



NEWS OF THE LIBRARY

Jim Peterson--the author of 'Tips from the Tigercub'--sent us an advance copy of his forthcoming catalog of public-domain software. The collection consists of about 160 SSSD disks, many full or nearly-so, of programs arranged by topic. There are 46 disks of games, 55 disks of music, 33 disks of educational topics, 9 disks of applications, and nearly 20 disks of utilities, demonstrations and tutorials. These are programs written by dozens of authors, some of them known as masters of their special craft.

Jim is offering these at \$1.50 per disk, which is far below the charges made by the Chicago, Boston and Los Angeles user-group libraries. The catalog is available to you for the trouble of copying it--it's on disk. I will have a print copy at the meeting if you're especially anxious to see what's available.

A program came my way a few days ago that really astonished me. It's Peter Kull's 'Compiler v1.1', sold by RYTE Data of Haliburton, Ont. It speeds up BASIC language by preparing a near-machine language version of your ordinary BASIC program and then runs it at a superfast pace. As a demo, I used the graphic program GARFIELD (see next paragraph) which draws a picture of that insufferable cat. When 'looped' for repetitive execution, 5 loops required about 187 seconds. The compiled version runs 5 loops in 18 seconds! If you want to see it work, ask...

The library has received the gift from King Forkner of a floppy disk of graphics and music programs. Here are the catalog listings:

Disk: GRAPHICS  
 Free: 69 \* Used: 289  
 FILENAME SIZE TYPE  
 ~~~~~  
 AMER/FLAG 19 Program U  
 BOWLING 15 Program U  
 GARFIELD 21 Program U  
 HANOI 11 Program U  
 HAPPYBIRTH 11 Program U  
 LOAD 20 Program U  
 MICKEYMSE 15 Program U  
 MONOPOLY 64 IntVar254 U  
 ROCKET 14 Program U  
 S/A/M 25 Program U  
 TIC\_T\_TOE 36 Program U  
 YOUR/MARK 38 Program U

Disk: MUSIC  
 Free: 18 \* Used: 340  
 FILENAME SIZE TYPE  
 ~~~~~  
 B/H/COP 58 IntVar254 U  
 LOAD 20 Program U  
 LORDPRAYER 33 Program U  
 MOZART 49 IntVar254 U  
 POLKA 35 Program U  
 PUPPYTOWN 34 Program U  
 RAINBOW 17 Program U  
 RISING/SUN 49 IntVar254 U  
 SPRITEDANC 45 Program U

Thanks again to Aubrey Johnson for the program 'TIA-SLIDES' which he demonstrated at the January meeting; it runs in XBASIC; the disk also includes a dozen images; there are other RLE pictures in the library that could be included in the slide-show. To get 'TIA-SLIDES' bring an initialized blank disk to the meeting!

\*\*\*\*\*

FOR SALE \* FOR SALE \* FOR SALE \* FOR SALE \* FOR SALE \* FOR SALE \* FOR SALE

- 1 - 32K MEMORY STAND-ALONE \$40.00
- 1 - DISK CONTROLLER STAND-ALONE \$45.00
- 1 - TI 99 4/A COMPUTER \$20.00
- 1 - EXTENDED BASIC \$25.00
- 1 - JOY STICK \$5.00
- 1 - AXIOM GP-700 COLOR PRINTER(EXTRA RIBBONS) \$125.00
- 1 - COMMODORE COLOR MONIOR(USED ONE WEEK) W/CABLES FOR TI \$160.00

CALL 217-423-4377  
 JERRY BRUNSON  
 2058 W. LEAFLAND  
 DECATUR, IL 62522

\*\*\*\*\*



FROM THE DESK OF VICE-PRESIDENT

INSTRUCTIONS FOR TI-99/4A COMPUTER  
 written by Jim Seitz - C.O.N.N.I.

1. Turn on television monitor.
2. Insert program module if necessary (e.g. Early Learning Fun).
3. Turn on computer console.
4. The title screen is the first screen to appear. Return to the title screen before turning the computer off. Modules can be inserted either when the console is off, or at the title screen.
5. Press any key as instructed by the title screen. The next screen is a Menu Screen; it indicates which program or computer language is currently available. The first choice is TI BASIC, the resident language; other choices depend on the module in use.
6. Make a selection from the menu screen. If you are using a module-based program, the program will begin and instruct you on how to use it. You may need to read the module's instruction manual for further details. If you are using TI BASIC you will now need to load your program from the cassette re-corder (unless you are writing your own program). Refer to the User's Reference Guide for more detailed information, pages 1-9 to 112, 11-42.
7. Press "Alpha Lock" down, type in "OLD CSI", press ENTER. Follow this sequence of events as instructed by the computer: (a) REMIND CASSETTE TAPE THEN PRESS ENTER, (b) PRESS CASSETTE PLAY THEN PRESS ENTER, (c) (READING), (d) (DATA OKAY), and (e) PRESS CASSETTE STOP THEN PRESS ENTER.
8. Wait for the cursor to return to the screen, then type "RUN", then ENTER.
9. You may now run and enjoy the program.
10. When you are ready to stop the program, follow its instructions to exit. If the program does not have an exit mode, try these two options.  
 Option 1. Press down on the FUNCTION key and the "4" key and release simultaneously (this causes the program to "break"). Now type in "BYE", ENTER.  
 Option 2. Press the FUNCTION key and the "+" key and release simultaneously.  
 Either of these options will return you to the Title Screen. You MUST be at the Title Screen before turning the computer off or inserting a new module. For some programs the Exit mode will display "DONE" when finished; type in "BYE", ENTER to return to the title screen.
11. If you are using a cassette-based program and want to run the next program, do the following: (a) Exit the program but you do not have to go back to the title screen. (b) At the "DONE" display or after pressing FUNCTION 4, type in "NEW", ENTER. (c) Type in "OLD CSI", ENTER to load the next program. Programs on a cassette must be run in order of their appearance on the cassette.
12. To turn the computer off, return to the Title Screen (step 10). Turn off the computer console, then the TV monitor.

