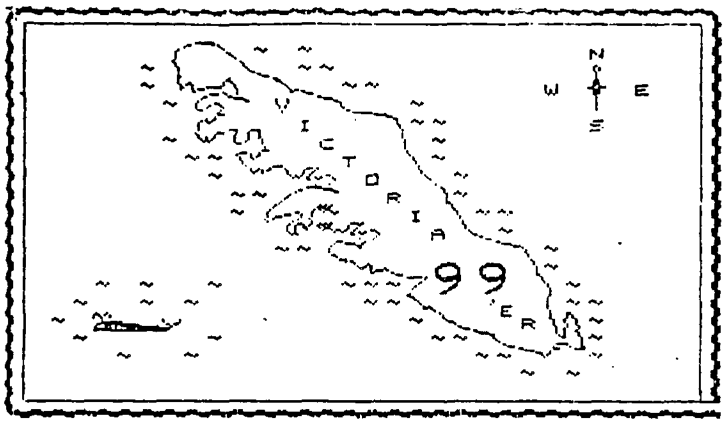

JUNE 1984

****
CHALLENGE : FOR EXTENDED BASIC USERS

THE TASK : ON A DARK SCREEN DISPLAY A SHRINKING RECTANGULAR OUTLINE

```
 -----------
!           !   --------
!           !  !        !     -----
!           !  !        !    !     !    --
!           !  !        !    !     !   !  !    =
!           !  !        !     -----     --
!           !   --------
 -----------
```

CHANGING THE COLOUR OF THE OUTLINE AS IT SHRINKS. REPEAT SEVERAL TIMES.

RESTRICTIONS : 1 PROGRAM LINE ONLY,NO USE OF EXTRA MEMORY OR DISK DRIVE,ONLY "RUN" TO BE ENTERED ON THE COMMAND LINE.
SUGESTIONS: USE CALL COLOR STATEMENTS TO DEFINE RECTANGLES(that is how I did it perhaps you will find a better way)

****
SOME HINTS FOR PROGRAMERS :
ASSIGNING COMMON VARIABLES;
EACH NUMBER IN A PROGRAM USES 2 BYTES + THE NUMBER OF CHARACTERS IN THE NUMBER EG. "1" USES 3 BYTES, "12" USES 4 BYTES, "123" USES 5 BYTES. A VARIABLE USES 1 BYTE FOR EACH CHARACTER IN ITS NAME WHEN USED IN A PROGRAM LINE AND IN ADDITION ANOTHER 8 BYTES IN MEMORY WHEN RUN. FOR EXAMPLE 1 USE OF "X" IN A PROGRAM TAKES 9 BYTES AND 100 USES OF "X" TAKES 108 BYTES. 1 USE OF "XX" USES 10 BYTES AND 100 USES OF "XX" WOULD USE 208 BYTES.
THIS KNOWLEDGE CAN BE PUT TO GOOD USE IN SAVING SPACE IN A PROGRAM. IN THE PROGRAM BATTLIN-SHIPS BY CRAWFORD AND ASSOCIATES THE NUMBER "1" IS USED 91 TIMES IF THEY HAD ASSIGNED A VARIABLE   O=1 AND SUBSTITUTED "O" FOR "1" THE WOULD HAVE SAVED @ 175 BYTES.

STRING NULLING;
IF YOU MAKE A STRING IN A ROUTINE AND WONT NEED IT FOR OTHER ROUTINES AND THE STRING IS 7 OR MORE CHARACTERS LONG,MAKE THE STRING INTO A NULLSTRING (X$="") BEFORE LEAVING THE ROUTINE. DO YOUR OWN GARBAGE COLLECTION!
TRY THIS EXAMPLE. ENTER AND RUN FROM EXTENDED BASIC WITHOUT DISK DRIVE.

```
100 DIM A(1670)
110 A$="AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
120 PRINT A$
130 B$="BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB"
140 PRINT B$
```

YOU SHOULD GET A BUNCH OF A's ON THE SCREEN AND A MEMORY FULL IN 130 MESSAGE.
NOW ADD,

```
125 A$=""
```

THE PROGRAM WILL NOW RUN PROPERLY.

