

INSIDE THIS ISSUE

3 X 5 CARD CATALOG by Ed Yorkt (CIN-DAY Users Group)

POSTCARD by John Behnke (Chicago TI-99/4A Users Group)

ASSEMBLY DISK CATALOG by Tom Freeman (LA 99ers)

CALANDER GENERATOR BY Norman Weiss (SFV 99ers)

PRESIDENT'S NOTES...by L. R. Livergood

Interest in home computing seems to be gaining momentum this spring and hopefully everyone will benefit. Thanks to the effort put forth by a few of the members, we have seen several good buys on full systems in addition to having one or two new members sign up during the last few months. Anyone who is still interested in purchasing a full system at a reasonable price, please talk to Steve Thorpe at the meeting.

The latest version of FUNLWRITER has arrived and will be available for distribution at the meeting. For those of you who are not familiar with this super program, you don't know what you are missing out on. This program combines TI-WRITER and DM-1000 along with several others to provide you with everything necessary to meet the majority of your programming needs. Minimum system requirements include the console, one disk drive, and 32K memory expansion. Two disk drives, a printer and a RAMdisk are recommended but not required.

For those looking to buy blank disks, the group has access to BONUS brand disks at a reasonable price. You may purchase ten for \$7.50. I believe they are produced by Verbatim and come with a warranty. The catch is that we will have to take orders prior to getting them and the order will have to total about \$100.00. Those who are interested talk to Steve Thorpe and place your order. If the group can afford it, we will go ahead and fill in what we are missing for future sales.

Next month is election month. We have already talked to a few of you and have put together a minimal slate of officers. It is imperative that everyone try to help out so that no one will be overwhelmed with too much work. Please let us know if you have the time and can help lighten the load.

We are interested in suggestions for topics for our monthly meetings. Several members have requested that the meetings become more structured and possibly include software reviews and maybe some programming tips. Since many of us are at various levels of understanding, we do not want to leave anyone out. The best way to handle this is by hearing from everyone. Don't be afraid to make a suggestion because even the simplest ideas can evolve into complex applications.

Finally, in order to continue the newsletter I feel we need some local input. Some of us I know are capable of writing a nice review or develop a handy little time-saving program which would benefit others. I suggest we take a minute to think about how we personally use our TI computers and write an article on it.

```

100 ! *****
110 ! * 3 x 5 CARD CATALOG *
120 ! *****
130 ! Original: Ed York
    Revamped: Rick Kellogg
    Revision: Ed York
140 !
150 DIM A$(12):: FOR A=1 TO
12 :: READ A$(A):: NEXT A ::
CALL CLEAR
160 DATA JAN,FEB,MAR,APR,MAY
,JUN,JUL,AUG,SEP,OCT,NOV,DEC
170 DISPLAY AT(9,9):"DISK LA
BELER": " ENTER TODAYS
DATE": :TAB(11);"MM/DD/YY":
: :TAB(11);" _/_/8_"
180 ACCEPT AT(16,11)BEEP SIZ
E(-2)VALIDATE(DIGIT):B$ :: I
F B$="_" THEN 180 ELSE IF (
VAL(B$)>12)+(VAL(B$)<1)THEN
170
190 ACCEPT AT(16,14)SIZE(-2)
VALIDATE(DIGIT)BEEP:C$ :: IF
C$="_" THEN 190 ELSE IF (V
AL(C$)>31)+(VAL(C$)<1)THEN 1
90
200 ACCEPT AT(16,18)SIZE(-1)
VALIDATE(DIGIT)BEEP:D$ :: D$
="0"&D$ :: GOSUB 610
210 E$=A$(VAL(B$))&"%C&%",
19"&D$ :: F$(1)="D/F" :: F$
(2)="D/U" :: F$(3)="I/F" ::
F$(4)="I/V" :: F$(5)="PROGRM
"
220 IMAGE "DISKNAME: *****
### FREE:#### USED:####"
230 IMAGE "***** SIDED/####
## DENSITY *****"
240 IMAGE "***** ###
#####
##### "
250 IMAGE "***** ###
#####
##### "
260 IMAGE "***** ###
#####
##### "
270 OPEN #1:"PIO",VARIABLE 1
00 :: PRINT #1:CHR$(15);CHR$(
27);"S";CHR$(0);CHR$(27);"A
";CHR$(5)
280 OPEN #2:"DSK"&STR$(8)&"
",INPUT ,RELATIVE,INTERNAL :
: INPUT #2:5,C,C,D
290 PRINT #1:RPT$(=" ",86);CH
R$(27);"A";CHR$(3):: E=0
300 PRINT #1:CHR$(27);"M";CH
R$(1):: PRINT #1,USING 220:5
$,D,C-D

```

```

310 IF C>720 AND C<1441 THEN
H$="DOUBLE" :: I$="DOUBLE"
320 IF C>360 AND C<721 THEN
H$="SINGLE" :: I$="DOUBLE"
330 IF C<361 THEN H$="SINGLE
" :: I$="SINGLE"
340 PRINT #1:CHR$(14):: PRIN
T #1,USING 230:H$,I$,E$ :: P
RINT #1:CHR$(27);"M";CHR$(0)
350 PRINT #1:RPT$(=" ",86);CH
R$(27);"A";CHR$(5)
360 PRINT #1,USING 240:" Fil
ename ", "Size", " Type ", "P",
" Filename ", "Size", " Type
", "P", " Filename ", "Size", " T
ype ", "P"
370 PRINT #1,USING 240:"----
-----", "-----", "-----", "-----",
"-----", "-----", "-----", "-----",
"-----", "-----", "-----", "-----",
"-----", "-----"
380 IF E=126 THEN 620
390 FOR F=1 TO 3 :: INPUT #2
:J$(F),G(F),H(F),I(F):: NEXT
F :: IF LEN(J$(1))=0 THEN 4
40 ELSE IF LEN(J$(2))=0 THEN
60SUB 490 ELSE 410
400 PRINT #1,USING 260:J$(1)
,H(1),K$(1),L$(1):: E=E+1 ::
GOTO 380
410 IF LEN(J$(3))=0 THEN 60S
UB 490 :: GOSUB 530 :: PRINT
#1,USING 250:J$(1),H(1),K$(
1),L$(1),J$(2),H(2),K$(2),L$(
2):: E=E+2 :: GOTO 380
420 GOSUB 490 :: GOSUB 530 :
: GOSUB 570
430 PRINT #1,USING 240:J$(1)
,H(1),K$(1),L$(1),J$(2),H(2)
,K$(2),L$(2),J$(3),H(3),K$(3)
,L$(3):: E=E+3 :: GOTO 380

