Second, the name of the file for the Disk Utilities User List has been changed from DS to D1. This is because the 80 column version of Disk Review now requires two files, DR and DS.

SYSCON files are fully compatible between versions 4.30 and 4.2x.

Upgrade is recommended if you have an 80 column card or if you are using version 4.1x or lower. Otherwise, wait for version 4.31!

MULTIPLE PRINTERS AND COMPUTERS

Over time you tend to collect lots of stuff. I have two operating computers and four printers. I wanted to set up two printers -- one for regular paper and one for labels (I do lots of labels).

The first thing I found was a ABab switch box. It works something like this:

```
 __________      __________
|          |    |          |
| Computer |    | Computer |
|    A     |    |    B     |
|__________|    |__________|
        \            /
         \          /
          \______  /
          | ABab |
          | Box  |
          |______|
         /        \
        /          \
       /            \
  _________      _________
 |         |    |         |
 | Printer |    | Printer |
 |    a    |    |    b    |
 |_________|    |_________|
```

The box has four standard, 36 wire centronics plugs on the back and a switch with two positions on the front. In one position (AaBb) computer A is attached to printer a and computer B to printer b. In the other (AbBa) computer A is attached to printer b and computer B to printer a. I found a box for under $25.

You have to buy two more cables with centronics connectors on both ends. Luckily, these can be found for under $10 each if you shop around.

No problem, or so I thought. I hooked everything up and it didn't work. The problem came because standards change. If you look in your RS232 book, it tells you to use pin 16 on the centronics side for the ground. All of my printers accept this as a ground. Unfortunately the cables are not 36 wire -- they just use the common wires and 16 is no longer used as a ground in the IBM world.

Out came my handy soldering iron. I switched a wire from a pin that the TI does not use to pin 16 on both ends. Now everything works like a dream.

It is a joy not to have to unload paper and load labels and then reverse the procedure. Net cost was about $35.

TI IN ENGLAND

Received a letter from Stephen Shaw, who is the Disk Librarian and VP of the TI 99/4A User Group in the United Kingdom (TIUGUK). He had some interesting things to say:

"TIUGUK is now into its 8th year. Back in 83, JUST before the plug was pulled, TI insisted that the amateur TI HOME group should pass to a commercial organization. Membership was then 4,000 and rising and getting a little heavy to handle on a non commercial basis. Then TI pulled the plug and there was no user group. And we could not get the mailing list. So we managed perhaps 400 members by the end of 83. A small local group in Brighton became the default national group and started from there, presently declined to about 140 odd, which is just about sustainable."

"[Our] annual meet [was] held at Chester Northgate Arena on Saturday, May 26th, 1990. The annual meet again confirmed the strong support of the 4A in the UK by a small handful of users."

"With a membership of between 140 and 170, scattered over the UK and with several living abroad, the meet was visited by maybe 50 odd members, who came together to elect group officials for the year and to see the latest in software and hardware."

"One unexpanded owner went away the proud owner of a mini memory, a stand alone 32k ram, a full Editor Assembler pack (for only five pounds) and the rare Miner 2049er sideways module. Your scribe picked up a Munchman II module to add to the collection. There was even the excellent SuperSketch peripheral on sale! And lots of spare bits and cables and so on."

"One display item was a photograph of young George Shaw (present with his favorite teddy bear Matthew) measuring two inches by two inches, and two printouts produced on a 4A, based on the photos, one measuring 7/8" by 5/8" and the other 8" by 8". These represent an interesting beta test of a possible new service for TI owners which may be offered by a famous US based owner (who at present wishes anonymity) - the photograph was scanned on a PC and cleaned up using Paintbrush (in particular the background was removed). Then the PC picture was translated to MacPaint format and transferred to the TI via RS232. From here it could be printed with MacFlix or PixPro and in fact was translated from MacPaint format to TI Artist format using PixPro."

"The TI Artist pictures was then printed as the 8" x 8" pictures using SmArtcopy by Alexander Hulpke. The smaller picture was made by using SQUEEZER to reduce the TI Artist picture to quarter size. SQUEEZER provides a choice of four densities and is the only really usable reducer for pictures. The small TI Artist pic was printed using Artist Photographic Vn 2, supplied with Harry Brashers Home Publishing on the 4A, Supplement #3. It is to be noted that while the TI programs could reproduce the scanned photo in the correct aspect ratios, the C could not -- the two computers being used together was a result better than either could do on their own!"

"Our newsletter is 60 pages or longer. On average each issues of TI*MES occupies about 400 sectors, unarchived."

"I continue to spend my computing time writing, trying to establish (and maintain!) order in a very large disk collection and playing with fractal graphics (very time consuming). My chosen language now is THE MISSING LINK. Fractals are very slow of course and I do look forward to seeing the Graphics extension to Turbo Pasc 99 should they ever come out. However, unlike TML, they will not have TI Artist file computability built in."

"Still get lots of pleasure from the machine and have very many more years of exploration to do, at least until the day comes when it breaks down irreparably!"

Best wishes, Stephen Shaw

Enjoy! ■

RELISTING PROGRAMS - 2

by Jim Peterson, Tigercub, USA

DV80 TO PROGRAM CONVERSION

John "Jeb" Hamilton of the Central Iowa User Group was the first to realize, several years ago, that a DV80 listing of a Basic or XBasic program could be converted to a DV163 file and then merged in and run as a program. I no longer have his program in my library, but this is a quick and dirty version of it -

```
100 DISPLAY AT(12,1)ERASE AL
L:"Input file? DSK":"":"Outp
ut file? DSK"
110 ACCEPT AT(12,16):A$ :: A
CCEPT AT(14,17):B$
120 OPEN #1:"DSK"&A$,INPUT :
: OPEN #2:"DSK"&B$,VARIABLE
163,OUTPUT :: LINPUT #1:M$
130 LINPUT #1:M$ :: IF LEN(M
$)>78 AND EOF(1)<>1 THEN LIN
PUT #1:M2$ :: M$=M$&M2$
140 X=POS(M$," ",1):: Y=VAL(
SEG$(M$,1,X-1))
150 PRINT #2:CHR$(INT(Y/256)
)&CHR$(Y-256*INT(Y/256))&"!"
&SEG$(M$,X+1,255)&CHR$(0)
160 IF EOF(1)<>1 THEN 130 EL
SE CLOSE #1 :: PRINT #2:CHR$
(255)&CHR$(255):: CLOSE #2
```

To try that out, key in this useless little program -

```
100 CALL CLEAR
110 FOR J=1 TO 10
120 PRINT J
130 NEXT J
140 END
```

List that to disk by LIST "DSK1.80". Then run the above converter program, answer the input prompt with 1.80 and the output prompt with 1.163. After it runs, enter NEW, then MERGE DSK1.163 and then LIST. This is what you should see

```
100 !CALL CLEAR
110 !FOR J=1 TO 10
120 !PRINT J
130 !NEXT J
140 !END
```

Type 100 and FCTN X to bring line 100 to the screen with the cursor on the "!". Type FCTN 1 to delete the "!" and repeat with FCTN X and FCTN 1 to delete all the others. Then enter RUN and it should do so!

All that the program does is delete the blank first line of the listing, convert each program line number to tokenized format, add a CHR$(0) end-of-line marker to each line, move the record to a DV163 file, and add the double CHR$(255) end-of-file marker.

The result is a merge format program composed of REM statements; when you delete the "!" REM indicator, these become program lines.

There is just one problem. A LISTed program is a DV80 file, consisting of records of 80 characters or less, but a program line in XBasic can be keyed in up to 140 characters long, and can be forced even longer (as I often do!) When such a line is LISTED, it is broken into 80-character records, which confuses the conversion program completely.

Line 130 of the conversion program attempts to resolve that problem. If a record is more than 78 characters long (because it could have been an 80-charcter line ending in a blank, which would become a 79-character record without the blank) it is taken to be most probably the first part of a long program line; another record is read in and tacked onto it.

However, this creates another problem, as you will find if you LIST the converter program and then try to convert it back to a program - line 140 will be tacked onto line 130 because line 130 is 79 characters long.

The best fix for this is to load the DV80 file into Funlweb and print out a hard copy; use a ruler to draw a vertical line after the 78th characters; mark any program line that ends on the 79th or 80th characters, delete those characters, save the listing, run it through the converter, merge it in and key those deleted characters back in - still much easier than keying in an entire listing.

After John Hamilton published his discovery, several authors wrote their own versions. It was suggested that programs could be written in text format, using the superior editing features of TI-Writer or Editor Assembler, and then converted to program format. Personally I was satisfied with the editing features of Basic and was not about to give up its syntax error-catching capability, so I never tried this method.

However, nowadays several hundred text files of newsletter articles are available on the Clearing House BBS, and other newsletters are being circulated on disk. Many of these articles contain program listings, and it would be much easier to extract and convert them than to print them out and key them in.

Later on, John Ford wrote a more complex converter called XLATE, which eliminates the need to delete all the "!" by converting the ASCII text file directly to tokenized merge format. It also checks for syntax errors and corrects them or reports them on-screen . If the LISTed program had regularly sequenced line numbers, it will check these to see whether records should be combined, which should greatly improve accuracy - I have not tested it enough to say how foolproof it is.

Blanks at the end of a DV record are dropped, so if the 80th character of a long program line is a blank, when the line is broken into two records and then recombined the blank will be missing. For instance, if the blank between FOR and J in FOR J=1 TO 10 happens to be the 80th character, it will recombine as FORJ=1 TO 10. This results in a SYNTAX ERROR referencing the line number, which is therefore easy to spot and correct. The same problem can cause the string "John Doe" to become "JohnDoe".

The above conversion programs are intended for listings in 80-column format. However, many of the listings within text articles have been reformatted to 28-column or 40-column width, or listed in those widths with Triton Super Extended Basic.

Fortunately, there is an alternative. Curtis Alan Provance has written a truly remarkable program in assembly, called TEXTLOADER, which will convert a DV80 file directly into a program in memory, and will handle the shorter line lengths, although with increased chance of error because the method of detecting new lines is far from foolproof.

I have not tested this program extensively, but have found only two major problems. The one is with records ending in a dropped blank, as described above; these are easy to correct. The other is with split referenced line numbers. For instance, if a line ends in GOSUB 120 :: GOTO 200 and splitting of records turns this into GOSUB 1 and 20 GOTO 200, you will find the line ending with GOSUB 1 and a new line 20 :: GOTO 200 at the beginning of the program. Comparison with the original listing makes this easy to correct.

TEXTLOADER loads into memory and remains there, so that you can load other text files by simply typing - CALL LINK("OLD"<"DSKn.filename"). The file loads and converts rapidly, displaying each line as it does so. Sometimes a line which has been corrupted will be reported as a syntax error and omitted, but sometimes it will be omitted without being reported, and sometimes it will not be detected until you try to run the program. Occasionally, especially when working with 28-character lines, you may get all sorts of invalid error messages. Apparently the program in memory differs from the screen display, and it may be impossible to debug in such cases.

Other features allow you to merge a converted text file into a programn in memory rather than overwriting it, and to read and run a batch file of command type instructions, such as -

```
CALL FILES(1)
NEW
RUN "DSKn.bigprogram"
```

An improved general-purpose memory image program loader is also included.

XLATE is a public domain program, available on my TI-PD disk #1083. TEXT- LOADER is a fairware program available on my TI-PD disk #1104. (XLATE se programlistning i detta numer av PB; TEXTLOAD finns på skiva i programbanken) ■

ARCHIVING – A COMPRESSION

by Andy Frueh, Lima Ohio UG, USA

A lot of people are puzzled by archiving and how to use Barry Boone's Archiver. What follows is both a reference guide and explanation of Archiver III. It is not meant to totally replace the documentation for this program. Actually, I haven't seen a distribution copy that comes with a set of instructions. There may be hidden features of ArcIII that aren't obvious to me (for example, Disk Utilities by John Birdwell has a feature to figure decimal-to-hex conversions).

What exactly is archiving? Putting it simply, when you archive you take a file or a set of files, and group them as one file then compress them so they take up less disk space. Some software comes archived. These ALMOST always include the archiving program. Examples are Jack Sughrue's PLUS! and the Complete Adventure disk set.

What is the purpose of archiving? Well it started out as a money saver for modem users. It is faster, and thus cheaper, to send 90 archived sectors as 1 file, than 120 sectors for 3 programs. Now it is also a means of backing up disks. You can save each of your disks as a one file, squashed archive. You can specify whether you want compressed files or not. The reason you have a choice is that some unusual files actually take up more space when they are compressed. Another useful application of archiving is when you have programs you want to keep, but don't need ready to use. You can keep archives of all these files instead of taking up disk space.

OK, now that you have the "what", here's the "how". As far as I know, the only archiver is Barry Boone's program. Its operation is completely different from Archiver II. Rather than add new features to past versions, Archiver was completely re-written. It usually contains an XB LOAD program, but may be loaded from E/A. The program's filename is usually ARC1. It can be found on almost all of the bulletin boards, as a commercial version with Geneve utilities, in user group libraries, with other Fairware programs or from the author. Chances are, you can definately get a copy.

First things first, so get the program loaded. After that, you should see a Fairware notice. Press any key to pass this. You then see a menu. Each menu option is described in detail below.