# GETTING THE MOST FROM YOUR CASSETTE SYSTEM: by MICKEY SCHMITT

## NUMBER 2: LOADING AND SAVING PROGRAMS

While LOADING and SAV(E)ing programs with the use of a cassette recorder is not a difficult process in itself - reading and understanding the instructions for the very first time can be quite confusing. With that thought in mind I have tried to keep the instructions as simple as possible.

### Instructions for LOADING programs:

1. Type: OLD CS1
2. Then: Press ENTER
3. Follow the directions as they appear on your monitor or TV screen:
  - 3.1 \* REWIND CASSETTE TAPE CS1  
THEN PRESS ENTER
  - 3.2 \* PRESS CASSETTE PLAY CS1  
THEN PRESS ENTER
  - 3.3 Computer displays message:  
\* READING
  - 3.4 Computer displays message:  
\* DATA OK
  - 3.5 \* PRESS CASSETTE STOP CS1  
THEN PRESS ENTER
4. Wait for the flashing cursor to appear in the lower Left-hand corner of your monitor or TV screen.
5. Type: RUN
6. Then: Press ENTER

### Instructions for SAV(E)ing programs:

1. Type: SAVE CS1
2. Then: Press ENTER
3. Follow the directions as they appear on your monitor or TV screen:
  - 3.1 \* REWIND CASSETTE TAPE CS1  
THEN PRESS ENTER
  - 3.2 \* PRESS CASSETTE RECORD CS1  
THEN PRESS ENTER
  - 3.3 Computer displays message:  
\* RECORDING
  - 3.4 \* PRESS CASSETTE STOP CS1  
THEN PRESS ENTER
4. Your program is now SAVEd - But you should get into the habit of checking all your programs to be sure that they were SAVEd without error.
5. Continue to follow the directions as they appear on your monitor or TV screen:
  - 5.1 Computer displays message:  
\* CHECK TAPE (Y OR N)?
  - 5.2 Type: Y
  - 5.3 Then: Press ENTER
  - 5.4 \* REWIND CASSETTE TAPE CS1  
THEN PRESS ENTER
  - 5.5 \* PRESS CASSETTE PLAY CS1  
THEN PRESS ENTER
  - 5.6 Computer displays message:  
\* CHECKING
  - 5.7 Computer displays message:  
\* DATA OK
  - 5.8 \* PRESS CASSETTE STOP CS1  
THEN PRESS ENTER
6. Your program is now SAVEd - Safely and without error! That's all there is to it!

Next month's topic will be how to keep your cassettes and programs organized.

Copyright 1988

TIGERCUB SOFTWARE  
156 Collingwood Ave.  
Columbus, OH 43213

Distributed by Tigercub Software to TI-99/4A Users Groups for promotional purposes and in exchange for their newsletters. May be reprinted by non-profit users groups, with credit to Tigercub Software.

Over 120 original programs in Basic and Extended Basic, available on cassette or disk, NOW REDUCED TO JUST \$1.00 EACH!, plus \$1.50 per order for cassette or disk and PP&M. Minimum order of \$10.00. Cassette programs will not be available after my present stock of blanks is exhausted. The Handy Dandy series, and Color Programming Tutor, are no longer available on cassette. Descriptive catalogs, while they last, \$1.00 which is deductible from your first order.

Tigercub Full Disk Collections, reduced to \$5 postpaid. Each of these contains either 5 or 6 of my regular catalog programs, and the remaining disk space has been filled with some of the best public domain programs of the same category. I am NOT selling public domain programs - they are a free bonus!

TIGERCUB'S BEST, PROGRAMMING TUTOR, PROGRAMMER'S UTILITIES, BRAIN GAMES, BRAIN TEASERS, BRAIN BUSTERS!, MANEUVERING GAMES, ACTION GAMES, REFLEX AND CONCENTRATION, TWO-PLAYER GAMES, KID GAMES, MORE GAMES, WORD GAMES, ELEMENTARY MATH, MIDDLE/HIGH SCHOOL MATH, VOCAB-

ULARY AND READING, MUSICAL EDUCATION, KALEIDOSCOPIES AND DISPLAYS

## NUTS &amp; BOLTS DISKS

These are full disks of 100 or more utility subprograms in MERGE format, which you can merge into your own programs and use, almost like having another hundred CALLS available in Extended Basic. Each is accompanied by printed documentation giving an example of the use of each. NUTS & BOLTS (No. 1) has 100 subprograms, a tutorial on using them, and 5 pp. documentation. NUTS & BOLTS No. 2 has 108 subprograms, 10 pp. of documentation. NUTS & BOLTS #3 has 140 subprograms and 11 pp. of documentation. NOW JUST \$15 EACH, POSTPAID.

## TIPS FROM THE TIGERCUB

These are full disks which contain the programs and routines from the Tips from the Tigercub newsletters, in ready-to-run program format, plus text files of tips and instructions. TIPS (Vol. 1) contains 50 original programs and files from Tips newsletters No. 1 through No. 14. TIPS VOL. 2 contains over 60 programs and files from Nos. 15 thru 24. TIPS VOL. 3 has another 62 from Nos. 25 through 32. TIPS VOL. 4 has 48 more from issues No. 33 through 41. NOW JUST \$10 EACH, POSTPAID.

```
*****
*           NOW READY           *
* TIPS FROM TIGERCUB VOL.5 *
* Another 49 programs and *
* files from issues No. 42 *
* through 50. Also $10 ppd *
*****
```

TIGERCUB CARE DISKS #1, #2, #3 and #4. Full disks of text files (printer required). No. 1 contains the Tips newsletters #42 thru #45, etc. Nos. 2 and 3 have articles mostly on Extended Basic

programming. No. 4 contains Tips newsletters Nos. 46-52. These were prepared for user group newsletter editors but are available to anyone else for \$5 each postpaid.