```

```

440 PRINT #1: : PRINT #1:CH
R$(18);CHR$(27);CHR$(50):: C
LOSE #2 :: CLOSE #1 :: DISPL
AY AT(20,1):"Want another Co
py or Disk? Y": " "
450 ACCEPT AT(20,28)SIZE(-1)
VALIDATE("YN")BEEP:M$ :: IF
M$="Y" THEN GOSUB 610 :: GOT
O 470 ELSE CALL CLEAR
460 OPEN #1:"PIO" :: PRINT #
1:CHR$(27);"T" :: CLOSE #1 :
: STOP
470 DISPLAY AT(20,1)BEEP:"
INSERT DISK INTO DRIVE "&STR
$(8): " PRESS ANY KEY TO
BEGIN "
480 CALL KEY(0,J,K):: IF K<1
THEN 480 ELSE 270
490 IF G(1)>0 THEN L$(1)=" "
ELSE L$(1)="Y"
500 IF ABS(G(1))=5 THEN K$(1)
)=F$(5):: RETURN ELSE A=LEN(
STR$(I(1)))
510 IF A=1 THEN K$(1)=F$(ABS
(G(1)))&" "&STR$(I(1)):: RE
TURN
520 IF A=2 THEN K$(1)=F$(ABS
(G(1)))&" "&STR$(I(1)):: RET
URN ELSE K$(1)=F$(ABS(G(1)))
&STR$(I(1)):: RETURN
530 IF G(2)>0 THEN L$(2)=" "
ELSE L$(2)="Y"
540 IF ABS(G(2))=5 THEN K$(2)
)=F$(5):: RETURN ELSE A=LEN(
STR$(I(2)))
550 IF A=1 THEN K$(2)=F$(ABS
(G(2)))&" "&STR$(I(2)):: RE
TURN
560 IF A=2 THEN K$(2)=F$(ABS
(G(2)))&" "&STR$(I(2)):: RET
URN ELSE IF A=3 THEN K$(2)=F
$(ABS(G(2)))&STR$(I(2)):: RE
TURN

```

```

570 IF G(3)>0 THEN L$(3)=" "
ELSE L$(3)="Y"
580 IF ABS(G(3))=5 THEN K$(3)
)=F$(5):: RETURN ELSE A=LEN(
STR$(I(3)))
590 IF A=1 THEN K$(3)=F$(AB
S(G(3)))&" "&STR$(I(3)):: RE
TURN
600 IF A=2 THEN K$(3)=F$(ABS
(G(3)))&" "&STR$(I(3)):: RET
URN ELSE IF A=3 THEN K$(3)=F
$(ABS(G(3)))&STR$(I(3)):: RE
TURN
610 DISPLAY AT(24,7):"CATALO
G DRIVE: 1" :: ACCEPT AT(24
,23)BEEP SIZE(-1)VALIDATE("1
234"):0 :: RETURN
620 INPUT #2:J$(1),G(1),H(1)
,I(1):: GOSUB 490 :: PRINT #
1,USING 250:J$(1),H(1),K$(1)
,L$(1):: GOTO 440
630 END