1) Archive Files - These options are largely self-explanatory. As you may have guessed, this option archives files. Pressing one will deliver a set of prompts. These are "Source Drive (1-Z)". Yes, you can have drive numbered from 1-9 and

A-Z. Then comes, "Output Drive (1-Z)". You may use one drive. Archiver will prompt you to change disks when needed. It is highly recommended that you use a blank output disk, since archives may fill or almost fill a disk. Next comes "Output Filename". This is usually the name of the disk you are archiving, or some related heading. For example, a set of D/V 80 articles may be named "ARTICLES". The following prompt is "Pack all Files? (Y/N)". If you answer "Y" then al the files on the source disk are archived. If you answer "N", then when Archiver is working, you are asked "Include filename? (Y/N)" If you answer "Y" then that file is archived, otherwise it is ignored. This is a handy feature if you have programs and files for example, and need them seperated. This process repeats for each of the files on the source disk. The final prompt is "Compress? (Y/N)". Saying "Y" and Archiver attempts to squash each file so it takes up less space. Remember that some unusual file types will actually get LARGER if compression is attempted. When all the prompts are answered, press REDO to correct an error in your answers, BACK to return to the menu, or any other key to continue. When Archiver is done performing any operation, pressing a key goes back to the main menu.

2) Extract Files - This is the opposite of archiving. It will let you pull (extract) files from an ARC file. You are first asked for the source drive. Next you input the source filename. After that, you are asked for the output drive. It must be stressed that the output drive for ALL operations of Archiver should be different than the input drive. You may run out of space or overwrite a file accidently. Output disks should be blank.

The next prompt asks, "Extract all files?" If you answer "Y" then every file stored in the ARC file will be taken out. If you answer "N" then when extracting starts, the program asks, "Include filename?" for every seperate file in the archive. Again, press REDO (to restart this option), BACK (returns to main menu), or any other key to continue.

3) Catalog Disk - This is fairly self explanatory. Simply input the source drive name. The program will ask if you want a printout. If you answer yes, then you are asked for the printer name. If there are more files than can be displayed, then [more] is printed on the screen and pressing a key advances the screen.

4) Catalog ARC File - If you aren't sure what files are contained in an archive file, than this option tells you. You are asked for the source drive, source filename, and whether or not you want a printout of the list of files.

5) File Copy - This option will copy a file (obviously). Simply supply the source drive and filename, and the output drive and filename.

6) File Rename - Again, this option should explain itself. Give the source drive and filename, then the output filename.

7) File Delete - Supply the source drive and filename.

8) File Un/Protect - You first supply the source drive and file-name. You are then asked "Protect?" If you answer "Y" the file is protected. Otherwise, file protection is lifted.

9) List Text File - This will display or print a D/V 80 file. Give the source drive and filename. You are then asked if you want the file printed or not.

10) Load FW - This returns to Funnelweb. Simply give the drive number on which the UTIL1 file is located.

NOTE: When an I/O error occurs, pressing a key returns to the main menu. If you have a Geneve, this is for you. Using a sector editor, find the string 04E08C00 and replace it with D8018C00.

Remember that it is fairware, so if you find it very useful, send the author (Barry Boone) a donation.

■

FROM BASIC TO ASSEMBLY - 5

by Bob August, Bug News, USA

This month we are going to do some graphics in assembly. Why do graphics? Because you need to if you are going to use windows or program games or just to spice up your program. When you do graphics you define your character the same way you do it in Basic or Extended Basic. But instead of CALL CHAR you use a label and data with the hex > in front of the numbers. You then assign the data to a character like 1 which is ASCII 49 or >31. This looks like
LTOP DATA >387C,>FEFF,>FF7F,>3F1F
in our program is the same as the Basic CALL CHAR(49,"387CFEFFFF7F3F IF"). The program in Extended Basic is:

```
100 ! Lesson Number 5
110 CALL CHAR(49,"387CFEFFFF
7F3F1F")
120 CALL CHAR(50,"387CFEFEFE
FCF8F0")
130 CALL CHAR(51,"0F07030100
000000")
140 CALL CHAR(52,"E0C0800000
000000")
150 GOSUB 240
160 CALL HCHAR(10,15,49)
170 CALL HCHAR(10,16,50)
180 CALL HCHAR(11,15,51)
190 CALL HCHAR(11,16,52)
200 GOSUB 260
210 CALL KEY(0,K,S):: IF S=0
 THEN 210
220 IF K<>13 THEN 210
230 STOP
240 CALL CLEAR
250 RETURN
260 DISPLAY AT(20,4):"PRESS
THE ENTER KEY TO QUIT"
270 RETURN
280 END
```

Both the above program and the assembly program will display a heart in the center of the screen. To do this in assembly we need to take the character we intend to use and multipy the ASCII number by 8. Add 2048 to the total and convert the answer to hex. We put this number into register zero, the label for our data in register one and an 8 into register two. You then write this to VDP ram. Next you put the screen location into register zero, the character number into register one and as register two already has an 8 we just write this as a single byte to the screen. Also for the second, third and forth characters we did not need to put the 8 into register two as it is already there.

Now enter you program and play with it. Also lets try to make windows. If you can't do it don't despair as we will show you how next month. (Lesson 6 see PB 91-5 p.14)

HAPPY ASSEMBLING!

```
***************************************
* BASIC TO ASSEMBLY  Lesson Number 5 *
***************************************
*
       DEF  START            Entry point of program
       REF  VSBW,VMBW,KSCAN  Utilities used in program
*
WRKSP  BSS  32               Workspace buffer
SAV11  BSS  2                Save return address buffer
*
LTOP   DATA >387C,>FEFF,>FF7F,>3F1F    Left top of heart
RTOP   DATA >387C,>FEFE,>FEFC,>F8F0    Right top of heart
LBOT   DATA >0F07,>0301,>0000,>0000    Left bottom of heart
RBOT   DATA >E0C0,>8000,>0000,>0000    Right bottom of heart
MSG1   TEXT 'PRESS THE ENTER TO QUIT'  Prompt to quit
*
       EVEN                Make sure we start on even byte
*
```

```
* Start of program
*
START  MOV  R11,@SAV11   Save return address
       LWPI WRKSP        Load the workspace
       BL   @CLEAR       GOSUB CLEAR to Clear the screen
*
* Put heart on screen
*
       LI   R0,>0988  ***Load pattern table address for a one
       LI   R1,LTOP      Load left top data
       LI   R2,8         load pattern descripter table
       BLWP @VMBW        Write it to VDP
       LI   R0,303       Load screen location (Row 10, Col. 14)
       LI   R1,>3100     Load ASCII 49 or a one
       BLWP @VSBW        Write it to the screen
*
       LI   R0,>0990  ***Load pattern table address for a two
       LI   R1,RTOP      Load right top data
       LI   R2,8       **Load pattern descripter table
       BLWP @VMBW        Write it to VDP
       LI   R0,304       Load screen location (Row 10, Col. 15)
       LI   R1,>3200     Load ASCII 50 or a two
       BLWP @VSBW        Write it to the screen
*
       LI   R0,>0998  ***Load pattern table address for a three
       LI   R1,LBOT      Load left bottom data
       LI   R2,8       **Load pattern descripter table
       BLWP @VMBW        write it to VDP
       LI   R0,335       Load screen location (Row 11, Col. 14)
       LI   R1,>3300     Load ASCII 51 or a three
       BLWP @VSBW        Write it to the screen
*
       LI   R0,>09A0  ***Load pattern table address for a four
       LI   R1,RBOT      Load right bottom data
       LI   R2,8       **Load pattern descripter table
       BLWP @VMBW        Write it to VDP
       LI   R0,336       Load screen location (Row 11, Col. 15)
       LI   R1,>3400     Load ASCII 52 or a four
       BLWP @VSBW        Write it to the screen
*
* Put message on screen
*
       BL   @DISPLY      Gosub to disp. message at Row 20, Col.4
       DATA 612,MSG1,24  Screen locat, Message,Length of message
*
* Call key routine
*
       CLR  @>8374       Clear to zero for CALL KEY(0,K,S)
       CLR  @>837C       Clear status to zero
       LI   R4,>2000                                (Ed.change)
KLOOP  BLWP @KSCAN       CALL KEY(0,K,S)
       CB   @>837C,R4    Check for key press        (Ed.change)
       JNE  KLOOP        IF S=0 THEN KLOOP          (Ed.change)
       MOV  @>8375,R0    Move Key press to register zero
       CI   R0,>0D       Compare to 13 or enter key
       JNE  KLOOP        If not enter key goto kloop
       CLR  @>837C       Clear status to zero
       MOV  @SAV11,R11   Put return address in register 11
       BLWP @0           Quit (FCTN =)
*
* Clear screen routine
*
```

```
CLEAR  LI   R1,>2000      Load Register one with space
       CLR  R0            Clear Regester zero to zero
CLOOP  BLWP @VSBW         Write blank space to screen
       INC  R0            Add one to register zero
       CI   R0,767        Compare contents to 767
       JLE  CLOOP         If less then 767 goto cloop
       RT                 Return to next line of calling area
*
* Display at routine
*
DISPLY MOV  *R11+,R0      Put screen location into Regester zero
       MOV  *R11+,R1      Put message into Regester one
       MOV  *R11+,R2      Put length into Regester two
       BLWP @VMBW         Write it to the screen
       RT                 Retrun to next line of calling area
*
* End program with auto start
*
       END  START
*
**  Actually not needed as R2 already contains an 8.
*
*** Formula to find address for pattern table
*
*   (49 * 8) + 2048 = 2440 = >0988
*   (50 * 8) + 2048 = 2448 = >0900
*   (51 * 8) + 2048 = 2456 = >0998
*   (52 * 8) + 2048 = 2464 = >09A0
```
■

PROGRAMS WRITE PROGRAMS -3

by Jim Peterson, Tigercub, USA

Let's start learning how to actually write a program that writes a program.

A MERGEd program is a D/V 163 file, so -

OPEN #1:"DSK1.(filename),VARIABLE 163,OUTPUT

Every program line begins with a line number, of course. In MERGE format the line number, whether 1 or 32767, is squished into two characters. We don't need to get into how this is done, but you can accomplish it with CHR$(INT(LN/256))&CHR$(LN-256*INT(LN/256)), where LN has ben predefined as the line number.

To print a statement or command, anything that is represented by a token in the token list, just print the CHR$ of its token ASCII. For instance, the token for DATA is 147, so you would print CHR$(147).

To print a variable name, either numeric or string, just enclose it in quotes, "A" or A$".

To print a value, or a string which is not in quotation marks (such as in a DATA statement), or the word which follows a CALL, you must print CHR$(200) followed by a token giving the number of characters to follow, such as CHR$(5) for a 5-letter word such as CLEAR, then the value in quotes. For instance, the token for CALL is 157, so CALL CLEAR is CHR$(157)&CHR$(200)&CHR$(5)&"CLEAR" .

Similarly, tokens for parentheses are 183 and 182, so the variable name A(1) is "A"&CHR$(183)&CHR$(200)&CHR$(1)&"1"&CHR$(182).

A quoted string is handled in the same way except that it is preceded by token 199, so PRINT "HELLO" is CHR$(156)&CHR$(199)&CHR$(5)&"HELLO". Don't worry about the quotation marks, the computer will handle that.

If you need to refer to a line number, as in GOTO 500, use token 201 followed by the line number formula, thus CHR$(134)&CHR$(201) &CHR$(INT(500/256))&CHR$(500-256* INT(500/256)).

Don't print more than 163 characters in a record. You can print multiple-statement XBasic lines, but be sure to use the double-colon token 130 as the separator, not two of the 181 colon tokens.

Each program line must end with CHR$(0) as the end-of-line indicator, and the last record you print must be CHR$(255)&CHR$ (255) as the end-of-file indicator.

If you get an I/O ERROR 25 when you try to merge your program, it means that you left off the final double-255. If the program merges, but crashes when you run it, you will probably be able to spot an obvious error in the line when you LIST it. If the line looks OK but gives you a DATA ERROR or SYNTAX ERROR, you left off a CHR$(0) or gave the wrong count of characters after token 199 or 200. The program published in Part 2 will help you to track down these bugs.

Now let's write a program. What is the longest possible one-liner program?

Well, RANDOMIZE is the longest statement that can stand alone. It is represented by the single token 149, and to repeat it must be followed by the double-colon token 130. Since any line number will take two bytes, let's use a 5-digit line number. And don't forget that final CHR$(0). That still leaves us 160 of the 163 bytes, so we can repeat tokens 149 and 130 for 79 times, followed by a final 149.

```
100 OPEN #1:"DSK1.LONG",VARI
ABLE 163,OUTPUT
110 FOR J=1 TO 79 :: M$=M$&C
HR$(149)&CHR$(130):: NEXT J
:: M$=CHR$(254)&CHR$(254)&M$
&CHR$(149)&CHR$(0):: PRINT #
1:M$ :: PRINT #1:CHR$(255)&C
HR$(255)
120 CLOSE #1
```

RUN, NEW, MERGE DSK1.LONG and LIST - over 34 lines long! But that one-liner doesn't do anything, so try this one -

```
100 OPEN #1:"DSK1.LONG",VARI
ABLE 163,OUTPUT
110 FOR J=1 TO 52 :: M$=M$&C
HR$(162)&"X"&CHR$(130):: NEX
T J :: M$=CHR$(254)&CHR$(254
)&M$&CHR$(162)&"X"&CHR$(0)::
 PRINT #1:M$
120 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1
```

Again RUN, enter NEW, then MERGE DSK1.LONG, then RUN. You'll get a message BREAKPOINT IN 32510 (don't ask me why!) but just enter RUN again.

