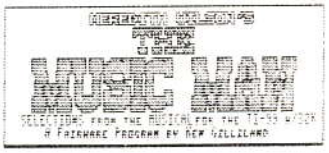
PROGRAM BITEN 94-4



ISSN 0281-1146

REDAKTÖREN

Detta är det sista nummer av Programbiten som jag sammanställer. Styrelsen har föreslagit att föreningen upplöses enligt kallelse till extra stämma som sändes ut på löst blad tillsammans med PB 94-3. Det finns dock andra TI-99/4A tidningar som fortfarande ges ut.

Micropendium, P.O.Box 1343
Round Rock, TX 78680, USA
Prenumerartion för ett år (12 nr) kostar med flygpost 52 dollar.

TI*MES
TI-99/4A User's Group (U.K.)
c/o Alasdair Bryce
51 Dumbuie Ave.
Silverton, Dumbarton
Scotland G82 2JH, Storbritannien
Medlemsavgift 15,50 pund per år och fyra nummer av TI*MES.

Cecure Electronics har nu övertagit alla reparationer i USA av TI-99/4A och CC-40. Cecure reparerar sedan tidigare även Myarc-produkter.
Adressen är:
Cecure Electronics
P.O.Box 132
MUSKEGO, WI 53150, USA
Telefon 414-679-4343

```
************************************
* User POWER-UP with Horizon RAMBO *
* Set the 9938 to 60 Hz            *
* by Jan Alexandersson, Sweden     *
* 1993-12-31                       *
************************************
       DEF  USRPWR   Tell CFG start
       REF  ROSWS    Ask CFG for WS
       REF  BO$OUT   Ask CFG exit
GPLWS  EQU  >83E0    GPL workspace

USRPWR LWPI ROSWS      Workspace
       LI   R0,>0900   60 Hz 9938
       BLWP @VWTR      VDP-register
       LWPI GPLWS      Reload GPL
       B    @BO$OUT    Exit

* Write to VDP-register
VWTR   DATA UTILWS,VWTR0
UTILWS BSS  32
VWTR0  MOV  *R13,R1
       MOVB @>0001(R13),@>8C02
       ORI  R1,>8000
       MOVB R1,@>8C02
       RTWP

       END                          ■
```


Föreningens adress:
Föreningen Programbiten
c/o Schibler
Wahlbergsgatan 9 NB
S-121 38 JOHANNESHOV, Sverige

Postgiro 430 01 59-3
Datainspektionens licens-nr 82100488

Föreningens tillbehörsförsäljning:
Följande tillbehör finns att köpa genom att motsvarande belopp insätts på föreningens postgiro(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:-
90-93 (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.

Redaktör: Jan Alexandersson
Springarvägen 5, 3tr
142 61 TRÅNGSUND
Tel. 08-771 0569

HORIZON RAM-DISK

av Jan Alexandersson

Jag har nu fått tag i en hårdvarumanual till Horizon 4000 och en ny version av operativsystemet ROS 8.14F från 04/22/93. 32 kbytes EM installeras med 32 kbytes för chip M32 samt en 74LS08 för chip U11. RAM-disken kan adressera 2 Mbytes med endast en 74HC154 tillsammans med 128 kbytes minnes-chips.

Phoenix-modifiering för Geneve kan ge två hårdvaru-RAM-diskar på samma kort genom att ROS-chip U9 ändras till 32 kbytes och en bygel flyttas.■

PUTTING IT ALL TOGETHER No.11

by Jim Peterson, Tigercub, USA

The hard part of learning to program is not in learning what the various commands do - it is learning how to put them together to do what you want them to do! Key in this little routine, run it to see what it does, then read the explanation of how it does it.

```
100 DISPLAY AT(12,1):"Input
filename?":"DSK" :: ACCEPT A
T(13,4):IF$
110 DISPLAY AT(15,1):"Output
 filename?":"DSK" :: ACCEPT
AT(16,4):OF$
120 DISPLAY AT(18,1):"Put bl
ank lines between     paragr
aphs? Y/N" :: ACCEPT AT(19,1
7)SIZE(1)VALIDATE("YN"):Q$
130 OPEN #1:"DSK"&IF$,INPUT
:: OPEN #2:"DSK"&OF$,OUTPUT
:: C$=CHR$(13)
140 IF EOF(1)THEN 170 :: LIN
PUT #1:M$ :: IF Q$="Y" THEN
160
150 IF M$="" THEN PRINT #2:C
$:M$;:: GOTO 140 ELSE IF ASC
(M$)<33 THEN PRINT #2:C$:M$;
:: GOTO 140 ELSE PRINT #2:""
:M$;:: GOTO 140
160 IF M$="" OR M$=" " THEN
PRINT #2:C$ :: GOTO 140 ELSE
 IF ASC(M$)<33 THEN PRINT #2
:C$:C$:M$;:: GOTO 140 ELSE P
RINT #2:"":M$;:: GOTO 140
170 PRINT #2:C$ :: CLOSE #1
:: CLOSE #2
```

This program will add carriage returns to a file, such as those which are nowadays being ported over from IBM. However, the file must have indented header lines and indented paragraphs, to give a clue as to where the CRs should be. You are also given the option of putting blank lines between paragraphs.

The first two lines get the name of the file to be worked on, and the filename to be used for the revised file. Note that the ACCEPT AT cursor is placed right after DSK, to make it plain that the input should be a drive number, period and filename.

In line 130 the files are opened and ASCII 13, the carriage return, is defined as C$ so it can be more conveniently referred to hereafter.

In line 140, the EOF end-of-file check is placed before the input, because execution keeps returning here until all the file is read. Records are read in by LINPUT rather than INPUT because if the record contains a string INPUT will stop reading it at that point. If the option to put blank lines between paragraphs was selected, execution jumps to line 160. All the work is done in 150 or 160.

A blank line may be a null string, containing nothing at all, or it may contain a single ASCII 32, the space character, depending on how it was created. In line 150, if the input record is a null string, a CR is printed to place a CR at the end of the previous record, which has always been left open. The colon starts a new record and the null string is printed to it, followed by the semi-colon to hold the record open. If the first character of the record is less than 33 (i.e., the space character 32), it is either a blank line or the first line of an indented paragraph, and the same action is taken. It is a peculiarity of XBasic that this cannot be written as IF M$="" OR ASC(M$)<33 - in spite of the OR, the program will attempt to find the ASCII of a null string and will crash.

If the line is not a null string and does not begin with a blank, it is the second or subsequent line of a paragraph. A null string is printed to close the previous open record, then the record is printed and held open in case it turns out to be the last line of a paragraph and needs to have a CR added next time.

Line 160 is similar. If the record is a null string or a single blank, a CR is printed to close the previous record. If the first character is a blank, the CR is followed by another CR, to place a blank line between paragraphs.

In all cases, execution goes back to line 140 for another input but first checks to see if the end of the file has been reached. In that case it jumps to 170 where a CR is printed to close the final pending record before the files are closed. ■

ALEX-SKIVOR FÖR UTLANDET

av Jan Alexandersson

Jag har översatt de flesta artiklar som jag skrivit i Programbiten till engelska. Skivorna innehåller även de program som jag skrivit. Alla de fem skivor som finns listade nedan kan fås av mig om du sänder in skivor och svarskuvert med frimärke. Observera att vissa skivor är dubbelsidiga (om USED är större än 360 sektorer), så sänd två om du endast kan läsa SS/SD 360 sektorer. Jag har även två skivor PRK- och Statistics-Basic: ALEX/PRK och ALEX/STA. Alla dessa sju skivor innehåller i stort sett allt jag gjort för TI-99/4A bortsett från modifiering av program som andra skrivit.

Först kommer en katalog över dessa skivor och sedan en kopia av README-filen på varje skiva med en kort beskrivning av varje fil.

ALEX/SWE-1 Sectors Used = 717 Free = 3 Filecount 51

Filename	Size	Type	Rec	P	Filename	Size	Type	Rec	P
-README1	21*	Dis/Var	80	P	PRK-MERGE	5	Program		P
AUTOSPRITE	6	Program		P	PRK-PATCH	7	Dis/Var	80	P
AVPC-1	58	Dis/Var	80	P	PRK-PRINT	3	Program		P
AVPC-2	26	Dis/Var	80	P	PRK-TEST	11	Program		P
CALENDAR	7	Dis/Var	80	P	PRK-TEXT0	15	Dis/Var	80	P
CATALOG	3	Program		P	PRK-TEXT1	40	Dis/Var	80	P
CATTEXT	7	Program		P	QUADDISK	33	Dis/Var	80	P
CHARFILTER	3	Dis/Var	163	P	REMOVER	2	Program		P
COINC/XB	2	Program		P	RITAKURVA	8	Program		P
CRU	9	Dis/Var	80	P	RITAKURVAK	6	Program		P
DISK-FILE	10	Dis/Var	80	P	RS232/PIO	28	Dis/Var	80	P
DM-1	24	Dis/Var	80	P	SECTOREDIT	45	Dis/Var	80	P
DM-2	9	Dis/Var	80	P	STA-MERGE	4	Program		P
FORIT/INST	4	Dis/Var	80	P	STA-TEST	6	Program		P
HFDC-1	91	Dis/Var	80	P	STA-TEXT	33	Dis/Var	80	P
HFDC-2	10	Dis/Var	80	P	STYRA-DM99	9	Program		P
HFDC-3	38	Dis/Var	80	P	TML-AUTO	4	Program		P
JOYST	2	Dis/Var	163	P	TML-RITA	5	Program		P
LADDA/XB	17	Program		P	TML-SPRITE	4	Program		P
LISTAORD	5	Program		P	XHI	46	Dis/Var	80	P
MULTISAY	3	Program		P	XHI-ART/G6	2	Program		P
PEEKV/XB	5	Program		P	XHI-LOD/G6	3	Program		P
PRINT-MP/C	4	Dis/Var	80		XHI-LOD/G7	3	Program		P
PRINT-RD/C	4	Dis/Var	80		XHI-MOT/G6	3	Program		P
PRK-CONV	5	Program		P	XHI-SCR/G7	3	Program		P
PRK-FILE	14	Program		P					

ALEX/SWE-2 Sectors Used = 384 Free = 336 Filecount 52

Filename	Size	Type	Rec	P	Filename	Size	Type	Rec	P
-README2	13*	Dis/Var	80		BUBBLE/SEA	2	Dis/Var	80	P
ALLOAD	2	Program		P	BUBBLE/SXB	3	Dis/Var	80	P
ALLOADM	2	Dis/Var	163	P	BUBBLE/SYS	12	Program		P
ALSAVE	6	Dis/Fix	80	P	BUBBLE/TXT	12	Dis/Var	80	P
BLWP-UTIL	14	Dis/Var	80	P	CALL_LINK	26	Dis/Var	80	P
BUBBLE	4	Program		P	GPLLNK	5	Dis/Var	80	P

BUBBLE/ALS	6	Program		P	HYBRID	10	Dis/Var	80	P
BUBBLE/BA	2	Program		P	INIT	10	Dis/Var	80	P
BUBBLE/BH	10	Program		P	INIT/O	6	Dis/Fix	80	P
BUBBLE/FWB	4	Program		P	INIT/S	2	Dis/Var	80	P
BUBBLE/L1	2	Dis/Var	80	P	LINK/BA	2	Program		P
BUBBLE/LA	2	Dis/Var	80	P	LINK/O	4	Dis/Fix	80	P
BUBBLE/LB	2	Dis/Var	80	P	LINK/S	4	Dis/Var	80	P
BUBBLE/O	11	Dis/Fix	80	P	LOADASMBAS	33	Program		P
BUBBLE/OBH	7	Dis/Fix	80	P	MAIN	13	Dis/Var	80	P
BUBBLE/OEA	3	Dis/Fix	80	P	MAIN/BH	6	Dis/Var	80	P
BUBBLE/OXB	6	Dis/Fix	80	P	NTSC/BA	3	Program		P
BUBBLE/PMM	6	Dis/Var	80	P	PROGRAM	25	Dis/Var	80	P
BUBBLE/PRG	28	Dis/Var	80	P	SAVE	13	Dis/Fix	80	P
BUBBLE/R1	3	Program		P	SCRON/BA	2	Program		P
BUBBLE/R2	2	Program		P	SYSTEX	11	Program		P
BUBBLE/RA	3	Program		P	SYSTEXDOC	11	Dis/Var	80	P
BUBBLE/RAG	4	Program		P	VDP-UTIL	5	Dis/Var	80	P
BUBBLE/RB	5	Program		P	VDPREG/BA	2	Program		P
BUBBLE/S	3*	Dis/Var	80	P	VDPREG/O	3	Dis/Fix	80	P
BUBBLE/SBH	3	Dis/Var	80	P	VDPREG/S	4	Dis/Var	80	P

ALEX/SWE-3 Sectors Used = 349 Free = 371 Filecount 46

Filename	Size	Type	Rec	P	Filename	Size	Type	Rec	P
-README3	8	Dis/Var	80	P	G2-TAB-4AG	3	Dis/Var	80	P
BLKVDP	3	Dis/Var	80	P	G2-TAB-4AT	3	Dis/Var	80	P
CRAYON/O	10	Dis/Fix	80	P	G238T	4	Dis/Var	80	P
CRAYON/S	38	Dis/Var	80	P	G24AG	7	Dis/Var	80	P
CRAYON/SCR	7	Program		P	G24AT	10	Dis/Var	80	P
DRAW	7	Dis/Var	80	P	G24AT-REV	10	Dis/Var	80	P
G1-PRG1/O	6	Dis/Fix	80	P	GPLLNK	5	Dis/Var	80	P
G1-PRG1/S	6	Dis/Var	80	P	GRAPHIC1	19	Dis/Var	80	P
G1-PRG2/O	5	Dis/Fix	80	P	GRAPHIC2G	12	Dis/Var	80	P
G1-PRG2/S	6	Dis/Var	80	P	GRAPHIC2T	19	Dis/Var	80	P
G1-PRG3/O	6	Dis/Fix	80	P	KSCAN	3	Dis/Var	80	P
G1-PRG3/S	7	Dis/Var	80	P	MULTICOLOR	7	Dis/Var	80	P
G1-TAB-38	3	Dis/Var	80	P	T1-PRG1/O	6	Dis/Fix	80	P
G1-TAB-4A	3	Dis/Var	80	P	T1-PRG1/S	6	Dis/Var	80	P
G138	4	Dis/Var	80	P	T1-TAB-4A	2	Dis/Var	80	P
G14A	9	Dis/Var	80	P	T14A	6	Dis/Var	80	P
G2-PRG1/O	6	Dis/Fix	80	P	T2	8	Dis/Var	80	P
G2-PRG1/S	6	Dis/Var	80	P	T2-PRG1/O	6	Dis/Fix	80	P
G2-PRG2/O	6	Dis/Fix	80	P	T2-PRG1/S	7	Dis/Var	80	P
G2-PRG2/S	7	Dis/Var	80	P	T2-TAB-38	3	Dis/Var	80	P
G2-PRG3/O	8	Dis/Fix	80	P	TEXT1+2	16	Dis/Var	80	P
G2-PRG3/S	9	Dis/Var	80	P	TSTKEY	6	Dis/Var	80	P
G2-TAB-38T	4	Dis/Var	80	P	VDP-UTIL	5	Dis/Var	80	P

ALEX/SWE-4 Sectors Used = 596 Free = 124 Filecount 37

Filename	Size	Type	Rec	P	Filename	Size	Type	Rec	P
-README4	13*	Dis/Var	80	P	HORIZONSXB	26	Program		P
BA-PROG-11	2	Program		P	HORIZON_P	25	Program		P
BA-PROG-12	2	Program		P	JANSYSTEM2	10*	Dis/Var	80	P
BA-PROG-13	2	Program		P	MULTIPLAN	25	Dis/Var	80	P
BA-PROG-14	2	Program		P	PG-CLK1	4	Program		P

```
BA-PROG-21     2 Program      P    PG-CLK2        3 Program      P
BA-PROG-22     2 Dis/Var 163  P    PG-CLK3/O      5 Dis/Fix   80 P
BA-PROG-23     2 Program      P    PG-CLK3/S     10 Dis/Var   80 P
BA-PROG-31     3 Program      P    PG-CLK3/XB     3 Program      P
BA-PROG-32     3 Program      P    PGRAM-1       33 Dis/Var   80 P
BA-TIPS-1     53 Dis/Var   80 P    PGRAM-2        4 Dis/Var   80 P
BA-TIPS-2     43 Dis/Var   80 P    PGRAM-3       33 Dis/Var   80 P
BA-TIPS-3     28 Dis/Var   80 P    WORD-1        39 Dis/Var   80 P
CLOCK-TIME     3 Program      P    WORD-2       28*Dis/Var   80 P
FILECOMPAR     3 Program      P    XB-1          20 Dis/Var   80 P
HFDC-4         7*Dis/Var    80 P   XB-2          22 Dis/Var   80 P
HORIZON-1     29 Dis/Var   80 P    XB-3           8 Dis/Var   80 P
HORIZON-2     53 Dis/Var   80 P    XB-SUB        11 Program      P
HORIZON-3     33 Dis/Var   80 P
```

ALEX/FORMA Sectors Used = 420 Free = 300 Filecount 33

```
 Filename  Size  Type    Rec P     Filename  Size  Type    Rec P
 --------- ---- ------- --- -     --------- ---- ------- --- -
 -README-FO   12*Dis/Var  80 P    CHR-MANUAL   50 Dis/Var  80 P
 ACCENTS-1    11 Dis/Var  80 P    CHR-NATION    4 Dis/Var  80 P
 ACCENTS-2    10 Dis/Var  80 P    CROSS-TL-1    2 Dis/Var  80 P
 ACCENTS-3E   11 Dis/Var  80 P    CROSS-TL-2    2 Dis/Var  80 P
 ACCENTS-3U    9 Dis/Var  80 P    CROSS-TL-3    2 Dis/Var  80 P
 ACCENTS-4     3 Dis/Var  80 P    CROSS-TL-4    2 Dis/Var  80 P
 ACCTEST-1     5 Dis/Var  80 P    NEWSLETTER   21 Dis/Var  80 P
 ACCTEST-2     5 Dis/Var  80 P    PRINTER      41 Dis/Var  80 P
 ACCTEST-3E    6 Dis/Var  80 P    TAB-EURO      2 Dis/Var  80 P
 ACCTEST-3U    6 Dis/Var  80 P    TAB-EURO-3    2 Dis/Var  80 P
 ACCTEST-4     2 Dis/Var  80 P    TAB-NASTY     2 Dis/Var  80 P
 ASCII         4*Dis/Var  80 P    TAB-USA       2 Dis/Var  80 P
 ASCII/BA      2 Program      P   TAB-USA-3     2 Dis/Var  80 P
 ASCII/ENLA    4 Dis/Var  80 P    TRANSLITER   62 Dis/Var  80 P
 ASCII/SUP     4 Dis/Var  80 P    WORD-1       39 Dis/Var  80 P
 CHR-LIMIT     9 Dis/Var  80 P    WORD-2       28*Dis/Var  80 P
 CHR-MAN/R    52*Dis/Var  80 P
```