```



DISKNAME: NUTS&BOLTS FREE: 12 USED: 346				SINGLE SIDED/SINGLE DENSITY FEB 16, 1987			
Filename	Size	Type	P	Filename	Size	Type	P
ALTERNATE	D/V/163			AVERAGE	D/V/163		
BISCHAR	D/V/163			BISMAR2	D/V/163		
BORDER	D/V/163			CALENDAR	D/V/163		
CARDS	D/V/163			CHARELEON	D/V/163		
CHARSAVE	D/V/163			CHARSAVEZ	D/V/163		
CHECK	D/V/163			CHECKSAY	D/V/163		
COLORTEXT	D/V/163			COLUMNIZER	D/V/163		
CRAWL	D/V/163			CURSOR	D/V/163		
CURTAIN	D/V/163			CURTAINZ	D/V/163		
FACTORIAL	D/V/163			FADE-IN	D/V/163		
FAILSAFE	D/V/163			FASTJOY	D/V/163		
FLASH	D/V/163			FLASH-ON	D/V/163		
HEX-DEC	D/V/163			HIGHLOW	D/V/163		
HIGHSCRAN	D/V/163			HOLD	D/V/163		
INVERSE	D/V/163			JISBLE	D/V/163		
JOYWRAP	D/V/163			KEYBOARD	D/V/163		
LARBECHAR	D/V/163			LOAD	PROGRM		
LONGSHELL	D/V/163			LOWERCASE	D/V/163		
HIGHSCALE	D/V/163			MIRROR	D/V/163		
MONTH	D/V/163			MUSIC	D/V/163		
NOTICE	PROGRM			OUTSIDE-IN	D/V/163		
PACKNUM	D/V/163			PASSWORD	D/V/163		
PLAY	D/V/163			PLAYORGAN	D/V/163		
QUICKSORT	D/V/163			QUICKBORN	D/V/163		
REARHUR	D/V/163			REARSPACE	D/V/163		
RESORTNUM	D/V/163			RESORTSORT	D/V/163		
SAVESTRIN	D/V/163			SCALE	D/V/163		
SCREENSAVE	D/V/163			SCRUNCH	D/V/163		
SHOEHORN	D/V/163			SCHOENH-N	D/V/163		
SLANT	D/V/163			SLASH	D/V/163		
SPRITEHOW	D/V/163			TASTURNS	D/V/163		
TITLE	D/V/163			TUTORIAL	PROGRM		
TWOMAY	D/V/163			UPDOWNMPE	D/V/163		
WAIT	D/V/163			WAITING	D/V/163		
WIPES	D/V/163						
				BELL	D/V/163		
				BINARYDEC	D/V/163		
				CALLBELL	D/V/163		
				CHARFACE	D/V/163		
				CHARSETZ	D/V/163		
				CLOCK	D/V/163		
				COUNTER	D/V/163		
				CURSORCONT	D/V/163		
				DIFFERENT	D/V/163		
				FADE-OUT	D/V/163		
				FLAG	D/V/163		
				FORMATTER	D/V/163		
				INTERLUSE	D/V/163		
				JOYSELECT	D/V/163		
				KILLSUIT	D/V/163		
				LONGSHLL	D/V/163		
				MAJORSKAL	D/V/163		
				NO-REPEAT	D/V/163		
				NO-REPEAT	D/V/163		
				PACKING	D/V/163		
				PAUSE	D/V/163		
				PRICE	D/V/163		
				REARSCREEN	D/V/163		
				REARSCREEN	D/V/163		
				RUBBIAN	D/V/163		
				SCREENRID	D/V/163		
				SECTORS	D/V/163		
				SHUTOFF	D/V/163		
				SLASHZER0	D/V/163		
				TIBER	D/V/163		
				TUBIE	D/V/163		
				UPSIDEOWN	D/V/163		
				WAITRUSIC	D/V/163		



```

THEN 970
980 IF H=88 OR H=120 OR H=65
OR H=101 THEN CALL HCHARC3+
AA*2,7,32J
990 IF H=88 OR H=120 THEN AA
-AA+1
1000 IF H=69 OR H=101 THEN A
A-AA-1
1010 IF AA=0 THEN AA=9
1020 IF AA=10 THEN AA=1
1030 IF H<13 THEN 960
1040 ON AA GOTO 1050,1050,13
10,550,1100,1270,1440,1830,1
280
1050 DISPLAY AT(24,1):"FILEN
AME: DSK1,CARD":ACCEPT AT
(24,14)BEEP SIZE(-15):K$
1060 K$="DSK"&K$:OPEN #1:
K$:FOR A=1 TO 21
1070 IF AA=1 THEN LINPUT #1:
L$:AS(A)-SEG$(L$,1,28):
BS(A)-SEG$(L$,29,28)
1080 IF AA=2 THEN PRINT #1:A
S(A):BS(A)
1090 NEXT A:CLOSE #1:G
OTO 840
1100 DISPLAY AT(1,8)ERASE AL
L:"PRINTER MODES":PRINTER
NAME:PID"
1110 ACCEPT AT(3,15)BEEP SIZ
EC-14):J$
1120 DISPLAY AT(5,1):"<N> NL
G ON":<N> NLQ OFF":<N> PIC
A MODE":<N> ELITE MODE"
1130 DISPLAY AT(9,1):"<N> CO
NDENSED MODE":<N> EXPANDED
MODE ON"
1140 DISPLAY AT(11,1):"<N> E
XPANDED MODE OFF":<N> DOUBL
E STRIKE ON"
1150 DISPLAY AT(13,1):"<N> D
OUBLE STRIKE OFF"
1160 DISPLAY AT(14,1):"<N> E
MPHASIZED ON":<N> EMPHASIZE
D OFF"
1170 DISPLAY AT(16,1):"<N> 1
/6 INCH LINE SPACING <N> 7
/72 INCH LINE SPACING"
1180 DISPLAY AT(18,1):"<N> 1
/8 INCH LINE SPACING <N> U
N1-DIRECTION PRINT ON"
1190 DISPLAY AT(20,1):"<N> U
N1-DIRECTION PRINT OFF <N> E
NTER OTHER LINE SPACING"
1200 FOR A=5 TO 20:ACCEPT
AT(A,2)VALIDATE(25)BEEP SIZ
EC-1):I$
1210 IF I$="Y" OR I$="U" THE
N OPEN #1:J$:PRINT #1:DSK
A-4):CLOSE #1
1220 NEXT A:ACCEPT AT(21,
2)VALIDATE(25)BEEP SIZE(-1):
I$
1230 IF I$="N" OR I$="n" THE
N 840
1240 DISPLAY AT(21,1):"NEW L
INE SPACING -12/72 inch"
1250 ACCEPT AT(21,19)VALIDAT
E(25)BEEP SIZE(-2):I:O
PEN #1:J$
1260 PRINT #1:HS&"A"&CHR$(1)
:CLOSE #1:GOTO 840
1270 FOR A=1 TO 21:AS(A),
BS(A):NEXT A:GOTO 8
40
1280 DISPLAY AT(24,1):"
ARE YOU SURE? N"
1290 ACCEPT AT(24,21)VALIDAT
E(25)BEEP SIZE(-1):I$:IF
I$="N" OR I$="n" THEN 84
0
1300 CALL CLEAR:END
1310 DISPLAY AT(1,7)ERASE AL
L:"SPECIAL GRAPHICS"
1320 DISPLAY AT(4,3):CHR$(12
6)8"FCIN W"CHR$(127)
8"FCIN U"
1330 DISPLAY AT(6,3):CHR$(12
8)8"CTRL"CHR$(129)
8"CTRL A"
1340 DISPLAY AT(8,3):CHR$(13
0)8"CTRL B"CHR$(131)
8"CTRL C"
1350 DISPLAY AT(10,3):CHR$(1
32)8"CTRL D"CHR$(133)
18"CTRL E"
1360 DISPLAY AT(12,3):CHR$(1
34)8"CTRL F"CHR$(135)
18"CTRL G"
1370 DISPLAY AT(14,3):CHR$(1
36)8"CTRL H"CHR$(137)
18"CTRL I"
1380 DISPLAY AT(16,3):CHR$(1
38)8"CTRL J"CHR$(139)
18"CTRL K"
1390 DISPLAY AT(18,3):CHR$(1
40)8"CTRL L"CHR$(141)
18"CTRL M"
1400 DISPLAY AT(20,3):CHR$(1
42)8"CTRL N"CHR$(143)
18"CTRL O"
1410 DISPLAY AT(24,4):"PRESS
ANY KEY TO EDIT"
1420 CALL KEY(0,D,H):IF H=
O THEN 1420
1430 GOTO 1430
1440 DISPLAY AT(1,9)ERASE AL
L:"CATALOG DISK":CALL HCH
AR(2,1,132,32)
1450 DISPLAY AT(5,1):"DRIVER?

```

```

(0-5):<1>"
1460 ACCEPT AT(5,16)VALIDATE
("012345")BEEP SIZE(-1):MS
1470 IF MS<>"0" THEN 1490 EL
SE DISPLAY AT(5,1):"DEVICE N
AME? <
1480 ACCEPT AT(5,15)BEEP SIZ
EC-13):MS
1490 MS="DSK"&MS&"":DISP
LAY AT(6,1):"OUTPUT TO PRINT
ERR (Y/N):N"
1500 ACCEPT AT(6,27)VALIDATE
(25)BEEP SIZE(-1):I$:IF I
$="U" THEN I$="Y"
1510 IF I$<>"Y" THEN 1530 EL
SE DISPLAY AT(6,1):"PRINTER
NAME:PID"
1520 ACCEPT AT(6,15)BEEP SIZ
EC-14):J$
1530 DISPLAY AT(8,1):"READIN
G...:":A TO ABORT"
1540 CALL HCHAR(10,2,126):
CALL HCHAR(10,3,127,28):CA
LL HCHAR(11,2,129)
1550 CALL HCHAR(10,31,128):CAL
L HCHAR(11,31,130):CAL
L HCHAR(12,2,131)
1560 CALL HCHAR(12,3,132,28)
:CALL HCHAR(12,31,133)
1570 OPEN #1:MS,INPUT #1,RELAT
IVE,INTERNAL:INPUT #1:MS,
G,G,D:FOR A=1 TO 127
1580 CALL KEY(0,J,H):IF J=
65 OR J=97 THEN 1640
1590 INPUT #1:OS,K,L,M:IF
LEN(OS)=0 THEN 1640
1600 GS(A)-OS&RPT$( " ",11-LE
N(OS))&STR$(L&RPT$( " ",5-LE
N(STR$(L)))&FS(A&BS(K)))
1610 IF ABS(K)<5 THEN PS=" "
&STR$(M):GS(A)-GS(A)&SEGS
P$,LEN(PS)-2,3)
1620 IF K<1 THEN GS(A)-GS(A)
&"Y"
1630 DISPLAY AT(11,1):GS(A):
NEXT A
1640 CLOSE #1
1650 CALL CLEAR:IF I$="Y"