Next month - using DEF to make it all easier. ■

TIME CALCULATOR

The Harrison Time Calculator is an Extended Basic program with built-in Assembly enhancement. Its purpose is to handle calculating numbers in Hours, Minutes, and Seconds. The time inputs may be made in either the "normal" 12 hour clock format or in the "military" 24 hour format. The six selections are:

1. ELAPSED TIME
2. CUMULATIVE SUM
3. TIME MULTIPLY
4. TIME DIVIDE
5. SET 12 OR 24
6. EXIT PROGRAM

The first selection, Elapsed Time, is for cases like "how much time is it from 10:22:35 to 3:30:46?"

Item 2 on the menu is for those cases such as trying to copy records or CDs onto cassette tapes.

Item 3 on the menu is designed for the old "cook-book" problem, where you find that the recipe says to roast the lamb at 350 degrees for 25 minutes per pound.

Du kan få en kopia av skivan om du sänder en skiva och frankerat svars-kuvert till redaktören.

■

EZ-KEYS PLUS - "HOT KEYS"

a review by Charles Good, Lima Ohio User Group, USA

Is EZ-KEYS PLUS just another a hot keys program that enables you, with a single keypress, to execute complex commands with a single keystroke? It is really hard to characterize this software. The title screen states that EZ-KEYS PLUS is "an enhanced environment for programming in extended basic." The publisher, ASGARD SOFTWARE, claims that EZ-KEYS PLUS is not offered primarily as a hot keys program, but in my opinion it beats all other hot keys programs for the TI hands down. EZ-KEYS PLUS works out of the extended basic environment and contains features that make programming in XB, or typing in programs from newsletter or magazine listings much easier. EZ-KEYS PLUS lets you create hot keys macros that do things in XB that are otherwise difficult or impossible. In terms of versatility, EZ-KEYS PLUS reminds me of the GRAM KRACKER. With both you can custom program the thing to do exactly what you want, and the custom programming possibilities are for the most part only limited by your imagination, needs, and programming skill.

EZ-KEYS PLUS is version 2 of what was originally called EZ-KEYS and reviewed in the January 1988 issue of Micropendium under that title. New features added to version 2 and not described in the Micropendium review include the following:

Automatic generation of checksums (if desired) when typing in programs.

Super easy printing to a printer of program listings in 28 colums with checksums added. This is great for newsletter editors!

Provision for you to INPUT data into the middle of a macro. This feature is very very useful, as you will subsequently see.

Automatic single density assembly language screen dump (without sprites) with the press of a key from almost anywhere within a running XB program or from command mode.

Display a disk catalog at any time without disturbing the XB program already in memory.

Create screens of text in 28 columns with a full screen editor, save the screen to disk, and load the screen back in for display from anywhere within an XB program or command mode.

Create your own custom character sets and load these into your XB programs.

You boot EZ-KEYS PLUS directly from XB usually as LOAD. It normally resides in Low memory expansion and is totally transparent to any XB program that does not have built in assembly language routines. Extensive efforts have been made by the program author, Harry Wilhelm, to make EZ-KEYS PLUS also transparent to XB programs that have assembly routines. Various techniques are available to the user to make EZ-KEYS PLUS compatible with XB programs that have assembly routines, and usually something can be worked out. EZ-KEYS PLUS doesn't interfere with other interrupt driven assembly routines such as a clock or BBS program. If necessary for compatibility with other XB assembly routines, EZ-KEYS PLUS can be loaded into Hi memory expansion.

The program comes unprotected, as does all software from ASGARD, and can be booted from a ramdisk. You can put customized versions of EZ-KEYS PLUS on your various user disks to combine EZ-KEYS PLUS's features with those of other programs. You can modify EZ-KEYS PLUS so that once loaded it will automatically RUN "DSKx.YOURPROG" to boot your application program. You can have EZ-KEYS PLUS boot FUNNELWEB v4.1x this way and have most (not all) of your previously defined EZ KEYS PLUS hot key macros available for use within any XB program you then boot from the FWB XB user list. If you do this, you loose all EZ-KEYS PLUS features that are accessed via CALL LINKs. These lost features include checksums, color changes, auto-saving, screen dump, full screen editing saving and loading, hilite, and the use of the macro editor to change macro definitions.

AIDS FOR CREATING AND/OR TYPING IN XB PROGRAMS

AUTOSAVE: You can designate a time interval in minutes and at the end of each interval EZ-KEYS PLUS will automatically save what you have typed to DSK1 alternately to files BACKUP1 and BACKUP2. This is insurance against XB lockup. You don't have to worry about an accidental QUIT. FCTN/= gives you a disk catalog instead.

CURSOR MOVEMENT: You can now move the cursor up and down within the text of a program line number. If the cursor is within the upper row of text you can move it instantly to the beginning of the LINE NUMBER and this line number can be changed if desired. If the cursor is in the last row of a displayed line number you can move it instantly to the end of the text in this line. CTRL/E and X let you perform these cursor manipulations.

HILITE: When you turn on this feature digits and arithmetic operators are displayed with foreground and background colors reversed. This makes it easy to distinguish zero from the upper case letter O, and the number one from the small case letter l.

PROGRAM LINE LENGTH: Your XB program lines are not limited to 5 screen rows any more. Just keep right on entering code after typing a line number and you can fill the entire screen (23 rows) with the code of one program line number. Sometimes, however, the XBASIC interpreter will refuse to accept extra long program lines.

SAVE AND LOAD TEXT SCREENS: With a full screen 28 column by 24 row editor you can create screens, save each screen to disk under a separate file name and then load them back into your program. This is an easy way to create help screens for viewing only if needed, or game screens.

CHECKSUMS: If you are typing in a published XB listing that contains checksums (such as those published in Micropendium and in most newsletters these days), just type CALL LINK("SUMON"), press FCTN/4, and start typing. A checksum is automatically generated each time you press ENTER. You can, if you want, create a hot key macro to do all the above CALL LINK, FCTN/4 typing for you with just a single keypress.

If you are a newsletter editor you know what a pain it is to add checksums to an existing program and then print a hard copy of the program with checksums for publication in the newsletter. Using the standard method of adding checksums to an existing program generates several intermediate disk files. The degree of complexity in doing this is similar to uncompressing and unpacking an archived file with Archiver v2.4. I am aware that commercial program SUPERBASIC makes the task of adding checksums easier than the standard method. I have used SUPERBASIC for this purpose. EZ-KEYS PLUS is by far the easiest of all. First load the extended basic program to which you wish to add checksums. Then load EZ-KEYS PLUS and type CALL LINK("SUMON"). Finally type the first line number of your XB program and then press CTRL/L. Thats all there is to it! When you press CTRL/L the printer proceeds to grind out a hardcopy list of your XB program printed in 28 column format with checksums added to the end of each line number. This is so easy it has to be seen to be believed. There is now no excese for any newsletter to publish XB code without checksums. User groups that publish a newsletter that sometimes contains XB program code should consider purchasing a copy of EZ-KEYS PLUS for use by the newsletter editor.

HOT KEYS and MACROS:

In this area EZ-KEYS PLUS really stands out from other hot keys software. You can from within EZ-KEYS PLUS easily define macros for up to 55 hot keys and then save this customized EZ-KEYS PLUS so that the defined hot keys are immediately active the next time you boot the customized software. A single macro can include up to 669 keystrokes. Approximately 1200 bytes of low memory expansion can be used to store these macros. To access a macro you press CTRL or FCTN and another key simultaneously from either XB command mode or from a running XB program. All CTRL/- and some FCTN/- keypresses are available use with your macros.

Unlike most other hot keys software packages for the TI, such as TI KEYS, you are not limited to just ASCII text. You can, for example, include <enter> in a macro definition. You can define a macro as LIST "PIO"<enter>. When the appropriate hot key is pressed, this text appears on the screen, the software presses <enter> for you, and the LIST is automatically sent to your printer. Very complex macros are available with <enter> and the following other special keys that can be included within a macro definition:
FCTN/1- delete character
FCTN/2- insert char
FCTN/3- erase entire program line
FCTN/4- CLEAR
FCTN/5- move cursor one space to the left
FCTN/6- move cursor up one screen row, or to the start of the program line
FCTN/7- move cursor down one screen line or to end of program line
FCTN/8- REDO
FCTN/9- erases all text to left of cursor
INPUT - stops execution in the middle of a macro until you input some text. The remainder of the macro will execute after you press <enter>.
HOLD - macro is ignored if the hot key is accidently pressed during a running XB program.

Small XB programs can be stored in low memory expansion and executed as macros with a single keypress. When used in this way, EZ keys acts as sort of a ram disk. The EZ-KEYS PLUS package includes such a program that will read any D/V80 file to the screen.

Macros can be chained, one macro calling another. You can embed macros within other macros, with many levels of embedding possible.

PROGLETS:

By now you are probably beginning to appreciate the extensive programming possibilities available with EZ-KEYS PLUS macros. I have saved the best for last. PROGLETS are sequences of XB commands without line numbers that are designed to run from XB command mode. They are like programs, but cannot be called programs because they lack line numbers. Many examples of useful PROGLETS are given in the EZ-KEYS PLUS docs, and some PROGLETS are already defined with the software as received. It is possible to write a proglet that will RUN an XB program, automatically insert data into the program such as answering Y or N to prompts within the program, and then do something else after the program ends. Wow! Here are some simple PROGLETS I have written for my own use:

```
OPEN #1:"PIO" :: PRINT #1
:CHR$(27)&&"G" :: CLOSE #1 !P
rinter DOUBLE PRINT<enter>
```

This macro displays the above text on the screen, presses enter, and executes the macro. The tail comment reminds me that the macro sets up my printer for doublestrike. I have other macros to set my SG10 printer for NLQ, condensed, emphasized, or expanded print, and a printer reset macro.

```
OPEN #1:"PIO" :: PRINT #1
:"<INPUT>" :: CLOSE #1<ENTER>
```

This macro allows me to type some text and have it immediately printed by my printer as soon as I hit ENTER. I can use my printer as a line by line typewriter with this macRO. The macro waits for me to input some text. After I type the text and manually press ENTER, the macro puts a closing quote at the end of my text, finishes typing the macro on the screen, then presses ENTER to automatically sends the text to the printer for printing.

CALCULATOR MACRO: By far the fastest way to use your computer as a calculator is from BASIC or XBASIC command mode. Here is my calculator macro. It appears very simple, but it allows easy data entry, can do any kind mathematical operation, and calculation time is very fast. Here is the macro:

```
! Calculator mode<enter>A=
<INPUT> :: PRINT A<ENTER>
```

The macro first prints on the screen a reminder that you are in calculator mode. You then input your digits and arithmetic operators, and then manually press ENTER. The macro then prints the answer on the screen. You can input a very long string of calculations as you might when balancing a checkbook. For example:
11290-56.98-58-2.50-436-99.95+450.
You can also input very complex operations such as (5*69.5)/6+98+.005*6/(.02*9). Press ENTER and the answer is immediately displayed on the screen. Of course you can do the same thing without EZ-KEYS PLUS directly from XB command mode, but with the macro there are fewer keypresses.

COMPATIBILITY PROBLEMS:

In addition to the occasional incompatibility with a few XB/assembly language programs, a few other problems have been discovered.

The disk directory routine will not recognize Horizon Ramdisks at high CRU addresses. Other popular software packages such as FUNNELWEB and DM1000 have been rewritten in recent years to solve this problem. I suspect that this problem can also be solved in EZ-KEYS PLUS with minor changes to its assembly code.

When using GK-EXTENDED BASIC (also known as GK UTILITY I) the screen display is confused with excess foreground colored dots if you try to automatically load another XB program from EZ-KEYS PLUS by including RUN "DSK1.ANOTHERPRG" within the EZ-KEYS PLUS code. This condition does not occur with regular TI EXTENDED BASIC. To avoid this problem, you have to wait for EZ-KEYS PLUS to fully load into GK-EXTENDED BASIC, and then load your XB application program from command mode. I suspect that the problem relates to the special character set of GK-EXTENDED BASIC being overwritten by the almost identical character set that is loaded in by EZ-KEYS PLUS. Since the SUPER EXTENDED BASIC module (version 120) is supposed to be almost the same as GK EXTENDED BASIC, users of this module may have similar problems.

Another minor problem that occurs only with the use of GK EXTENDED BASIC (and maybe also with the SUPER EXTENDED BASIC module) is the lack of special the screen display which you are supposed to see in the macro editor when you use FCTN, CTRL, INPUT, and HOLD as part of a macro definition. The defined macros execute correctly, so this is only a minor problem.

FINAL COMMENTS:

I am impressed enough with EZ-KEYS PLUS that after finishing this review I sent my money off to ASGARD and purchased my review copy. If you think you can use some of the features described in this review and/or if you enjoy experimenting with XB programming and would like to try your hand at PROGLETS, then by all means give EZ-KEYS PLUS a try.

EZ-KEYS PLUS
$14.95 plus $7.00 air mail
Asgard Software, 1423 Flagship Dr.
Woodbridge, VA 22192, USA (new addr)
Telephone: 703-491-1267 ■

MICROPENDIUM har höjt prenumerationen till 52 dollar per år med flygpost för 12 nummer. Adress: Micropendium, P.O.Box 1343, ROUND ROCK, TX 78680, USA.