ALEX/SWE-1, rev 1993-07-26

AUTOSPRITE	PB 84-4	program basic MM/EA, 32 sprites CALL POKEV
AVPC-1	PB 89-1	review of DIJIT AVPC 80 column card V9938
AVPC-2	PB 89-4, 90-2,3,4	update of DIJIT AVPC, new EPROM, CALL COINC
CALENDAR	PB 92-1	text about julian and gregorian calendar
CATALOG	PB 85-1	program basic, disk catalog, only 3 sectors
CATTTEXT	PB 84-4, 85-3	program basic for MM/EA, dsk cat text mode
CHARFILTER	PB 85-4	XB SUB-program, transliterate char
COINC/XB	PB 90-4	program XB, test of CALL COINC with AVPC
CRU	PB 90-6	list of CRU addresses for cards Micropendium Aug 1992 page 29

DISK-FILE	PB 90-6	DSK.DISKNAME.FILENAME (Micropendium Oct 1994 page 30)
DM-1	PB 91-1	Disk Manager review DM2, DM1000, Myarc DM V, FW DR40 and DR80
DM-2	PB 91-3	Disk Manager 1000 clarification
FORIT/INST	-	Instruction for loading FORIT by Bo Carleoe
HFDC-1	PB 88-4, 89-1	Myarc HFDC hard disk controller review
HFDC-2	-	Update of HFDC review
HFDC-3	PB 90-4	Mixup of sectors with HFDC and DS/DD disks Important bytes in sector zero for Myarc
JOYST	PB 84-4	XB SUB-program replaces JOYST with KEY
LADDA/XB	PB 86-5	Load AL program from XB
LISTAORD	PB 85-1	View DIS/VAR 80 files, counts bytes/lines
MULTISAY	PB 85-3	XB program for Speech Synthesizer
PEEKV/XB	PB 85-4	XB program for PEEKV, not a good solution
PRINT-MP/C	PB 89-4	TI-BASE command file, prints my Microp. list
PRINT-RD/C	PB 89-4	TI-BASE command file, prints Microdex R/D
PRK-CONV	-	program converts PRK file to DIS/VAR 80
PRK-FILE	PB 85-2 (v.1)	program PRK Basic(or STA), database program this version 2 from 1988 is compressed
PRK-MERGE	PB 88-3	program creates MERGE file for PRK-TEST
PRK-PATCH	PB 89-3	patch for 132 columns output with GRAM
PRK-PRINT	-	program prints PRK-Basic list files
PRK-TEST	PB 88-3	program demo for hidden PRK CALLs
PRK-TEXT0	PB 85-2, 4	PRK/Statistics CALL D, A, P, L, S, H, G
PRK-TEXT1	PB 88-3	PRK hidden Basic CALL >04 - >0C (in total 9) Micropendium Oct 1988 page 20
QUADDISK	PB 90-1	Quad Density disks and disk managers Micropendium April 1991 page 27
REMOVER	PB 90-5	program removes TAB-setting from TIWR file
RITAKURVA	PB 85-2	program plots a sine curve in Basic
RITAKURVAK	PB 85-4	the same as above after compression by SMASH
RS232/PIO	PB 90-1	TI and Myarc RS232 card, review
SECTOREDIT	PB 89-4	Sector Editors, review Disk Patch FW, DSKU, Sector One, Hard Master Micropendium June 1993 page 29

STA-MERGE	PB 88-4	program creates MERGE file for STA-TEST
STA-TEST	PB 88-4	program demo for hidden Statistics CALLs
STA-TEXT	PB 88-4	STA hidden Basic CALL >04 - >09 (in total 6)
STYRA-DM99	PB 89-3	Menu program for DM99
TML-AUTO	PB 90-4	The Missing Link program, creates 32 sprites
TML-RITA	PB 90-4	The Missing Link program, plots a sine curve
TML-SPRITE	PB 90-4	The Missing Link program, sprite tests
XHI	PB 89-2, 4	Review of XHI 3.6 for 80 column V9938
XHI-ART/G6	PB 89-2	XHI program converts TI-Artist to Myart G6
XHI-LOD/G6	PB 89-2	XHI program loads Myart G6 picture
XHI-LOD/G7	PB 89-2	XHI program loads Myart G7 picture
XHI-MOT/G6	PB 89-4	XHI program auto motion of sprite
XHI-SCR/G7	PB 89-2	XHI program test of FILSCR in G7 mode

ALEX/SWE-2 rev. 1992-05-03

PB 91-2 Bubble, how to assemble a short AL program with Funnelweb
text file: BUBBLE/TXT
program file: BUBBLE/PRG is the complete program
BUBBLE/S(/O) is the same program but loads the following files:
MAIN + INIT + VDP-UTIL + GPLLNK
BUBBLE/BA is a Basic program that loads BUBBLE/O
BUBBLE/SEA(/OEA) for EA module, REF to subprogram
BUBBLE/SXB(/OXB) for XB module, EQU to subprogram

PB 91-3 Call AL program from Basic and XB
text file: CALL_LINK
program file: LINK/S(/O) is for CALL LINK of BEEP, HONK and POWER
LINK/BA is a Basic program that loads LINK/O
VDPREG/S(/O) is for CALL LINK of VDPREG
VDPREG/BA is a Basic program that loads VDPREG/O and sets screen
SCRON/BA is a Basic program that make CALL SCRON and CALL SCROFF
NTSC/BA, Basic program, sets 50/60 Hz, PAL/NTSC, 192/212 lines

PB 91-4 AL program in PROGRAM format
text file: PROGRAM
program file: BUBBLE is made with EA SAVE from BUBBLE/O
BUBBLE/FWB is made with Funnelweb FSAVE from BUBBLE/O
BUBBLE/RAG is made with RAG Linker from BUBBLE/O
BUBBLE/L1, control file for RAG Linker, creates /R1 and /R2
BUBBLE/R1 + /R2 is made with RAG Linker from BUBBLE/OEA
BUBBLE/LA, control file for RAG Linker, creates BUBBLE/RA
BUBBLE/RA is made with RAG Linker as one single file
INIT/S(/O) is used by BUBBLE/LB to create BUBBLE/RB
BUBBLE/LB, control file for RAG Linker, creates BUBBLE/RB
BUBBLE/RB this transformed BUBBLE/OEA may load from any module
help file: SAVE from EA will create BUBBLE in PROGRAM format

PB 91-5 Hybrid program XB and AL
text file: HYBRID
program file: BUBBLE/ALS is made with ALSAVE
BUBBLE/SYS is made with SYSTEX
BUBBLE/BH is made with LOADASMBAS from BUBBLE/OBH
MAIN/BH + BUBBLE/SBH + BUBBLE/OBH used for LOADASMBAS
help file: ALLOAD + ALLOADM + ALSAVE
SYSTEX + SYSTEXDOC
LOADASMBAS

PB 91-5 Assembler REF and addresses for EA, XB, MM and FW
text file: BLWP-UTIL

PB 91-5 BUBBLE/PMM program listing of BUBBLE for Mini Memory

ALEX/SWE-3 rev. 1992-12-27

GRAPHIC MODES FOR 99/4A(9918A/9929A) AND 9938

PB 91-6
text file: GRAPHIC1

program file: G1-PRG1/S (/O) + G1-TAB-4A + G14A + BLKVP + KSCAN
Print message on the screen for 99/4A in Graphic1 mode

G1-PGR2/S (/O)
Print big characters on the screen for 99/4A in Graphic1 mode

G1-PRG3/S (/O) + G1-TAB-38 + G138
Graphic1 mode with 26.5 rows for 9938

help file: VDP-UTIL + GPLLNK

PB 92-1
text file: TEXT1+2

program file: T1-PRG1/S (/O) + T1-TAB-4A + T14A
Print message on the screen for 99/4A in Text1 mode, 40 column

T2-PRG1/S (/O) + T2-TAB-38 + T2
Text2 mode with 26.5 rows for 9938, 80 column, 4 colours

PB 92-2
text file: GRAPHIC2T

program file: G2-PRG1/S (/O) + G2-TAB-4AT + G24AT
Print message on the screen for 99/4A in Graphic2 mode

G2-PRG2/S (/O) + G2-TAB-38T + G238T + G24AT-REV
Graphic2 mode with 26.5 rows for 9938 with text message

PB 92-3
text file: GRAPHIC2G

program file: G2-PRG3/S (/O) + G2-TAB-4AG + G24AG + DRAW + TSTKEY
Draw graphics for 99/4A in Graphic2 mode

PB 92-4
text file: MULTICOLOR

program file: CRAYON/S (/O) + CRAYON/SCR
From 99'er 82-06

ALEX/SWE-4, rev 1993-08-24

-README4	-	This file
BA-PROG-11	PB 93-1	Test of maximal Basic line length
BA-PROG-12	PB 93-1	Test of garbage collection
BA-PROG-13	PB 93-1	Test of ACCEPT AT with more than 28 characters
BA-PROG-14	PB 93-1	Test of functions with DEF
BA-PROG-21	PB 93-2	Test of loop with POS
BA-PROG-22	PB 93-2	CALL JOYST with arrow keys ESDX
BA-PROG-23	PB 93-2	CALL SOUND for 50 and 60 Hz consoles
BA-PROG-31	PB 88-2	PRINT with 14 digits in Basic
BA-PROG-32	PB 88-2	PRINT with powers up to 127 in Basic
BA-TIPS-1	PB 93-1	Basic and XB Tips: general statements
BA-TIPS-2	PB 93-2	Basic and XB Tips: CALL statements
BA-TIPS-3	PB 88-2	14 decimals, power of 127, line edit, cassette
CLOCK-TIME	PB 92-3	Reads P-GRAM CLOCK and writes to HFDC TIME
FILECOMPAR	PB 92-3	Compare two DIS/VAR 80 files
HFDC-4	PB 93-1,-3	Myarc HFDC: Move SUBDIR DSK file, CRU >1800
HORIZON-1	PB 87,88,89	Review of original Horizon RAM disk
HORIZON-2	PB 92-4	General article about all Horizon version
HORIZON-3	PB 93-1	Review of new Horizon 4000 RAM disk
HORIZONSXB	PB 93-1	Super-XB program which plots HORIZON_P
HORIZON_P	PB 92-4	Picture of Horizon prices 1986-1992
JANSYSTEM2	-	My own TI-99/4A system
MULTIPLAN	PB 93-2	Versions of Multiplan and HFDC, PGRAM and HRD
PG-CLK1	PB 93-2	Reads PGRAM CLOCK as Corcomp and MBP
PG-CLK2	PB 93-2	PGRAM CLOCK time difference, errored readings
PG-CLK3/S/O	PB 93-2	AL-program to read CLOCK twice and compare
PG-CLK3/XB	PB 93-2	Compare PGRAM CLOCK readings with CALL LINK

PGRAM-1	PB 92-3	Review of P-GRAM+
PGRAM-2	PB 92-4	PGRAM with Mini Memory and GRAM Packer
PGRAM-3	PB 93-2	PGRAM CLOCK reading and PGRAM+ pages
WORD-1	PB 90-5	Word Processing, TI-writer, Characters, Margins
WORD-2	PB 93-3	Funnelweb 40 & 80 col Editor 5.00, review
XB-1	PB 93-3	Extended Basic VDP memory
XB-2	PB 93-3	Extended Basic Super SUB programs
XB-3	PB 93-3	Extended Basic references
XB-SUB	PB 93-3	Super SUB programs for Extended Basic

ALEX/FORMA, rev 1993-11-16 Reviewed in Micropendium Oct 1993 p.26

-README-FO	This file
ACCENTS-1	Transliteration Euro-Writer to a 7-bit printer
ACCENTS-2	Transliteration Euro-Writer to an 8-bit printer
ACCENTS-3E	Transliteration FW All Char to 7-bit with Euro-Formatter
ACCENTS-3U	Transliteration FW All Char to 7 bit with USA-Formatter
ACCENTS-4	Transliteration Sweden 7-bit to 8-bit ASCII
ACCTEST-1	Test print with ACCENTS-1
ACCTEST-2	Test print with ACCENTS-2
ACCTEST-3E	Test print with ACCENTS-3E
ACCTEST-3U	Test print with ACCENTS-3U
ACCTEST-4	Test print with ACCENTS-4
ASCII	Character 128-254 listed in one page
ASCII/BA	A Basic program to create the ASCII file with char 128-254
ASCII/ENLA	ASCII file with vertical and horizontal enlargement
ASCII/SUP	ASCII file with superscript, char set different on my printer
CHR-LIMIT	Available characters with FW All Char and 7-bit printer
CHR-MAN/R	Revised version of CHR-MANUAL with names of Greek letters. Check this with your printer. See the PRINTER file about differences.
CHR-MANUAL	Available characters with FW All Char and 8-bit printer
CHR-NATION	Test print of national european characters

CROSS-TL-1	Cross transliterate A-B with USA-Formatter
CROSS-TL-2	Cross transliterate A-B with Euro-Formatter
CROSS-TL-3	Permutate A-B-C with USA-Formatter
CROSS-TL-4	Permutate A-B-C with Euro-Formatter
NEWSLETTER	The front page of the Swedish newsletter PROGRAMBITEN
PRINTER	7 and 8 bit printers, differences in IBM char and national char
TAB-EURO	Euro-TAB with Euro-Writer
TAB-EURO-3	Euro-TAB with Funnelweb Editor 5.00
TAB-NASTY	This text line start with char 252 which cannot print from any formatter, but it will load into the FW Editor 5.00
TAB-USA	USA-TAB with USA TI-Writer or Funnelweb 4.40
TAB-USA-3	USA-TAB with Funnelweb Editor 5.00
TRANSLITER	Text file which describes USA-Formatter, Euro-Formatter, TAB-Line, Transliteration, Backspace char 08
WORD-1	Word processing with TI-99/4A, from PB 90-5
WORD-2	Funnelweb 40 & 80 col Editor 5.00, from PB 93-3 ■

SAMLINGSSKIVA PROGBIT-94

(sänd skiva och svarskuvert med frimärke till redaktören)

PROGBIT-94 Sectors Used = 360 Free = 0 Filecount 29

Filename	Size	Type	Rec	P	Filename	Size	Type	Rec	P
28-80	3	Program		P	HISTO	8	Program		P
AIRDEF	6	Program		P	INTERRUPT	7	Program		P
AIRDEFENCE	22	Program		P	JACKPOT	28	Program		P
BUG-AL-7/O	5	Dis/Fix	80	P	MATRIS	37	Program		P
BUG-AL-9/O	5	Dis/Fix	80	P	MG-LSIZE	4	Program		P
BUG-AL10/O	5	Dis/Fix	80	P	NUMTALK	8	Program		P
BUG-AL11/O	6	Dis/Fix	80	P	POWERUP/O	3	Dis/Fix	80	P
CAT/MYARC	6	Program		P	PRINT/DOC	7	Program		P
CHGE/TO/0	4	Program		P	RADKONV/R	5	Program		P
COLUMNS	5	Program		P	SOUND2	5	Program		P
DIS/ASS	44	Program		P	SPEECH2	2	Program		P
DISPLAY/AT	31	Program		P	TL	2	Dis/Var	163	P
FORCE1	30	Program		P	TRULY/XB	18	Program		P
GOBBLEGOOK	15	Program		P	WORM	27	Program		P
GOBBLER	10	Program		P					■

```
100 REM MULTISAY XB SS
130 CALL CLEAR
140 DIM A$(100),X$(100)
150 FOR I=1 TO 100
160 ACCEPT A$(I):: IF A$(I)=
"" THEN 160
170 IF A$(I)="." THEN M=I-1
:: GOTO 200
180 CALL SPGET(A$(I),X$(I))
190 NEXT I
200 FOR I=1 TO M
210 CALL SAY(,X$(I))
220 NEXT I
230 GOTO 130
```
■

FROM BASIC TO ASSEMBLY -11

by Bob August, Bug News, USA

This month we have a complete program that does something more than put a message on the screen. The program is from Ira McComic's book "Learning TI994/A home computer assembly language programming". This is a good book and if you don't have one, I would suggest that you bye it. The program in this newsletter is a modified version of the program on page 203. We have some new commands that we are using this month.

```
       IDT  'MORSE'
GETKEY SB   @>8374,@>8374
       COC  @KEYMSK,R0
       ANDI R0,>7F00
       SETO R1
       AI   R3,-39
       SLA  R3,1
       SRL  R3,8
       JNC  DOT
DELAY  SRC  R12,15
```

I think those are the only ones that are new to you. Here is what they mean:

The IDT gives your program a name. The name can not be more than 8 characters long and the IDT directive must be the first executable line of code. The IDT directive is optional. If you look at your object code you will find the name at the front of your code.

The SB means SUBTRACT BYTES and this subtracts a byte in the first operand form a byte in the second operand. In our program we simply set the byte at >8374 to zero or CALL KEY(0,K,S) to scan the entire keyboard.

The COC means COMPARE ONES CORRESPONDING and checks to see if both KEYMSK and R0 contain a one.

The ANDI means AND IMMEDIATE and performs a logical and on two operands. This means a one and one equals a one and a one and a zero equals a zero and a zero and a zero equals a zero. In our program we check to see if we have an alphabetic character or not.

Te SETO means SET TO ONE and initializes a word to minus one or >FFFF. In our program we set register zero to -1.

The AI means ADD IMMEDIATE and adds the first operand to the second operand. In our program it subtracts 39 from the contents of register three.

The SLA means SHIFT LEFT ARITHMETIC and shifts the contents of the register left 8 bits. In our program this multiplies the result in register three by two.