```

```

THEN OPEN #1:J$
1660 PRINT TAB(9):"A TO ABOR
T":TAB(9):"P TO PAUSE":
1670 PRINT MS:"-DISKNAME-
":NS:"AVAILABLE":D:"USED-":
G-D
1680 IF I$="Y" THEN PRINT #1
:MS:"-DISKNAME-":NS:"AVAI
LABLE":D:"USED-":G-D
1690 PRINT #1:FILENAME SIZE
FILENAME P-----
1700 IF I$="N" THEN 1720
1710 PRINT #1:FILENAME SI
ZE FILETYPE P:"-----"
1720 FOR A=1 TO A:CALL KE
Y(0,D,H):IF D=65 OR D=97 T
HEN 1760
1730 IF D=80 OR D=112 THEN 1
810
1740 PRINT GS(A):IF I$="Y"
THEN PRINT #1:GS(A)
1750 NEXT A
1760 IF I$="Y" THEN CLOSE #1
1770 PRINT #1:LIST AGAIN (Y/N
)?"
1780 CALL KEY(0,D,H):IF H=
O THEN 1780
1790 IF D=89 THEN 1650
1800 GOTO 840
1810 FOR D=1 TO 50:NEXT D
:CALL KEY(0,D,H):IF H=O
THEN 1810
1820 GOTO 1740
1830 DISPLAY AT(24,1):"SCREE
N COLOR? (2-16):<2>"
1840 ACCEPT AT(24,24)VALIDAT
E(25)BEEP SIZE(-2):I
1850 DISPLAY AT(24,1):"LETTE
R COLOR? (2-16):<16>"
1860 ACCEPT AT(24,24)VALIDAT
E(25)BEEP SIZE(-2):A
1870 IF A=1 THEN 1830
1880 CALL SCREEN(I):FOR I=
1 TO 14:CALL COLOR(I,A,1)
:NEXT I
1890 GOTO 840
1900 CALL CLEAR:PRINT "ER
ROR, RE-BOOTING..." :RUN

```

# ASSEMBLY DISK CATALOG

by Tom Freeman

My subject for this month is the mysterious cataloging routine present in most storage devices. It should be instructive as to how this routine works, and also with regard to floating point numbers, which are not easy to understand.

Do you all remember typing in that "catalog" contained in the Disk Drive Manual? I remember actually believing at that time (in my primitive frame of mind) that there was a REAL file on the disk that contained the catalog, with real records etc. Boy did I get weird results when I tried to manipulate it in any way except the standard way that it was given to us. Well the reason is... Of course there is no file! What there IS, is a subprogram in the DSR of the card involved (floppy disk controller, ram disk, etc.) that is accessed when you open a dummy "file" that contains the device name and ends in period. What is returned to the PAB buffer with each record is a string and three numbers. THESE are in a standard format as if there were a real file. Each is preceded by a length byte (in the case of numbers, which are always stored in floating point format, this is 8). The first record contains the name of the directory or disk, a zero, the total number of sectors possible on the disk, and the number available. This information is contained in sector 9. The program actually has to compute the numbers. Each subsequent record will yield a file name, the file type (represented by a number from 1 to 5), the number of sectors used, and the logical record length. This information is obtained from the alphabetical list of pointers in sector 1, and each FDR it points to. My program takes advantage of this built in routine. I decided not to use direct sector access, because this requires a "drive number" and I wished to retain the ability to use device names not assigned to numbers (e.g. RD for Myarc RAMdisk). It was also meant to be able to catalog a hard disk, unfortunately this will require some additional code which I will publish within the next few months.

The second item of interest is "floating point numbers." These are always represented by 8 bytes. The number is expressed in RADIX 100 notation. Think of it as a BASE 100 number. Each byte in the floating point representation is a "place" which can be expressed, in decimal terms, by a single byte. For example the number 123 would have a "1" in the 2nd place to the left of the decimal, and a "23" in the 1st place. Places to the right of the decimal are expressed similarly. HOWEVER the 1st byte of the 8 is the "exponent" plus 40. E.g. the number 123 would have an exponent of 1, and thus would be represented as >41,>01,>17,... (the last 5 bytes don't matter). The 99/4A handles most "arithmetic" this way. There are a number of routines in the console that manipulate this type of number, as well as convert to ASCII etc. They are mostly accessed through XMLLNK, as

well as GPLLNK.

The program below is meant to be used out of command mode in XBASIC. It is not reliable from a running program because of its extensive use of VDP RAM, which is used for strings in running programs. There are a number of lines which can be eliminated if the program is to be attached to an assembly program, if that is the way you wish to use it. However you may have to "AORG" it so as not to interfere with the rest of the program. It works on an interrupt, always waiting for you to press CTRL C. When you do so, the screen clears and you are asked for the device name. After this is entered, you will get brief information on the screen as to the total volume, then nothing more until ALL the programs are read into memory. When this is done, the first page will flash onto the screen. You may then page up or down using the X and E keys. Paging is circular in both directions. Return to command mode is effected by pressing enter (which is also an escape from the input line).

Genial Computerware will shortly be publishing this program in an already assembled format, with several different AORG'd versions to fit with any different programs. The disk will also contain a patch for DISKASSEMBLER so that disks may be cataloged from any point within that program (this was done by popular request).