I believe this word game is totally different from anything you have ever seen, and very challenging if you don't use the AID key. The first time you run it, pick option 3 to create a file of phrases and give it the file name COMPUTE. This will then become the computer's file, option 1, and you can create as many of your own files as you want. Recommend phrases of several to as many as 20 words - short ones are too difficult.

```
100 DIM W$(20):: DIM D$(20)
110 GOTO 150
120 Q$,K,S,Q,F$,E,FLAG,X,J,X
$,Y$,A,B,M$,DY$,V,A$(),C$,CH
,C$,Y,W$(),L,M,D$(),F,Z,C,R
,H
130 CALL CHAR :: CALL KEY ::
CALL SOUND :: CALL CLEAR ::
CALL CHARPAT :: CALL COLOR
:: CALL SCREEN :: CALL VCHAR
:: CALL SPRITE :: CALL LOCA
TE :: CALL DELSPRITE
140 !?P-
150 CALL CHAR(94,"3C4299A1A1
99423C"):: DISPLAY AT(2,1)ER
ASE ALL:"TIGERCUB SHUTTLESEA
RCH V.1.1": ""^^ Tigercub So
ftware for free": "distributi
on but no price"
160 DISPLAY AT(6,1): "or copy
ing fee to be charged": "" "I
f you should feel moved to":
"send me a few bucks for my"
:"work, I won't be offended!"
"
170 DISPLAY AT(12,1): "Jim Pe
terson": "156 Collingwood Ave
.": "Columbus, OH 43213"
180 DISPLAY AT(16,5): "Instru
ctions? (Y/N) N" :: ACCEPT A
T(16,25)SIZE(-1)VALIDATE("YM
"):Q$ :: IF Q$="N" THEN 260
190 DISPLAY AT(2,1)ERASE ALL
:" The computer will display
a": "phrase or saying concea
led": "within a grid of rando
```

```
 letters."
200 DISPLAY AT(6,1): "The wo
rds will be horizon-": "tal,
one word per line and": "on c
onsecutive lines, but": "not
necessarily beginning on"
210 DISPLAY AT(10,1): "the to
p line, and the phrase": "may
'wrap around' from the": "bo
ttom row to the top."
220 DISPLAY AT(13,1): "You c
an find the phrase by": "shut
tling columns of letters": "u
p and down, looking for": "co
nsecutive rows with letter"
230 DISPLAY AT(17,1): "combin
ations that could be": "parts
of words." : "A cheat key is
available," : "if you are rea
lly stuck, but"
240 DISPLAY AT(21,1): "try no
t to use it!"
250 DISPLAY AT(23,8): "PRESS
ANY KEY" :: DISPLAY AT(23,8)
:"press any key" :: CALL KEY
(0,K,S):: IF S=0 THEN 250
260 DISPLAY AT(3,2)ERASE ALL
:"Do you want to - 1": "" : (
1) Solve a saving from my
file?": "" : (2) Solve a p
hrase from your file
?"
270 DISPLAY AT(11,2): "(3) Cr
eate a file of": " phrase
s?": "" : (4) Have someone ty
pe in a phrase to solve
?"
280 ACCEPT AT(3,19)SIZE(-1)V
ALIDATE(DIGIT):Q :: IF Q<1 O
R Q>4 THEN 280
290 ON Q GOTO 300,310,410,47
0
300 F$="1.COMPUTE" :: E=1 ::
GOTO 320
310 DISPLAY AT(18,1): "Filena
me? DSK" :: ACCEPT AT(18,14)
:F$ :: E=2
320 ON ERROR 370
330 IF FLAG=1 THEN 350 :: FL
AG=1 :: OPEN #1:"DSK"&F$,FIX
ED,RELATIVE,INPUT :: ON ERRO
R STOP
340 INPUT #1,REC 0:X :: CLOS
E #1 :: FOR J=1 TO X :: X$=X
&CHR$(J):: NEXT J :: Y$=X$
350 RANDOMIZE :: A=INT(RND*LE
N(Y$)+1):: B=ASC(SEG$(Y$,A,
1)):: Y$=SEG$(Y$,1,A-1)&SEG$
(Y$,A+1,255):: IF LEN(Y$)=0
THEN Y$=X$
```

```

360 OPEN #1:"DSK"&F$,FIXED,R
ELATIVE,INPUT :: ON ERROR ST
OP :: INPUT #1,REC B:M$ :: C
LOSE #1 :: GOTO 490
370 FOR J=1 TO 10 :: DISPLAY
AT(20,1):"" :: DISPLAY AT(2
0,1):"CANNOT OPEN FILE!" ::
CALL SOUND(-99,110,5,-4,5)::
NEXT J
380 ON ERROR 390 :: CLOSE #1
390 FLAG=0 :: INPUT "CHECK D
ISK AND DRIVE, PRESS ANY KEY
":DY$
400 IF E=1 THEN RETURN 260 E
LSE IF E=2 THEN RETURN 310 E
LSE RETURN 410
410 DISPLAY AT(8,1)ERASE ALL
:"Filename? DSK" :: ACCEPT A
T(8,14):F$
420 E=3 :: ON ERROR 370 :: O
PEN #1:"DSK"&F$,FIXED 124,RE
LATIVE,OUTPUT :: ON ERROR ST
OP :: X=0
430 DISPLAY AT(12,1):"Enter
END when finished": ""::"Ty
pe phrases, not more than 20
words and 124 characters"
440 X=X+1 :: ACCEPT M$ :: IF
LEN(M$)>124 THEN PRINT "TOO
LONG!" :: X=X-1 :: GOTO 440
450 IF M$<>"END" THEN PRINT
#1,REC X:M$ :: GOTO 440
460 PRINT #1,REC 0:X :: CLOS
E #1 :: GOTO 260
470 CALL KEY(3,K,S):: DISPLA
Y AT(12,1)ERASE ALL:"Type a
phrase of less than 20 word
s and press Enter"
480 ACCEPT M$ :: CALL CLEAR
490 DISPLAY AT(3,2)ERASE ALL
:"Choose skill level - 1": ""
:" (1) All words begin in": "
first column"
500 DISPLAY AT(8,2):"(2) All
words begin in same": "
column": "" (3) Each word m
ay appear in": " a differ
ent column"
510 DISPLAY AT(14,2):"(4) As
No. 3 but AID key is": "
disabled": "" (5) Quit"
520 ACCEPT AT(3,23)SIZE(-1)V
ALIDATE(DIGIT):V :: IF V<1 O
R V>5 THEN 520 :: IF V=5 THE
N CALL CLEAR :: STOP
530 DISPLAY AT(12,6)ERASE AL
L:"SCRAMBLING....."
540 A$(1)="jkzae klmpr vgaho
nceci sdfuy bqijw astrf urd
sa nvjxe blbig trakv nobth w