TIPS FROM THE TIGERCUB #68

Tigercub Software
156 Collingwood Ave.
Columbus, OH 43213, USA
****

My three Nuts & Bolts disks, each containing 100 or more subprograms, have been reduced to $5.00 each. I am out of printed documentation so it will be supplied on disk.

My TI-PD library now has almost 600 disks of fairware (by author's permission only) and public domain, all arranged by category and as full as possible, provided with loaders by full program name rather than filename, Basic programs converted to XBasic, etc. The price is just $1.50 per disk(!), post paid if at least eight are ordered. TI-PD catalog #5 and the latest supplement is available for $1 which is deductible from the first order.

When I have finished reading Barry Traver's column in Computer Monthly, I like to take a look at whatever Dr. Michael Ecker is up to in his "Recreational Computing" column, although much of his math is beyond me and I can't always translate his GW Basic into TI Basic.

In the February issue, he had a routine to play Fibonacci modular music. This is the TI version; it is not very musical, but the notes are in the chromatic scale.

```
100 A=0 :: B=1 :: M=51
110 C=A+B :: C=C-M*INT(C/M):
: CALL SOUND(-100,110*2^(C/1
2),5):: A=B :: B=C :: GOTO 1
10
```

Dr. Ecker also had a challenge to swap two numbers without using a third variable or the SWAP command - which TI Basic doesn't have anyway. The practical way, of course, is to use the 3rd variable, T=A :: A=B :: B=T, but just for the fun of it, if we are dealing with one-digit numbers -

```
100 A=1 :: B=2 :: A=A+B/10 :
: B=INT(A):: A=(A-INT(A))*10
 :: PRINT A;B
```

But suppose we are dealing with numbers of any length - we can still do it with a one-liner, or a two-liner if we want to input the numbers from the keyboard -

```
100 INPUT A :: INPUT B
110 B=B/10^(LEN(STR$(B)))::
A=A+B :: B=INT(A):: A=A-INT(
A):: A=A*10^(LEN(STR$(A))-1)
:: PRINT A;B :: GOTO 110
```

So you got smart and tried a negative number or a decimal? OK, how about this -

```
100 INPUT A$ :: INPUT B$
110 A$=A$&" "&B$ :: B$=SEG$(
A$,1,POS(A$," ",1)-1):: A$=S
EG$(A$,POS(A$," ",1)+1,255):
: PRINT A$;" ";B$ :: GOTO 11
0
```

And another challenge was to alternately assign X the value of A and B, without using IF...THEN or any outside help. That seems to require a two-liner -

```
100 A,X=77 :: B=132
110 X=ABS(X=A)*B+ABS(X=B)*A
:: PRINT X :: GOTO 110
```

The only honest way to compute interest on a loan is on the unpaid balance, although the banks and finance companies have devised more complicated and profitable ways. If you want to make an honest loan, here is how to do it -

```
100 DISPLAY AT(3,1)ERASE ALL
:"SIMPLE INTEREST CALCULATOR
":"":"For interest to be cal
cu-   lated monthly on unpai
d      balance."
110 DISPLAY AT(9,1):"Printer
? PIO" :: ACCEPT AT(9,10)SIZ
E(-20):P$
120 DISPLAY AT(11,1):"Amount
 loaned? $" :: ACCEPT AT(11,
17)VALIDATE(NUMERIC):A
130 DISPLAY AT(13,1):"Intere
st rate?     %" :: ACCEPT AT
(13,16)SIZE(4)VALIDATE(NUMER
IC):X
140 IF X<1 THEN DISPLAY AT(1
2,1):"Enter as a percentage"
 :: GOTO 130
150 DISPLAY AT(15,1):"Monthl
y payments of $" :: ACCEPT A
T(15,22)VALIDATE(NUMERIC):P
160 DISPLAY AT(17,1):"Beginn
ing in month (1-12)   of yea
r"
170 ACCEPT AT(17,27)VALIDATE
(DIGIT):M :: ACCEPT AT(18,9)
VALIDATE(DIGIT):Y
180 DATA JAN,FEB,MAR,APR,MAY
,JUN,JUL,AUG,SEP,OCT,NOV,DEC
190 X=X/100 :: DIM M$(12)::
FOR J=1 TO 12 :: READ M$(J):
: NEXT J
200 OPEN #1:P$,VARIABLE 254
:: PRINT #1:CHR$(27)&"E"&CHR
$(27)&"G"&CHR$(27)&"N"&CHR$(
6)&CHR$(27)&"M";
210 PRINT #1:"$";STR$(A);" F
INANCED AT ";STR$(X*100);"%
WITH MONTHLY PAYMENTS OF $";
STR$(P);" BEGINNING ";M$(M);
Y:""
220 I=A*X/12 :: TI=TI+I :: A
=A+I-P
230 PRINT #1:M$(M);Y;" PAYME
NT $";STR$(P);" OF ";
240 PRINT #1,USING "$###.##"
:I;:: PRINT #1:" INTEREST AN
D ";
250 PRINT #1,USING "$####.##
":P-I;:: PRINT #1:" PRINCIPA
L - BALANCE OF ";
260 PRINT #1,USING "$####.##
":A
270 M=M+1 :: IF M=13 THEN M=
1 :: Y=Y+1
280 IF A>=P THEN 220
290 PRINT #1,USING "FINAL PA
YMENT $###.##":A :: PRINT #1
,USING "TOTAL INTEREST PAYED
 $####.##":TI
```

Thanks to Bruce Harrison, here is a neat subprogram to sort strings into sequence as they are entered -

```
100 CALL CLEAR :: DIM W$(100
)
110 FOR J=1 TO N :: W$(J)=""
 :: NEXT J :: INPUT "N=? ":N
120 INPUT I$ :: IF I$="" THE
N 130 ELSE CALL INSORT(W$(),
I$,N):: GOTO 120
130 FOR J=1 TO N :: PRINT W$
(J):: NEXT J :: GOTO 110
30020 SUB INSORT(W$(),I$,N):
: FOR T=1 TO N :: IF I$>W$(T
)THEN 30030 ELSE 30040
30030 NEXT T :: GOTO 30050
30040 FOR J=N TO T STEP -1 :
: W$(J+1)=W$(J):: NEXT J
30050 W$(T)=I$ :: N=N+1 :: S
UBEND
```

In the test routine in lines 100-130, give N the value of 0, input some words and then just press enter.

To start a new array, use FOR J=1 TO N :: W$(J)="" :: NEXT J, then reset N to 0. If you want to sort in reverse sequence, change the > to <. If you need to sort numbers, delete all the $, change the "" in line 120 to 0, and input a 0 when you are when finished inputting.

Someone sent me a program to figure days between dates but it would not count leap dates, so I decided to write one that would.

(See the listed program in PROGRAMBITEN 92-3 page 21)

A leap year is a year that is evenly divisible by 4 unless it is evenly divisible by 100 but not evenly divisible by 400. The subprogram in lines 270-280 will give X a value of -1 if Y is a leap year.

Gene Hitz of Arcade Action Software reports another undocumented feature of TI Extended Basic. The manual says that you can only enter a subprogram by a CALL and only leave it by a SUBEXIT or SUBEND, but the manual is wrong. You can GOSUB to a subroutine within a subprogram, providing it does not contain a SUBEXIT, and return; and you can GOSUB from within a subprogram to a subroutine in the main program, and return. In this way, you can transfer variables in and out of a subprogram without putting them in a parameter list. See for yourself -

```
100 CALL CLEAR
110 INPUT M$ :: CALL SUB(M$)
:: PRINT M$ :: GOSUB 140 ::
PRINT "M$ IS";X;"CHARACTERS
LONG" :: GOTO 110
120 M$="SEE WHAT I TOLD YOU?
" :: RETURN
130 SUB SUB(M$):: GOSUB 120
:: GOSUB 140 :: SUBEXIT
140 X=LEN(M$):: RETURN
150 SUBEND
```

If you are among the lonely few who have purchased my TI-PD disks, you will know that most of them load from a menu by full program name, not those abbreviated filenames. Those menus are prepared quickly and easily by my Catwriter program which was published in Tips #47 and in MICROpendium and is available on TI-PD 1105.2.

I was asked if there was a way to dump those full program names to the printer. There is, but it requires a big program - like this -

```
1 OPEN #1:"DSK2.TI-PD/CAT",A
PPEND
2 DISPLAY AT(12,1)ERASE ALL:
"TI-PD# ?" :: ACCEPT AT(12,1
0):N
14 FOR J=1 TO X-1 :: READ X$
 :: PRINT #1:X$;TAB(30);N ::
 NEXT J :: CLOSE #1 :: STOP
17 REM
```

Save that on an empty disk by SAVE DSK2.C,MERGE. Put your TI-PD disk in drive 1, boot its LOAD program, break it with FCTN 4 and enter MERGE DSK2.C, then RUN. Put in the next TI-PD disk and do the same. You will have a D/V80 file of all the programs, followed by their TI-PD disk number. Run the file through Sort Experiment or TI-Sort or whatever, and you can print them out in alphabetical sequence.

If you have only one drive just change that DSK2. to DSK1. and swap disks after breaking the LOAD program.

Of course, this won't work wth fairware disks which have the author's own loader or some other disks which do not have my Catwriter load for one reason or another. You'll have to type those into the file.

Another user asked me if there was anyway to key in the ASCII above 127 into TI-Writer's Editor. Many of those ASCII can be entered from the keyboard by using the CTRL and FCTN keys - try this -

```
100 INPUT N$ :: PRINT ASC(N$
):: GOTO 100
```

- but the Editor has been programmed to refuse them because so many of those FCTN and CTRL combinations are used as edit commands.

I had a bright idea - I thought. I wrote a little program to create 127 files, named 128 through 255, each containing just the ASCII of the same number. Now, I thought, when I want to put in such an ASCII I will just LF that file into the next line and CTR 2 to pop it into place. But the Editor refused to even load a file that began with an ASCII above 127!

I'll fool you, I thought. I created those files again, but with an asterisk before the high ASCII. Now they loaded alright - but each ASCII above 127 became an ASCII 128 numbers lower! It is too bad that the Editor does not have a command to add 127 to an ASCII, just as CTRL U subtracts 64, but if you want those graphics characters in your text you will just have to transliterate them and print through the Formatter.

Folks take it for granted that my Nuts & Bolts disks are only useful for programmers, but they contain many routines so simple to use that anyone can use them to dress up their favorite program. For instance -

```
20083 SUB TITLE(S,T$):: CALL
 SCREEN(S):: L=LEN(T$):: CAL
L MAGNIFY(2)
20084 FOR J=1 TO L :: CALL S
PRITE(#J,ASC(SEG$(T$,J,1)),J
+1-(J+1=S)+(J+1=S+13)+(J>14)
*13,J*(170/L),10+J*(200/L)):
: NEXT J
20085 SUBEND
```

Key that in and save it by SAVE DSK1.TITLE,MERGE . Load your favorite program. Enter MERGE DSK1.TITLE . Make sure your program does not have a line 1 or 2 - if so, RES it. Type in -

```
1 CALL CLEAR :: CALL TITLE(5
,"MY PROGRAM")
2 FOR D=1 TO 1000 :: NEXT D
:: CALL DELSPRITE(ALL)
```

And try it. Instead of "MY PROGRAM", put the name of your program. Instead of 5, put the number of whatever screen color you would like, from 2 to 16 - check your Basic manual. Change 1000 to whatever delay you want - if you have selected a screen color that will leave text legible, use -

```
2 DISPLAY AT(24,1):"PRESS AN
Y KEY" :: DISPLAY AT(24,1):"
press any key" :: CALL KEY(0
,K,S):: IF S=0 THEN 2 ELSE C
ALL DELSPRITE(ALL)
```

You might also need a CALL SCREEN(8) to restore normal screen color.
Oops! Memory full! - Jim P ■

TI*MES(UK) 15,50 pund per år: TI99/4a User's Group(UK), c/o Alasdair Bryce, 51 Dumbuie Ave., Silverton, Dumbarton, Scotland, G82 2JH