I hope you enjoy using and understanding this program.

```

DEF START,OFF,ON
PABLOC EQU >1950
FAC EQU >834A
ARG EQU >833C
STATUS EQU >837C
KEY EQU >8375
GPLNS EQU >83E0
VBF1 EQU >1000
VBF2 EQU >1000
VBF3 EQU >1400
VBF4 EQU >1800
*****
OFF CLR @>83C4      Clear ISR hook-turn off interpts
RT
ON LI 0,START      Load start of routine in ISR
MOV 0,@>83C4
RT
START CB @KEY,@CTRLC Is CTRL C pressed?
JEQ S2            Yes, begin
RT               No, return
S2 LWPI >83C8      Interrupt workspace
MOV 13,@SAV13     R13-15 need to be saved for rtn
MOV 14,@SAV14     -destroyed by FSLB routine below
MOV 15,@SAV15

```

```

CLR @83C4      Turn off interrupts temporarily
LWPI HS        Our Workspace
* The next 8 lines are needed in Basic only, because all
* text must have basic bias of >6@ added
ABS @BIASCK    Have we modified text already?
JNE S1A        Yes, skip next part
SETO @BIASCK   Mark the change
LI @,DEV       Beginning of text to be changed
LI 2,TYPES-DEV Length (end-beginning)
S1  AB @BIAS,+@ Add the bias one byte at a time
    DEC 2        More?
    JNE S1       Yes, go back
S1A LI @,>9295    Change screen image table from @
    BLMP @VNTR   to >149@ (>49@*5)-saves orig scr
    LI @,>@1F@   Text mode
    BLMP @VNTR

*A copy of value in VDP REB 1 (now in LSB of R@) must be
*placed @>83D4 because the value there is transferred to
*VDP REB 1 at every key press
    MOVB @NS+1,@>83D4
    LI @,>@717    1=FB color,7=86 change if you
                    like
    BLMP @VNTR
S3  LI @,>@954    WRITE address for VDP >149@
    LI 2,>@C92    VDPMA reversed
    MOVB @,>2     Move LSB of >549@ first
    SMPB @
    MOVB @,>2     Now MSB
    DECT 2       VDP@D (>8C9@) - as each byte is
                    * moved here, the address at >8C92
                    * auto-increments - Handy!
                    *
                    * LI 1,>136@ clear buffer(127 files bytes)
                    * LI @,>@99@  space with basic bias,use >25@
                    * if not in basic
S4  MOVB @,>2     Because of auto-increment each
    DEC 1        byte written goes to next, with-
    JNE S4       out changing R2
    LI @,>142A    3rd Row, Col. 1@
    LI 1,DEV     Text
    LI 2,9       Remember new screen image table
    BLMP @VMBW   write on screen
    AI @,9       Prepare for input
    CLR 2        Counter
    LI 3,PROBUF+1 For storage
S5  BLMP @KSCAN  Look for key press
    MOVB @STATUS,1 Key pressed?
    JEQ S5       No, go back
    MOVB @KEY,1  Yes put value in MSB of R1
    CB 1,@ENTER Enter Key?
    JEQ S6       Yes, process
    MOVB 1,+3+   Store value,increase buffer pos.
    AB @BIAS,1  Add Basic Bias to R1
    INC @        Next position
    BLMP @VSBW  Write on screen
    INC 2        Increase the counter
    JMP S5       Go for more
S6  MOV 2,2      Enter pressed without text?
    JNE S7       No, go on
    B @ENDEND@  Yes, branch to end
S7  BLMP @VSBR  Read the last character
    
```

```

CB 1,@PERIOD   Is it a period?
JNE S3         No, go all the way back
SMPB 2         Count in MSB of R2
MOVB 2,@PROBUF Put len byte at start of storage
LI @,VBF3     This is the screen image table
LI 1,TIT1     Text
LI 2,12@     3 lines
BLMP @VMBW    Write
LI @,PABLOC   Open mode
LI 1,DSKPAB
LI 2,1@
BLMP @VMBW    Write first part of PAB
AI @,9        Where len byte,dev.name will go
LI 1,PROBUF   Where they are now
MOVB +1,2     len byte into R2 (MSB)
SRL 2,8       Now in LSB
INC 2         Dev.name PLUS len byte
BLMP @VMBW    Write rest of PAB
MOV @PABPT,@>8356 Pointer to location of len byte
BLMP @DSRLNK  Open the file
DATA @
JEQ S3        Error, go back
BL @RECORD    "Read" 1st record (will contain
                    * disk name,then 3 #'s (@,total sectors,number available)
BL @CLRBUF    Clear buffer space
LI 1,PROBUF   Where string will go
BLMP @VMBR    Read string into it-see RECORD to
                    * see what R@,R2 have become
                    *
                    * A 2,@ Next item is a number(fltng pt)
                    * AI @,19 The 1st @ was @(9 bytes),the 2nd
                    * we will use below(9 bytes)plus 1
                    * more byte to get to actual @
                    *
                    * LI 4,PROBUF+15
                    * Look at routine GETNUM to see what it does
BL @GETNUM    AVAIL in FAC & @PROBUF+15
AI @,-9       Now back to 1st @(TOTAL)
LI 1,ARB      This will put TOTAL in ARB
LI 2,8        AVAIL will still be in FAC
BLMP @VMBR    both as floating point numbers
LWPI >83E@
LI 7,14       XLINK table is @>@CFA,FSUB is
A @>@CFA,7    7th entry (ARB-FAC)result in FAC
MOV +7,7      R7 contains address of routine
BL +7         Now USED in FAC!
LWPI HS
LI 4,PROBUF+3@
BL @GETNUM1   Convert to ASCII,put #R4
BL @ADD@     Add Basic Bias to the text
LI @,VBF3+11 Next three BLMP @VMBW instruc-
LI 1,PROBUF   tions place the DISKNAME, AVAIL
LI 2,1@      and USED in proper locations on
BLMP @VMBW    screen
AI @,15
AI 1,16
LI 2,4
BLMP @VMBW
AI @,9
AI 1,15
BLMP @VMBW
    
```

```

LI 9,VBF4      Initialize buffer to hold files
CLR @TOTAL    File counter
GETPRO BL @CLRBUF  Clear PROBUF
* Each "record" will now produce a string which is the
* Filename, then 3 floating point numbers:1) file type
* negative if protected,2)size in sectors 3)record length
* if not "program"
BL @RECRD
JEQ END1      Null string=no more, jump to end
LI 1,PROBUF+3  Read the name into PROBUF
BLWP @VMBR
A 2,0        Get to 1st number
CLR 1
* 1st number is 1-5(+ or -),so 1st byte is ALWAYS >40
INCT 0       therefore 2nd byte is IT
BLWP @VSR
LI 3,>5920    "y" or " "
SRA 1,8      Number in LSB,but sign bit there
JLT GP1      If negative, leave R3 alone
SWPB 3       Put " " in MSB of R3
GP1 ABS 1      Now get the positive & 1-5
MOV 1,8      Save R1
MOV8 3,@PROBUF+31 Put "y" or " " in proper loc.
SLA 1,3      multiply by 8
AI 1,TYPES-8 Index to file type
MOV 1,3
LI 4,PROBUF+19
LI 5,7
BL @MOV34     Move the TYPE to PROBUF
AI 0,8       Next number
LI 4,PROBUF+13 Where to put
BL @GETNUM    Convert to ASCII
CI 8,5       S="progras", skip record length
JEQ GP2
AI 0,9       Next number
LI 4,PROBUF+25 Where to put
BL @GETNUM    Convert to ASCII
GP2 BL @ADD60    Now add Basic Bias to all
MOV 9,9      Location in VDP buffer
LI 1,PROBUF
LI 2,40
BLWP @VMBW    Write it to the buffer
A 2,9        Next position in buffer
INC @TOTAL    Counter
JMP GETPRO    Back for more
END1 LI 0,PABLOC
LI 1,>0100    Opcode for close
BLWP @VSRW
MOV @PABPT,@08356
BLWP @DSRLNK  Close the file
DATA 8
LI 4,21      Divide one R (4) into a contig-
CLR 5        uous 2 word area(R5-6),integer
MOV @TOTAL,6 result in R5,remainder in R6
DIV 4,5      R5=number of full pages of 21
MOV 5,6      files
MOV 6,@TOTAL Now TOTAL=no. of full pages
* Note,we need one more "page" for end,but begin at 0!
CLR 6        Initialize to first page