The SRL means SHIFT RIGHT LOGICAL and the bits in the register to the right the number of positions of the second operand. In our program we put 8 zeros in the left half of register three.

The JNC means JUMP IF NO CARRY and jumps if the carry status bit is zero. In our program we jump to dot if the status bit is zero meaning that it is not a dash.

The SRC means SHIFT RIGHT CIRCULAR and does just what it says. The right most bit becomes the left most bit and does this the number of time set out in the second operand. In our program this just kills time.

The program is not that hard to figure out so we will keep this short as the the program is long with all the data statements.

Until next month

HAPPY ASSEMBLING! (program next page)

SVENSKA BOKSTÄVER

(repris från PB 85-4.23)

```
8000 CALL CHAR(91,"002800384
47C44440028007C4444447C00382
838447C4444")
8010 CALL CHAR(123,"00002800
38447C44000028007C44447C0000
382838447C44")
```

```
***********************************
* Program to learn Morse Code from *
* page 203 of book by Ira McComic  *
* "LEARNING TI 99/4A HOME COMPUTER *
*  ASSEMBLY LANGUAGE PROGRAMMING"  *
*  in editor assembly environment  *
* Revised by R.W. August May, 1990 *
* for BUG News -  Lesson number 11 *
***********************************
*
       IDT  'MORSE'         Assigns a name to the program
       DEF  START           Defines entry point of program
       REF  KSCAN,SOUND,VSBW,VMBW  Utilities used in program
CR     BYTE >0D             Code value for Enter key
DOTIME EQU  4200            Length of tone
KEYMSK DATA >2000           Key mask
SBUF   BSS  2               Screen buffer
RKEY   BYTE 42,43,60,61,62,64,91,92,93,94  Dummy codes to remove
APOS   TEXT ''''            Lowest number is (' or 39)
UNDER  TEXT '_'             Highest number is(_ or 95)
MESG1  TEXT 'MORSE CODE'
MESG2  TEXT 'TUTORIAL'
MESG3  TEXT 'CODING [ ]'
MESG4  TEXT 'Press the key to hear Code'
MESG5  TEXT 'Press ENTER key to quit'
       EVEN                 Make next byte even
START  LWPI >8300           Load workspace FAST RAM
* Clear screen
       LI   R0,0            Put zero in R0 for location
       LI   R1,>2000        Put space character in R1
       LI   R2,767          Put screen max in R2
       BLWP @VSBW           ($-8) Write a space to the screen
       INC  R0              Add 1 to R0
       DEC  R2              Subtract 1 from R2
       JGT  $-8             Go back to loop if R2 is not zero
* Put Messages on the screen
       BL   @PRINT          Goto print routine
       DATA 330,MESG1,10    Data to print message #1
       BL   @PRINT          Goto print routine
       DATA 395,MESG2,8     Data to print message #2
       BL   @PRINT          Goto print routine
       DATA 458,MESG3,10    Data to print message #3
       BL   @PRINT          Goto print routine
       DATA 547,MESG4,26    Data to print message #4
       BL   @PRINT          Goto print routine
       DATA 611,MESG5,23    Data to print message #5
* Main program
BLANK  BL   @PRINT          Put a blank on screen between [ ]
       DATA 466,>20,1       Data to print blank
* Accept key
GETKEY SB   @>8374,@>8374   Select entire keyboard CALL KEY(0,K,S)
       BLWP @KSCAN          Check for key press
       MOVB @>837C,R0       Read key pressed
       COC  @KEYMSK,R0      Check keyboard status
       JNE  GETKEY          Jump if no key pressed
       MOVB @>8375,R0       Put ASCII code into left byte of R0
       ANDI R0,>7F00        Strip off parity bit
       CB   R0,@APOS        Compare code to "'" (Apostrophe)
       JL   LOWER           Jump if less then 39
       CB   R0,@UNDER       Compare code to '_' (Underline)
       JH   HIGHER          Jump if greater then 95
       SETO R1              Set R1 to -1 or >FFFF
```

```
        INC  R1             ($-14) Increment R1
        CB   R0,@RKEY(R1)   Compare for key to remove
        JEQ  HIGHER         Jump if key compares
        CI   R1,>A          See if we have tested all codes
        JGT  $+4            If so jump to program
        JMP  $-14           If not go back for more
        MOV  R0,R3          ($+4) Copy code to R3 (left side)
        MOV  R0,@SBUF       Move character to sbuf
        BL   @PRINT         Jump to print routine
        DATA 466,SBUF,1     Data to print
        SWPB R3             Put code in right side of R3
        AI   R3,-39         Subtract 39 from it
        SLA  R3,1           Multiply the result in R3 by 2
        MOV  @CODE(R3),R4   Put table entry in R4
        MOV  R4,R3          Copy it to R3
        SRL  R3,8           Right justify element count
* Make dash sound
DASH    LI   R10,>9100      Turn on sound
        MOVB R10,@SOUND     Make a tone
        CLR  R2             Put a zero in R2
        SRL  R4,1           Shift next element into carry
        JNC  DOT            Jump to dot if not a dash
        AI   R2,DOTIME*2    Add delay for dash
* Make dot sound
DOT     AI   R2,DOTIME      Add delay for dot
        BL   @DELAY         Get delay and end tone
        LI   R2,DOTIME      Get inter-element delay time
        BL   @DELAY         Delay after element
        DEC  R3             Subtract 1 from element count
        JNE  DASH           Jump to dash if more to go
        JMP  BLANK          GO back to main program
* Check for enter key
LOWER   CB   R0,@CR         Is it ASCII 13
        JNE  HIGHER         If yes go on else jump to higher
        BLWP @0             Go to power up screen
* Error routine
HIGHER  LI   R10,>F400      Turn on error tone
        MOVB R10,@SOUND     Make a noise
        LI   R2,DOTIME*2    Set delay time
        BL   @DELAY         Gosub delay and turn off noise
        JMP  BLANK          Go back to main program
* Put it on the screen
PRINT   MOV  *R11+,R0       Screen location
        MOV  *R11+,R1       Message to write
        MOV  *R11+,R2       Length of message
        BLWP @VMBW          Put it on the screen
        RT                  Return
* Delay routine
DELAY   SRC  R12,15         Kill time
        DEC  R2             Subtract 1 from R2 for count
        JNE  DELAY          Jump to delay if not zero
        LI   R10,>9FFF      Turn off
        MOVB R10,@SOUND     Tone
        SWPB R10            Turn off
        MOVB R10,@SOUND     noise
        B    *R11           Return to calling routine
* Translation look-up table
*           Hex #       ASCII  Code
CODE    DATA >061E      39 ' = .----.
        DATA >062D      40 ( = -.--.-    >06 = number of elements in code
        DATA >062D      41 ) = -.--.-    >2D = 101101 = 45 or >2D
        DATA >0000      42 * dummy code       101101 = 32+8+4+1
```

```
        DATA >0000      43 + dummy code
        DATA >0633      44 , = --..--     >33 = 110011 = 51 or >33
        DATA >0621      45 - = -....-     >21 = 100001 = 33 or >21
        DATA >062A      46 . = .-.-.-     >2A = 101010 = 42 or >2A
        DATA >0509      47 / = -..-.      >09 = 01001  =  9 or >09
        DATA >051F      48 0 = -----      >1F = 11111  = 31 or >1F
        DATA >051E      49 1 = .----      >1E = 11110  = 30 or >1E
        DATA >051C      50 2 = ..---      >1C = 11100  = 28 or >1C
        DATA >0518      51 3 = ...--      >18 = 11000  = 24 or >10
        DATA >0510      52 4 = ....-      >10 = 10000  = 16 or >10
        DATA >0500      53 5 = .....      >00 = 00000  =  0 or >00
        DATA >0501      54 6 = -....      >01 = 00001  =  1 or >01
        DATA >0503      55 7 = --...      >03 = 00011  =  3 or >03
        DATA >0507      56 8 = ---..      >07 = 00111  =  7 or >07
        DATA >050F      57 9 = ----.      >0F = 01111  = 15 or >0F
        DATA >0607      58 : = ---...     >07 = 000111 =  7 or >07
        DATA >0615      59 ; = -.-.-.     >15 = 010101 = 21 or >15
        DATA >0000      60 < dummy code
        DATA >0000      61 = dummy code
        DATA >0000      62 > dummy code
        DATA >060C      63 ? = ..--..     >0C = 001100 = 12 or >0C
        DATA >0000      64 @ dummy code
        DATA >0202      65 A = .-
        DATA >0401      66 B = -...       >01 = 0001   =  1 or >01
        DATA >0405      67 C = -.-.
        DATA >0301      68 D = -..
        DATA >0100      69 E = .
        DATA >0404      70 F = ..-.
        DATA >0303      71 G = --.
        DATA >0400      72 H = ....
        DATA >0200      73 I = ..
        DATA >040E      74 J = .---
        DATA >0305      75 K = -.-
        DATA >0402      76 L = .-..
        DATA >0203      77 M = --
        DATA >0201      78 N = -.
        DATA >0307      79 O = ---
        DATA >0406      80 P = .--.
        DATA >040B      81 Q = --.-
        DATA >0302      82 R = .-.
        DATA >0300      83 S = ...
        DATA >0101      84 T = -
        DATA >0304      85 U = ..-
        DATA >0408      86 V = ...-
        DATA >0306      87 W = .--
        DATA >0409      88 X = -..-
        DATA >040D      89 Y = -.--
        DATA >0403      90 Z = --..
        DATA >0000      91 [ dummy code
        DATA >0000      92 \ dummy code
        DATA >0000      93 ] dummy code
        DATA >0000      94 ^ dummy code
        DATA >062C      95 _ = ..--.-     >2C = 101100 = 44 or >2C
* End with auto start
        END  START                                                    ■
```

SVERIGES STÄDER

Programmet på sid 17 och 18 innehåller ASCII-koder mellan 128 och 143. Dessa kan knappas in i XB men syns först efter det att programmet körs. Rad 710 och 720. PB 92-4.03 hjälper att hitta ASCII-kod.■