EKONOMIRAPPORT

```
100 CALL CHAR(91,"0028003844
7C4444")
110 CALL CHAR(92,"0028007C44
44447C")
120 CALL CHAR(93,"0038283844
7C4444")
130 CALL CHAR(123,"000028003
8447C44")
140 CALL CHAR(124,"000028007
C44447C")
150 CALL CHAR(125,"000038283
8447C44")
170 REM TAX DEDUCTION FILER
180 OPTION BASE 1 :: DIM A$(
500),T(17),N$(17)
190 FOR Z=1 TO 8 :: READ M$(
Z):: NEXT Z :: FOR Z=1 TO 17
 :: READ N$(Z):: NEXT Z :: N
=2 :: GOSUB 850
200 CALL CLEAR :: DISPLAY AT
(0,10):"HUVUDMENY
           ————————" :: FOR Z=
1 TO 8 :: DISPLAY AT(4+Z*2,1
):STR$(Z);") ";M$(Z):: NEXT
Z
210 GOSUB 860 :: IF K<49 OR
K>56 THEN CALL SOUND(50,220,
0):: GOTO 210
220 CALL CLEAR :: ON K-48 GO
TO 230,290,450,510,530,690,7
50,940
230 GOSUB 820
239 DISPLAY AT(24,1):"Tryck
enter för meny"
240 DISPLAY AT(21,3):"<" ::
ACCEPT AT(21,1)VALIDATE(DIGI
T)SIZE(2):C$ :: IF C$="" THE
N 200 ELSE C=VAL(C$)
250 IF C<1 OR C>17 THEN CALL
 SOUND(50,220,0):: GOTO 240
260 DISPLAY AT(20,1):"Sakbes
krivning:         " :: ACCEPT
 AT(21,1):D$ :: IF D$="" THE
N 200
270 DISPLAY AT(23,1):"Skriv
värdet:     " :: ACCEPT AT(24
,1)VALIDATE(NUMERIC):V$ :: I
F V$="" THEN 200
275 IF C=1 THEN V$="-"&V$
280 R=R+1 :: A$(R)=CHR$(100+
C)&CHR$(LEN(D$)+100)&D$&V$ :
: CALL HCHAR(20,1,32,160)::
GOTO 230
290 GOSUB 820
291 DISPLAY AT(24,1):"Tryck
enter för meny"
300 DISPLAY AT(21,3):"<" ::
ACCEPT AT(21,1)VALIDATE(DIGI
T)SIZE(2):C$ :: IF C$="" THE
N 200 ELSE C=VAL(C$)
310 IF C<1 OR C>17 THEN CALL
 SOUND(50,220,0):: GOTO 300
320 CALL CLEAR :: L=1 :: FOR
 Z=1 TO R :: GOSUB 810 :: IF
 C=AC THEN DISPLAY AT(L,1):S
TR$(Z);") ";D$;" ";V$ :: L=L
+1
330 IF L/20<>INT(L/18)AND Z<
>R THEN 440 ELSE DISPLAY AT(
23,1):"Skriv nummer:":"Enter
 för oförändrat."
340 ACCEPT AT(23,14)VALIDATE
(DIGIT)SIZE(3):E$ :: IF E$="
" THEN 200 ELSE E=VAL(E$)::
IF E>R OR E<1 THEN CALL SOUN
D(50,220,0):: GOTO 340
350 Z=E :: GOSUB 810 :: IF A
C<>C THEN 340 ELSE CALL HCHA
R(20,1,32,160):: DISPLAY AT(
23,1):"————————————————
————————"
355 DISPLAY AT(22,1):"/D/ fö
r radering" :: DISPLAY AT(24
,1):"Ny text:";D$ :: ACCEPT
AT(24,9)SIZE(-26):DN$
370 IF DN$="" THEN 410
380 IF DN$<>"/D/" THEN 400
390 FOR N=E TO N :: A$(N)=A$
(N+1):: NEXT N :: R=R-1 :: G
OTO 200
400 D$=DN$
410 DISPLAY AT(22,1):"Obs! M
inus för inkomst" :: DISPLAY
 AT(24,1):"Ny summa för:";E
:: ACCEPT AT(24,18)VALIDATE(
NUMERIC):NV$
420 IF NV$<>"" THEN V$=NV$
430 A$(E)=CHR$(100+C)&CHR$(L
EN(D$)+100)&D$&V$ :: GOTO 20
0
440 NEXT Z :: GOTO 290
450 GOSUB 820
459 DISPLAY AT(24,1):"Tryck
enter för meny"
460 DISPLAY AT(22,3):"<" ::
ACCEPT AT(22,1)VALIDATE(DIGI
T)SIZE(2):C$ :: IF C$="" THE
N 200 ELSE C=VAL(C$)
470 IF C<1 OR C>17 THEN CALL
 SOUND(50,220,0):: GOTO 460
480 CALL CLEAR :: L=1 :: FOR
 Z=1 TO R :: GOSUB 810 :: IF
 C=AC THEN DISPLAY AT(L,1):S
TR$(Z);") ";D$;" Kr:";VAL(V$
)*-1 :: L=L+1
490 IF L/20<>INT(L/18)AND Z<
>R THEN 440 ELSE DISPLAY AT(
24,1):"Tryck enter för meny
" :: GOSUB 860
500 NEXT Z :: GOTO 200
510 DISPLAY AT(1,8):"TOTALSU
MMA" :: FOR Z=1 TO 17 :: T(Z
)=0 :: NEXT Z :: FOR Z=1 TO
R :: GOSUB 810 :: V=VAL(V$):
: T(AC)=T(AC)+V :: NEXT Z
520 FOR Z=1 TO 17 :: DISPLAY
 AT(Z+2,1):N$(Z);TAB(21);STR
$(T(Z)*-1):: ZC=ZC+T(Z):: NE
```

```
XT Z :: DISPLAY AT(22,0):"Ne
tto:"&STR$(ZC*-1):: ZC=0
521 DISPLAY AT(24,1):"Tryck
enter för meny"
522 CALL KEY(0,K,S):: IF S=0
 THEN 522 ELSE 200
530 CALL CLEAR :: DISPLAY AT
(1,1):"RS232.BA=110.LF" :: A
CCEPT AT(1,1)SIZE(-28):DV$ :
: OPEN #2:DV$
540 DISPLAY AT(4,1):"
                 ": : : :"1)
Nettorapport      ": :"2) Se
parata kategorier    ": :"3
) Samtliga poster"
541 DISPLAY AT(22,1):"Tryck
enter för meny"
550 GOSUB 860 :: IF K=13 THE
N CLOSE #2 :: GOTO 200 ELSE
IF K<49 OR K>51 THEN CALL SO
UND(50,220,0):: GOTO 550
560 DISPLAY AT(18,1):"Skriv
in datum:":RT$
569 ACCEPT AT(19,1)SIZE(-28)
:RT$
570 PRINT #2: : :TAB(18);"EK
ONOMIRAPPORT:";TAB(34);RT$;D
T$ :: PRINT #2:TAB(18);"————
————————————————————————————
————————————————"

571 ON K-48 GOTO 580,610,670
580 PRINT #2:TAB(18);"Nr. Ka
tegorier";TAB(56);"Totalsumm
a:":TAB(18);"———————————————
————————————————————————————
——————"

589 FOR Z=1 TO 17 :: T(Z)=0
:: NEXT Z
590 FOR Z=1 TO R :: GOSUB 81
0 :: T(AC)=T(AC)+VAL(V$):: N
EXT Z :: FOR Z=1 TO 17 :: PR
INT #2:TAB(17);Z;" ";TAB(22)
;N$(Z);TAB(56);"Kr ";T(Z)*-1
 :: ZC=ZC+T(Z)
591 NEXT Z
594 PRINT #2:"":TAB(22);""
595 PRINT #2:TAB(18);"
                         Net
to: Kr  "&STR$(ZC*-1):: ZC=0
596 PRINT #2:TAB(18);"——————
————————————————————————————
——————————————"

600 CLOSE #2 :: GOTO 200
610 CALL CLEAR :: K=53 :: GO
SUB 820 :: DISPLAY AT(21,1):
"       Kategori:"
620 ACCEPT AT(21,18)VALIDATE
(DIGIT)SIZE(3):C$ :: IF C$="
" THEN CLOSE #2 :: GOTO 200
630 C=VAL(C$):: IF C<1 OR C>
17 THEN CALL SOUND(50,220,0)
:: GOTO 620
640 PRINT #2:TAB(18);"Katego
rislag: ";N$(C):"
      Delposter:————————————
——————————————————————————————"

650 FOR Z=1 TO R :: IF C=ASC
(A$(Z))-100 THEN GOSUB 810 :
: PRINT #2:TAB(32);D$;TAB(55
);"Kr ";VAL(V$)*-1 :: ZC=ZC+
T(Z)
660 NEXT Z :: CLOSE #2 :: GO
TO 200
670 PRINT #2:TAB(18);"Samtli
ga poster:" :: PRINT #2:TAB(
18);"Nr. Sakbeskrivning:
                Summa:"
672 PRINT #2:TAB(18);"——————
————————————————————————————
——————————————"

680 FOR Z=1 TO R :: GOSUB 81
0 :: PRINT #2:;" ";TAB(18);A
C;TAB(22);D$;TAB(55);"Kr ";V
AL(V$)*-1 :: NEXT Z :: CLOSE
 #2 :: GOTO 200
690 DISPLAY AT(1,1):M$(6): :
 : : : : : :"1) DSK1.
  ": :"2) Kassett     ": :"
3) Annan kod"
691 DISPLAY AT(24,1):"Tryck
enter för meny"
700 GOSUB 860 :: IF K<49 OR
K>51 THEN 200 ELSE ON K-48 G
OTO 720,710,730
710 OPEN #1:"CS1",INPUT ,FIX
ED :: GOTO 740
720 DISPLAY AT(22,1):"Filnam
n:";FN$;TAB(22);" " :: ACCEP
T AT(22,9)SIZE(-10):FN$ :: O
PEN #1:"DSK1."&FN$,VARIABLE
60,INPUT :: GOTO 740
730 DISPLAY AT(22,1):"Kodnam
n:":DEV$ :: ACCEPT AT(22,9):
DEV$ :: OPEN #1:DEV$,VARIABL
E 60,INPUT
740 INPUT #1:R :: FOR Z=1 TO
 R :: INPUT #1:A$(Z):: NEXT
Z :: CLOSE #1 :: GOTO 200
750 DISPLAY AT(1,1):M$(7): :
 : : : : : :"1) DSK1.
  ": :"2) Kassett
  ": :"3) Annan kod   "
751 DISPLAY AT(24,1):"Tryck
enter för meny"
760 GOSUB 860 :: IF K<49 OR
K>51 THEN 200 ELSE ON K-48 G
OTO 780,770,790
770 OPEN #1:"CS1",OUTPUT,FIX
ED :: GOTO 800
780 DISPLAY AT(22,1):"Filnam
n:";FN$;TAB(22);"" :: ACCEPT
 AT(22,9)SIZE(-18):FN$ :: OP
EN #1:"DSK1."&FN$,VARIABLE 6
0,OUTPUT :: GOTO 800
790 DISPLAY AT(22,1):"Kodnam
n:";DEV$ :: ACCEPT AT(22,9)S
IZE(-18):DEV$ :: OPEN #1:DEV
$,VARIABLE 60,OUTPUT
800 PRINT #1:R :: FOR Z=1 TO
 R :: PRINT #1:A$(Z):: NEXT
Z :: CLOSE #1 :: GOTO 200
810 AC=ASC(A$(Z))-100 :: AL=
ASC(SEG$(A$(Z),2,1))-100 ::
D$=SEG$(A$(Z),3,AL):: V$=SEG
$(A$(Z),AL+3,LEN(A$(Z))-AL-2
):: RETURN
820 DISPLAY AT(1,(28-LEN(M$(
K-48)))/2+1):M$(K-48):: FOR
Z=1 TO 17 :: DISPLAY AT(2+Z,
1):STR$(Z);") ";N$(Z):: NEXT
 Z :: RETURN
830 FOR Z=1 TO L-F+1 :: DISP
LAY AT(2+Z,1):STR$(Z);") ";N
$(F+Z-1):: NEXT Z :: GOTO 86
0
840 FOR Z=1 TO N :: READ X,Y
,P$ :: DISPLAY AT(X,Y):P$ ::
 NEXT Z :: GOTO 860
850 CALL CLEAR :: FOR Z=1 TO
 N :: READ X,Y,P$ :: DISPLAY
 AT(X,Y):P$ :: NEXT Z :: CAL
L SOUND(10,440,0)
860 CALL KEY(0,K,S):: IF S=0
 THEN 860 ELSE RETURN
870 DATA SKRIVA IN DATA,ÄNDR
A DATA,VISA UPPGIFTER,TOTALS
UMMA,PRINTER,LADDA DATA,SPAR
A DATA,LÄMNA PROGRAMMET
881 DATA Inkomster
882 DATA Kläder & Skor
883 DATA Matköp,Huskostnader
,Läkare & medicin
890 DATA Bil & Båt,Amorterin
gar,Kontokort,El & Vatten,Re
sor & gåvor,Sparkonton,Rep &
 Underhåll
900 DATA Facket,Försäkringar
,Semesterpengar,Trädgård,Övr
iga utgifter
910 DATA 12,6,* EKONOMIRAPPO
RTER *          ————————————
————,24,3,Tryck enter
920 DATA CS1,DSK1,OTHER
930 DATA Skriv ny sak,Ändra
uppgift,Visa uppgift,Till Me
ny
940 CALL CLEAR :: DISPLAY AT
(12,1):"Data raderas om Du
        lämnar programmet oc
h       Loadprogrammet körs!
"
950 DISPLAY AT(24,1):"Skriv
(J/N):J" :: ACCEPT AT(24,14)
SIZE(1)VALIDATE("JN"):P$ ::
IF P$="N" THEN 200 ELSE RUN
"DSK1.LOAD"                ■
```