```

```

END2 LI 7,840  21 lines of 40 characters
MOV 6,8        The page number
MPY 7,8
* In the instruction MPY RX,RY the result will be con-
* in the 2 word sequence RY,RY+1 as a 32 bit number.What
* was in RY+1 before is wiped out.Thus in this case,R9
* will contain the result of the multiplication.
AI 9,>1800     Now we know which page in buffer
BL @SCRD      Write to screen
END3 BLWP @KSCAN  Look for key press
MOV8 @STATUS,1
JEQ END3      None pressed
MOV8 @KEY,1   Move value of key press to R1
CB 1,@ENTER   Is it "enter"?
JEQ ENDEND    Yes jump to end
CB 1,0X       Is it "X"?
JNE END4      No,jump ahead
DEC 6         Yes,down one page
JLT END5      Don't let it be <0
JMP END2      Go back and write new screen
END5 MOV @TOTAL,6  Last page
JMP END2      And go back
END4 CB 1,0E       Is it "E"?
JNE END3      No,go back(no others allowed)
INC 6         Up one page
C 6,@TOTAL    Don't let it be >last page
JLE END2      OK,go back and write
CLR 6         1st page
JMP END2      go back and write
ENDEND LI 0,>0200  Reset original screen image tabl
BLWP @WTR
LI 0,>01E0     Reset 32 col.mode for Basic
BLWP @WTR
MOV8 @MS+1,@083D4 Remember it needs to be saved
* Next 2 instructions return to original colors
LI 0,>07F4     if needed (your choice)
BLWP @WTR     F4 are colors in GK IBASIC
LI 0,START    Reload the ISR hook
MOV 0,@083C4
CLR 0
MOV8 0,@STATUS  Clear GPL status byte
LWPI >083C0
MOV @SAV13,13  Restore the lost registers!
MOV @SAV14,14
MOV @SAV15,15
RTMP          And back to basic!
*****
* SUBROUTINES
*****
* Clears PROBUF to spaces (without basic BIAS)
CLRBUF LI 3,>2020
LI 4,PROBUF
LI 5,20       20 words=40 bytes
CB MOV 3,+4+
DEC 5
JNE CB
RT
* adds the basic BIAS to all 40 positions of PROBUF
ADD60 LI 1,PROBUF

```

```

A6  LI 2,40
    AB @BIAS,+1+
    DEC 2
    JNE A6
    RT
* Moves R5 bytes from R3 to R4
MOV34 MOVB #3+,#4+
    DEC 5
    JNE MOV34
    RT
* Read a record,assume a string with length byte is
* first,get LEN into R2 and INC R0
RECRD LI 0,PABLOC
    LI 1,>200 READ OP code
    BLWP @VSBW
    MOV @PABPT,@8336
    BLWP @DSRLNK
    DATA 8
    LI 0,VBF1 location of read buffer
    BLWP @VSRB read 1st byte (length)
    INC 0 Beginning of string
    SRL 1,8 To LSB
    MOV 1,2 Put it in R2
    RT
* Sub to convert a floating point # at VOP addr in R0
* to ASCII and place it at addr. in R4,right justified
* GTNUM1 does same if # already in FAC
* The GPLLNK routine with DATA >14 takes a FP # in FAC
* converts it to ASCII starting at location pointed to
* by FAC+11 (must add >8300). If byte at FAC+11 is #
* then number will be in Basic format
* the length of the string is returned in FAC+12
GETNUM LI 1,FAC
    LI 2,8
    BLWP @VMBR Put the Number in FAC
GTNUM1 MOVB 2,@FAC+11 MSB of R2=#
    MOV 11,10 Save return
    MOVB 2,@STATUS Clear STATUS byte
    BLWP @GPLLNK execute the routine
    DATA >14
    MOVB @FAC+11,3 Address of result,LSB
    SHPB 3
    MOVB @H83,3 MSB is >83
    MOVB @FAC+12,5 Length,includes leading space
    SRL 5,8 Put in LSB of R5
    LI 6,5
    S 5,6 This right justifies result
    A 6,4
    INC 3 These 3 instructions adjust for
    INC 4 the leading space
    DEC 5
    BL @MOV34 Move to location in R4
    B Return
* MOVE 21 LINES FROM BIG BUFFER POINTED
* TO BY R9 TO VBF3
SCRD LI 3,VBF3+120 Start on 4th line of screen
    LI 1,PROBUF To transfer one line at a time
    LI 2,40 40 bytes
    LI 5,21 21 lines