SVERIGES STÄDER

av Sören Bernle

(Se även förklaring sid 16)

```
110 CALL CLEAR
120 CALL SCREEN(1)
130 GOTO 150 :: R,C,FEL,SPO,
Q,W,ST$,MI$,T$,CH$,NAM$,O,PO
,X,DEL,K,S,A$,TE,RU,MI,ST,I
140 CALL COLOR :: CALL HCHAR
 :: CALL SOUND :: CALL KEY :
: CALL SPRITE :: CALL PATTER
N :: CALL LOCATE :: CALL MAG
NIFY
150 CALL CHAR(39,RPT$("0",16
)):: CALL SOUND(400,622,0)::
 CALL CHAR(40,RPT$("F",16))
160 CALL SOUND(800,622,0)
170 FOR I=1 TO 8 :: CALL COL
OR(I,2,5):: NEXT I
180 CALL SOUND(400,494,0)
190 CALL COLOR(2,12,1):: CAL
L SOUND(400,494,0):: CALL CO
LOR(9,5,1)
200 CALL SOUND(800,494,0)
210 CALL HCHAR(1,1,39,768)
220 CALL SOUND(400,554,0)
230 CALL HCHAR(11,1,40,128):
: CALL VCHAR(1,12,40,72)
240 CALL SOUND(400,622,0)
250 DIM L$(132)
260 CALL CHAR(59,"000000007C
")
270 CALL SOUND(800,622,0)
280 CALL CHAR(64,"0010101010
100010")
290 CALL SOUND(400,554,0)
300 CALL CHAR(91,"0044384444
7C444400447C444444447C003828
38447C4444FE8188887E09897E")
310 CALL SOUND(400,494,0)
320 !@P-
325 CALL SOUND(800,466,0)
330 DISPLAY AT(3,1):"SVERIGE
S ((((" :: DISPLAY AT(5,1):"S
TÄDER  ((((" :: DISPLAY AT(7
,1):"AR 1969  ((("
335 CALL SOUND(400,20000,29)
340 DISPLAY AT(16,15):"PROGR
AM AV ^" :: DISPLAY AT(18,15
):"SÖREN BERNLE" :: DISPLAY
AT(20,15):"PL 280" :: DISPLA
Y AT(22,15):"260 38 KATTARP"
345 CALL SOUND(400,554,0)
347 CALL SOUND(800,554,0)
350 FOR I=9 TO 14
360 CALL COLOR(I,12,1)
370 NEXT I
375 CALL SOUND(400,466,0)
380 CALL CHAR(47,"3F3FBFFFFF
FFFFFF")
385 CALL SOUND(400,494,0)
390 CALL CHAR(43,"0303010101
0303031F1F3F7F7F3F7FFF0F1F0F
0F07070F1F0000010303070707")
395 CALL SOUND(400,554,0)
400 CALL CHAR(35,"0303030707
1F1F1F0707070303010103 0F0703
010000010 10F0707070707070F")
405 CALL SOUND(400,466,0)
410 CALL CHAR(140,"2F0F0F0F0
70703035F5F3F3F3F1F1F1FFCFCF
EFCFDB91F5F3F3F1F1F1F3FBEBE"
)
415 CALL SOUND(400,622,0)
420 CALL CHAR(136,"FFFFFFFFF
FFF8142FFFEFE7EFFFF720001010
0000001000001000003050203 03"
)
425 CALL SOUND(400,494,0)
430 CALL CHAR(132,"E0E0A0C0E
0C0E0E0383030102000000C8C8C
8C8C8D090B0B0B0A02020000000"
)
435 CALL SOUND(1600,415,0)
440 CALL CHAR(128,"8830F0D0E
0A040008901F8FFFFFDFDFDFCF00
0E080E0E0E003061C1C18383C38"
)
450 CALL CHAR(124,"E8C0E0E0E
0E0E0E0E0F0F0E0FAFEFFFF00000
000004000C0D0F0F0F8F0F8E0E0"
)
460 CALL CHAR(120,"E0C0C0808
0800020FEFCF8E8E0C04000FFFEF
7FAF8FCFCE8E0E0F8F0F0E0E0F0"
)
470 CALL CHAR(116,"FCFEFEFFF
FFFFFFF00000080C0C0E040F6F0D
0E0E0804080C0C08080008080C0"
)
480 CALL CHAR(112,"FFBF3F3F3
F7FFFFFFFFEFFE0F9FFFFF00000
080E0F0F0F8FCF8F8FCFCFCFEFE"
)
485 CALL SOUND(800,370,0)
490 CALL CHAR(108,"F9FDFDFFF
FFFFFFFFCFCFDFDFFFFFFFFFFFFE
FA783874145170 75FBEFEFEFCFC"
)
500 CALL CHAR(104,"FE8282828
282FE00FFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFBBFFFFFFFFFFFFFFF3"
)
510 CALL CHAR(100,"FFFEFFFFF
FFFFFFF0000001000000000000003
82838000000007C4444447C0000"
)
520 CALL CHAR(96,"0000000020
3E3F3F70787EFF7F7FFFFF000000
000000E060FF7F3FFFFFFFFFFF")
530 CALL SPRITE(#1,100,2,200
,1)
540 DISPLAY AT(4,15):"DITT F
ÖRNAMN:" :: ACCEPT AT(7,15)V
ALIDATE(UALPHA,"ÅÄÖ")SIZE(14
)BEEP:NAM$
550 DISPLAY AT(17,1):"INSTRU
K? ((((" :: DISPLAY AT(19,1):
"<J>=JA  ((((" :: DISPLAY AT
(21,1):"<N>=NEJ  ((("
560 CALL KEY(0,K,S):: IF K=7
8 THEN 570 :: IF K<>74 THEN
560
570 CALL CLEAR :: CALL SCREE
N(5):: CALL CHAR(39,"0F07070
707070F0F010303070707070F1F0
F070707070FFF0303030707070F0
F")
580 CALL COLOR(1,12,1):: CAL
L COLOR(2,12,1)
590 IF K=78 THEN 670
600 CALL CLEAR :: PRINT "PRO
GRAMMET BYGGER PÅ 1969  ÅRS
 FOLKRÄKNING OCH BESTÅR  AV
RIKETS ALLA STÄDER"
610 PRINT :: PRINT "STÄDERNA
 ÄR RANGORDNADE FRÅNSTÖRSTA
<1> ; MINSTA <132>  STORSTAD
 = f ANNAN STAD = e 4:A STÄD
ER TÄTT NORR STHLM"
620 PRINT "MARKERAS MED NAMN
ETS LÄNGD" :: PRINT :: PRINT
 "OBS SKANÖR MED FALSTERBO"
:: PRINT :: PRINT "RÄTT 1:A
SVAR=  3 POÄNG"
630 PRINT "RÄTT 2:A SVAR=  1
 POÄNG     RÄTT 3:E SVAR=  0
 POÄNG     FEL  3:E SVAR= ;3
 POÄNG  OCH" :: PRINT "
            SVAR VISAS"
640 PRINT :: PRINT "SPELET Ä
R SLUT OM DATORN HARGETT DIG
 SVARET PÅ 2 STÄDER" :: PRIN
T :: PRINT "I SLUTET UTDELAS
 BETYG@" :: PRINT
650 DISPLAY AT(24,1):"FÖRSTÅ
TT? TRYCK DÅ PÅ <F>  ^"
660 CALL KEY(0,K,S):: IF K<>
70 THEN 660
670 CALL CLEAR
680 DISPLAY AT(1,6):"b" :: D
ISPLAY AT(2,5):"`ar" :: DISP
LAY AT(3,4):"./is" :: DISPLA
Y AT(4,4):",iit" :: DISPLAY
AT(5,4):",iiiu"
690 DISPLAY AT(6,3):"+iiiv"
:: DISPLAY AT(7,3):"*iiiw" :
: DISPLAY AT(8,3):")iiix" ::
 DISPLAY AT(9,2):"(kiiy" ::
DISPLAY AT(10,2):"'liz"
700 DISPLAY AT(11,2):"&iiä"
:: DISPLAY AT(12,2):"%iiö" :
: DISPLAY AT(13,2):"$dcå~" :
: DISPLAY AT(14,2):"#iij"&CH
R$(127)
710 DISPLAY AT(15,2):"Ängü"&
```

```
CHR$(128):: DISPLAY AT(16,2)
:"Äopé" :: DISPLAY AT(17,2):
"ìmiää" :: DISPLAY AT(18,2):
"îiiåå" :: DISPLAY AT(19,2):
"ïiêç"
720 DISPLAY AT(20,2):"èë"
730 DISPLAY AT(1,8):"SVERIGE
S STÄDER 1969"
740 DISPLAY AT(13,9):NAM$ ::
 DISPLAY AT(15,9):"VÄLJ DET
INTERVALL" :: DISPLAY AT(17,
9):"DU VILL TÄVLA INOM"
750 DISPLAY AT(3,28):"^"
760 GOTO 2100
770 L$(1)="STOCKHOLM,112,54"
780 L$(2)="GÖTEBORG,130,24"
790 L$(3)="MALMÖ,154,30"
800 L$(4)="VÄSTERÅS,109,46"
810 L$(5)="UPPSALA,106,52"
820 L$(6)="NORRKÖPING,120,45
"
830 L$(7)="ÖREBRO,113,40"
840 L$(8)="HELSINGBORG,149,2
8"
850 L$(9)="LINKÖPING,122,43"
860 L$(11)="BORÅS,130,29"
870 L$(12)="ESKILSTUNA,113,4
7"
880 L$(14)="SKELLEFTEÅ,50,61
"
890 L$(13)="SUNDSVALL,78,48"
900 L$(10)="GÄVLE,97,49"
910 L$(18)="JÖNKÖPING,130,35
5"
920 L$(21)="KARLSTAD,112,32"
930 L$(19)="UMEÅ,61,60"
940 L$(15)="SÖDERTÄLJE,113,5
2"
950 L$(20)="LUND,153,32"
960 L$(22)="HALMSTAD,142,29"
970 L$(24)="KRISTIANSTAD,149
,36"
980 L$(23)="TROLLHÄTTAN,124,
27"
990 L$(25)="KARLSKOGA,112,37
"
1000 L$(29)="KARLSKRONA,147,
44"
1010 L$(26)="KALMAR,142,46"
1020 L$(27)="UDDEVALLA,123,2
5"
1030 L$(16)="LULEÅ,40,65"
1040 L$(30)="TRELLEBORG,156,
31"
1050 L$(32)="FALUN,99,43"
1060 L$(34)="LANDSKRONA,151,
29"
1070 L$(31)="VÄXJÖ,140,39"
1080 L$(36)="PITEÅ,44,63"
1090 L$(37)="NYKÖPING,118,49
"
1100 L$(38)="KARLSHAMN,147,4
0"
1110 L$(44)="AVESTA,103,45"
1120 L$(40)="RONNEBY,147,42"
1130 L$(43)="KIRUNA,17,56"
1140 L$(39)="BORLÄNGE,100,41
"
1150 L$(41)="MOTALA,121,40"
1160 L$(42)="SKÖVDE,123,34"
1170 L$(47)="ÖSTERSUND,70,37
"
1180 L$(50)="OSKARSHAMN,135,
47"
1190 L$(49)="SANDVIKEN,98,47
"
1200 L$(51)="BODEN,38,62"
1210 L$(52)="VÄSTERVIK,130,4
8"
1220 L$(57)="LUDVIKA,104,40"
1230 L$(54)="KRISTINEHAMN,11
3,35"
1240 L$(56)="KATRINEHOLM,117
,45"
1250 L$(55)="KÖPING,110,44"
1260 L$(59)="NÄSSJÖ,131,38"
1270 L$(58)="VÄNERSBORG,123,
26"
1280 L$(60)="ALINGSÅS,128,28
"
1290 L$(33)="LIDKÖPING,122,3
1"
1300 L$(64)="TRANÅS,127,40"
1310 L$(61)="VARBERG,137,26"
1320 L$(65)="HUSKVARNA,130,3
7"
1330 L$(62)="VISBY,130,56"
1340 L$(68)="HEDEMORA,102,44
"
1350 L$(63)="ENKÖPING,109,49
"
1360 L$(46)="HÄRNÖSAND,76,51
"
1370 L$(67)="BOLLNÄS,90,45"
1380 L$(69)="FAGERSTA,106,43
"
1390 L$(71)="ÖRNSKÖLDSVIK,68
,53"
1400 L$(77)="FALKÖPING,126,3
2"
1410 L$(76)="ARVIKA,108,29"
1420 L$(75)="HUDIKSVALL,86,4
8"
1430 L$(73)="HÄSSLEHOLM,148,
34"
1440 L$(72)="VÄRNAMO,136,35"
1450 L$(70)="MARIESTAD,120,3
4"
1460 L$(78)="KUMLA,115,40"
1470 L$(66)="ESLÖV,151,32"
1480 L$(79)="OXELÖSUND,119,4
9"
1490 L$(81)="YSTAD,155,34"
1500 L$(82)="ÄNGELHOLM,147,3
0"
1510 L$(84)="SÖDERHAMN,91,48
"
1520 L$(80)="HÖGANÄS,147,27"
1530 L$(85)="FALKENBERG,140,
27"
1540 L$(86)="MJÖLBY,124,41"
1550 L$(88)="SÄFFLE,114,29"
1560 L$(91)="ARBOGA,111,43"
1570 L$(95)="KRAMFORS,74,50"
1580 L$(93)="SALA,105,47"
1590 L$(90)="LJUNGBY,140,35"
1600 L$(87)="NORRTÄLJE,106,5
7"
1610 L$(92)="KUNGÄLV,128,25"
1620 L$(97)="NYNÄSHAMN,117,5
3"
1630 L$(53)="NYBRO,141,45"
1640 L$(96)="VETLANDA,134,40
"
1650 L$(98)="SKARA,123,32"
1660 L$(102)="STRÖMSTAD,117,
22"
1670 L$(99)="EKSJÖ,131,40"
1680 L$(101)="SOLLEFTEÅ,70,4
8"
1690 L$(106)="HAPARANDA,36,7
2"
1700 L$(103)="ÅMÅL,116,28"
1710 L$(105)="NORA,111,39"
1720 L$(100)="STRÄNGNÄS,112,
49"
1730 L$(104)="FLEN,115,48"
1740 L$(107)="HAGFORS,105,33
"
1750 L$(108)="ÖSTHAMMAR,101,
55"
1760 L$(110)="ULRICEHAMN,130
,32"
1770 L$(74)="SIMRISHAMN,154,
37"
1780 L$(111)="LYSEKIL,124,22
"
1790 L$(112)="FILIPSTAD,108,
35"
1800 L$(109)="TORSHÄLLA,112,
47"
1810 L$(113)="TIDAHOLM,125,3
4"
1820 L$(114)="VIMMERBY,131,4
4"
1830 L$(83)="LINDESBERG,110,
41"
1840 L$(117)="VADSTENA,123,3
9"
1850 L$(94)="KUNGSBACKA,133,
25"
1860 L$(115)="SÖLVESBORG,149
,38"
1870 L$(118)="LYCKSELE,53,54
"
1880 L$(119)="SÖDERKÖPING,12
```

```
2,46"
1890 L$(120)="SÄVSJÖ,134,38"
1900 L$(122)="HJO,124,36"
1910 L$(123)="SÄTER,101,43"
1920 L$(121)="VAXHOLM,110,56
"
1930 L$(124)="SKÄNNINGE,124,
39"
1940 L$(127)="ASKERSUND,118,
39"
1950 L$(125)="SIGTUNA,108,53
"
1960 L$(126)="LAHOLM,144,31"
1970 L$(128)="GRÄNNA,127,38"
1980 L$(130)="MARIEFRED,113,
50"
1990 L$(116)="BORGHOLM,139,4
9"
2000 L$(129)="SKANÖR MED FAL
STERBO,155,29"
2010 L$(131)="TROSA,117,51"
2020 L$(132)="MARSTRAND,129,
22"
2030 L$(17)="SOLNA,111,54"
2040 L$(28)="LIDINGÖ,111,55"
2050 L$(35)="MÖLNDAL,131,25"
2060 L$(45)="SUNDBYBERG,111,
53"
2070 L$(48)="NACKA,113,55"
2080 L$(89)="DJURSHOLM,110,5
4"
2090 RETURN
2100 GOSUB 3050
2110 K=0
2120 DISPLAY AT(22,1):"STÖRS
TA STAD="
2130 ACCEPT AT(22,14)BEEP VA
LIDATE(DIGIT)SIZE(3):ST
2140 IF ST>132 OR ST<1 THEN
2130
2150 DISPLAY AT(24,1):"MINST
A STAD="
2160 ACCEPT AT(24,13)BEEP VA
LIDATE(DIGIT)SIZE(3):MI
2170 IF MI>132 OR MI<ST THEN
 2160 :: IF MI<1 THEN 2160
2180 GOSUB 770
2190 IF ST>2 AND ST<21 THEN
2210
2200 ST$=SEG$(STR$(ST),LEN(S
TR$(ST)),1):: IF ST$="1" OR
ST$="2" THEN ST$=STR$(ST)&":
A" :: GOTO 2220
2210 ST$=STR$(ST)&":E"
2220 IF MI>2 AND MI<21 THEN
2240
2230 MI$=SEG$(STR$(MI),LEN(S
TR$(MI)),1):: IF MI$="1" OR
MI$="2" THEN MI$=STR$(MI)&":
A" :: GOTO 2250
2240 MI$=STR$(MI)&":E"
2250 IF MI-ST+1>1 THEN 2280
2260 DISPLAY AT(22,1):"  DEN
 "&ST$&" STÖRSTA STADEN" ::
DISPLAY AT(23,1):RPT$(" ",20
):: DISPLAY AT(24,1):"  BLIN
KAR NU PÅ KARTAN@"
2270 GOTO 2300
2280 DISPLAY AT(22,1):STR$(M
I-ST+1)&" STÄDER: DEN "&ST$&
" ; "&MI$ :: DISPLAY AT(23,1
):"STÖRSTA STADEN KOMMER NU"
2290 DISPLAY AT(24,1):"SLUMP
VIS UPP PÅ KARTAN@      "
2300 GOSUB 3050
2310 RANDOMIZE
2320 DISPLAY AT(15,9):RPT$("
 ",11):: DISPLAY AT(17,9):RP
T$(" ",20)
2330 DISPLAY AT(15,9):"DU HA
R 0   " :: DISPLAY AT(17,9):
"AV "&STR$((MI-ST+1)*3)&" MÖ
JLIGA POÄNG"
2340 FOR Q=MI-ST+1 TO 1 STEP
 -1
2350 W=INT(RND*Q+ST)
2360 T$=SEG$(L$(W),1,POS(L$(
W),",",1)-1)
2370 O=LEN(T$)+2
2380 R=VAL(SEG$(L$(W),O,POS(
L$(W),",",O)-O))
2390 C=VAL(SEG$(L$(W),LEN(ST
R$(R))+1+O,2))
2400 GOSUB 2530
2410 L$(W)=L$(Q+ST-1)
2420 NEXT Q
2430 FOR DEL=110 TO 990 STEP
 11 :: CALL SOUND(-10,DEL,3)
:: NEXT DEL
2440 GOSUB 3010
2450 CALL LOCATE(#1,200,3)
2460 DISPLAY AT(22,1):"SAMMA
 OMGÅNG IGEN = <I>     VÄLJA
 NY OMGÅNG = <V>       SLUTA
 SPELA = <S>           "
2470 CALL KEY(0,K,S):: IF S=
0 THEN 2470
2480 IF K=73 OR(K=86 OR K=83
)THEN 2490 ELSE 2470
2490 CALL SOUND(50,440,0)::
CALL SOUND(1,44000,30)
2500 IF K=73 THEN 2180
2510 IF K=86 THEN 2100
2520 RUN "DSK1.MENY"
2530 CALL LOCATE(#1,R,C)
2540 PO=3
2550 RU=2
2560 IF T$="SOLNA" OR T$="LI
DINGÖ" OR T$="SUNDBYBERG" OR
 T$="DJURSHOLM" THEN 2570 EL
SE 2580
2570 DISPLAY AT(8,9):RPT$("h
",LEN(T$))&" <"&STR$(LEN(T$)
)&">"
2580 CALL MAGNIFY(RU):: CALL
 LOCATE(#1,R-4,C-4)
2590 FOR I=102+RU TO 100+RU*
1.4 STEP -1
2600 CALL PATTERN(#1,I)
2610 CALL SOUND(-50,4000,5)
2620 FOR DEL=1 TO 100
2630 NEXT DEL
2640 NEXT I
2650 IF RU=1 OR T$="STOCKHOL
M" OR T$="GÖTEBORG" OR T$="M
ALMÖ" THEN 2700 ELSE RU=1
2660 CALL LOCATE(#1,200,C)
2670 CALL MAGNIFY(RU)
2680 CALL LOCATE(#1,R,C)
2690 GOTO 2590
2700 CALL PATTERN(#1,32)
2710 CALL PATTERN(#1,101)
2720 CALL KEY(0,K,S):: IF S=
0 THEN 2700
2730 ACCEPT AT(8,9)VALIDATE(
UALPHA,"ÅÖÄ")BEEP:A$
2740 IF A$="" THEN 2550
2750 IF T$=A$ THEN 2880
2760 CALL SOUND(-75,222,4,-4
,4)
2770 DISPLAY AT(8,9):"iiii F
 E L iiii "
2780 IF PO=-1 THEN 2810
2790 PO=PO-2
2800 GOTO 2550
2810 PO=0
2820 CALL SOUND(-75,444,4,-2
,4)
2830 FEL=FEL+1
2840 SPO=SPO-3
2850 IF SPO<0 THEN SPO=0
2860 DISPLAY AT(8,9):T$
2870 GOTO 2920
2880 IF PO<>-1 THEN 2900
2890 PO=0
2900 SPO=SPO+PO
2910 CALL SOUND(-100,3520,0,
-8,8)
2920 DISPLAY AT(15,16):USING
 "###":SPO
2930 IF FEL=2 THEN 2950
2940 RETURN
2950 FOR DEL=1100 TO 220 STE
P -20 :: CALL SOUND(-5,DEL,5
,DEL*2,5,DEL*4,5,-8,9):: NEX
T DEL
2960 DISPLAY AT(8,9)SIZE(20)
:NAM$
2970 IF FEL<2 THEN 3010
2980 DISPLAY AT(9,9):"DU FÅR
 BAKLÄXA@ "
2990 CALL SOUND(175,110,0,13
0,0,211,0,-8,0)
3000 GOTO 2460
3010 ST$=STR$(INT(10*SPO/((M
I-ST+1)*3)))
```

```
3020 DISPLAY AT(8,9):NAM$ ::
 DISPLAY AT(9,9):"DU FÅR EN"
 :: DISPLAY AT(10,9):ST$&":A
 I BETYG@ "
3030 FOR DEL=1 TO 5 :: CALL
SOUND(-75,110,0,115,0,120,0,
-4,0):: CALL SOUND(-100,128,
0,199,0,137,0,-5,0):: NEXT D
EL
3040 RETURN
3050 PO,SPO,FEL=0 :: CALL HC
HAR(8,11,32,20):: CALL HCHAR
(9,11,32,16):: CALL HCHAR(10
,11,32,13)
3060 IF K=86 THEN 3080
3070 RETURN
3080 CALL HCHAR(22,1,32,96)
3090 RETURN                    ■
```