```

```

ehey vnijo oherq umbai rtika
opleg nosve tarkh zeski "
550 A$(2)="!boiu m.fqt krac,
pjjp? tn-un osheg kar,q ibl
.o tons! idrix ?uhig ebarf u
ks,k ,,jhge vifyt kibrn taga
, .!ry lakle ilf.! inst"
560 C=A$(1)&A$(2)
570 FOR CH=65 TO 90 :: CALL
CHARPAT(CH,CH$):: CALL CHAR(
CH+32,CH$):: NEXT CH :: CALL
CHAR(42,"82444428281010")
580 CALL CHAR(143,"18243C4A4
A3C2418"):: CALL COLOR(14,16
,1)
590 M$=M$&" " :: Y=1
600 X=POS(M$," ",1):: M$(Y)=
SEG$(M$,1,X):: L=LEN(M$(Y)):
: M=MAX(M,L):: RANDOMIZE ::
M$(Y)=M$(Y)&SEG$(C$,INT(230$
RND+1),20-L)
610 Y=Y+1 :: IF Y=21 THEN 62
0 :: M$=SEG$(M$,X+1,255):: I
F LEN(M$)>0 THEN 600
620 FOR J=Y TO 20 :: M$(J)=S
EG$(C$,INT(230$RND+1),20)::
NEXT J
630 ON V GOTO 670,640,650,65
0
640 X=INT(RND*(20-M))+M+1 ::
FOR J=1 TO Y :: M$(J)=SEG$(
M$(J),X,255)&SEG$(M$(J),1,X-
1):: NEXT J :: GOTO 670
650 FOR J=1 TO Y :: X=INT(RN
D*(20-M))+M+1 :: M$(J)=SEG$(
M$(J),X,255)&SEG$(M$(J),1,X-
1):: NEXT J :: GOTO 670
660 ! the string
670 FOR J=1 TO 20 :: FOR L=1
TO 20 :: D$(J)=D$(J)&SEG$(M
$(L),J,1):: NEXT L :: NEXT J
680 IF V=1 THEN F=M ELSE F=2
0
690 FOR J=1 TO F :: Z=INT(20
$RND+1):: D$(J)=SEG$(D$(J),Z
,255)&SEG$(D$(J),1,Z-1):: NE
XT J
700 CALL CLEAR :: CALL SCREE
N(5):: FOR S=1 TO 13 :: CALL
COLOR(S,5,16):: NEXT S :: C
ALL VCHAR(1,31,1,96)
710 CALL VCHAR(4,5,143,20)::
CALL VCHAR(4,28,143,20)
720 FOR C=1 TO 20 :: FOR R=1
TO 20 :: CALL VCHAR(R+3,C+6
,ASC(SEG$(D$(C),R,1))): NEX
T R :: NEXT C
730 DISPLAY AT(1,1):"s&d to
select, e&x to scrollfctn 7
aid, fctn 8 restart"

```

```

740 H=1 :: C=48 :: CALL SPRI
TE(#1,42,7,18,C)
750 CALL KEY(3,K,S):: IF S=0
THEN 750 ELSE ON POS("EXSD"
&CHR$(1)&CHR$(6),CHR$(K),1)+
1 GOTO 750,800,810,820,830,7
60,840
760 IF V=4 THEN 750
770 FOR S=5 TO 8 :: CALL COL
OR(S,16,5):: NEXT S
780 CALL KEY(3,K,S):: IF S=-
1 THEN 780
790 FOR S=5 TO 8 :: CALL COL
OR(S,5,16):: NEXT S :: GOTO
750
800 D$(H)=SEG$(D$(H),2,19)&S
EG$(D$(H),1,1):: FOR R=1 TO
20 :: CALL VCHAR(R+3,H+6,ASC
(SEG$(D$(H),R,1))): NEXT R
:: GOTO 750
810 D$(H)=SEG$(D$(H),20,1)&S
EG$(D$(H),1,19):: FOR R=1 TO
20 :: CALL VCHAR(R+3,H+6,AS
C(SEG$(D$(H),R,1))): NEXT R
:: GOTO 750
820 C=C-8-(C=48)*8 :: H=C/8-
5 :: CALL LOCATE(#1,18,C)::
GOTO 750
830 C=C+8+(C=200)*8 :: H=C/8
-5 :: CALL LOCATE(#1,18,C)::
GOTO 750
840 CALL CLEAR :: FOR J=1 TO
20 :: D$(J)="" :: NEXT J ::
M=0 :: CALL DELSPRITE(#1)::
IF Q=1 OR Q=2 THEN 350 ELSE
470