GRAPHICS MANIPULATION XBASIC SUBROUTINES AND JOYSTICK&KEY SUBROUTINE

BY TONY BIGRAS

```
9999 !turns parameter character(C) upsidedown.          SIZE 111 bytes run.

10000 SUB INVERT(C):: CALL CHARPAT(C,C$):: T$="" :: FOR L=15 TO 1 STEP -2 :: T$=
T$&SEG$(C$,L,2):: NEXT L :: CALL CHAR(C,C$):: SUBEND


10099 !makes parameter character(C) into negative image.SIZE 189 bytes run.

10100 SUB NEG(C):: F$="" :: CALL CHARPAT(C,C$):: FOR L=1 TO 16 :: N=ASC(SEG$(C$,
L,1)):: IF N<57 THEN N=63-N ELSE N=70-N
10110 IF N<10 THEN T$=CHR$(N+48)ELSE T$=CHR$(N+55)
10120 F$=F$&T$ :: NEXT L :: CALL CHAR(C,F$):: SUBEND


10399 !scrolls character definition to make character scroll down itself. SIZE 9
2 bytes run.

10400 SUB SCROLL(C):: CALL CHARPAT(C,C$):: C$=SEG$(C$,15,2)&SEG$(C$,1,14):: CALL
 CHAR(C,C$):: SUBEND


10499 !SP=speed, R=display at row number, M$=string to be printed max len 225 ch
aracters.  SIZE 134 bytes run.

10500 SUB TICKER1(SP,R,M$):: M$=RPT$(" ",27)&M$&"  " :: FOR L=1 TO LEN(M$)-1 ::
DISPLAY AT(R,1):SEG$(M$,L,28):: FOR S=1 TO SP :: NEXT S :: NEXT L :: SUBEND


10599 !SP=speed,RT=rate of letter shift,R=row,C=column,L=length of window,M$=str
ing to write,variable max=250 -L display max=116.

10600 SUB TICKER2(SP,RT,R,C,L,M$):: T2$=M$ :: T2$=RPT$(" ",L)&T2$&RPT$(" ",1+RT)
:: FOR T2=1 TO LEN(T2$)-1 STEP RT :: DISPLAY AT(R,C)SIZE(L):SEG$(T2$,T2,L)
10610 FOR T1=1 TO SP :: NEXT T1 :: NEXT T2 :: SUBEND


10798 !Y$ of "Y"= preserve scroll,SP=speed,RT=rate of letter shift,R=row,C=colum
n,L=length of window,M$=string to write
10799 !max len of string       255 -(L+4). SIZE

10800 SUB TICKER3(Y$,SP,RT,R,C,L,M$):: IF Y$<>"Y" THEN 10820
10810 D$="" :: FOR L3=C+2 TO C+L+2 :: CALL GCHAR(R,L3,T):: D$=D$&CHR$(T):: NEXT
L3
10820 M$=RPT$(" ",L)&M$&RPT$(" ",1+R):: FOR L2=1 TO LEN(M$)-1 STEP RT :: DISPLAY
 AT(R,C)SIZE(L):SEG$(M$,L2,L)
10830 FOR S=1 TO SP :: NEXT S :: NEXT L2 :: IF Y$<>"Y" THEN SUBEXIT
10840 DISPLAY AT(R,C)SIZE(L):D$ :: SUBEND


19999 !this accepts input from keypad or joystick and produces output in a form
identical to CALL KEY(SIDE.KEY.STATUS)

20000 SUB JOYSTK(SIDE.KEY,STATUS)
20010 CALL JOYST(SIDE,X,Y)
20020 IF X=0 AND Y=0 THEN STATUS=0 :: GOTO 20120
20030 STATUS=1
20040 IF X=0 AND Y=4 THEN KEY=5 :: GOTO 20120
20050 IF X=4 AND Y=4 THEN KEY=6 :: GOTO 20120
20060 IF X=4 AND Y=0 THEN KEY=7 :: GOTO 20120
20070 IF X=4 AND Y=-4 THEN KEY=14 :: GOTO 20120
20080 IF X=0 AND Y=-4 THEN KEY=0 :: GOTO 20120
20090 IF X=-4 AND Y=-4 THEN KEY=15 :: GOTO 20120
20100 IF X=-4 AND Y=0 THEN KEY=2 :: GOTO 20120
20110 IF X=-4 AND Y=4 THEN KEY=4 :: GOTO 20120
20120 CALL KEY(SIDE,K,S):: IF S=0 THEN 20150
20130 IF SIDE=1 THEN KEY=K :: STATUS=S
20140 IF SIDE=2 THEN KEY=K :: STATUS=S :: IF KEY=18 THEN KEY=11
20150 SUBEND
```