MATRISBERÄKNING

av Bengt Karlsson

```
100 CALL CLEAR
110 CALL CHAR(128,"181818181
8181818")
120 CALL CHAR(129,"03070C181
8181818")
130 CALL CHAR(130,"181818181
80C0703")
140 CALL CHAR(131,"183060C0C
0603018")
150 CALL CHAR(132,"FFFF00000
0000000")
160 CALL CHAR(133,"000000000
000FFFF")
170 CALL CHAR(134,"C0C0C0C0C
0C0C0C0")
180 CALL CHAR(135,"030303030
3030303")
190 CALL CHAR(136,"FFFFC0C0C
0C0C0C0")
200 CALL CHAR(137,"C0C0C0C0C
0C0FFFF")
210 CALL CHAR(138,"FFFF03030
3030303")
220 CALL CHAR(139,"030303030
303FFFF")
230 CALL CHAR(91,"0010003844
7C4444")
240 CALL CHAR(93,"0028003844
7C4444")
250 CALL CHAR(95,"0028003844
444438")
260 CALL CLEAR
270 PRINT "     MATRISBER]KN
INGAR"
280 PRINT "     ═══════════
═══"
290 PRINT : : : : : : : : :
: : : : :
300 PRINT "          AV BEN
GT KARLSSON":"          (J
UNI-JULI 1983)"
310 FOR DELAY=1 TO 700
320 NEXT DELAY
330 CALL CLEAR
340 CALL SCREEN(4)
350 PRINT "        M E N Y"
360 PRINT "        ═══════"
: : : : :
370 PRINT "1  -  MATRISADDIT
ION": :
380 PRINT "2  -  MATRISSUBTR
AKTION": :
390 PRINT "3  -  MATRISMULTI
PLIKATION": :
400 PRINT "4  -  MATRISINVER
TERING": :
410 PRINT "5  -  L_SNING AV
EKV.SYSTEM": :
420 PRINT "6  -  AVSLUTA!":
:
430 PRINT : : :
440 PRINT "TRYCK SIFFRA (1-6
)!": :
450 CALL KEY(0,K,S)
460 IF S=0 THEN 450
470 IF K<49 THEN 450
480 IF K>54 THEN 450
490 IF K=51 THEN 1040
500 IF K=49 THEN 550
510 IF K=54 THEN 4020
520 IF K=50 THEN 590
530 IF K=52 THEN 1950
540 IF K=53 THEN 2770
550 CALL CLEAR
560 PRINT "      MATRISADDI
TION":"      ═══════════
": : : : : : : : : : : : : :
 : :
570 CHECK=1
580 GOTO 620
590 CALL CLEAR
600 PRINT "      MATRISSUBTR
AKTION":"      ═══════════
═══": : : : : : : : : : : :
 : : : :
610 CHECK=0
620 FOR DELAY=1 TO 400
630 NEXT DELAY
640 CALL CLEAR
650 PRINT "MATRISDIMENSION?
(R,K)"
660 INPUT R,K
670 CALL CLEAR
680 PRINT "MATRIS A":"------
--": :
690 FOR I=1 TO K
700 PRINT "KOLUMN";I
710 FOR J=1 TO R
720 INPUT A(J,I)
730 NEXT J
740 PRINT :
750 NEXT I
760 CALL CLEAR
770 PRINT "MATRIS B":"------
--": :
780 FOR I=1 TO K
790 PRINT "KOLUMN";I
800 FOR J=1 TO R
810 INPUT B(J,I)
820 NEXT J
830 PRINT :
840 NEXT I
850 CALL CLEAR
860 IF CHECK=1 THEN 890 ELSE
 870
870 PRINT "DIFFERENSEN AV A
& B BLIR:": :
880 GOTO 900
890 PRINT "SUMMAN AV MATRIS
A & B BLIR:": :
900 FOR J=1 TO R
910 FOR I=1 TO K
920 IF CHECK=1 THEN 950 ELSE
 930
930 S=A(J,I)-B(J,I)
940 GOTO 960
950 S=A(J,I)+B(J,I)
960 PRINT S;
970 NEXT I
980 PRINT : :
990 NEXT J
1000 PRINT :
1010 PRINT "MENYN?-TRYCK N[G
ON TANGENT!"
1020 GOSUB 1910
1030 GOTO 330
1040 CALL CLEAR
1050 PRINT "    MATRISMULTIP
LIKATION"
1060 PRINT "    ═══════════
═══════": : : : : : : : : :
 : : : : : :
1070 FOR DELAY=1 TO 400
1080 NEXT DELAY
1090 CALL CLEAR
1100 PRINT "DIMENSION HOS MA
TRIS A?(R,K)"
1110 INPUT R1,K1
1120 PRINT
1130 PRINT "DIMENSION HOS MA
TRIS B?(R,K)"
1140 INPUT R2,K2
1150 IF K1=R2 THEN 1260
1160 CALL SOUND(-1200,110,0,
120,0,130,0,-3,0)
1170 CALL CLEAR
1180 CALL SCREEN(10)
1190 PRINT "    OMV]RDERA DI
NA DATA!": :"      A OCH B K
AN INTE": :"      MULTIPLIC
ERAS!": :"      FEL DIMENSI
ON!"
```

```
1200 PRINT : : : : : : : : :
1210 FOR DELAY=1 TO 1000
1220 NEXT DELAY
1230 CALL SCREEN(4)
1240 CALL CLEAR
1250 GOTO 1090
1260 CALL CLEAR
1270 PRINT "MATRIS A"
1280 PRINT "--------":
1290 FOR I=1 TO K1
1300 PRINT "KOLUMN";I
1310 FOR J=1 TO R1
1320 INPUT A(J,I)
1330 NEXT J
1340 PRINT
1350 NEXT I
1360 CALL CLEAR
1370 PRINT "MATRISEN A=": :
1380 GOSUB 1670
1390 PRINT "MATRIS B"
1400 PRINT "--------":
1410 FOR I=1 TO K2
1420 PRINT "KOLUMN";I
1430 FOR J=1 TO R2
1440 INPUT B(J,I)
1450 NEXT J
1460 PRINT
1470 NEXT I
1480 CALL CLEAR
1490 PRINT "MATRIS B=": :
1500 GOSUB 1790
1510 PRINT "MAT(A*B)=": :
1520 FOR J=1 TO R1
1530 PRINT
1540 FOR I=1 TO K2
1550 S=0
1560 FOR K=1 TO R2
1570 S=S+A(J,K)*B(K,I)
1580 NEXT K
1590 PRINT S;
1600 NEXT I
1610 PRINT
1620 NEXT J
1630 PRINT : :
1640 PRINT "MENYN?-TRYCK N[G
ON TANGENT!"
1650 GOSUB 1910
1660 GOTO 330
1670 REM *****UTSKRIFT AV MA
T A*****
1680 FOR J=1 TO R1
1690 PRINT : :
1700 FOR I=1 TO K1
1710 PRINT A(J,I);
1720 NEXT I
1730 NEXT J
1740 PRINT : :
1750 FOR DELAY=1 TO 1000
1760 NEXT DELAY
1770 CALL CLEAR
1780 RETURN
1790 REM *****UTSKRIFT AV MA
T B*****
1800 FOR J=1 TO R2
1810 PRINT : :
1820 FOR I=1 TO K2
1830 PRINT B(J,I);
1840 NEXT I
1850 NEXT J
1860 PRINT : :
1870 FOR DELAY=1 TO 1000
1880 NEXT DELAY
1890 CALL CLEAR
1900 RETURN
1910 REM *****PAUS*****
1920 CALL KEY(0,K,S)
1930 IF S=0 THEN 1920
1940 RETURN
1950 REM *****4 - MATRISINVE
RTERING*****
1960 CALL CLEAR
1970 PRINT "      MATRISINVE
RTERING":"      ==========
==========": : : : : : : : : :
: : : : :
1980 FOR DELAY=1 TO 400
1990 NEXT DELAY
2000 CALL CLEAR
2010 CALL SCREEN(4)
2020 PRINT "F_LJANDE PROGRAM
 BER]KNAR   DEN INVERTERADE
MATRISEN    TILL A OCH SKRIV
ER UT RESUL-"
2030 PRINT "TATET AVRUNDAT T
ILL TV[ DE- CIMALER.": : : :
 :
2040 PRINT "F_R ATT EN MATRI
S SKA KUNNA INVERTERAS M[STE
 DEN VARA   KVADRATISK - DVS
 ANTALET "
2050 PRINT "RADER M[STE VARA
 LIKA MED   ANTALET KOLUMNER
.": : : : :"ANGE MATRISDIMEN
SION R !"
2060 INPUT R
2070 CALL CLEAR
2080 PRINT "MATRIS A":"-----
----"
2090 FOR I=1 TO R
2100 PRINT
2110 PRINT "KOLUMN";I
2120 FOR J=1 TO R
2130 INPUT A(J,I)
2140 NEXT J
2150 PRINT
2160 B(I,I)=1
2170 NEXT I
2180 GOSUB 2720
2190 FOR J=1 TO R
2200 FOR I=J TO R
2210 IF A(I,J)<>0 THEN 2330
2220 NEXT I
2230 CALL CLEAR
2240 CALL SCREEN(10)
2250 PRINT "MATRISEN SINGUL]
R - KAN EJ  INVERTERAS!!!":
: : : : : : : :"VILL DU [TER
G[ TILL MENYN? -TRYCK 1"
2260 PRINT "VILL DU INVERTER
A EN NY     MATRIS? - TRYCK
2"
2270 CALL KEY(0,KEY,STATUS)
2280 IF STATUS=0 THEN 2270
2290 IF KEY<49 THEN 2270
2300 IF KEY>50 THEN 2270
2310 IF KEY=49 THEN 330
2320 IF KEY=50 THEN 2000
2330 FOR K=1 TO R
2340 S=A(J,K)
2350 A(J,K)=A(I,K)
2360 A(I,K)=S
2370 S=B(J,K)
2380 B(J,K)=B(I,K)
2390 B(I,K)=S
2400 NEXT K
2410 T=1/A(J,J)
2420 FOR K=1 TO R
2430 A(J,K)=T*A(J,K)
2440 B(J,K)=T*B(J,K)
2450 NEXT K
2460 FOR L=1 TO R
2470 IF L=J THEN 2530
2480 T=-A(L,J)
2490 FOR K=1 TO R
2500 A(L,K)=A(L,K)+T*A(J,K)
2510 B(L,K)=B(L,K)+T*B(J,K)
2520 NEXT K
2530 NEXT L
2540 NEXT J
2550 CALL CLEAR
2560 CALL SCREEN(4)
2570 PRINT "DEN INVERTERADE
MATRISEN =": :
2580 FOR I=1 TO R
2590 FOR J=1 TO R
2600 PRINT INT(B(I,J)*100+.5
)/100;
2610 NEXT J
2620 PRINT : :
2630 NEXT I
2640 PRINT
2650 PRINT "MENYN?          -
 TRYCK 1":"NY INVERTERING? -
 TRYCK 2"
2660 CALL KEY(0,KEY,STATUS)
2670 IF STATUS=0 THEN 2660
2680 IF KEY<49 THEN 2660
2690 IF KEY>50 THEN 2660
2700 IF KEY=49 THEN 330
2710 IF KEY=50 THEN 2000
2720 REM *****"PAUS-BILD"***
**
2730 CALL CLEAR
2740 CALL SCREEN(12)
2750 PRINT "V ] N T A   E T
T   T A G !": : : : : : : :
```

```
: : : : : : : : :
2760 RETURN
2770 REM
2780 REM
2790 REM *****LOESNING AV EK
VATIOSSYSTEM*****
2800 REM
2810 CALL CLEAR
2820 PRINT " L_SNING AV EKVA
TIONSSYSTEM":" ═══════
═══════": : : : : : :
: : : : : : : : :
2830 FOR DELAY=1 TO 500
2840 NEXT DELAY
2850 CALL CLEAR
2860 PRINT "BETRAKTA F_LJAND
E ALLM]NA   EKVATIONSSYSTEM"
: : :"  a11x1+a12x2+....+a1N
xN=c1"
2870 PRINT "  a21x1+a22x2+..
..+a2NxN=c2":"     :    :
       :  :":"     :    :
       :  :"
2880 PRINT "  aN1x1+aN2x2+..
..+aNNxN=cN": : :"D]R x1....
.xN ]R OBEKANTA.  MED HJ]LP
AV MATRISER KAN "
2890 PRINT "EKVATIONSSYSTEME
T SKRIVAS P[F_LJANDE S]TT":
:"       A * X = C": : : : :
 :"(TRYCK N[GON TANGENT!)"
2900 CALL VCHAR(4,4,129,1)
2910 CALL VCHAR(5,4,128,2)
2920 CALL VCHAR(7,4,131,1)
2930 CALL VCHAR(8,4,128,2)
2940 CALL VCHAR(10,4,130,1)
2950 CALL KEY(0,KE,STA)
2960 IF STA=0 THEN 2950
2970 CALL CLEAR
2980 PRINT "D]R": : :"   a11
 a12...a1N       x1":"   a21
 a22...a2N       x2":"A=  :
  :     :     X= :"
2990 PRINT "    :   :     :
       :":"   aN1 aN2...aNN
      xN": : : :
3000 PRINT "   c1    L_SNING
EN TILL":"   c2    EKVATIONS
SYSTEMET":"C= :     ,SKRIVEN
 I MAT-"
3010 PRINT "   :     RISFORM
, BLIR":"   cN    X = A^(-1)
 * C":"         DVS A INVERT
ERAS"
3020 PRINT "         OCH MUL
TIPLICERAS":"         D]REFT
ER  MED C.": : :"(TRYCK N[GO
N TANGENT!)"
3030 CALL VCHAR(4,5,136)
3040 CALL VCHAR(5,5,134,6)
3050 CALL VCHAR(10,5,137)
3060 CALL VCHAR(4,19,138)
3070 CALL VCHAR(5,19,135,6)
3080 CALL VCHAR(10,19,139)
3090 CALL VCHAR(4,25,136)
3100 CALL VCHAR(5,25,134,6)
3110 CALL VCHAR(10,25,137)
3120 CALL VCHAR(4,28,138)
3130 CALL VCHAR(5,28,135,6)
3140 CALL VCHAR(10,28,139)
3150 CALL VCHAR(12,5,136)
3160 CALL VCHAR(13,5,134,6)
3170 CALL VCHAR(18,5,137)
3180 CALL VCHAR(12,8,138)
3190 CALL VCHAR(13,8,135,6)
3200 CALL VCHAR(18,8,139)
3210 CALL KEY(0,K,S)
3220 IF S=0 THEN 3210
3230 CALL CLEAR
3240 INPUT "ANTAL OBEKANTA?
 ":R
3250 CALL CLEAR
3260 PRINT "MATRIS A":"─────
──": :
3270 FOR I=1 TO R
3280 PRINT "KOLUMN";I
3290 FOR J=1 TO R
3300 INPUT A(J,I)
3310 NEXT J
3320 PRINT
3330 B(I,I)=1
3340 NEXT I
3350 CALL CLEAR
3360 PRINT "VEKTOR C":"─────
──": :
3370 FOR L=1 TO R
3380 INPUT Y(L)
3390 NEXT L
3400 CALL CLEAR
3410 CALL SCREEN(14)
3420 PRINT "V ] N T A   E T
T   T A G  !": : : : : : : :
 : : : : : : :
3430 FOR J=1 TO R
3440 FOR I=J TO R
3450 IF A(I,J)<>0 THEN 3580
3460 NEXT I
3470 CALL CLEAR
3480 CALL SCREEN(4)
3490 PRINT "EKVATIONSSYSTEME
T EJ L_SBART"
3500 PRINT : : : : : : : : :
 : : : : : :
3510 PRINT "NYTT EKV.SYST? -
 TRYCK 1":"MENYN?         -
TRYCK 2"
3520 CALL KEY(0,Q,S)
3530 IF S=0 THEN 3520
3540 IF Q<49 THEN 3520
3550 IF Q>50 THEN 3520
3560 IF Q=49 THEN 3230
3570 IF Q=50 THEN 260
3580 GOTO 3760
3590 PRINT "EKVATIONSSYSTEME
TS L_SNING:"
3600 FOR I=1 TO R
3610 S=0
3620 FOR K=1 TO R
3630 S=S+B(I,K)*Y(K)
3640 NEXT K
3650 PRINT
3660 PRINT "X";I;"=";S
3670 NEXT I
3680 PRINT : : :
3690 PRINT "NYTT EKV.SYSTEM?
 - TRYCK 1":"MENYN?
  - TRYCK 2"
3700 CALL KEY(0,Q,S)
3710 IF S=0 THEN 3700
3720 IF Q<49 THEN 3700
3730 IF Q>50 THEN 3700
3740 IF Q=49 THEN 3230
3750 IF Q=50 THEN 330
3760 REM *****FORTSAETTNING
EKV.SYST.*****
3770 FOR K=1 TO R
3780 S=A(J,K)
3790 A(J,K)=A(I,K)
3800 A(I,K)=S
3810 S=B(J,K)
3820 B(J,K)=B(I,K)
3830 B(I,K)=S
3840 NEXT K
3850 T=1/A(J,J)
3860 FOR K=1 TO R
3870 A(J,K)=T*A(J,K)
3880 B(J,K)=T*B(J,K)
3890 NEXT K
3900 FOR L=1 TO R
3910 IF L=J THEN 3970
3920 T=-A(L,J)
3930 FOR K=1 TO R
3940 A(L,K)=A(L,K)+T*A(J,K)
3950 B(L,K)=B(L,K)+T*B(J,K)
3960 NEXT K
3970 NEXT L
3980 NEXT J
3990 CALL SCREEN(4)
4000 CALL CLEAR
4010 GOTO 3590
4020 CALL COLOR(13,16,2)
4030 Z=0
4040 FOR E=1 TO 32
4050 Z=Z+10
4060 CALL SOUND(-1000,110+Z,
0)
4070 CALL VCHAR(1,E,134,24)
4080 NEXT E
4090 FOR F=1 TO 24
4100 Z=Z-10
4110 CALL SOUND(-1000,110+Z,
0)
4120 CALL HCHAR(F,1,132,32)
4130 NEXT F
4140 CALL CLEAR
4150 END                    ■
```

TIPS FROM THE TIGERCUB #56

by Jim Peterson, USA

In Tips #55, I showed you some quick and easy ways to create new character sets. Since folks nowadays don't like to key in long programs, let's continue with "tinygram" programming, and at the same time show you how to manipulate strings, and teach you the value of using MERGE format.

First, let's make a screen to display our new characters. Some of them will have to be double-spaced horizontally or vertically, so -

```
100 CALL CLEAR :: X=1 :: FOR
 CH=48 TO 159 :: PRINT CHR$(
CH)&" ";:: X=X+2 :: IF X<29
THEN 110 ELSE PRINT "":"":""
;:: X=1
110 NEXT CH
```

Save it- SAVE DSK1.100,MERGE

Now, you might like to move the common punctuation marks into the same character sets as the characters, so that you will not have to reidentify so many sets, also so you can color them easier.

```
120 DATA 32,33,34,44,46
130 FOR J=1 TO 5 :: READ CH
:: CALL CHARPAT(CH,CH$):: CA
LL CHAR(J+90,CH$):: CALL CHA
R(J+122,CH$)
140 NEXT J :: CALL CHARPAT(6
3,CH$):: CALL CHAR(64,CH$)::
 :: CALL CHAR(96,CH$)
```

If you want to program in Basic, or use BXB with characters all the way up to ASCII 159, add CALL CHAR(J+154,CH$) to the end of line 130 and CALL CHAR(128,CH$) to the end of line 140.

Save by SAVE DSK1.120,MERGE

If you are using that transliteration, you must remember that with upper case characters the ? is @, space is [, ! is \, " is], comma is , period is _. With the lower case they are FCTN keys C, F, A, G, W and V and for the 3rd set (ASCII 129 to 154) they are CTRL comma, period,;,=,* and (.

You can transfer upper case to lower by -
CALL CHARPAT(CH,CH$) and then CALL CHAR(CH+32,CH$) or the opposite by CH-32 and if you have BXB merged in you can create a 3rd set by CH+64.

The following are all incompatible with each other, so give them all line number 150 and save them in merge format as 150A, 150B, etc.