```

Here are three screen display subprograms of the type you will find on my Nuts and Bolts disks. Note that subprograms can read DATA from the main program. The double colons in the DATA statement cause input of null strings of data for spacing between the lines. The M\$( ) in the subprogram parameter lists is necessary, even though the array is not passed from the main program, in order to DIMension the array in the subprogram - unless you prefer to place the DIM in the subprogram itself. T is the number of DATA items to be read.

```

100 CALL CLEAR
110 DATA THIS IS A DEMO,,OF

```

```

THREE SCREEN PRINTING,,SUBPR
OGRAMS PUBLISHED IN,,TIPS FR
OM THE TIGERCUB,,No. 51,,BY
TIGERCUB SOFTWARE
120 DIM M$(11):: CALL DOWNPR
INT(M$( ),11):: FOR D=1 TO 10
00 :: NEXT D :: CALL CLEAR ::
RESTORE 110 :: CALL DIAGPR
INT(M$( ),11)
130 FOR D=1 TO 1000 :: NEXT
D :: CALL CLEAR :: RESTORE 1
10 :: CALL INWARD(M$( ),11)
1000 SUB DOWNPRINT(M$( ),T)
1001 FOR J=1 TO T :: READ M$
(J):: L=INT(LEN(M$(J))+.5)::
M$(J)=RPT$(" ",14-INT(L/2))
&M$(J):: M$(J)=M$(J)&RPT$("
",28-LEN(M$(J))): NEXT J
1002 FOR J=1 TO 28 :: FOR L=
1 TO T
1003 DISPLAY AT(L,1):SEG$(M$
(L),1,J):: NEXT L
1004 NEXT J :: SUBEND
2000 SUB INWARD(M$( ),T):: FO
R J=1 TO T :: READ M$(J):: M
EXT J :: R=1 :: FOR A=1 TO T
2001 L=INT(LEN(M$(A))): F=1
3-L/2 :: G=L+L
2002 FOR J=1 TO INT(L/2+.5):
: DISPLAY AT(R,F+1):SEG$(M$(
A),J,1):: DISPLAY AT(R,6):S
EG$(M$(A),L-J+1,1):: F=F+1
:: G=G-1 :: NEXT J :: R=R+1
:: NEXT A :: SUBEND
3000 SUB DIAGPRINT(M$( ),T)::
FOR J=1 TO T :: READ M$(J):
: L=INT(LEN(M$(J))+.5):: M$(
J)=RPT$(" ",14-(L/2))&M$(J):
: M$(J)=M$(J)&RPT$(" ",28-LE
N(M$(J))): NEXT J
3001 FOR J=1 TO 28+L :: FOR
L=1 TO T
3002 IF J<L THEN 3004
3003 DISPLAY AT(L,1):SEG$(M$
(L),1,J-L):: NEXT L
3004 NEXT J :: SUBEND

```

Just in case you didn't know - to jump directly to the first or last line in a TI-Writer file, use FCTN 9 and S(earch) and 1 for the first line or E for the last.

MEMORY ALMOST FULL...

Jim Peterson



**TI-Base - TUTORIAL**  
**Extended Basic Background 2.1**  
**NorthCoast 99'ers**  
**Copyright 1988 By Martin A. Smoley**

I am reserving the copyright on this material, but I will allow the copying of this material by anyone under the following conditions. (1) It must be copied in its entirety with no changes. (2) If it is retyped, credit must be given to myself and the NorthCoast 99ers, as above. (3) The last major condition is that there may not be any profit directly involved in the copying or transfer of this material. In other words, Clubs can use it in their newsletters and you can give a copy to your friend as long as its free.

```

100 ! ##### 5ML=>D/V80
101 ! Copyright 1988 By Martin A. Smoley
102 !
300 LNFL1$="^"      ^ :: FL1=15
310 FNFL2$="^"      ^ :: FL2=15
320 SAFL3$="^"      ^ :: FL3=25
330 CTFL4$="^"      ^ :: FL4=20
340 ZPFL5$="^"     ^ :: FL5=5
350 PHFL6$="^"     ^ :: FL6=12
360 XPFL7$="^"     ^ :: FL7=5
500 OPEN #1:"DSK6.NOCOTEST",INTERNAL,FIXED 150,INPUT
600 OPEN #2:"DSK6.NOCO-TIBX",DISPLAY ,VARIABLE 80,OUTPUT
990 C=0
1000 IF EOF(1)THEN CLOSE #1 :: CLOSE #2 :: STOP
1100 IF C<1 THEN INPUT #1:N
1110 C=C+1
1200 INPUT #1:LN$,FN$,CH$,SA$,CT$,ZP$,PH$,XP$
1280 CALL CLEAR
1290 IF C<2 THEN PRINT "  N: : : "
1400 IF C>0 AND C<10 THEN NM$="+00"&STR$(C)
1410 IF C>9 AND C<100 THEN NM$="+0"&STR$(C)
1420 IF C>99 THEN NM$="+"&STR$(C)
1500 SF=LEN(LN$):: IF SF>=FL1 THEN SF=FL1
1510 LNT$=SEG$(LN$,1,FL1)&SEG$(LNFL1$,SF+1,FL1-SF)
1600 SF=LEN(FN$):: IF SF>=FL2 THEN SF=FL2
1610 FNT$=SEG$(FN$,1,FL2)&SEG$(FNFL2$,SF+1,FL2-SF)
1710 MIT$="^^" ! ##### Create new space for MI
1800 SF=LEN(SA$):: IF SF>=FL3 THEN SF=FL3
1810 SAT$=SEG$(SA$,1,FL3)&SEG$(SAFL3$,SF+1,FL3-SF)
1900 SF=LEN(CT$):: IF SF>=FL4 THEN SF=FL4
1910 CTT$=SEG$(CT$,1,FL4)&SEG$(CTFL4$,SF+1,FL4-SF)
2010 STT$="OH" ! ##### Create space and fill with OH
2100 SF=LEN(ZP$):: IF SF>=FL5 THEN SF=FL5
2110 ZPT$=SEG$(ZP$,1,FL5)&SEG$(ZPFL5$,SF+1,FL5-SF)
2200 SF=LEN(PH$):: IF SF>=FL6 THEN SF=FL6
2210 PHT$=SEG$(PH$,1,FL6)&SEG$(PHFL6$,SF+1,FL6-SF)
2300 SF=LEN(XP$):: IF SF>=FL7 THEN SF=FL7
2310 XPT$=SEG$(XP$,1,FL7)&SEG$(XPFL7$,SF+1,FL7-SF)
2410 GPT$="NOCO^" ! ### Create space and fill with NOCO^
2510 IDT$="^" ^ ! ##### Create space for ID number
2990 PRINT NM$:LNT$:FNT$:MIT$:SAT$:CTT$:STT$:ZPT$:PHT$:XPT$:GPT$:IDT$
4000 PRINT #2:NM$:LNT$:FNT$:MIT$:SAT$:CTT$:STT$:ZPT$:PHT$:XPT$:GPT$:IDT$
4500 GOTO 1000
5000 CLOSE #1 :: CLOSE #2
5050 ! SAVE DSK6.5ML=>D/V80
5550 END