FEMTONSPEL -XB

Flytta siffror eller bokstäver i ordning med ESDX-tangenter eller joystick. Antalet drag räknas. Normalt flyttas den tomma rutan men detta kan inverteras med I så att siffra/bokstav flyttas. Tryck N så stängs dragljudet av. Tryck minus (-) eller Fire på joystick så backar dragen upp till 250 drag. FCTN(REDO) ställer in samma startskärm. FCTN(BACK) tillbaka till huvudmenyn. FCTN(BEGIN) ger nytt spel. FCTN(ERASE) avslutar programmet.

```
160 REM MOSAIC PUZZLE
170 REM BY R.ROTHSTEIN
180 REM COMPUTE 83-10 XB
190 CALL MAGNIFY(4):: RANDOM
IZE :: DIM TILE(16),TEMP(16)
:: TEMP(16)=16 :: FR=153
200 CALL CLEAR :: CALL SCREE
N(11):: CALL CHARSET :: CALL
 CHAR(35,"0",71,RPT$("0",12)
&"FF0000FF")
210 DISPLAY AT(1,9):RPT$("G"
,12):: DISPLAY AT(2,9):"OPTI
ON##MENU"
220 DISPLAY AT(3,9):RPT$("H"
,12):: DISPLAY AT(10,4):"PRE
SS####FOR":"   HHHHH####HHH"
230 DISPLAY AT(13,6):"1#####
#NUMBER#PUZZLE" :: DISPLAY A
T(16,6)BEEP:"2######LETTER#P
UZZLE"
240 WASTE=RND :: CALL KEY(3,
K,ST):: IF ST=0 THEN 240
250 IF K=49 THEN I=0 ELSE IF
 K=50 THEN I=1 ELSE IF K=7 T
HEN 700 ELSE CALL SOUND(150,
110,0):: GOTO 240
260 CALL CLEAR :: CALL SCREE
N(4):: IF I=0 THEN RESTORE 7
10 ELSE RESTORE 790
270 FOR I=80 TO 136 STEP 4 :
: READ A$,B$ :: CALL CHAR(I,
"FFFF"&A$&"FFFFFFFF"&B$&"FFF
F"):: NEXT I :: CALL CHAR(14
0,RPT$("0",64))
280 CALL CHAR(71,"0000000000
000000303030303030303000000FF
FF")
290 CALL CHAR(74,"00003F3F30
3030300000FCFC0C0C0C0C303030
303F3F00000C0C0C0CFCFC",78,R
PT$("0C",8)&"00000000FFFF")
300 RESTORE 870 :: FOR I=38
TO 47 :: READ A$ :: CALL CHA
R(I,A$):: NEXT I
310 CALL COLOR(5,15,2,6,15,2
):: CALL HCHAR(2,7,71,20)::
DISPLAY AT(3,5):"GJ"&RPT$("I
",16)&"KG"
320 FOR I=4 TO 16 STEP 4 ::
DISPLAY AT(I,1):RPT$("####GH
"&RPT$("G",16)&"NG####",4)::
 NEXT I
330 DISPLAY AT(20,5):"GL"&RP
T$("O",16)&"MG" :: CALL HCHA
R(21,7,71,20)
340 GOSUB 890 :: FOR I=1 TO
15 :: TILE(I)=I :: NEXT I
350 FOR J=1 TO 15 :: R=1+INT
(RND*(16-J)):: TEMP(J)=TILE(
R):: TILE(R)=TILE(16-J):: NE
XT J
360 N=0 :: FOR I=1 TO 14 ::
FOR J=1 TO 15-I :: IF TEMP(I
)>TEMP(I+J)THEN N=N+1
370 NEXT J :: NEXT I :: IF N
/2<>INT(N/2)THEN TEMP(16)=TE
MP(15):: TEMP(15)=TEMP(14)::
 TEMP(14)=TEMP(16):: TEMP(16
)=16
380 FOR I=1 TO 16 :: TILE(I)
=TEMP(I):: NEXT I :: N=0 ::
SP=16
390 FOR I=22 TO 124 STEP 34
:: FOR J=62 TO 164 STEP 34 :
: N=N+1 :: CALL LOCATE(#TILE
(N),I,J):: NEXT J :: NEXT I
400 MOVE$="" :: TOTAL=0 :: D
ISPLAY AT(24,1):CHR$(32+6*DI
R)&"#####()*+#,-(.+/:#0#####
##"&CHR$(39-7*NO):: CALL SOU
ND(150,666,0):: GOTO 430
410 CALL SOUND(150,110,0)
420 WASTE=RND :: IF K=73 OR
K=78 OR K=105 OR K=110 THEN
450
430 CALL KEY(1,KK,ST):: IF K
K=18 THEN 670 ELSE CALL JOYS
T(1,X,Y):: IF ABS(X)+ABS(Y)=
8 OR X+Y=0 THEN 450
440 IF X/4=(-1)^(1-DIR)THEN
510 ELSE IF X/4=(-1)^(2-DIR)
THEN 550 ELSE IF Y/4=(-1)^(2
-DIR)THEN 630 ELSE IF Y/4=(-
1)^(1-DIR)THEN 590
450 CALL KEY(3,K,ST):: IF ST
=0 THEN 430 ELSE IF K=45 THE
N 670 ELSE IF K=83-HORZ THEN
 510
460 IF K=68+HORZ THEN 550 EL
SE IF K=88-VERT THEN 590 ELS
E IF K=69+VERT THEN 630
470 IF K=6 THEN CALL HCHAR(2
4,4,32,26):: GOSUB 890 :: GO
TO 380 ELSE IF K=7 THEN 700
480 IF K=15 THEN CALL DELSPR
ITE(ALL):: GOTO 200 ELSE IF
K=14 THEN CALL HCHAR(24,4,32
,26):: GOTO 340 ELSE IF ST=-
1 THEN 420
490 IF K=78 THEN NO=1-NO ::
FR=153+NO*30000 :: CALL HCHA
R(24,30,39-7*NO):: GOTO 420
500 IF K=73 THEN HORZ=15-HOR
Z :: VERT=19-VERT :: DIR=1-D
IR :: CALL HCHAR(24,3,32+6*D
IR):: GOTO 420 ELSE 410
510 IF SP=1 OR SP=5 OR SP=9
OR SP=13 THEN 410 ELSE SP=SP
-1 :: CALL POSITION(#TILE(SP
),ROW,COL):: CALL SOUND(4000
,FR,14*NO)
520 IF MINUS=0 THEN MOVE$="L
"&MOVE$ ELSE MINUS=0
530 FOR I=COL TO COL+34 STEP
 2 :: CALL LOCATE(#TILE(SP),
ROW,I):: NEXT I
540 TILE(SP+1)=TILE(SP):: TI
LE(SP)=16 :: CALL SOUND(-1,F
R,30):: GOTO 690
550 IF SP=4 OR SP=8 OR SP=12
 OR SP=16 THEN 410 ELSE SP=S
P+1 :: CALL POSITION(#TILE(S
P),ROW,COL):: CALL SOUND(400
0,FR,14*NO)
560 IF MINUS=0 THEN MOVE$="R
"&MOVE$ ELSE MINUS=0
570 FOR I=COL TO COL-34 STEP
 -2 :: CALL LOCATE(#TILE(SP)
,ROW,I):: NEXT I
580 TILE(SP-1)=TILE(SP):: TI
LE(SP)=16 :: CALL SOUND(-1,F
R,30):: GOTO 690
590 IF SP>12 THEN 410 ELSE S
P=SP+4 :: CALL POSITION(#TIL
E(SP),ROW,COL):: CALL SOUND(
4000,FR,14*NO)
600 IF MINUS=0 THEN MOVE$="D
"&MOVE$ ELSE MINUS=0
610 FOR I=ROW TO ROW-34 STEP
 -2 :: CALL LOCATE(#TILE(SP)
,I,COL):: NEXT I
620 TILE(SP-4)=TILE(SP):: TI
LE(SP)=16 :: CALL SOUND(-1,F
R,30):: GOTO 690
630 IF SP<5 THEN 410 ELSE SP
=SP-4 :: CALL POSITION(#TILE
(SP),ROW,COL):: CALL SOUND(4
000,FR,14*NO)
640 IF MINUS=0 THEN MOVE$="U
"&MOVE$ ELSE MINUS=0
650 FOR I=ROW TO ROW+34 STEP
 2 :: CALL LOCATE(#TILE(SP),
I,COL):: NEXT I
660 TILE(SP+4)=TILE(SP):: TI
LE(SP)=16 :: CALL SOUND(-1,F
R,30):: GOTO 690
670 IF MOVE$="" THEN 410 ELS
E MINUS=1 :: K=ASC(MOVE$)::
MOVE$=SEG$(MOVE$,2,250):: TO
TAL=TOTAL-2
680 IF K=82 THEN 510 ELSE IF
 K=76 THEN 550 ELSE IF K=85
THEN 590 ELSE IF K=68 THEN 6
30
690 TOTAL=TOTAL+1 :: DISPLAY
 AT(24,19)SIZE(4)BEEP:TOTAL
:: MOVE$=SEG$(MOVE$,1,250)::
 GOTO 420
700 CALL DELSPRITE(ALL):: CA
LL CLEAR :: END
710 DATA FEFEFEFEFEFEFEFEFEF
EFEFE,7F7F7F7F7F7F7F7F7F7F7F
7F,F8F0F3FFFFFFFEFCF8F1F0F0,
1F0FCFCF8F1F3F7FFFFF0F0F
720 DATA F8F0F3FFFFFEFEFFFFF
3F0F8,1F0FCFCF8F1F1F8FCFCF0F
1F,FFFFFEFCF8F1F0F0FFFFFFFF,
8F0F0F4FCFCF0F0FCFCFCFCF
730 DATA F0F0F3F3F0F0FFFFFFF
3F0F8,1F1FFFFF3F1F8FCFCF8F1F
3F,FCF8F1F3F0F0F1F3F3F1F8FC,
1F1FFFFF3F1F8FCFCF8F1F3F
740 DATA F0F0F3FFFFFFFFFEFEF
EFEFE,0F0FCF8F9F1F3F3F7F7F7F
7F,FCF8F9F9FCF8F1F3F3F1F8FC,
3F1F9F9F3F1F8FCFCF8F1F3F
750 DATA FCF8F1F3F3F1F8FCFFF
FF8F8,3F1F8FCFCF8F0F0FCF8F1F
3F,CFCECCCCCCCCCCCCCCCCCECF,
0F0763F3F3F3F3F3F363070F
760 DATA F3F3F3F3F3F3F3F3F3F
3F3F3,CFCFCFCFCFCFCFCFCFCFCF
CF,CECCCCCFCFCFCFCFCECCCCCC,
0703F3F3E3C78F1F3F7F0303
770 DATA CECCCCCFCFCFCFCFCFC
CCCCE,070373F3E38787E3F3F303
07,CFCFCFCFCECCCCCCCFCFCFCF,
E3C3831333730303F3F3F3F3
780 DATA CCCCCCCCCCCCCFCFCFC
CCCCE,0707FFFF0F07E3F3F3E307
0F
790 DATA FEFCF8F1E3E7E0E0E7E
7E7E7,7F3F1F8FC7E70707E7E7E7
E7,E0E0E7E7E7E0E0E7E7E7E0E0,
1F0FC7E7C70F0FC7E7C70F1F
800 DATA F8F0E3E7E7E7E7E7E7E
3F0F8,0F07E7FFFFFFFFFFFFE707
0F,E0E0E7E7E7E7E7E7E7E7E0E0,
3F1F8FC7E7E7E7E7C78F1F3F
810 DATA E0E0E7E7E7E0E0E7E7E
7E0E0,0707FFFFFF1F1FFFFFFF07
07,E0E0E7E7E7E0E0E7E7E7E7E7,
0707FFFFFF1F1FFFFFFFFFFF
820 DATA F8F0E3E7E7E7E7E7E7E
3F0F8,0F07E7FFFFFF8787E7E707
07,E7E7E7E7E7E0E0E7E7E7E7E7,
E7E7E7E7E70707E7E7E7E7E7
830 DATA F8F8FEFEFEFEFEFEFEF
EF8F8,1F1F7F7F7F7F7F7F7F7F1F
1F,FFFFFFFFFFFFFFFFFE7E3F0F8,
E7E7E7E7E7E7E7E7E7C70F1F
840 DATA E7E7E7E6E4E0E0E0E6E
7E7E7,C78F1F3F7FFFFF7F3F1F8F
C7,E7E7E7E7E7E7E7E7E7E7E0E0,
FFFFFFFFFFFFFFFFFFFFFF0707
850 DATA E7E3E1E0E4E6E7E7E7E
7E7E7,E7C787072767E7E7E7E7E7
E7,E7E3E1E0E4E6E6E7E7E7E7E7,
E7E7E7E7E76767270787C7E7
860 DATA F8F0E3E7E7E7E7E7E7E
3F0F8,1F0FC7E7E7E7E7E7E7C70F
1F
870 DATA 3060FF0000FF060C,0E
09080868F8F86,00446C54544444
44,007C4444444444447C,00444444
2828101,007C40407840407C
880 DATA 00446464544C4C44,00
44444444444438,0078242438242
478,0078444478504844,@
890 FOR I=1 TO 16 STEP 2 ::
CALL SPRITE(#I,76+4*I,16,193
,1,#I+1,80+4*I,11,193,1):: N
EXT I :: RETURN                ■
```