The numerals and the upper case letters all have the topmost pixel row blank to provide spacing between lines of text. We can make taller letters by deleting the top row and doubling the 7th row -

```
150 FOR CH=48 TO 126 :: CALL
 CHARPAT(CH,CH$):: CALL CHAR
(CH,SEG$(CH$,3,12)&SEG$(CH$,
13,4)):: NEXT CH
151 REM
```

Or, you can double the 3rd row -

```
150 FOR CH=48 TO 95 :: CALL
CHARPAT(CH,CH$):: CALL CHAR(
CH,SEG$(CH$,3,4)&SEG$(CH$,5,
12)):: NEXT CH
151 REM
```

The lower case letters are really small upper case with the upper 3 rows blank. All their vertical bars are in the 4th, 6th and 8th rows, so let's drop the first 3 rows and quadruple the 7th.

```
150 FOR CH=97 TO 127 :: CALL
 CHARPAT(CH,CH$):: CALL CHAR
(CH,SEG$(CH$,7,6)&RPT$(SEG$(
CH$,13,2),4)&SEG$(CH$,15,2))
:: NEXT CH
151 REM
```

Or, for topheavy letters, quadruple the 5th row -

```
150 FOR CH=97 TO 127 :: CALL
 CHARPAT(CH,CH$):: CALL CHAR
(CH,SEG$(CH$,7,2)&RPT$(SEG$(
CH$,9,2),4)&SEG$(CH$,11,6)):
: NEXT CH
151 REM
```

Or, if you want line spacing -

```
150 FOR CH=97 TO 122 :: CALL
 CHARPAT(CH,CH$):: CH$=SEG$(
CH$,5,8)&RPT$(SEG$(CH$,13,2)
,3)&SEG$(CH$,15,2):: CALL CH
AR(CH,CH$):: NEXT CH
151 REM
```

Or, for something silly -

```
150 FOR CH=48 TO 90 :: CALL
CHARPAT(CH,CH$):: CALL CHAR(
CH,SEG$(CH$,3,2)&RPT$(SEG$(C
H$,5,2),4)&SEG$(CH$,9,4)&SEG
$(CH$,15,2)):: NEXT CH
151 REM
```

For some good blocky characters -

```
150 FOR CH=48 TO 90 :: CALL
CHARPAT(CH,CH$):: CALL CHAR(
CH,RPT$(SEG$(CH$,3,2),2)&SEG
$(CH$,5,8)&RPT$(SEG$(CH$,15,
2),2)):: NEXT CH
151 REM
```

Or, if you would prefer them shorter for single-line spacing -

```
150 FOR CH=48 TO 90 :: CALL
CHARPAT(CH,CH$):: CALL CHAR(
CH,"00"&RPT$(SEG$(CH$,3,2),2
)&SEG$(CH$,7,6)&RPT$(SEG$(CH
$,15,2),2)):: NEXT CH
151 REM
```

If you would like numerals the same size as lower case,

```
150 FOR CH=48 TO 57 :: CALL
CHARPAT(CH,CH$):: CALL CHAR(
CH,"0000"&SEG$(CH$,1,6)&SEG$
(CH$,9,4)&SEG$(CH$,15,2)):: 
NEXT CH
151 REM
```

You can even shrink the lower case to only 4 rows high, although some letters are not very legible -

```
150 FOR CH=97 TO 122 :: CALL
 CHARPAT(CH,CH$):: CALL CHAR
```

```
(CH,SEG$(CH$,1,6)&SEG$(CH$,5
,4)&SEG$(CH$,11,6)):: NEXT C
H
151 REM
```

Something modernistic -

```
150 A$="00" :: FOR CH=48 TO
90 :: CALL CHARPAT(CH,CH$)::
 CALL CHAR(CH,SEG$(CH$,1,4)&
A$&SEG$(CH$,7,6)&A$&SEG$(CH$
,15,2)):: NEXT CH
151 REM
```

Or perhaps even better -

```
150 A$="00" :: FOR CH=48 TO
90 :: CALL CHARPAT(CH,CH$)::
 CH$=SEG$(CH$,3,10)&RPT$(SEG
$(CH$,13,2),2)&SEG$(CH$,15,2
)
151 CALL CHAR(CH,SEG$(CH$,1,
4)&A$&SEG$(CH$,7,2)&A$&SEG$(
CH$,11,2)&A$&SEG$(CH$,15,2))
:: NEXT CH
```

I call this one "Spooky".

```
150 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: CH$=SEG$(
CH$,3,14)&SEG$(CH$,1,2):: X$
=SEG$(CH$,1,1)&"0"
151 FOR J=3 TO 15 STEP 2 ::
X$=X$&SEG$(CH$,J,1)&SEG$(CH$
,J-1,1):: NEXT J :: CALL CHA
R(CH,X$):: X$="" :: NEXT CH
```

And "Spooky" backward -

```
150 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 15 STEP 2 :: CH2$=CH2$&SEG
$(CH$,J,1)&SEG$(CH$,J+3,1)::
 NEXT J :: CALL CHAR(CH,CH2$
):: CH2$="" :: NEXT CH
151 REM
```

Now, clear the memory with NEW, then -

```
MERGE DSK1.100
MERGE DSK1.120
Add a line 500 GOTO 500
```

And start MERGEing in your series of "150" routines and running them to see what you have created.

Then, save these next routines in MERGE format as 160A, 160B, etc.

All normal characters have the leftmost column of pixels and the two rightmost columns blank, for spacing between letters. We can widen the character into the left column -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 15 STEP 2
161 CH2$=CH2$&SEG$("014589CD
",POS("01234567",SEG$(CH$,J,
1),1),1)&SEG$(CH$,J+1,1):: N
EXT J :: CALL CHAR(CH,CH2$):
: CH2$="" :: NEXT CH
162 REM
163 REM
```

Or widen it both left and right -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 15 STEP 2
161 CH2$=CH2$&SEG$("014589CD
",POS("01234567",SEG$(CH$,J,
1),1),1)&SEG$("028A",POS("04
8C",SEG$(CH$,J+1,1),1),1)
162 NEXT J :: CALL CHAR(CH,C
H2$):: CH2$="" :: NEXT CH
163 REM
```

Or even a full 8 columns wide by just changing the "028A" in line 161 to "0129"

For darker characters, we can shade them into the 7th column -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=2 T
O 16 STEP 2 :: IF SEG$(CH$,J
-1,1)="1" THEN CH2$=CH2$&"18
" :: GOTO 163
161 IF CH=67 OR CH=71 OR CH=
99 OR CH=103 THEN 162 :: IF
SEG$(CH$,J-1,1)="4" AND SEG$
(CH$,J,1)="0" THEN CH2$=CH2$
&"60" :: GOTO 163
162 CH2$=CH2$&SEG$(CH$,J-1,1
)&SEG$("0367CBEF",POS("02468
ACE",SEG$(CH$,J,1),1),1)
163 NEXT J :: CALL CHAR(CH,C
H2$):: CH2$="" :: NEXT CH
```

Or shade them both left and right -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 15 STEP 2 :: A$=SEG$(CH$,J
,1):: P=POS("0123456789ABCDE
F",A$,1)
161 A$=SEG$("0367CDEF89ABCDE
F",P,1):: B$=SEG$(CH$,J+1,1)
:: P=POS("02468ACE",B$,1)::
B$=SEG$("0367CBEF",P,1):: CH
2$=CH2$&A$&B$
162 NEXT J :: CALL CHAR(CH,C
H2$):: CH2$="" :: NEXT CH
163 REM
```

Or shaded into both of the rightmost columns -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=2 T
O 16 STEP 2 :: CH2$=CH2$&SEG
$(CH$,J-1,1)&SEG$("0377EBFF"
,POS("02468ACE",SEG$(CH$,J,1
),1),1):: NEXT J :: CALL CHA
R(CH,CH2$):: CH2$="" :: NEXT
 CH
161 REM
162 REM
163 REM
```

Or into all 8 columns -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 15 STEP 2 :: P=POS("012345
6789ABCDEF",SEG$(CH$,J,1),1)
161 A$=SEG$("0367CDEF89ABCDE
F",P,1):: P=POS("02468ACE",S
EG$(CH$,J+1,1),1):: B$=SEG$(
"0367EBFF",P,1):: CH2$=CH2$&
A$&B$
162 NEXT J :: CALL CHAR(CH,C
H2$):: CH2$="" :: NEXT CH
163 REM
```

More neatly, shaded inward at right -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$)
161 FOR J=1 TO 15 STEP 2 ::
CH2$=CH2$&SEG$(CH$,J,1)&SEG$
("0C8C",POS("048C",SEG$(CH$,
J+1,1),1),1):: NEXT J
162 CALL CHAR(CH,CH2$):: CH2
$="" :: NEXT CH
163 REM
```

Or inward at right, outward at left -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 15 STEP 2
161 CH2$=CH2$&SEG$("0367CBEF
",POS("01234567",SEG$(CH$,J,
1),1),1)&SEG$("0C8C",POS("04
8C",SEG$(CH$,J+1,1),1),1)::
```

```
NEXT J
162 CALL CHAR(CH,CH2$):: CH2
$="" :: NEXT CH
163 REM
```

Here's a weirdo -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=9 T
O 15 STEP 2
161 CH2$=CH2$&SEG$("014589CD
",POS("01234567",SEG$(CH$,J,
1),1),1)&SEG$("028A",POS("04
8C",SEG$(CH$,J+1,1),1),1)
162 NEXT J :: CALL CHAR(CH,S
EG$(CH$,1,8)&CH2$):: CH2$=""
 :: NEXT CH
163 REM
```

Try changing that to FOR J =1 TO 7 and CALL CHAR(CH,CH2 $&SEG$(CH$,9,8))

And one more -

```
160 FOR CH=48 TO 122 :: CALL
 CHARPAT(CH,CH$):: FOR J=1 T
O 7 STEP 2
161 A$=SEG$("02468ACE",POS("
01234567",SEG$(CH$,J,1),1),1
):: B$=SEG$("0808",POS("048C
",SEG$(CH$,J+1,1),1),1):: CH
2$=CH2$&A$&B$ :: NEXT J
162 CALL CHAR(CH,CH2$&SEG$(C
H$,9,8)):: CH2$="" :: NEXT C
H
163 REM
```

Now, clear the memory, MERGE in 100 and 120, put in a holding line 500 GOTO 500 and start MERGEing in all of the different combinations of the 150 and 160 lines and see how many different character sets you can make!

Memory full, Jim Peterson ■

JACKPOT ENARMAD BANDIT

by Rick Rothstein, USA

Before play begins, the number 1 is displayed in each window, and the prompt ENTER LEVEL appears under the payout chart. If you press the space bar, the displayed level number will change. Press ENTER to begin the game at the displayed level. The payout chart continually prompts you to press the letter P to play and S to stop the game. In addition, pressing AID (FCTN-7) will allow you to enter a new level of play, and pressing REDO (FCTN-8) will reset your money status to even while retaining the same level of play.

```
80 REM JACKPOT
90 REM COMPUTE 84-08 XB
100 CALL CLEAR :: CALL SCREE
N(12):: CALL COLOR(0,12,12):
: CALL HCHAR(1,1,30,768)
110 CALL COLOR(1,5,16,2,7,16
,3,2,16,4,2,16,5,2,16,6,7,16
,7,2,16,8,7,16)
120 CALL COLOR(9,13,16,10,14
,16,11,14,16,12,5,16,13,13,1
6,14,13,16)
130 RANDOMIZE :: LEVEL=49 ::
 TOTAL=0 :: GOTO 310
135 REM P,S OR AID PRESS
140 RANDOM=RND :: CALL KEY(3
,KEY,STATUS):: IF STATUS=0 T
HEN 140
150 IF KEY=83 THEN CALL CLEA
R :: CALL COLOR(1,1,1):: END
 ELSE IF KEY=80 THEN TOTAL=T
OTAL-1 :: GOTO 200
160 IF KEY=6 THEN TOTAL=0 ::
 GOSUB 810 :: GOTO 140 ELSE
IF KEY<>1 THEN 140
170 GOSUB 770 :: CALL COLOR(
#1,2,#2,2,#3,2):: DISPLAY AT
(24,1)BEEP:RPT$(CHR$(30),8)&
"ECTERwFEVEF"&RPT$(CHR$(30),
9)
175 REM CHANGE LEVEL
180 CALL KEY(3,KEY,STATUS)::
 IF STATUS<1 THEN 180 ELSE I
F KEY=13 THEN GOSUB 780 :: G
OTO 140 ELSE IF KEY<>32 THEN
 180
190 LEVEL=LEVEL+1+3*(LEVEL>5
0):: DISPLAY AT(1,2)SIZE(1)B
EEP:"K" :: CALL PATTERN(#1,L
EVEL,#2,LEVEL,#3,LEVEL):: GO
TO 180
195 REM PICK 3 SHAPES
200 CALL SOUND(50,-2,0):: GO
SUB 810 :: GOSUB 770 :: FOR
I=1 TO 3 :: PICK(I)=VAL(SEG$
(WHEEL$(LEVEL-48,I),INT(20*R
ND+1),1)):: NEXT I
205 REM DISPLAY SHAPES
210 FOR I=4 TO 20 STEP 8 ::
FOR J=3 TO 7 :: DISPLAY AT(J
,I)SIZE(5):SHAPE$(PICK((I+4)
/8),J-2):: NEXT J
220 CALL SOUND(50,-6,0):: NE
XT I :: CALL SOUND(100,44000
,30)
225 REM CHECK FOR WIN
230 IF PICK(1)=1 THEN IF PIC
K(2)=1 THEN COINS=5 :: GOTO
270 ELSE COINS=2 :: GOTO 270
240 IF PICK(1)<>PICK(2)THEN
140 ELSE IF ((PICK(2)<>PICK(
3)OR PICK(2)=5)AND PICK(3)<>
5)OR(PICK(2)=6 AND PICK(3)=5
)THEN 140
250 IF PICK(1)<5 THEN COINS=
2+4*PICK(1):: GOTO 270
260 IF PICK(1)=5 THEN COINS=
100 :: GOTO 280 ELSE COINS=2
00 :: GOTO 290
265 REM UPDATE MONEY STATUS
270 FOR I=1 TO COINS :: TOTA
L=TOTAL+1 :: GOSUB 810 :: CA
LL SOUND(25,1000,0,3250,0,67
50,0):: NEXT I :: GOTO 140
280 FOR I=5 TO COINS STEP 5
:: TOTAL=TOTAL+5 :: GOSUB 81
0 :: CALL SOUND(35,1000,0,32
50,0,6750,0):: NEXT I :: GOT
O 140
290 FOR I=40 TO COINS STEP 4
0 :: FOR SIREN=700 TO 900 ST
EP 10 :: CALL SOUND(-99,SIRE
N,0):: NEXT SIREN :: TOTAL=T
OTAL+40 :: GOSUB 810
300 FOR SIREN=900 TO 700 STE
P -20 :: CALL SOUND(-200,SIR
EN,0):: NEXT SIREN :: NEXT I
 :: GOTO 140
305 REM DEFINE GRAPHICS
310 FOR I=32 TO 140 STEP 4 :
: READ CH$ :: CALL CHAR(I,CH
$):: NEXT I
312 DATA FFFFFFFFFFFFFFFF181
81818181818FF010307070F0F0F0
F80C0E0E0F0F0F0F0
315 DATA 0F1F1F1F1F1F3F3FF0F
8F8F8F8F8FCFC0000000000030F1
F3F7F7FFFFFFFFFF
320 DATA 00000000001C3E7F000
0000078FCFEFE7FFFFFFFFFFFFF7
FC0F0F8F8FCFCFCFC
330 DATA 030F1F1F3F3F3F3FFFF
FFFFFFFFFFEFE7F3F3F0F0300000
0F8F8F0E08
340 DATA 0038444444443800001
0301010103800003840810207C0
000384418044438
350 DATA 00081828487C0800007
840780444380000384078444438
0007C040810202
360 DATA 003844384444380000
3844443C047800000000000003F3F3
F0000000000FCFCFC
370 DATA 3F3F3F0000000000FCF
```