T I 9 9 E R   G R O U P

STATEMENT OF ACCOUNTS

OCT 83 - SEP 84

	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	TOTAL
REVENUES													
C-F PREV. MO.	54.82	190.02	402.66	344.00	344.75	294.87	287.12	301.40					54.82
MEMBERSHIPS	150.00	225.00	0.00	0.00	175.00	0.00	30.00	25.00					605.00
INTEREST EARNED	0.20	0.64	1.17	0.75	0.63	0.65	0.89	0.34					5.27
DONATIONS	0.00	3.00	0.00	0.00	0.00	0.00	0.00	0.00					3.00
AUCTIONS-RAFFLES	0.00	0.00	0.00	5.00	0.00	0.00	0.00	0.00					5.00
OTHER	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00					0.00
TOTAL REVENUES	205.02	418.66	403.83	349.75	520.38	295.52	318.01	326.74					673.09
EXPENDITURES													
RENT	15.00	15.00	0.00	5.00	20.00	0.00	0.00	0.00					55.00
HARD-SOFTWARE	0.00	0.00	0.00	0.00	123.05	0.00	0.00	150.00					273.05
NEWS LETTER	0.00	0.00	0.00	0.00	0.00	8.40	16.61	16.80					41.81
SUPSCRIPTIONS	0.00	0.00	59.83	0.00	82.46	0.00	0.00	0.00					142.29
MISC.	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00					1.00
TOTAL EXPENSES	15.00	16.00	59.83	5.00	225.51	8.40	16.61	166.80					513.15
NET BALANCE	190.02	402.66	344.00	344.75	294.87	287.12	301.40	159.94					159.94

TI-Wycove FORTHs: a QUICK COMPARISON. BY JOHAN VAN IMSCHOOT

The TI release may be considered superior to the Wycove version 1.0 simply because a lot more extra Forth source code is supplied with it. The same is not true of Wycove's version 2.0, just recently published. In fact, this version offers even more programs than the TI version. The two products, TI's and Wycove's, cover more or less the same material. Both are excellent. Both are Fig Forth, and their core is fairly compatible. (Unfortunately, they are not entirely so.) I cannot help suspect that Wycove saw the TI product before they completed their new FORTH, and that they designed it to match TI's, and to better it. I do not mean to imply that they used TI code, only that they seem to provide the same features. (For example, they included a new compatible CASE word, and redid their disk access system to be faster, like TI's.)

Forth being what it is, even Wycove's first version could be used to make any sort of system support material. Users prefer to have things ready made, however, and now both products offer complete filehandling, graphics, etc... In two areas does Wycove offer code, and TI did not: sound, and speech. The Wycove second version includes sample music writing support software, using soundlists (much as LOGO does), but entering the music directly, not as complex hexadecimal codes, as in their first version. A user is of course, as always with Forth, free to make up his own system support software if the provided material is not to his liking. The speech software is capable both of using the speech synthesizer's built-in speech, and the speech production capabilities of the Terminal Emulator II module (TI product). To use the latter either the console or the module must be modified, or else an item like the Widget (from Navarone Industries) must be in use, in order to avoid the automatic reset which inserting the TEII cartridge would otherwise cause.

Wycove's version 1.0 editor, although full screen, was a poor one. I am not overly fond of either their new one, or the TI ones, and have written my own as a result. Nevertheless, the supplied 40 character and 60 character products for both TI and Wycove are reasonable editors. The 64 char editor for both is supported by bit-map mode control software, but whereas TI provides two screen modes, Wycove provides a single full-screen one. Potentially they are all useful.

One could go on about the mass of support software supplied by both TI and Wycove, but there is little point. No matter which product one has, it should be possible to adapt this software to the other, given one has enough understanding of Forth. Although the disk-reading is not fully compatible, one can transfer screens between them: I have already done so using Wycove Forth itself. When I did so for Wycove's original version I used Extended Basic to do the actual transfers, but this is not necessary.