```

```

SR1 MOV 9,0 Exact location in buffer
    BLWP @VMBR Read to PROBUF
    MOV 3,0 Screen location
    BLWP @VMBW Write on screen
    A 2,9 Change buffer location
    A 2,3 And screen location
    DEC 5 Go back for more
    JNE SR1 If any!
    RT
*****
* DATA AND BUFFERS
*****
DSKPAB DATA >000D,VBF1,0,0,0 INTERNAL,RELATIVE,FIXED
DEV TEXT 'DEV-NAME:'
* 0 1 2 3 4
TIT1 TEXT ' DISKNAME= AVL= USD=
TIT2 TEXT ' FILENAME SIZE TYPE P
TIT3 TEXT '
TYPES TEXT 'DIS/FIX DIS/VAR INT/FIX INT/VAR PROGRAM'
EVEN
MS BSS 32
PROBUF BSS 40
TOTAL BSS 2
SAV4 BSS 2
SAV11 BSS 2
SAV13 BSS 2
SAV14 BSS 2
SAV15 BSS 2
BIASCK DATA 0
D100 DATA 100
PABPT DATA >1059
CTRLC BYTE 131
READ BYTE 2
CLOSE BYTE 1
H40 BYTE >40
H83 BYTE >83
ENTER BYTE 13
E BYTE 'E'
PERIOD BYTE '.'->60
X BYTE 'X'
BIAS BYTE >60
*****
* GPLLNK AND DSRLNK M6 VERSION,E/A GPLLNK won't work
*****
* Sorry, some of Craig's text is cut off!
GR4 EQU GPLWS+8 GPL workspace R4
GR6 EQU GPLWS+12 GPL workspace R6
STKPNT EQU >8373 GPL Stack pointer
LDGADD EQU >60 Load & Execute GROM address ent
XTAB27 EQU >200E Low Mem XML table location 27
GETSTK EQU >166C
GPLLNK DATA GLNKWS R7 Set 'p BLWP Vectors
    DATA GLINK1 R8
RTNAD DATA XMLRTN R9 Address where GPL XML returns t
GXMLAD DATA >176C R10 GROM Address for GPL XML (@F 27
    DATA >50 R11 Initialized to >50 where PUTSTK
GLNKWS EQU #->18 GPLLNK's workspace of which
    BSS >08 R12-R15 registers R7 through R15 are
GLINK1 MOV #R11,@GR4 Put PUTSTK Address into R4 o

```

```

MOV #R14,@6R6      Put GPL Routine Address in R
MOV @XTAB27,R12     Save the value at >200E
MOV R9,@XTAB27     Put XMLRTN Address into >200
LWPI GPLWS         Load GPL WS
BL #R4             Save current Grom Address on
MOV @GXMLAD,@8302(R4) Push GPL XML Add on stack
INCT @STKPNP       Adjust the stack pointer
B @LDGADD          Execute our GPL Routine
XMLRTN MOV @GETSTK,R4 Get GETSTK pointer
BL #R4             Restore GROM address off the
LWPI GLNKWS        Load our WS
MOV R12,@XTAB27    Restore >200E
RTWP              All Done - Return to Caller

PUTSTK EQU >50     Push Grom Add to stack pointer
TYPE EQU >836D     DSRLNK Type byte for GPL DSSLNK
NAMLEN EQU >8356   Device name length pointer in V
VMA EQU >8C02      VDP Write Arress location
VRD EQU >8800      VDP Read Data byte location
GR4LB EQU >83E9    GPL Workspace R4 Lower byte
GSTAT EQU >837C   GPL Status byte location
DSRLNK DATA DSRWS,DLINK1 Set BLMP Vectors
DSRWS EQU $        Start of DSRLNK workspace
DR3LB EQU $+7      R3 lower bytes of DSRLNK work
DLINK1 MOV R12,R12 R0 Have we already looked up the LI
JNE DLINK3 R1 YES! Skip look up routine
LWPI GPLWS R2,R3 Else load GPL workspace
MOV @PUTSTK,R4 R4,R5 Store current GROM address
BL #R4 R6
LI R4,>11 R7,R8 Load R4 with address of LINK r
MOVB R4,@402(R13) R9,R10 Set up GROM with addr
JMP DLINK2 R11 Jump around R12-R15
DATA 0 R12 contains >2000 flag when set
DATA 0,0,0 R13-R15 contains WS, PC & ST for RT
DLINK2 MOVB @GR4LB,@402(R13) Finish setting up GROM add
MOV @GETSTK,R5 Take some time & set up GETSTK po
MOVB #R13,@DSRAD1 Get the GPL DSR LINK vector
INCT @DSRADD       Adjust it to get past GPL FETC
BL #R5            Restore the GROM Address off t
LWPI DSRWS         Reload DSRLNK workspace
LI R12,>2000       Set flag to signify DSRLNK add
DLINK3 INC R14     Adjust R14 to point to Callers
MOVB #R14,@TYPE   Move it into >836D for GPL DSR
LI 3,>9F00
MOVB 3,@8400      TURN OFF SOUND GENERATOR
MOV @NAMLEN,R3     Save VDP address of Name Lengh
AI R3,-0          Adjust it to point to PAB Flag
BLWP @GPLLNK      Execute DSR LINK
DSRADD BYTE >03   High byte of GPL DSRLNK address
DSRAD1 BYTE >00   Lower byte of GPL DSRLNK addre
MOVB @DR3LB,@VMA Set up LSB of VDP Add for Erro
MOVB R3,@VMA      Set up MSB of VDP Add for Erro
SZCB R12,R15      Clear EQ bit for Error Report
MOVB @VRD,R3      Get PAB Error Flag
SRL R3,5          Adjust it to 0-7 error code
MOVB R3,#R13     Put it into Callers R0 (msb)
JNE SETEQ        If its not zero set EQ bit
ODC @GSTAT,R12   Else test CND bit for Link Err
JNE DSREND       No Error Just return
    
```

```

SETEQ SOCB R12,R15 Error so set Callers EQ bit
DSREND RTWP      All Done - Return to
*****
* VDP UTILITIES *
*****
* modified by Tom Freeman to save a few bytes, and cor-
* for errors in case R0=0 in the multiple byte utilities
*****
VSBW DATA VDPWS,SBW
VMBW DATA VDPWS,MBW
VSBR DATA VDPWS,SBR
VMBR DATA VDPWS,MBR
VMTR DATA VDPWS,WTR
KSCAN DATA VDPWS,KSC
VDPWS EQU $->14   20 BYTES NOT USED
BSS XC          ONLY NEED R10-15
SBW BL @WRITST
MOVB @2(R13),@8C00 HR1
RTWP
MBW BL @WRITST
JEQ VRTN
MOREVM MOVB #R10+,@8C00
DEC R12
JNE MOREVM
VRTN RTWP
SBR BL @READST
MOVB @8800,@2(R13) HR1
RTWP
MBR BL @READST
JEQ VRTN
MOREVR MOVB @8800,#R10+
DEC R12
JNE MOREVR
RTWP
WRITST LI R10,>4000
JMP ADDRESS
READST CLR R10
ADDRESS MOV #R13,R12
MOVB @VDPWS+25,@8C02 LR2
SOC R10,R12
MOVB R12,@8C02
MOV @2(R13),R10 OLD R1
MOV @4(R13),R12 OLD R2
RT
WTR MOV ,12
MOVB @1(13),@8C02
ORI 12,>8000
MOVB 12,@8C02
RTWP
KSC LWPI >83E0
MOV 11,@VDPWS+>16
BL @800E
LWPI VDPWS
MOV 11,@83F6
LIST
RTWP
END
    
```