```

I'm taking up where I left off with Background 1.2, last month. If you haven't read 1.2, most of this won't make any sense to you. There are many programming ideas which I will not re-explain at this time.

Last month I tried to demonstrate how a file was compressed when it was saved by XBasic. We have a file named NOCOTEST, which we really haven't done anything with. We just looked at it so far. The XBasic program 5ML=>D/V80 will make a bunch of changes to the data in NOCOTEST and create a new file named NOCO-TIBX. NOTE: For the rest of this article I will refer to the program 5ML=>D/V80 as 5M. I have placed a copy of the file 5M produced in the lower right portion of this page. The idea here is to set definite lengths for every field that exists in the NOCOTEST file. In 5M lines 300 through 360, I have set up blank string variables and their lengths to be used as

fill in our new file. For example in line 300, LNFL1\$ is filled with 15 characters. It starts with a circumflex, then there are 13 spaces and last another circumflex. This is the length I want LM to end up with in the new database. The circumflexes will not allow XBasic to collapse the file. FL1=15, is the length of this field. I set this up at the beginning of 5M so I could make changes by adding spaces and changing FL1. This will then change those variables where ever they are used in the rest of 5M.

See Next Page.

**NOCO-TIBX**

+001		+004
Smoley	^	Aardvark
Martin A.	^	Willard
^^		^^
6149 Bryson Drive	^	No Newsletter
Mentor	^	^
OH		OH
44060		^ ^
216-257-1661		1-465-7689 ^
02-89		09-88
NOCO^		NOCO^
^ ^		^ ^
+002		+005
Whitman	^	Vivannovitch
Raymond (Slim)^	^	Elexxie
^^		^^
2574 East 254th.	^	111 E. 98th. St.
Eastlake OH.	^	Cleveland
OH		OH
44094		91023
951-2345	^	541-5415 ^
09-88		05-88
NOCO^		NOCO^
^ ^		^ ^
+003		+006
Aardvark	^	JONES
Grant E.	^	QUINCY W.
^^		^^
9995 State Rt. 84	^	37285 BURGANDY LAINE
Geneva	^	Mentor-on-the-Lake O
OH		OH
44014		44060
1-465-9876 ^		257-1029 ^
02-88		08-88
NOCO^		NOCO^
^ ^		^ ^

**TI-Base - TUTORIAL**  
**Extended Basic Background 2.2**  
**NortCoast 99'ers**  
**Copyright 1988 By Martin A. Sooley**

"Remember, this is an XBasic article. We'll get back to TI-Base later." Lines 500 and 600 open our disk files. Line 990 creates C and puts 0 into it. Line 1000 will check to see if we have reached the end of NOCOTEST. If you recall, the old program saved the number of records as the first item in the data file. Line 1100 pulls that number out, and thus gets it out of our way, so we can read the important data. Line 1100 only executes once, because this is the only time C will be less than 1, as you can see by line 1110. Line 1200 reads or inputs 1 complete record, Last Name, First Name, etc. Next the screen is CLEARed and N, or the number of records, is printed on the screen. Line 1290 is also only executed once. Line 1400 combines "+00" and C together as long as C is from 1 to 9. Line 1410 combines "+0" and C together as long as C is from 10 to 99, and 1420 combines "+" and C from 100 to 999. This is strictly a demonstration of how to add a plus sign and leading zeros. As you will see later, it is of no real value to what we are doing. Lines 1500 and 1510 are important. SF=LEN(LN\$) tells XBasic to find the length of LN\$, the Last Name, and place that value in SF. The next part of that line says, IF SF is greater than or equal to FL1 THEN make sure they are equal to each other. NOTE: IF SF is not greater than or equal to FL1 then it will be left whatever it is (Not Changed). Remember FL1=15, from line 300. Now the biggie, line 1510: SEG\$(LN\$,1,FL1) is quite confusing for no reason. It says, take LN\$ and extract the character from number 1 through FL1, which is 15. In most cases the length is longer than we need, but trailing spaces will be thrown out by XBasic anyway. I did this because some of my variables were longer than the allotted space. SEE +006 >Mentor-on-the-Lake 0<, for an example. SEG\$(FNL1\$,SF+1,FL1-SF) is tough but it works. It says, take the blank line with circumflexes on each end, that we created in line 300, and extract characters starting with the length of LN\$+1 (SF), and continue to the end. The length to the end from that point would be FL1-SF. We're still on line 1510. Now, the & in the middle of the line says put those two oddball pieces together into one string, and last, put it all into LNT\$. "Like I said, it does work." FYI: The experienced programmers out there will look at some of these routines and say, "Wow, this guy is really sloppy". That's true, but if I kept refining these things until they were great, it would take all year for one article and the people who needed help wouldn't get it. Lines 1710, 2010, 2410 and 2510 all create space that didn't exist previously. They also put something in that space to make sure the size of the space remains constant. Line 2990 prints what we have created to the screen and line 4000 prints it to the new data file (NOCO-TIBX). Line 4500 loops back to line 1000 until there is no data left in NOCOTEST, and the program actually ends with the STOP in line 1000. Line 5050 is a trick I use. If you edit a lot, saving a program with a long and intricate name can be troublesome. And 5ML=>D/V80 is one of those. I place this line near the end of my program with a line number that is easy to remember. 5050 is pretty good. When I want to resave the program because of editing changes, I enter this. 5050 <FCTN X>, <ENTER>, <FCTN B>, Press <FCTN 2> (Delete), until the line number and the ! have been deleted and all that is left is SAVE