PRINT D/V 80

```
100 REM PRINT ORD XB NEC
110 REM JAN ALEXANDERSSON
120 REM VERSION 1987-03-08
130 CALL CLEAR :: MARG$="l"&
CHR$(10)
140 CALL CHAR(91,"0028003844
7C44440028007C4444447C003828
38447C4444")
150 INPUT "\NSKAD FIL ":FIL$
210 INPUT "KONTINUERLIG ELLE
L STEG ":VAL$
220 OPEN #1:FIL$,INPUT
230 OPEN #2:"PIO"
260 PRINT #2:CHR$(27);MARG$;
270 IF ASC(VAL$)=83 THEN 390
280 FOR I=1 TO 60
290 LINPUT #1:TEXT$
310 PRINT #2:TEXT$
320 IF EOF(1)=1 THEN 360
330 NEXT I
340 PRINT #2:CHR$(12)
350 GOTO 280
360 CLOSE #1
370 CLOSE #2
380 END
390 LINPUT #1:TEXT$
400 PRINT TEXT$
420 CALL KEY(3,KEY,STA):: IF
 STA<1 THEN 420 :: IF KEY=70
 THEN PRINT #2:CHR$(12);
430 PRINT #2:TEXT$
440 IF EOF(1)=1 THEN 360
450 GOTO 390                   ■
```

STATISTICS

- medelvärde
- standardavvikelse
- medianvärde
- max/min intervall

```
70 REM STATISTICS
80 REM BY BURKE LUTTICH
90 REM COMPUTE 84-07
100 DIM SA(300)
110 CALL CLEAR
120 PRINT TAB(10);"STATISTIC
S": : : :TAB(13);"FOR": : :
:TAB(7);"NON-STATISTICANS":
: : : : : :
180 FOR K=1 TO 400
190 NEXT K
200 CALL CLEAR
210 PRINT "THIS PROGRAM CALC
ULATES THE": : :"FOLOWING VA
LUES FROM DATA": : :"YOU INP
UT:": : :TAB(4);"1. MEAN": :
 :
270 PRINT TAB(4);"2. STANDAR
D DEVIATION": : :TAB(4);"3.
MEDIAN": : :TAB(4);"4. RANGE
": : : :TAB(2);"PRESS ANY KE
Y TO CONTINUE"
350 GOSUB 2170
360 SUM=0
370 MEAN=0
380 DFF=0
390 SDDEV=0
400 RG=0
410 REM INSTR REQUEST
420 PRINT TAB(6);"INSTRUCTIO
NS (Y/N)?": : : : : : : : :
: :
440 GOSUB 2170
450 IF (K<>89)*(K<>78)THEN 4
40
460 IF K=78 THEN 490
470 GOSUB 1330
480 REM DATA ENTRY
490 CALL CLEAR
500 INPUT "  ENTRY SAMPLE SI
ZE ":N
520 IF (N>300)+(N<=1)THEN 49
0
530 CALL CLEAR
540 PRINT "  ENTER YOUR DATA
 ONE VALUE": :"AT A TIME, TH
EN PRESS": :"ENTER.": : : :"
  IF YOU MAKE AN ERROR,": :
580 PRINT "CONTINUE WITH DAT
A ENTRY.": :"YOU WILL BE ABL
E TO MAKE": :"CORRECTIONS LA
TER.": : : : :" PRESS ANY KE
Y TO CONTINUE"
620 GOSUB 2170
630 FOR I=1 TO N
```

```
640 CALL CLEAR
650 PRINT "DATA ENTRY #";I;
660 INPUT R$
670 SA(I)=VAL(R$)
680 NEXT I
690 REM ERROR CORR REQUEST
700 CALL CLEAR
710 PRINT "  ANY CORRECTIONS
 (Y/N) ?": : : : : : : : : :
 :
730 GOSUB 2170
740 IF K<>89 THEN 770
750 GOSUB 1800
760 REM CALC MEAN & STD DEV
770 PRINT TAB(9);"PLEASE WAI
T": : : :"STATISTICS BEING C
ALCULATED": : : : : : : : :
:
800 FOR I=1 TO N
810 SUM=SUM+SA(I)
820 NEXT I
830 MEAN=SUM/N
840 FOR I=1 TO N
850 DFF=DFF+(SA(I)-MEAN)^2
860 NEXT I
870 SDDEV=SQR(DFF/(N-1))
880 REM SORT DATA NUMERIC
890 FL=0
900 FOR I=1 TO N-1
910 IF SA(I)<=SA(I+1)THEN 96
0
920 Q=SA(I)
930 SA(I)=SA(I+1)
940 SA(I+1)=Q
950 FL=1
960 NEXT I
970 IF FL=1 THEN 890
980 REM CALC OF RANGE
990 RG=SA(N)-SA(1)
1000 LR=SA(1)
1010 HR=SA(N)
1020 REM CALC OF MEDIAN
1030 IF N/2<>INT(N/2)THEN 10
90
1040 IF SA(N/2)<>SA(N/2+1)TH
EN 1060
1050 MDD=SA(N/2)
1060 IF SA(N/2)=SA(N/2+1)THE
N 1080
1070 MDD=(SA(N/2)+SA(N/2+1))
/2
1080 GOTO 1110
1090 MDD=SA(INT(N/2+1))
1100 REM PRINT RESULT
1110 CALL CLEAR
1120 PRINT TAB(5);"CALCULATI
ON RESULTS": :"*************
***************": : :"SAMPLE
 SIZE";TAB(19);N: :
1150 PRINT "MEAN (X BAR)";TA
B(19);INT(MEAN*10000+.5)/100
00: :"STD. DEVIATION";TAB(19
);INT(SDDEV*10000+.5)/10000
1170 PRINT :"MEDIAN ";TAB(19
);INT(MDD*10000+.5)/10000: :
"RANGE";TAB(19);INT(RG*10000
+.5)/10000: :
1190 PRINT "LOWEST VALUE";TA
B(19);LR: :"HIGHEST VALUE";T
AB(19);HR: : :"PRESS ANY KEY
"
1220 GOSUB 2170
1230 REM CON OR END
1240 PRINT " WISH TO PROCESS
 MORE DATA": : :TAB(12);"(Y/
N)?": : : : : : : : : :
1260 GOSUB 2170
1270 IF K=78 THEN 1320
1280 FOR I=1 TO N
1290 SA(I)=0
1300 NEXT I
1310 GOTO 360
1320 END
1330 PRINT "  THE MAXIMUM NU
MBER OF EN-": :"TRIES YOU CA
N MAKE IS 300.": :"THE MINIM
UM NUMBER IS 2.": : :"  THE
MEAN IS THE ARITH-": :
1370 PRINT "METIC AVERAGE OF
 THE NUMBERS": :"YOU ENTER."
: : :"  STANDARD DEVIATION I
S A": :"MEASURE OF HOW WIDLE
Y YOUR": :"NUMBERS SPREAD FR
OM THE": :"AVERAGE.": : :
1430 GOSUB 2160
1440 CALL CLEAR
1450 PRINT "  SINCE THE VALU
ES YOU ENTER": :"TEND TO FOR
M A BELL CURVE": :"(NORMAL D
ISTRIBUTION), THE": :"STD. D
EVIATION IS A MEASURE": :
1490 PRINT "OF THE AREA UNDE
R THE BELL": :"CURVE.": : :"
   NO. OF STD.    % AREA":"
   DEV. (+/-)":"   ---------
--    ------": :
1540 PRINT "        1
   68.3":"        2
 95.5":"        3          9
9.7":"        4          99.
9": : :
1580 GOSUB 2160
1590 PRINT "  THE MEDIAN IS
THE VALUE AT": :"THE MID-POI
NT OF YOUR DATA.": : :"  THE
 RANGE IS THE DIF-": :"FEREN
CE BETWEEN YOUR LOWEST": :
1630 PRINT "DATA VALUE AND T
HE HIGHEST.": :"IT IS A QUIC
K-AND-DIRTY": :"ESTIMATE OF
THE SPREAD.": :"STANDARD DEV
IATION IS MORE": :
1670 PRINT "RELIABLE, HOWEVE
R.": : : :"  PRESS ANY KEY T
O START"
1690 GOSUB 2170
1700 RETURN
1710 REM CORR OPTION
1720 GOSUB 2170
1730 IF (K<>67)*(K<>78)*(K<>
81)THEN 1720
1740 FL=0
1750 IF K<>78 THEN 1780
1760 FL=1
1770 GOTO 1980
1780 IF K=81 THEN 770
1790 REM ERROR CORR SUBR
1800 PRINT "REMEMBER INCORRE
CT SAMPLE #": :TAB(11);"(Y/N
) ?": : : : : : : : : :
1820 GOSUB 2170
1830 IF K=78 THEN 1980
1840 INPUT "WHAT IS THE SAMP
LE # ? ":EN$
1850 EN=VAL(EN$)
1860 IF (EN>N)+(EN<1)+(EN<>I
NT(EN))THEN 1840
1870 PRINT : :"SAMPLE";EN;"
  ";"VALUE=";SA(EN): :
1900 INPUT "ENTER YOUR NEW V
ALUE : ":SA(EN)
1920 PRINT : : : : : :TAB(3)
;"ANY MORE CHANGES (Y/N)?":
: : : :
1940 GOSUB 2170
1950 CALL CLEAR
1960 IF K=78 THEN 770
1970 GOTO 1800
1980 IF FL=1 THEN 2020
1990 PRINT "THESE ARE THE FI
RST TEN": :
2000 L=1
2010 GOTO 2040
2020 CALL CLEAR
2030 PRINT "THESE ARE THE NE
XT TEN": :
2040 PRINT "VALUES.": : :TAB
(5);"ENTRY";TAB(15);"VALUES"
: :
2060 FF=0
2070 FOR L=L TO L+9
2080 FF=FF+1
2090 IF L>300 THEN 770
2100 PRINT TAB(5);L;TAB(15);
SA(L)
2110 NEXT L
2120 PRINT : :"C=CHANGE DATA
   N=NEXT TABLE": :TAB(12);"
Q=QUIT"
2150 GOTO 1720
2160 PRINT "  PRESS ANY KEY
FOR MORE";
2170 CALL KEY(3,K,S)
2180 IF S=0 THEN 2170
2190 CALL CLEAR
2200 RETURN
```
■