NOTE: Turn catalog on and off from Command level with CALL LINK("ON") and CALLLINK("OFF")

CALENDAR GENERATOR, from Norm Weiss, SFV 99ers

>>> This CALENDAR GENERATOR program is a typical example of a good program of it's type. However, it is just a "bare bones" program as now written, and can be modified and added to in a variety of ways.  
 For instance, sound could be inserted into the program so that when an error message is displayed a tone is heard. You could change the color of the type displaying the month, and/or the background color.  
 A design within a border could be placed around the edge of the screen, or surrounding the month being displayed.  
 A musical accompaniment, playing while the calendar is being displayed might be a nice touch.  
 More importantly, the addition of programming lines that would give you one or more hard copies of the calendar currently being displayed on the screen.  
 These are just a few suggestions to improve this program. Experiment! Try a few or all of the above ideas, and add some of your own, to shape and personalize this program to suit you. The Road to Adventure stretches out before you!

```

430 PRINT " DO YOU WANT TO GENERATE "
440 PRINT " ANOTHER CALENDAR? (Y/N) "
450 PRINT
460 CALL KEY(0,K,S)
470 IF S=0 THEN 460
480 IF K=89 THEN 100
490 STOP
500 L=LEN(M$)
510 I=INT((28-L)/2)+2
520 PRINT TAB(1);M$;:
530 PRINT " SUN MON TUE WED THU FRI SAT"
540 PRINT " -----"
550 FOR R=1 TO N
560 ON D GOTO 570,590,610,630,650,670,690
570 PRINT TAB(-2+R*4);R;
580 GOTO 710
590 PRINT TAB(2+R*4);R;
600 GOTO 710
610 PRINT TAB(6+R*4);R;
620 GOTO 710
630 PRINT TAB(10+R*4);R;
640 GOTO 710
650 PRINT TAB(14+R*4);R;
660 GOTO 710
670 PRINT TAB(18+R*4);R;
680 GOTO 710
690 PRINT TAB(22+R*4);R;
710 NEXT R
720 PRINT :
730 RETURN
999 END
    
```

```

100 CALL CLEAR
110 PRINT "JANUARY FEBRUARY MARCH APRIL"
120 PRINT " MAY JUNE JULY AUGUST"
130 PRINT " SEPTEMBER OCTOBER NOVEMBER"
140 PRINT TAB(11);"DECEMBER";:
150 PRINT TAB(11);"MONTHLY"
160 PRINT " CALENDAR GENERATOR";:
170 PRINT " TYPE THE NAME OF THE MONTH"
180 INPUT " TO BE DISPLAYED: ";M$
190 CALL CLEAR
200 PRINT "JAN=31 FEB=28 MAR=31 APR=30"
210 PRINT "MAY=31 JUN=30 JUL=31 AUG=31"
220 PRINT "SEPT=30 OCT=31 NOV=30 DEC=31";:
230 PRINT " 1988 IS THE NEXT LEAP YEAR"
240 PRINT TAB(10);"(FEB=29)";:
250 PRINT TAB(11);"MONTHLY"
260 PRINT " CALENDAR GENERATOR";:
270 INPUT " NO. OF DAYS IN MONTH? ";N
280 IF (N>27)*(N<32) THEN 310
290 PRINT "WRONG NO. OF DAYS-TRY AGAIN";:
300 GOTO 270
310 CALL CLEAR
320 PRINT " THE FIRST DAY OF THE MONTH";
330 PRINT "BEGINS ON ONE OF THE FOLLOW ING DAYS: (1=SUN, 2=MON.)";
340 PRINT "3=TUE..... 7=SAT)";:
350 PRINT " TYPE THE NUMBER OF THE DAY"
360 INPUT "OF THE WEEK, THAT THE MONTH STARTS AT: ";D
370 IF (D<=0)+(D>7) THEN 390
380 GOTO 410
390 PRINT "WRONG ENTRY! TRY 1 THRU 7";:
400 GOTO 320
410 CALL CLEAR
420 GOSUB 500
    
```

A QUICKIE QUICKIE!

Well, here's some aerobics for your sound chip!

```

100 CALL INIT
110 FOR C=1 TO 4
120 FOR Z=1 TO 400 STEP 8
130 CALL LOAD(-31744,Z*(1-C))
140 NEXT Z
150 NEXT C
160 CALL SOUND(1,1000,8)
170 GOTO 100
    
```

TIC TAC  
 USERS GROUP  
 Santa. Monica



Cedar Valley 99'er U. G.  
 288 Windsor Dr. NE  
 Cedar Rapids, IA 52402



NEXT MEETING DATE:  
 THURSDAY, MAY 14, 1987

ALL MEETING DATES:  
 6:30 PM TO 8:30 PM  
 DECATUR PUBLIC LIBRARY  
 SECOND FLOOR BOARD ROOM

>>>>MAY MEETING DATE<<<<<

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13>>	14<<	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

\*\*\*\*\*  
 \* DECATUR 99er HOME COMPUTER USERS GRP \*  
 \* APPLICATION FOR MEMBERSHIP \*  
 \* \* \* \* \*  
 \* DATE / /87 \*  
 \* \* \* \* \*  
 \* NAME \_\_\_\_\_ \*  
 \* \* \* \* \*  
 \* ADDRESS \_\_\_\_\_ \*  
 \* \* \* \* \*  
 \* CITY \_\_\_\_\_ ZIP \_\_\_\_\_ \*  
 \* \* \* \* \*  
 \* PHONE \_\_\_\_\_ \*  
 \* \* \* \* \*  
 \* WORK PHONE \_\_\_\_\_ \*  
 \* \* \* \* \*  
 \* DUES: MEM, STUDENT \$15 \*  
 \* ADD'L FAMILY \$ 5 \*  
 \* OR NEWSLETTER ONLY \$12 \$ \_\_\_\_\_ \*  
 \* (25 MAX) \*  
 \*\*\*\*\*

Decatur 99er H. C. U. G.  
 P. O. Box 726  
 Decatur, IL 62525