DSKx.5ML=>D/V/80, and press <ENTER> to save the program. As long as I'm doing tips and tricks, I'll keep going. We have created NOCO-TIBX which can be loaded into FunnelWeb's editor. I loaded it to do a lot of editing. All of my old files were entered in upper case only, like +006 JONES, QUINCY W., etc. I wanted to change that first. In FunnelWeb you can place the cursor on any character, and pressing <CTRL and period> will change that character to lower case. CTRL and semicolon will change it to upper case. This process will auto-repeat to do a complete word or sentence. This trick really helped me a lot. Next, I retyped some middle initials in the space below the first name. Then I spaced over the middle initials located after the first name. This brings up a point. When editing this type of file always press CTRL zero to get out of wordwrap mode. If you accidentally reformat this thing you'll, be amazed at the garbage that is produced. If you want to remove something, space over it, do not delete it. If you must delete something like the I in LAINE, you must then move to the end of that item and add an equal number of spaces to return the circumflex to the proper length position. The circumflexes will hold our field length, much the way the tabs did at the bottom of page 1.1 last month. The 0 at the end of Mentor-on-the-Lake 0 is where the circumflex should be, because that field was longer than the allocated space. You can replace it with a circumflex or leave it, we'll chop it off later. If you edit your file, as I have, do not save it, but print it to disk. Type <FCTN 9>, then <PF>, and then DSK1.NOCO-TIBX, instead of PIO or RS232. This will keep FunnelWeb from putting those trailing characters in the file which will cause trouble for us later. And now that that file is taken care of and printed to a disk file, let's get to the next program. The program is "D/V=>I/FX", as listed below.

```
100 ! ##### D/V=>I/FX
101 ! Copyright 1988 By Martin A. Sooley
500 OPEN #1:"DSK1.NOCO-TIBX",DISPLAY ,VARIABLE 80,INPUT
600 OPEN #2:"DSK1.NOCO-I/FX",INTERNAL,FIXED 150,OUTPUT
700 OPEN #9:"PIO",VARIABLE 136 :: PRINT #9:CHR$(15)
800 ON ERROR 5000
1000 IF EOF(1)THEN CLOSE #1 :: CLOSE #2 :: STOP
2000 INPUT #1:NM$
2010 INPUT #1:LN$
2020 INPUT #1:FN$
2030 INPUT #1:MI$
2040 INPUT #1:SA$
2050 INPUT #1:CT$
2060 INPUT #1:ST$
2070 INPUT #1:ZP$
2080 INPUT #1:PH$
2090 INPUT #1:XP$
2100 INPUT #1:GP$
2110 INPUT #1:ID$
2300 XPT$=SEG$(XP$,4,2)&"/"&SEG$(XP$,1,2)
2500 P$=NM$&LN$&FN$&MI$&SA$&CT$&ST$&ZP$&PH$&XPT$&GP$&ID$
3000 PRINT #9:P$
3500 PRINT #2:P$
4000 GOTO 1000
5000 CLOSE #1 :: CLOSE #2
5010 ! ##### D/V=>I/FX
5050 ! SAVE DSK6.D/V=>I/FX
5500 END
```

See Next Page.



**TI-Base - TUTORIAL**  
**Extended Basic Background 2.3**  
**NortCoast 99'ers**  
**Copyright 1988 By Martin A. Smoley**

D/V=>I/FX should be a snap for you by this time. First, you need to have your printer turned on for this one. In lines 500, 600 and 700 we are going to open NOCO-TIBX, our D/V 80 file, and NOCO-I/FX, a new I/F 150 file, and the printer. Line 800 is just a safety device that closes everything in case something goes wrong. Line 1000 checks for the EOF in NOCO-TIBX, and lines 2000 through 2110 read or input each of the string variables we printed out in line 4000 in 5M. At last! line 2300. Line 2300 is a much better example of SE6\$(X\$,x,x). Take a look at the expiration dates in NOCO-TIBX on the last page. They are month-year (02-89). This does not sort well. I want them to be year/month (89/02). We just input XP\$ in line 2090, this is a good time to make the change. Remember XP\$ contains 02-89. It is 5 characters in length. In 2300, SE6\$(XP\$,4,2) is saying take XP\$ and starting with character 4, pull out 2 characters. In other words pull out characters 4 and 5, or (89). Because this is to the left side of that total group, within line 2300, it will become the left part of our new variable. In that line we are also saying SE6\$(XP\$,1,2). This means extract 2 characters from XP\$, starting with character 1, or (02). This will wind up on the right side of our new variable. We are not taking out character 3 (-). Now, we take the piece on the left (89), stick it together with a new piece for the middle (/), and stick those together with the new piece on the right (02), and put the whole thing (89/02) into XPT\$. I hope you get this, because this example is pretty clean and straightforward. In line 2500 we are putting all of our string variables together, into one long string variable. The reason for this is to eliminate the hidden length character XBasic places at the beginning of every variable it outputs to a disk file. We will still have one length character at the beginning of P\$ that we must allow for. I have printed the new I/F 150 file named NOCO-I/FX at the bottom of this page. The only thing that you don't see is a "u" just before the plus sign at the beginning of each line. The "u" stands for a length of 117. You will notice that the circumflexes hold the spacing we will need for the TI-Base CONVERT function. This is where we move into the area of TI-Base Version 2.0. We have run the XBasic program named D/V=>I/FX and it has both printed a listing like the one at the bottom of this page and created a disk file named NOCO-I/FX. Use your disk manager to copy NOCO-I/FX to the disk you will use for your TIB DATDISK. Your next step is to load TI-Base Version 2.0. With NOCO-I/FX on the DATDIST type:

**CONVERT NOCO-I/FX NC-DB9 GO <E>**

This will throw you into TIBs CREATE screen. At that point you should enter all the information at the top of the next column. You are CREATEing the database for TIB to pull NOCO-I/FX into.

arrows to move, enter to advance  
**FIELD    DESCRIPTOR    TYPE    WIDTH    DEC**

1	NM	C	5	
2	LN	C	15	
3	FN	C	15	
4	MI	C	2	
5	SA	C	25	
6	CT	C	20	
7	ST	C	2	
8	ZP	C	5	
9	PH	C	12	
10	XP	C	5	
11	GP	C	5	
12	ID	N	7	0

**[ NC-DB9    STRUCTURE ]**

Notice that NM has a length of 5. We need one more space in NM then we can see columns for it at the bottom of the page. +001 is 4 columns, so we make NM 5 columns as above. The first field is the only one that must accept an extra character. The rest of the fields will be whatever we made them back in 5M. When the last item in row 12 has been entered press <FCTN 8> and TIB will do the rest for you. TIB will (by brute force), chop up NOCO-I/FX into the fields you requested, and jam the pieces into NC-DB9. The trouble is when it's done you can't use the database as is. Before anything else, you must type USE NC-DB9 <E>, and right after it looks like TIB has opened the database for you, type RECOVER <E>. After TIB has RECOVERed the file you will be able to USE NC-DB9 as a normal database. You can use EDIT to look around in NC-DB9, but don't bother to change anything because we still have to run it through a COMMAND FILE to clean it up a bit. The two CFs are listed on the next page. MOVEM1 and MOVEM2 are modified versions of MOVED1 and MOVED2 from Tutorial 4. I started with those CFs and added lines to get the end result I wanted. Place the DB named NEWNAMES, from last month on your DATDISK and DO CLEAR to empty it. When MOVEM1 is executed, it, along with MOVEM2, will copy NC-DB9 to NEWNAMES and make a bunch of changes. Here are a couple of important highlights. In MOVEM1 we create a bunch of LOCALs to match fields in NC-DB9 that can have the circumflex chopped off the end. We make these variables one character shorter than their matching fields. Therefore, REPLACE LNT WITH 1.LN will chop off the last character no matter what it is (15 CHARS =>INTO=> 14 CHARS). The next line, REPLACE 2.LN WITH LNT, will copy LNT to the new database, NEWNAMES; however, the fieldlength is 15 again so a space will be added to the end of LN, (14 CHARS =>INTO=> 15 CHARS). We have managed to chop off most of the circumflexes and replace the with blank spaces. As you should be able to see, we have thrown away the NM field and REPLACed it with a completely new number, NUMT. Remember, these tutorials are to teach you, so you can write CFs of your own, not do everything for you.

+001Smoley	^Martin	^A.6149 Bryson Drive	^Mentor	^OH44060216-257-166189/02NOCO^^	^	
+002Whitman	^Raymond (Slim)	^^^2574 East 254th.	^Eastlake OH.	^OH44094951-2345	^88/09NOCO^^	^ See
+003Aardvark	^Grant	^E.9995 State Rt. 84	^Geneva	^OH440141-465-9876	^88/02NOCO^^	^ Next
+004Aardvark	^Willard	^^^No Newsletter	^^	^OH^ ^1-465-7689	^88/09NOCO^^	^ Page.
+005Vivannovitch	^Elexxie	^^^111 E. 98th. St.	^Cleveland	^OH91023541-5415	^88/05NOCO^^	^
+006Jones	^Quincy	^W.37285 Burgandy Lane	^Mentor-on-the-Lake	00H44060257-1029	^88/08NOCO^^	^

```

*****
*
*
*
*          STUDENT $15
*
*          DUES: MEMBERSHIP $20
*
*-----*
*          WORK PHONE
*-----*
*          PHONE
*-----*
*          CITY
*          ZIP
*-----*
*          ADDRESS
*-----*
*          NAME
*
*          DATE __/__/89
*
*          APPLICATION FOR MEMBERSHIP
*
*          DECATUR 99er HOME COMPUTER USERS GROUP
*****

```

```

-----
26 27 28
19 20 21 22 23 24 25
12 13 14 15>>16<<17 18
5 6 7 8 9 10 11
1 2 3 4
-----
S M T W T F S
-----
>>>FEBRUARY MEETING DATE<<<<<
-----
DECATUR, ILL
3465 NORTH MacARTHUR RD.
FIRST CONGRAGATIONAL CHURCH
6:30 PM TO 7:30 PM
ALL MEETING DATES:
THURSDAY, FEBRUARY 16, 1989
NEXT MEETING DATE:

```

DECATUR 99er H.C.U.G.  
 P.O. BOX 726  
 DECATUR, IL 62525

TIME DATED MATERIAL