```
CFC0000000000003854301854380
0FFFFFFFFFFFFFFF
380 DATA 0038101010103800003
844447C444400003C223C22223C0
000446454544C44
390 DATA 0078444444447800007
C407840407C000040404040407C0
0003844404C4438
400 DATA 1F1F0F0300000000FCF
8F0C000000000183C7E7E7E3C180
0FFFFFFFFFFFFFFF
410 DATA 0000070704000000000
0F8F818303060000000010101000
060C0C080808
420 DATA 0078444478404000004
4442810101000007844447848440
000384430084438
430 DATA 007C101010101000004
4444444443800004444444428100
000444444545428
440 DATA FFFFFFFFFFFF8F8FFF
FFFFFFFF0000FFFFFFFFFFFFEF
C01010307070F1F1F
450 DATA FCF8F0E0C0C080803F3
F3F3E7E7E7E7E7CFCFCFCFCFCFCF
CFCFCFCFCFCFCFCFC,FF7F1F0700
000000FFFFFFFFFF000000FFFEF8
E000000000FFFFFFFFFFFFFFFF
460 DATA 0000030F1F1F3F7F000
0C0F0F8F8FCFE7F3F1F1F0F03000
0FEFCF8F8F0C
470 DATA 00030F1F3F7FFFFF00C
0F0F8FCFEFFFF0101010303070707
7808080C0C0E0E0E
480 DATA 070F0F0F0F0F0F07E0F
0F0F0F0F0F0E0070707030301010
1E0E0E0C0C08080808
490 DATA FFFF7F3F1F0F0300FFF
FFEFCF8F0C000FFFFFFFFFFFFFFF
F0000001070F0F1F1F
500 DATA 000080E0F0F0F8F81F1
F0F0F07010000F8F8F0F0E080000
00
510 DATA FCFEFEFFFFFFFFFF000
0000000C0F0F8000101010103070
70080808080C0E0E
520 DATA 07070F1F3F3F0000E0E
0F0F8FCFC0000C000000000000000
00000003
530 DATA 00010608102040803FC
1030204040808010202040408080
8101020204040404
540 DATA 4020201008040201000
0000001020408000064881020201
0000000000071F7FFF
550 DATA 000000FFFFFFFFFF000
00000E0F8FEFF01030307070F0F1
F80C0C0E0E0F0F0F8
560 DATA 1F3FFFFFFFFF3F1FF8F
CFFFFFFFFFCF81F0F0F070703030
1F8F0F0E0E0C0C08
565 REM CREATE SHAPES
570 SHAPE$(1,1)="ww"&CHR$(12
8)&CHR$(129)&"w" :: SHAPE$(1
,2)="w"&CHR$(130)&"w"&CHR$(1
31)&"w" :: SHAPE$(1,3)="("&C
HR$(131)&"w"&CHR$(132)&")"
580 SHAPE$(1,4)="*+w,-" :: S
HAPE$(1,5)="./wHI" :: SHAPE$
(2,1)="w"&CHR$(135)&CHR$(136
)&CHR$(137)&"w"
590 SHAPE$(2,2)=CHR$(138)&"c
cc"&CHR$(139):: SHAPE$(2,3)=
CHR$(140)&"ccc"&CHR$(141)::
SHAPE$(2,4)=CHR$(142)&"ccc"&
CHR$(143)
600 SHAPE$(2,5)="w`abw" :: S
HAPE$(3,1)="whriw" :: SHAPE$
(3,2)="jrrrk" :: SHAPE$(3,3)
="lrrrm" :: SHAPE$(3,4)="nrr
ro" :: SHAPE$(3,5)="wprqw"
610 SHAPE$(4,1)="ww!ww" :: S
HAPE$(4,2)="w"" #w" :: SHAPE
$(4,3)="w$ %w" :: SHAPE$(4,4
)="&' xy" :: SHAPE$(4,5)="'
  x"
620 SHAPE$(5,1)="wwwww" :: S
HAPE$(5,2)="?????" :: SHAPE$
(5,3)="?BAR?" :: SHAPE$(5,4)
="?????" :: SHAPE$(5,5)="www
ww"
630 SHAPE$(6,1)="wXYZw" :: S
HAPE$(6,2)="ww[\w" :: SHAPE$
(6,3)="ww]ww" :: SHAPE$(6,4)
="ww^ww" :: SHAPE$(6,5)="ww_
ww"
635 REM DISPLAY GRAPHICS
640 DISPLAY AT(1,2)SIZE(25):
RPT$("K",25):: GOSUB 770 ::
DISPLAY AT(9,2)SIZE(25):RPT$
("K",25)
650 DISPLAY AT(11,2)SIZE(25)
:"wwwOCEwDOFFARwPERwPFAQwww"
660 DISPLAY AT(12,2)SIZE(25)
:"w"&CHR$(133)&CHR$(134)&"ww
wwwwwwwwstststwwwwww"
665 DISPLAY AT(13,2)SIZE(25)
:"wJJw"&CHR$(127)&"w"&CHR$(1
27)&"w>w2wwuvuvuvw>w14w"
670 DISPLAY AT(14,2)SIZE(25)
:"w"&CHR$(133)&CHR$(134)&CHR
$(133)&CHR$(134)&"wwwwwwwwz{
z{:;wwwwww"
680 DISPLAY AT(15,2)SIZE(25)
:"wJJJJw"&CHR$(127)&"w>w5ww{
}|}<=w>w18w" :: DISPLAY AT(1
6,2)SIZE(25):"wdede:;wwwwwwz
{z{z{wwwwww"
690 DISPLAY AT(17,2)SIZE(25)
:"wfgfg<=w>10ww{}|}|}w>w18w"
 :: DISPLAY AT(18,2)SIZE(25)
:"wdededewwwwww:;:;:;wwwwww"
700 DISPLAY AT(19,2)SIZE(25)
:"wfgfgfgw>10ww<=<=<=w>100w"
 :: DISPLAY AT(20,2)SIZE(25)
:"wstst:;wwwwwwLMLMLMwwwwww"
710 DISPLAY AT(21,2)SIZE(25)
:"wuvuv<=w>14wwNONONOw>200w"
 :: DISPLAY AT(22,2)SIZE(25)
:"PRESSwPwTOwPFAQ"&CHR$(127)
&"SwTOwSTOP"
720 CALL MAGNIFY(2):: CALL S
PRITE(#1,LEVEL,1,29,53,#2,LE
VEL,1,29,117,#3,LEVEL,1,29,1
81)
725 REM PUT SHAPES ON WHEEL
730 FOR I=1 TO 3 :: FOR J=1
TO 3 :: READ ORDER$ :: WHEEL
$(I,J)=ORDER$ :: NEXT J :: N
EXT I :: KEY=1 :: GOTO 160
735 REM ORDER OF SHAPES
740 DATA 2531236424531425323
4,14216313156425213132,23424
325424364234324
750 DATA 1265312413512431524
6,62543512136423146352,24352
463523423542364
760 DATA 5213464612153153624
1,56231534146213125645,23456
246356254352634,@
765 REM CLEAR WINDOWS
770 FOR I=2 TO 8 :: DISPLAY
AT(I,2)SIZE(25):"KwwwwwwwKww
wwwwwKwwwwwwwwK" :: NEXT I ::
 RETURN
775 REM INITIAL SHAPES
780 FOR I=1 TO 3 :: PICK(I)=
VAL(SEG$(WHEEL$(LEVEL-48,I),
INT(20*RND+1),1)):: NEXT I
790 CALL COLOR(#1,1,#2,1,#3,
1):: TOTAL=0 :: FOR I=4 TO 2
0 STEP 8 :: FOR J=3 TO 7
800 DISPLAY AT(J,I)SIZE(5):S
HAPE$(PICK((I+4)/8),J-2):: N
EXT J :: CALL SOUND(35,-6,0)
:: NEXT I :: CALL SOUND(100,
40000,30)
805 REM DISPLAY MONEY STATUS
810 IF TOTAL=0 THEN DISPLAY
AT(24,1):RPT$(CHR$(30),5)&"Q
OUwAREwCOWwEVEC"&RPT$(CHR$(3
0),7):: RETURN
820 TOTAL$=STR$(ABS(TOTAL)):
: LENGTH=LEN(TOTAL$):: COLUM
N=6+(TOTAL>0)-INT(.5+LENGTH/
2)
830 IF TOTAL>0 THEN DISPLAY
AT(24,COLUMN)SIZE(20+LENGTH)
:CHR$(30)&"QOUwAREwW@CC@CGw>
"&TOTAL$&RPT$(CHR$(30),4)::
RETURN
840 IF TOTAL<0 THEN DISPLAY
AT(24,COLUMN)SIZE(18+LENGTH)
:CHR$(30)&"QOUwAREwFOS@CGw>"
&TOTAL$&RPT$(CHR$(30),4):: R
ETURN
```

COMPUTE GUIDE SOUND&GRAPHICS

Nedanstående skiva med alla program från boken kan fås från redaktören genom att sända in skiva och frankerat svarskuvert till redaktören.

```
COM:GRAPH
FIL=38 LED=0 ANV=360
filnamn     sekt typ     längd

ADV:AIRDEF     6 PROGRAM 1208
ADV:CHASE      4 PROGRAM  714
ADV:MAZE      20 PROGRAM 4793
ADV:METEOR     7 PROGRAM 1503
ALL:ADD       24 PROGRAM 5803
ALL:ALPHA     20 PROGRAM 4649
ALL:BUNNY     17 PROGRAM 4067
ALL:MIMIC     12 PROGRAM 2683
ALL:SHOOT     20 PROGRAM 4720
ALL:SLOT      17 PROGRAM 3869
ALL:ZONE      11 PROGRAM 2559
CATALOG        3 PROGRAM  498
GRA:BLINKY     5 PROGRAM  939
GRA:CAT        5 PROGRAM  994
GRA:GRAPH     11 PROGRAM 2335
GRA:HISTO      8 PROGRAM 1682
GRA:PLANE      6 PROGRAM 1253
GRA:SHAPES     8 PROGRAM 1657
GRA:TANK       7 PROGRAM 1290
LOAD          10 PROGRAM 2242
SOU:BACH      32 PROGRAM 7853
SOU:DEMO1      4 PROGRAM  601
SOU:DEMO2      5 PROGRAM  835
SOU:DEMO3      7 PROGRAM 1351
SOU:OCTAVE     3 PROGRAM  368
SOU:TUTOR     13 PROGRAM 2854
SPE:ALLNUM     4 PROGRAM  517
SPE:ALLSTR     4 PROGRAM  618
SPE:DEMO1      4 PROGRAM  598
SPE:DEMO2      2 PROGRAM  255
SPE:TALKER     3 PROGRAM  457
SPE:WORD      16 PROGRAM 3754
SPR:BIRD       4 PROGRAM  702
SPR:EDIT      13 PROGRAM 2858
SPR:EXEMP1     3 PROGRAM  333
SPR:EXEMP2     3 PROGRAM  463
SPR:GOBB      10 PROGRAM 2157
SPR:KALEID     7 PROGRAM 1286
```

```
80 ! ADV:AIRDEF XB
90 ! COMPUTE SOUND&GRAPHICS
100 CALL CLEAR
110 RANDOMIZE
120 PAT=97 :: SHOT=.1
130 CALL CHAR(96,"0000C06030
187EFF0000181818187EFF000003
060C187EFF")
160 CALL CHAR(100,"0000003C7
EFF7E3C187EFFFF2442810000007
E7EFF7E7E000000000000181800"
,104,"FFFFFFFFFFFFFFFF"):: C
ALL COLOR(10,13,1)
210 CALL CHAR(112,"010000101
80040811080810082828114")
230 CALL HCHAR(22,1,104,96)
260 CALL SPRITE(#1,97,2,162,
125)
270 CALL SPRITE(#3,100+INT(R
ND*3),5,20+INT(RND*125),255,
0,-(10+INT(RND*12)))
280 CALL KEY(3,K,S)
290 RATE=HIT/SHOT*100
300 DISPLAY AT(1,8):USING "#
###### ###":"RATING:",RATE
310 IF S=0 THEN 280
320 IF K=83 THEN PAT=MAX(PAT
-1,96)
330 IF K=68 THEN PAT=MIN(PAT
+1,98)
340 IF K=32 THEN 370
350 CALL PATTERN(#1,PAT)
360 GOTO 280
370 CALL SPRITE(#2,103,7,156
,125-((97-PAT)*6))
380 SHOT=SHOT+1
390 R=-30 :: C=((PAT-97)*30)
400 CALL MOTION(#2,R,C)
410 CALL POSITION(#2,X,Y)::
IF X<12 OR Y<5 OR Y>250 THEN
 CALL DELSPRITE(#2):: GOTO 2
80
420 CALL COINC(#2,#3,5,CO)
430 IF CO=0 THEN 410
440 CALL COLOR(#3,9)
450 CALL DELSPRITE(#2)
460 CALL PATTERN(#3,112)
470 FOR L=1 TO 30 :: NEXT L
480 CALL PATTERN(#3,113)
490 FOR L=1 TO 30 :: NEXT L
500 CALL DELSPRITE(#3)
510 HIT=HIT+1
520 GOTO 270
```

```
80 ! GRA:HISTO XB
90 ! COMPUTE SOUND&GRAPHICS
100 CALL CLEAR
110 DIM V(12)
120 CALL CHAR(96,"FFFFFFFFFF
FFFFFF0000FFFFFFFFFFFF000000
00FFFFFFFF0000000000000FFFF")
160 CALL CHAR(42,"FFFFFFFFFF
FFFFFFFFF0F0F0F0F0F0F0",104,
"FF",112,"FF")
200 CALL COLOR(9,5,1,10,3,1,
11,7,1)
210 DISPLAY AT(4,1):"1 - ENT
ER VALUES MANUALLY": :"2 - R
UN DEMONSTRATION": :"WHICH O
NE? -->_"
230 ACCEPT AT(8,15)VALIDATE(
"12")SIZE(-1)BEEP:OPT
240 ON OPT GOSUB 510,600
250 CALL CLEAR
260 DISPLAY AT(2,10):"COMPUT
ER SALES" :: DISPLAY AT(4,13
):"BY MONTH"
270 CALL HCHAR(18,6,42,25)
280 CALL VCHAR(8,6,43,10)
290 FOR L=8 TO 12 :: CALL HC
HAR(L,7,104,24):: NEXT L
300 FOR L=13 TO 17 :: CALL H
CHAR(L,7,112,24):: NEXT L
310 DISPLAY AT(19,6):"J F M
A M J J A S O N D"
320 DISPLAY AT(8,1):"10";::
DISPLAY AT(13,2):"5";:: DISP
LAY AT(17,2):"1";
330 MX=0 :: FOR L=1 TO NV ::
 MX=MAX(V(L),MX):: NEXT L
340 IF MX<=10 THEN S=1
350 IF MX>10 AND MX<=1000 TH
EN S=100 :: LIT$="HUNDREDS"
360 IF MX>1000 THEN S=1000 :
: LIT$="THOUSANDS"
370 FOR L=1 TO NV
380 BAR=INT(V(L)/S):: FR=V(L
)/S-INT(V(L)/S)
390 CALL VCHAR(18-BAR,6+L*2,
96,BAR)
400 FRP=18-BAR-1
410 IF FR>=.15 AND FR<.35 TH
EN CALL VCHAR(FRP,6+L*2,99)
420 IF FR>=.35 AND FR<.65 TH
EN CALL VCHAR(FRP,6+L*2,98)
430 IF FR>=.65 AND FR<.9 THE
N CALL VCHAR(FRP,6+L*2,97)
440 IF FR>=.9 THEN CALL VCHA
R(FRP,6+L*2,96)
450 NEXT L
460 IF S>1 THEN DISPLAY AT(2
1,7):"IN ";LIT$;" OF UNITS"
470 DISPLAY AT(24,9):"PRESS
ANY KEY"
480 CALL KEY(3,K,S):: IF S=0
 THEN 480
490 CALL CLEAR
500 GOTO 210
510 CALL CLEAR
520 DISPLAY AT(4,1):"NUMBER
OF VALUES 1-12? -->"
530 ACCEPT AT(4,27)VALIDATE(
NUMERIC)BEEP:NV
540 IF NV>12 THEN 530
550 FOR L=1 TO NV
560 DISPLAY AT(6+L,1):"VALUE
 # ";L
570 ACCEPT AT(6+L,13)VALIDAT
E(NUMERIC)BEEP:V(L):: IF V(L
)>10000 THEN 570
580 NEXT L
```

```
590 RETURN
600 FOR L=1 TO 12
610 V(L)=INT(1+RND*10)
620 NEXT L
630 NV=12 :: RETURN


80 ! SOU:DEMO2 XB
90 ! COMPUTE SOUND&GRAPHICS
100 T=500*50/60
110 LC=131 :: LD=147 :: LE=1
65 :: LF=175 :: LG=196 :: LA
F=208 :: LA=220 :: LBF=233 :
: LB=247
120 C=262 :: D=294 :: E=330
:: F=349 :: G=392 :: A=440 :
: B=494 :: HC=523
130 CALL SOUND(T,HC,0,LC,0)
140 CALL SOUND(T/2,HC,0,LE,0
)
150 CALL SOUND(T/2,B,0,LE,0)
160 CALL SOUND(T,A,0,LF,0)
170 CALL SOUND(T,A,0,LF,0)
180 CALL SOUND(T,G,0,LG,0)
190 CALL SOUND(T/2,G,0,LB,0)
200 CALL SOUND(T/2,F,0,LB,0)
210 CALL SOUND(T,E,0,C,0)
220 CALL SOUND(T/2,E,0,C,0)
230 CALL SOUND(T/2,F,0,C,0)
240 CALL SOUND(T,G,0)
250 CALL SOUND(T,C,0,LBF,0)
260 CALL SOUND(T,D,0,LA,0)
270 CALL SOUND(T,F,0,LAF,0)
280 CALL SOUND(T,E,0,LG,0)
290 CALL SOUND(T/2,E,0,LF,0)
300 CALL SOUND(T/2,D,0,LF,0)
310 CALL SOUND(T,C,0,LE,0)


80 ! SPE:DEMO2 XB SS
90 ! COMPUTE SOUND&GRAPHICS
100 CALL SPGET("READ",W$)
110 CALL SPGET("MOVE",W2$)
120 TR=LEN(W$)-50-3
130 NW$=SEG$(W$,1,2)&CHR$(TR
)&SEG$(W$,4,TR)
140 NEWWORD$=NW$&W2$
150 CALL SAY("NOW, YOU SHOUL
D",NEWWORD$,"THE1 CASSETTE")


80 ! SPR:GOBBLER XB
90 ! COMPUTE SOUND&GRAPHICS
100 CALL CLEAR
110 RANDOMIZE
120 CALL CHAR(96,"001F3F7F7F
7F7F7F7F7F7F7F3F1F0000F8FC
FEFEFEFEFEFEFEFEFEFEFCF800")
130 CALL CHAR(100,"001F3F7F7
F7F7F7F7E787E7F7F3F1F0000F8F
CF6FEFEFE00000002FEFEFCF800"
)
140 CALL CHAR(104,"001F3F6F7
F7F7F010000407F7F3F1F0000F8F
CFEFEFEFEFEFE3EFEFCFEFCF800"
)
150 CALL CHAR(108,"001C38787
87878787C7D7D7F7F3F1F0000787
C767E7E7E7EFEFEFEFEFEFCF800"
)
160 CALL CHAR(112,"001F3F7F7
F7D7D7C7878787878381C0000F8F
CFEFEFEFEFE7E7E7E7E767C7800"
)
170 CALL CHAR(120,"FFFFFFFFF
FFFFFFF"):: CALL COLOR(12,14
,1)
180 CALL CHAR(116,"000000181
8000000")
190 CALL HCHAR(1,1,120,32)::
 CALL HCHAR(21,1,120,32):: C
ALL VCHAR(1,1,120,20):: CALL
 VCHAR(1,32,120,20)
200 FOR L=2 TO 20 :: FOR L2=
2 TO 31
210 IF RND>.3 THEN CALL HCHA
R(L,L2,116)
220 NEXT L2 :: NEXT L
230 CALL HCHAR(11,16,116,2)
240 CALL MAGNIFY(3)
250 CALL SPRITE(#1,100,5,76,
124)
260 CR=11 :: CC=17 :: SR=76
:: SC=124
270 DISPLAY AT(22,1):"E,S,D,
X TO MOVE - Q TO QUIT"
280 DISPLAY AT(24,8):"SCORE:
 "
290 CALL KEY(3,K,S):: IF S=0
 THEN 290
300 IF K=68 THEN 360
310 IF K=83 THEN 450
320 IF K=69 THEN 540
330 IF K=88 THEN 630
340 IF K=81 THEN CALL DELSPR
ITE(ALL):: SCOR=0 :: GOTO 10
0
350 GOTO 290
360 CALL GCHAR(CR,CC+1,CH)
370 IF CH<>116 THEN 290
380 SC=SC+8 :: CALL LOCATE(#
1,SR,SC)
390 CALL PATTERN(#1,96)
400 FOR L=1 TO 10 :: NEXT L
410 CALL PATTERN(#1,100)
420 CC=CC+1 :: CALL HCHAR(CR
,CC,32)
430 SCOR=SCOR+1 :: DISPLAY A
T(24,16):SCOR
440 GOTO 290
450 CALL GCHAR(CR,CC-1,CH)
460 IF CH<>116 THEN 290
470 SC=SC-8 :: CALL LOCATE(#
1,SR,SC)
480 CALL PATTERN(#1,96)
490 FOR L=1 TO 10 :: NEXT L
500 CALL PATTERN(#1,104)
510 CC=CC-1 :: CALL HCHAR(CR
,CC,32)
520 SCOR=SCOR+1 :: DISPLAY A
T(24,16):SCOR
530 GOTO 290
540 CALL GCHAR(CR-1,CC,CH)
550 IF CH<>116 THEN 290
560 SR=SR-8 :: CALL LOCATE(#
1,SR,SC)
570 CALL PATTERN(#1,96)
580 FOR L=1 TO 10 :: NEXT L
590 CALL PATTERN(#1,108)
600 CR=CR-1 :: CALL HCHAR(CR
,CC,32)
610 SCOR=SCOR+1 :: DISPLAY A
T(24,16):SCOR
620 GOTO 290
630 CALL GCHAR(CR+1,CC,CH)
640 IF CH<>116 THEN 290
650 SR=SR+8 :: CALL LOCATE(#
1,SR,SC)
660 CALL PATTERN(#1,96)
670 FOR L=1 TO 10 :: NEXT L
680 CALL PATTERN(#1,112)
690 CR=CR+1 :: CALL HCHAR(CR
,CC,32)
700 SCOR=SCOR+1 :: DISPLAY A
T(24,16):SCOR
710 GOTO 290                    ■
```

PRINT DOC DV80