DIS/VAR 80 TILL BASIC

```
10 OPTION BASE 1 ! XLATE
 JOHN FORD - DEC '84
20 DIM T$(24,12),T(24),G(24,
12),H(17):: DISPLAY AT(10,2)
ERASE ALL:"TEXT TO PROGRAM C
ONVERTER": : :" writes a mer
ge format file  from text."
30 DATA 8,ABS,203,APPEND,249
,ASC,220,ATN,204,AT,240,AND,
187,ALL,236,ACCEPT,164,3,BAS
E,241,BREAK,142,BEEP,238
40 S$=RPT$(" ",80):: GOTO 14
0 :: A$,B$,P$,A,B,C,E,I,L,M,
P,X,Y,Z :: !@P-
50 DATA 4,CHR$,214,CLOSE,160
,COS,205,CALL,157,5,DEF,137,
DELETE,153,DIM,138,DISPLAY,1
62,DIGIT,233,6,ELSE,129
60 DATA END,139,EOF,202,EXP,
206,ERASE,239,ERROR,165,2,FO
R,140,FIXED,250,3,GO,133,GOT
O,134,GOSUB,135,0,4,IF,132
70 DATA INPUT,146,INT,207,IN
TERNAL,245,0,0,4,LEN,213,LET
,141,LOG,208,LINPUT,170,2,MA
X,223,MIN,224,3,NEXT,150
80 DATA NUMERIC,232,NOT,189,
5,ON,155,OPEN,159,OPTION,158
,OUTPUT,247,OR,186,4,PRINT,1
56,POS,217,PERMANENT,251
90 DATA PI,221,0,9,RND,215,R
EAD,151,RETURN,136,RESTORE,1
48,RANDOMIZE,149,REC,222,REL
ATIVE,244,RPT$,225,RUN,169
100 DATA 12,SEG$,216,STR$,21
9,STEP,178,SGN,209,SIN,210,S
QR,211,STOP,152,SEQUENTIAL,2
46,SUB,161,SIZE,235
110 DATA SUBEND,168,SUBEXIT,
167,5,THEN,176,TO,177,TAB,25
2,TAN,212,TRACE,144,5,UNBREA
K,143,UNTRACE,145
120 DATA UPDATE,248,USING,23
7,UALPHA,234,3,VAL,218,VARIA
BLE,243,VALIDATE,254,1,WARNI
NG,166,1,XOR,188
130 DATA 179,180,190,183,182
,253,191,192,193,194,196,195
,197,184
140 FOR X=1 TO 24 :: READ T(
X):: FOR Y=1 TO T(X):: READ
T$(X,Y),G(X,Y):: NEXT Y :: N
EXT X :: FOR X=4 TO 17 :: RE
AD H(X):: NEXT X
150 DISPLAY AT(2,2)ERASE ALL
:"INPUT FILE NAME:": : :" OU
TPUT FILE NAME:"
160 ON ERROR 160 :: ACCEPT A
T(3,2):A$ :: OPEN #1:A$ :: O
N WARNING NEXT
170 ON ERROR 170 :: ACCEPT A
T(6,2):A$ :: OPEN #2:A$,VARI
ABLE 163
180 DISPLAY AT(8,2):"enter 1
 for basic":" enter 2 for ex
tended basic":"    --->2"
190 ON ERROR 630 :: ACCEPT A
T(10,8)SIZE(-1)VALIDATE("12"
):A$ :: IF A$="2" THEN E=1 :
: GOTO 210 ELSE IF A$="" THE
N 180
200 T(1),T(4),T(5),T(15)=4 :
: T(2),T(22)=2 :: T(14)=1 ::
 T(18)=7 :: T(12),T(16),T(21
)=3 :: T(13),T(23),T(24)=0 :
: T(19)=9
210 DISPLAY AT(12,1):"SELECT
 MODE OF OPERATION:B": :"A.
Lines with any character
in the 80th column to be
joined with the next line"
220 DISPLAY AT(18,1):"B. Lin
es regularly sequenced" :: A
CCEPT AT(12,26)VALIDATE("AB"
)SIZE(-1):A$
230 IF A$="" THEN 220 ELSE I
F A$="A" THEN 260 ELSE DISPL
AY AT(19,4):"Enter lineincre
ment:0010"
240 ACCEPT AT(19,25)VALIDATE
(DIGIT)SIZE(-4):I :: IF I=0
THEN 240
250 IF EOF(1)THEN 290 ELSE L
INPUT #1:P$ :: IF P$="" OR P
$=" " THEN 250 ELSE L=VAL(SE
G$(P$,1,POS(P$," ",1)))
260 IF I=0 THEN 290 ELSE IF
EOF(1)THEN I=0 :: A$=P$ :: G
OTO 310 ELSE LINPUT #1:B$ ::
 IF B$="" OR B$=" " THEN 260
270 IF SEG$(B$,1,LEN(STR$(L+
I))+1)=(STR$(L+I)&" ")THEN A
$=P$ :: P$=B$ :: L=L+I :: GO
TO 310
280 P$=SEG$(P$&S$,1,INT((LEN
(P$)-1)/80+1)*80)&B$ :: IF L
EN(P$)=255 THEN DISPLAYAT(24
,1):"* error - line is too l
ong" :: GOTO 260 ELSE 260
290 IF EOF(1)THEN PRINT #2:C
HR$(255);CHR$(255):: CLOSE #
1 :: CLOSE #2 :: STOP ELSE A
$=""
300 LINPUT #1:B$ :: IF B$=""
 OR B$=" " THEN 290 :: A$=A$
&B$ :: IF LEN(B$)=80 THEN 30
0
310 Z=POS(A$," ",1):: A=LEN(
A$):: P,C,M,B=0 :: B$=SEG$(A
$,1,Z):: DISPLAY A$ :: GOSUB
 590
320 P=Z :: IF M>3 THEN PRINT
 #2:CHR$(H(M));:: IF M<>4 TH
EN B=0 :: GOTO 360 ELSE360
330 IF M<>2 THEN B=0 ELSE 36
0
340 IF M=1 THEN B$="" :: GOS
UB 600 :: GOTO 360
350 IF M=3 THEN IF SEG$(A$,P
+1,1)=":" AND E THEN PRINT #
2:CHR$(130);:: P=P+1 :: C=0
ELSE PRINT #2:CHR$(181);
360 IF P>=A THEN PRINT #2:CH
R$(0):: GOTO 260
370 FOR Z=P+1 TO A :: M=POS(
"""" :,;=()#<>+-/*^&",SEG$(A$
,Z,1),1)
380 IF M THEN IF Z=P+1 THEN
320 ELSE 400
390 NEXT Z
400 B$=SEG$(A$,P+1,Z-P-1)::
X=ASC(B$)-64 :: IF X<-18 OR
X>-7 THEN 430
410 IF B AND C=0 AND B<>132
AND B<>222 AND B<>241 AND B<
>149 AND B<>155 AND B<>156 A
ND B<>162 THEN PRINT #2:CHR$
(201);:: GOSUB 590 :: GOTO 3
20
420 PRINT #2:CHR$(200)&CHR$(
LEN(B$))&B$;:: GOTO 320
430 IF B=157 OR B=161 AND E
THEN 420 ELSE IF X<1 OR X>24
 THEN 520
440 FOR Y=1 TO T(X)
450 IF T$(X,Y)=B$ THEN B=G(X
,Y):: PRINT #2:CHR$(B);:: IF
 B=140 OR B=159 THEN C=1 ::
GOTO 320 ELSE IF B=129 THEN
C=0 :: GOTO 320 ELSE 320
460 NEXT Y
470 IF B$="REM" THEN PRINT #
2:CHR$(154)&SEG$(A$,Z,A-Z+1)
&CHR$(0):: GOTO 260
480 IF B$="IMAGE" AND E THEN
 PRINT #2:CHR$(163);ELSE 520
490 X=ASC(SEG$(A$,Z,1)):: IF
 X=32 THEN Z=Z+1 :: GOTO 490
500 IF X=34 THEN B$="" :: P=
Z :: GOSUB 600 :: PRINT #2:C
HR$(0):: GOTO 260
510 PRINT #2:CHR$(200)&CHR$(
A-Z+1)&SEG$(A$,Z,A-Z+1)&CHR$
(0):: GOTO 260
520 IF X=-31 AND E THEN PRIN
T #2:CHR$(131)&SEG$(A$,P+2,A
-P-1)&CHR$(0):: GOTO 260
530 IF B$="DATA" THEN PRINT
#2:CHR$(147);:: P=Z+1 ELSE P
RINT #2:B$;:: GOTO 320
540 IF P>A THEN PRINT #2:CHR
$(0):: GOTO 260 ELSE X=ASC(S
EG$(A$,P,1)):: IF X=32 THEN
P=P+1 :: GOTO 540
```

```
550 IF X=44 THEN PRINT #2:CH
R$(179);:: P=P+1 :: GOTO 540
560 IF X=34 THEN B$="" :: Z=
P :: GOSUB 600 :: P=Z+1 :: G
OTO 540
570 Z=POS(A$,",",P+1):: IF Z
=0 THEN PRINT #2:CHR$(200);C
HR$(A-P+1);SEG$(A$,P,A-P+1);
CHR$(0):: GOTO 260
580 IF SEG$(A$,Z-1,1)=" " TH
EN Z=Z-1 :: GOTO 580 ELSE PR
INT #2:CHR$(200);CHR$(Z-P);S
EG$(A$,P,Z-P);:: P=Z :: GOTO
 540
590 X=VAL(B$):: PRINT #2:CHR
$(INT(X/256))&CHR$(X-256*INT
(X/256));:: RETURN
600 Z=POS(A$,"""",Z+1):: IF
Z=0 THEN DISPLAY BEEP:"*unma
tched quotes":"*quote added
at end of line" :: Z=A+1
610 B$=B$&SEG$(A$,P+1,Z-P-1)
:: IF SEG$(A$,Z+1,1)="""" TH
EN B$=B$&"""" :: P,Z=Z+1 ::
GOTO 600
620 PRINT #2:CHR$(199)&CHR$(
LEN(B$))&B$;:: P=Z :: RETURN
630 DISPLAY BEEP:"*error in
above line":"*processing con
tinuing" :: ON ERROR 630 ::
IF P THEN PRINT #2:CHR$(0)::
 RETURN 260 ELSE RETURN 260
```

Beställ program från
PROGRAMBANKEN.

KATALOG AV MÅNGA SKIVOR

```
100 ! MULTICAT XB NEC
110 ! BY LARRY HUGHES
120 ! FÖR NEC P6
130 ! SKRIVSKYDDADE FILER
140 ! REV JAN ALEXANDERSSON
150 ! 1987-04-07
160 DIM PRG$(300),DSK$(300),
SEK(300),T(300)
170 TYP$(1)="DIS/FIX"
180 TYP$(2)="DIS/VAR"
190 TYP$(3)="INT/FIX"
200 TYP$(4)="INT/VAR"
210 TYP$(5)="PROGRAM"
220 CALL CHAR(91,"0028003844
7C44440028007C4444447C003828
38447C4444")
230 CALL SCREEN(12)
240 DISPLAY AT(2,11)ERASE AL
L:"MULTICAT": :"    AV LARRY
 HUGHES 1982":"      REVIVE
RAT 1987": : :"DETTA PROGRAM
 LÄSER AV":"KATALOGEN FRÅN F
LERA DISKAR,"
250 DISPLAY AT(10,1)BEEP:"SO
RTERAR OCH SKRIVER UT":"UPP
TILL 300 PROGRAM SOM ÄR";"SK
RIVSKYDDADE MED P": : : :"
 STOPPA IN FÖRSTA DISK": :"
     (TRYCK PÅ ENTER)"
260 TOT=0
270 CALL KEY(3,K,S):: IF K<>
13 THEN 270
280 OPEN #1:"DSK1.",INPUT ,R
ELATIVE,INTERNAL
290 INPUT #1:B$,J,J,K
300 FOR LOOP=1 TO 127
310 INPUT #1:A$,A,J,K
320 IF LEN(A$)=0 THEN 380
330 IF A>0 THEN 370
340 TOT=TOT+1
350 IF TOT>300 THEN TOT=300
:: DISPLAY AT(10,2)ERASE ALL
:"300 FILNAMN": :" MINNET FU
LLT" :: GOTO 420
360 PRG$(TOT)=A$ :: DSK$(TOT
)=B$ :: SEK(TOT)=J :: T(TOT)
=ABS(A)
370 NEXT LOOP
380 CLOSE #1
390 DISPLAY AT(8,3)ERASE ALL
:TOT;" FILER (MAX=300)"
400 DISPLAY AT(14,1)BEEP:"(E
NTER)  FÖR NÄSTA DISK": :"
'S'     FÖR SORTERING"
410 CALL KEY(3,K,S):: ON 1-(
K=13)-2*(K=83)GOTO 410,280,4
20
420 DISPLAY AT(23,1):"SORTER
AR..."
430 Y=TOT
440 Y=INT(Y/2):: IF Y=0 THEN
 510
450 Z=TOT-Y :: X=1
460 V=X
470 W=V+Y :: IF PRG$(V)<=PRG
$(W)THEN 500
480 PP$=PRG$(V):: PRG$(V)=PR
G$(W):: PRG$(W)=PP$ :: PP$=D
SK$(V):: DSK$(V)=DSK$(W):: D
SK$(W)=PP$ :: SS=SEK(V):: SE
K(V)=SEK(W):: SEK(W)=SS :: S
S=T(V):: T(V)=T(W):: T(W)=SS
490 V=V-Y :: IF V>=1 THEN 47
0
500 X=X+1 :: IF X>Z THEN 440
 ELSE 460
510 DISPLAY AT(10,1)ERASE AL
L:"PRINTER  PIO" :: ACCEPT A
T(10,10)SIZE(-19)BEEP:FIL$
520 DISPLAY AT(12,1):"KOLUMN
ER 2  (2-3)" :: ACCEPT AT(12
,10)VALIDATE("23")SIZE(-1):K
B
530 DISPLAY AT(14,1):"RUBRIK
" :: ACCEPT AT(14,10)SIZE(19
)BEEP:RUBRIK$
540 DISPLAY AT(16,1):"DATUM"
 :: ACCEPT AT(16,10)SIZE(19)
BEEP:A$
550 IF KB=2 THEN BRED=80 ::
L$=CHR$(27)&"M"&CHR$(27)&"l"
&CHR$(10):: TT=43
560 IF KB=3 THEN BRED=132 ::
 L$=CHR$(15)&CHR$(27)&"P"&CH
R$(27)&"l"&CHR$(12):: TT=62
570 OPEN #2:FIL$,VARIABLE BR
ED
580 PRINT #2:L$
590 PRINT #2: : :CHR$(14);TA
B((TT-LEN(RUBRIK$))/2);RUBRI
K$
600 PRINT #2:CHR$(14);TAB((T
T-LEN(A$))/2);A$: :
610 HUV$=" FILENAMN    DISKN
AMN   SEKT   TYP"
620 FOR L=1 TO KB-1 :: PRINT
 #2:HUV$&RPT$(" ",6+KB);:: N
EXT L :: PRINT #2:HUV$
630 STRECK$="—————————  ——
—————  ———  ——————"
640 FOR L=1 TO KB-1 :: PRINT
 #2:STRECK$&RPT$(" ",3+KB);:
: NEXT L :: PRINT #2:STRECK$
650 I=TOT
660 JJ=(I/KB)-INT(I/KB)
670 IF JJ=0 THEN 690
680 I=I+1 :: GOTO 660
690 JJ=I/KB
700 FOR I=1 TO JJ
710 PRINT #2:PRG$(I);TAB(13)
;DSK$(I);TAB(27-LEN(STR$(SEK
(I))));SEK(I);TAB(31);TYP$(T
(I));
720 K=I+JJ
730 IF K>TOT THEN 790
740 PRINT #2:TAB(41+KB);PRG$
(K);TAB(53+KB);DSK$(K);TAB(6
7+KB-LEN(STR$(SEK(K))));SEK(
K);TAB(71+KB);TYP$(T(K));
750 IF KB=2 THEN 790
760 K=K+JJ
770 IF K>TOT THEN 790
780 PRINT #2:TAB(87);PRG$(K)
;TAB(99);DSK$(K);TAB(113-LEN
(STR$(SEK(K))));SEK(K);TAB(1
17);TYP$(T(K));
790 PRINT #2:
800 NEXT I
810 PRINT #2: : :TAB(TT-6);T
OT;" FILENAMN":CHR$(12)
820 CLOSE #2
830 DISPLAY AT(10,5)BEEP ERA
SE ALL:"FORTSÄTTA  J" :: ACC
EPT AT(10,16)SIZE(-1):A$ ::
IF A$="J" THEN 240
```
■