```
10 CALL CLEAR
20 !TI*MES LIBRARY U32+
 this will read dis/var 80
 files on disk and dump to
the screen or your printer
25 !
30 DISPLAY AT(12,7):"DIS/VAR
 80 UTILITY":"   ADAPTED by
CLIVE SCALLY":"for TI99/4a E
XCHANGE Library"
40 FOR DLY=1 TO 500 :: NEXT
DLY
50 INPUT "PRINTER ":PR$ ! if
 you have a printer
60 INPUT "
          ENTER
          FILENAME DSK1.":FN
$
70 CALL KEY(0,K,S)::
80 PRINT : :"SEND TO": :"1=S
CREEN": :"2=PRINTER": :
90 INPUT A :: IF A<1 OR A>2
THEN 90
100 CALL CLEAR
110 OPEN #1:"DSK1."&FN$,INPU
T ,VARIABLE
```

```
120 ON A GOTO 130,200
130 LINPUT #1:X$
140 PRINT X$
150 CALL KEY(0,K,S):: IF K=3
2 THEN 160 ELSE 180
160 FOR DL=1 TO 500 :: NEXT
DL
170 CALL KEY(0,K,S):: IF S=0
 THEN 170 ELSE IF K=32 THEN
180 ELSE 170
180 IF EOF(1)THEN 340
190 GOTO 130
200 PRINT "1=TEXT": :"2=GRAP
HICS"
210 INPUT B :: IF B<1 OR B>2
 THEN 210 ELSE CALL CLEAR
220 OPEN #3:PR$,VARIABLE 132
 :: PRINT #3:CHR$(27)&"@";
230 ON B GOTO 240,250
240 PRINT #3:CHR$(27);CHR$(7
8);CHR$(3):: GOTO 280
250 PRINT "1=COMPRESSED": :"
2=FULL SIZE": :"3=FULL SIZE,
EMPHASIZED" :: INPUT C
260 IF C<1 OR C>3 THEN 250 E
LSE IF C=1 THEN 270 ELSE IF
C=3 THEN PRINT #3:CHR$(27)&"
E" :: GOTO 280 ELSE 280
270 PRINT #3:CHR$(27);CHR$(6
5);CHR$(7);CHR$(15);
280 LINPUT #1:X$
290 IF B=2 AND C=1 THEN PRIN
T #3:"
       ";X$ :: GOTO 310
300 PRINT #3:X$
310 IF EOF(1)THEN 330
320 GOTO 280
330 PRINT #3:CHR$(27);CHR$(7
9):: CLOSE #1 :: CLOSE #3 ::
 INPUT "FINISHED?(Y/N)":AN$
:: IF AN$="N" THEN 350 ELSE
STOP
340 CLOSE #1 :: END
350 INPUT "FORM FEED?(Y/N)":
AN$ :: IF AN$="Y" THEN OPEN
#1:PR$ :: PRINT #1:CHR$(12):
: CLOSE #1
360 INPUT "DSK"&FN$&" AGAIN?
(Y/N)":AN$ :: IF AN$="Y" THE
N 80 ELSE 60                ■
```

HEX DEC BIN

(repris fr. PB 84-5.13)

```
90 !HEX-DEC-BI.Ett program f
oer konvertering av tal.
95 !Input i en form ger utsk
rift i de tva andra.
96 !Ur The Smart Programmer.
100 ON WARNING NEXT :: CALL
CLEAR :: H$="0123456789ABCDE
F" :: PRINT "LAAS ALFA LOCK-
TANGENTEN!": :"VAELJ INPUT-B
AS- D,H ELLER B": :
110 PRINT : :"D=DEC #  H=HEX
 #  B=BIN #": : :: CALL SOUN
D(40,660,9)
120 CALL KEY(0,K,S):: IF S<1
 THEN 120 ELSE ON POS("DHB",
CHR$(K),1)+1 GOTO 110,130,14
0,150
130 INPUT "DEC #=":DEC :: IF
 DEC<-32768 OR DEC>65535 THE
N 130 ELSE A,DEC=INT(DEC-655
36*(DEC<0)):: GOSUB 200 :: G
OSUB 220 :: GOTO 160
140 PRINT "HEX #=" :: ACCEPT
 AT(23,7)BEEP SIZE(4)VALIDAT
E(H$):HEX$ :: GOSUB 180 :: G
OSUB 200 :: GOTO 160
150 PRINT "BIN #=" :: ACCEPT
 AT(23,7)BEEP SIZE(16)VALIDA
TE("10"):BIN$ :: GOSUB 190 :
: GOSUB 220 :: GOSUB 210
160 A=INT(DEC/256):: PRINT :
"D=";DEC;TAB(12);A;DEC-A*256
 :: IF DEC>32767 THEN PRINT
"  ";DEC-65536
170 PRINT "H= ";HEX$:"B= ";S
EG$(BIN$,1,8)&" "&SEG$(BIN$,
9,8):: HEX$,BIN$="" :: A,DEC
=0 :: GOTO 110
180 HEX$=SEG$("0000",1,4-LEN
(HEX$))&HEX$ :: FOR I=1 TO 4
 :: A,DEC=DEC+(POS(H$,SEG$(H
EX$,I,1),1)-1)*16^(4-I):: NE
XT I :: RETURN
190 FOR I=1 TO LEN(BIN$):: D
EC=DEC-2^(I-1)*(SEG$(BIN$,(L
EN(BIN$)+1-I),1)="1"):: NEXT
 I :: RETURN
200 A=A/2 :: BIN$=STR$(-(A-I
NT(A)<>0))&BIN$ :: A=INT(A):
: IF A THEN 200
210 BIN$=SEG$(RPT$("00",8),1
,16-LEN(BIN$))&BIN$ :: RETUR
N
220 A=DEC+65536*(DEC>32767)
230 HEX$=SEG$(H$,(INT(A/4096
)AND 15)+1,1)&SEG$(H$,(INT(A
/256)AND 15)+1,1)&SEG$(H$,(I
NT(A/16)AND 15)+1,1)&SEG$(H$
,(A AND 15)+1,1):: RETURN   ■
```

MUSIK 1

av Lars-Erik Svahn

(repris fr. PB 86-2.14)

```
100 GOSUB 340 !INITIALIZE
110 !
120 FOR I=1 TO 2
130 CALL TRIO(2,A2,E2,C2)
140 CALL TRIO(2,B2,E2,G1)
150 CALL TRIO(2,C3,A1,E2)
160 CALL TRIO(2,D3,A2,F2)
170 CALL TRIO(4,E3,A2,C2)
180 CALL TRIO(4,C3,E2,A1)
190 CALL TRIO(4,B2,E2,D2)
200 CALL TRIO(4,A2,E2,C2)
210 CALL TRIO(2,B2,F@2,D1)
220 CALL TRIO(2,F@2,D1,B1)
230 CALL TRIO(2,C3,F@2,D1)
240 CALL TRIO(2,B2,G@2,D2)
250 CALL TRIO(4,A2,C2,A0)
260 CALL TRIO(4,E2,E1,E1)
270 CALL TRIO(4,C2,C1,C1)
280 CALL TRIO(4,A1,A0,A0)
290 NEXT I
300 CALL QUIET
310 STOP
320 !
330 !INITIALIZE
340 A0=110 :: A@0=117
350 B0=123
360 C1=131 :: C@1=139
370 D1=147 :: D@1=156
380 E1=165
390 F1=175 :: F@1=185
400 G1=196 :: G@1=208
410 A1=220 :: A@1=233
420 B1=247
430 C2=262 :: C@2=277
440 D2=294 :: D@2=311
450 E2=330
460 F2=349 :: F@2=370
470 G2=392 :: G@2=415
480 A2=440 :: A@2=466
490 B2=494
500 C3=523 :: C@3=554
510 D3=587 :: D@3=622
520 E3=659
530 F3=698 :: F@3=740
540 G3=784 :: G@3=831
550 A3=880 :: A@3=932
560 B3=988
570 PSE=20000
580 RETURN
590 !
600 SUB TRIO(T,P,H,C)
610 FOR A=0 TO 28 STEP 0.9*T
 :: CALL SOUND(-500,P,A,H,20
,C,16):: NEXT A
620 SUBEND
630 !
640 SUB QUIET
650 CALL SOUND(1,110,29,110,
29,110,29)
660 SUBEND
670 !
680 END                     ■
```

LOVE YOU TRULY

by Bob August, USA

The program this month is a sing along. Hope you enjoy.

```
100 ! I LOVE YOU TRULY
    ! IN TI EXTENDED BASIC
    ! BY R.W. AUGUST
110 CALL CLEAR :: CALL SCREE
N(16):: CALL COLOR(13,1,7)::
 CALL CHAR(128,"002255494122
1408")
120 DISPLAY AT(12,7):"I LOVE
 YOU TRULY" :: CALL VCHAR(1,
31,128,96):: CALL HCHAR(24,1
,128,64)
130 LLA=110 :: LLB=123 :: LC
=131 :: LD=147 :: LE=165 ::
LFS=185 :: LG=196 :: LA=
220 :: LB=247 :: C=262 :: D=
294 :: EF=311 :: E=330
140 FS=370 :: G=392 :: A=440
 :: B=494 :: HC=523 :: N2=30
00 :: N4=1500 :: N8=750 :: N
16=375 :: FOR DELAY=1 TO 500
 :: NEXT DELAY
150 DISPLAY AT(12,5):"     I
 LOVE YOU" :: CALL SOUND(N8,
D,0,LB,5,LD,8):: CALL SOUND(
N8,E,0,LB,5,LD,8):: CALL SOU
ND(N8,G,0,LB,5,LD,8)
160 DISPLAY AT(12,5):"
 TRU-LY," :: CALL SOUND(N4,G
,0,C,5,LLA,8):: CALL SOUND(N
8+N16,FS,0,C,5,LD,8):: CALL
SOUND(N16,FS,0,C,5,LD,8)
170 DISPLAY AT(12,5):" "
180 CALL SOUND(N4,HC,0,E,5,L
LA,8):: DISPLAY AT(12,12):"T
RU-LY," :: CALL SOUND(N8,HC,
0,E,5,LLA,8):: CALL SOUND(N8
,HC,0,EF,5,LD,8)
190 DISPLAY AT(12,11):"  DEA
R." :: CALL SOUND(N2,B,0,LD,
5)
200 CALL SOUND(N8,D,0,LB,5,L
LA,8):: DISPLAY AT(12,8):"LI
FE WITH ITS" :: CALL SOUND(N
8,E,0,LB,5,LD,8):: CALL SOUN
D(N8,G,0,D,5,LD,8)
210 DISPLAY AT(12,8):"  SOR
-ROW," :: CALL SOUND(N4,G,0,
C,5,LLA,8):: CALL SOUND(N4,F
S,0,C,5,LD,8)
220 DISPLAY AT(12,8):"LIFE W
ITH ITS" :: CALL SOUND(N4,HC
,0,C,5,LLA,8):: CALL SOUND(N
8,HC,0,E,5,LLA,8):: CALL SOU
ND(N8,HC,0,EF,5,LD,8)
230 DISPLAY AT(12,8):"     T
EAR," :: CALL SOUND(N2,B,0,G
,5,LG,8)
240 CALL SOUND(N4,B,0,FS,5,L
B,8):: DISPLAY AT(12,10):"FA
DE IN-TO" :: CALL SOUND(N8+N
16,B,0,FS,5,LB,8):: CALL SOU
ND(N16,A,0,EF,5,LLB,8)
250 DISPLAY AT(12,8):"DREAMS
 WHEN I" :: CALL SOUND(N4,G,
0,E,5,LE,8):: CALL SOUND(N8,
G,0,LB,5,LE,8):: CALL SOUND(
N8,G,0,LB,5,LE,8)
260 DISPLAY AT(12,8):" FEEL
 YOU ARE" :: CALL SOUND(N4,B,
0,E,5,LA,8):: CALL SOUND(N8,
G,0,E,5,LA,8):: CALL SOUND(N
8,G,0,E,5,LE,8)
270 DISPLAY AT(12,9):"    NE
AR," :: CALL SOUND(N2,E,0,LE
,5,LA,8)
280 CALL SOUND(N8,D,0,LB,5,L
D,8):: DISPLAY AT(12,8):"FOR
 I LOVE YOU" :: CALL SOUND(N
8,D,0,LB,5,LD,8):: CALL SOUN
D(N8,E,0,LB,5,LE,8)
290 CALL SOUND(N8,E,0,LB,5,L
E,8):: DISPLAY AT(12,8):"
 TRU-LY," :: CALL SOUND(N8,B
,0,D,5,LD,8):: CALL SOUND(N4
+N16,B,0,D,5,LFS,8)
300 DISPLAY AT(12,12):" " ::
 DISPLAY AT(12,12):" "
310 DISPLAY AT(12,12):"TRU-L
Y," :: CALL SOUND(N4,D,0,LC,
5,LLA,8):: CALL SOUND(N4,B,0
,LC,5,LLA,8)
320 DISPLAY AT(12,12):" DEAR
." :: CALL SOUND(N2+N8,G,0,L
B,5,LD,8)
330 CALL SOUND(N8,D,0,LB,5,L
D,8):: DISPLAY AT(12,8):"AH!
 LOVE, 'TIS" :: CALL SOUND(N
8,E,0,LB,5,LD,8):: CALL SOUN
D(N8,G,0,LB,5,LD,8)
340 DISPLAY AT(12,8):"SOME-T
HING TO," :: CALL SOUND(N4,G
,0,C,5,LLA,8):: CALL SOUND(N
8+N16,FS,0,C,5,LD,8):: CALL
SOUND(N16,FS,0,C,5,LD,8)
350 DISPLAY AT(12,8):"FEEL Y
OUR KIND" :: CALL SOUND(N4,H
C,0,E,5,LLA,8):: CALL SOUND(
N8,HC,0,E,5,LLA,8):: CALL SO
UND(N8,HC,0,EF,5,LD,8)
360 DISPLAY AT(12,8):"     H
AND." :: CALL SOUND(N2,B,0,L
D,5)
370 CALL SOUND(N8,D,0,LB,5,L
LA,8):: DISPLAY AT(12,8):"AH
! YES, 'TIS" :: CALL SOUND(N
8,E,0,LB,5,LD,8):: CALL SOUN
D(N8,G,0,D,5,LD,8)
380 DISPLAY AT(12,8):"SOME-T
HING BY" :: CALL SOUND(N4,G,
0,C,5,LLA,8):: CALL SOUND(N4
,FS,0,C,5,LD,8)
390 DISPLAY AT(12,8):" YOUR
SIDE TO" :: CALL SOUND(N4,HC
,0,C,5,LLA,8):: CALL SOUND(N
8,HC,0,E,5,LLA,8):: CALL SOU
ND(N8,HC,0,EF,5,LD,8)
400 DISPLAY AT(12,8):"    ST
AND," :: CALL SOUND(N2,B,0,G
,5,LG,8)
410 CALL SOUND(N4,B,0,FS,5,L
B,8):: DISPLAY AT(12,8):" GO
NE IS THE" :: CALL SOUND(N8+
N16,B,0,FS,5,LB,8):: CALL SO
UND(N16,A,0,EF,5,LLB,8)
420 DISPLAY AT(12,9):" SOR-
ROW," :: CALL SOUND(N4,G,0,E
,5,LE,8):: CALL SOUND(N8,G,0
,LB,5,LE,8):: CALL SOUND(N8,
G,0,LB,5,LE,8)
430 DISPLAY AT(12,8):"GONE D
OUBT AND" :: CALL SOUND(N4,B
,0,E,5,LA,8):: CALL SOUND(N8
,G,0,E,5,LA,8):: CALL SOUND(
N8,G,0,E,5,LE,8)
440 DISPLAY AT(12,8):"     F
EAR" :: CALL SOUND(N2,E,0,LE
,5,LA,8)
450 CALL SOUND(N8,D,0,LB,5,L
D,8):: DISPLAY AT(12,7):"FOR
 YOU LOVE ME" :: CALL SOUND(
N8,D,0,LB,5,LD,8):: CALL SOU
ND(N8,E,0,LB,5,LE,8)
460 CALL SOUND(N8,E,0,LB,5,L
E,8):: DISPLAY AT(12,7):"
  TRU-LY," :: CALL SOUND(N8,
B,0,D,5,LD,8):: CALL SOUND(N
4+N16,B,0,D,5,LFS,8)
470 DISPLAY AT(12,12):" "
480 DISPLAY AT(12,12):" " ::
 DISPLAY AT(12,12):"TRU-LY,"
 :: CALL SOUND(N4,D,0,LC,5,L
LA,8):: CALL SOUND(N4,B,0,LC
,5,LLA,8)
490 DISPLAY AT(12,12):" DEAR
." :: CALL SOUND(N2+N8,G,0,L
B,5,LD,8):: DISPLAY AT(12,5)
:"Play it again? Y/N Y"
500 ACCEPT AT(12,24)VALIDATE
("YyNn")SIZE(-1):YN$ :: IF Y
N$="Y" OR YN$="y" THEN 150 E
LSE CALL CLEAR :: END
```

■

KOM IHÅG

Extra stämma
21 jan 1995 kl 13.00.
Ordinarie stämma
11 mars 1995 kl 13.00 ■