What is much less easy to change (although no doubt not impossible), is the core of the two Forths). This is especially the case for TI, since it seems to have devoted a great deal of memory to writing support routines in machine code. Needless to say these would not be easy to decode and modified, especially as the source assembly code does not appear to be available. It is for this reason that I give the Wycove product an edge: its core system is more flexible and easier to use, generally speaking. It has the following in interactive mode:
-- full line editing (with TI one can only backspace)
   (delete,insert,left and right arrows, and function clear:
   only REDO is missing.)
-- screen width control (very useful for most of us using televisions for
   monitors: the system can skips columns on either side of the screen.
   TI does not offer this convenience.)
-- the possibility of entering hex numbers in decimal mode by preceding
   them with the hex sign: this avoids many errors, clearly distinguishes
   between hex and decimal numbers, which in standard Forth can only be
   done by searching earlier code to see what the current base is. Move
   a line of code containing 12 from one spot to another and it may no
   longer work because the base is different there, but with the Wycove
   method, it will always work if one maintains decimal mode by convention
   and enters hex numbers as >.... inside the code.
   (Of course, the hex mode is still available, and necessary for printing
   out hex numbers, etc.... As usual almost any base is available.)

-- stack size prompting: this is a minor convenience but is helpful: when
   a routine finishes one is told if it has left something on the stack: the
   ok prompt includes the number of items on the stack.
-- single-command system saving can create a new version of FORTH
   containing all words currently resident.
-- disk write protection. The Wycove version appears to provide some file
   protection. When a diskette is initially inserted, and the screens file
   accessed, the system will use the catalogue to locate the screens file,
   or create one if not there. It may be possible to share a diskette with
   non-FORTH material (I would not recommend it. Further, it may be that the
   screens file would have to be consecutive on the diskette: this would in
   any case be preferable.). If a new diskette is inserted the user must
   himself close the old diskette file, however, in order to retain the
   protection (or in some case even to correctly access the screens file--
   depending on its location).
   TI offers no protection: as I was only familiar with
   Wycove's version 1 when I received my TI Forth, I inadvertently used it
   to destroy part of an Extended Basic program. This could not have
   happened in Wycove's first version. Of course, now that I am aware of
   the possibility, I am actually glad these new FORTHs will allow me to look
   at any disk (disk fixing, and searches, become possible, as long as the
   screens are not limited to part of the disk).
   The TI method appears to bypass the catalogue entirely: it probably
   accesses the disk directly somehow, and I suspect, always beginning the
   screens file in the same location. (It may look different to Basic due to
   a different catalogue header having been written.)
   In any case, there is no protection. Diskettes for TI FORTH should be
   reserved for TI FORTH only, and others used with caution.
-- cassette useage: all that is required to run Wycove Forth is extra
   memory, the cassette recorder, and one of 3 modules. Ti Forth will
   only load from diskette. (This could probably be modified with much work.)
-- module compatibility: the Wycove Forth version 2 (unlike its version 1)
   is independent of the module used to load it. It can be loaded with
   Extended Basic, Mini Memory, or Editor Assembler modules, and once loaded
   runs independently from the module. (All 3 modules are TI products.)
   (It is even possible to remove the module.)
   TI FORTH requires the Editor/Assembler module for loading, and APPEARS
   to require it even after loading, although it has a number of its own
   utilities: perhaps TI was not consistent in the design here.
   This limits its use somewhat.
-- as a result of the above feature, Wycove FORTH can be used with any TI module
   in place; this, as with the TEII speech program, can have important benefits.
-- possible recovery, or Forth restart from EASYBUG in Minimem;
   without having to reload Forth (in some situations one can even restore
   Forth with the material that was already loaded in still accessible.)
   TI FORTH cannot be started with the EASYBUG Execute option as its address
   interpreter resides in the PAD, which gets cleaned out by the system reset
   code.

The main advantage that the TI system offers is the binary image overlay
system. I have by now adapted this to the Wycove system with practically
no modification. It shows that many of the differences between the systems
can be resolved: both systems can benefit from the other.
Another possible slight advantage in TI's core system is the PAUSE word.
The same function exists in the Wycove product, but I have not yet found
(nor searched for) a user accessible word for it.
Split-screen graphics, as set up by the TI system, have some great advantages
for some applications. However, this should be readily transferable to the
Wycove system, as it is supplied in source code. (No doubt modifications
would have to be made.)

Wycove Forth, for fifty dollars, comes with a 180 page manual, single-
spaced for the most part, and thus at least as large as the TI manual.
This is a definite improvement over their first version, as is the product
generally. For the price, one diskette and one cassette tape are also
included, as are mailing charges. I consider this eminently reasonable.

The TI product, being in the public domain, is FREE. What does that mean?
It can mean that you supply your own diskette, your own paper, and get
your own photocopying done. The price will therefore
depend on the circumstances. (For example, you may need
extra equipment.)

One last possible advantage of the Wycove version: Wycove made me aware of, and offered to me, as past owner, their new Forth at a reduced price. That is, with Wycove, you can get updating support. Who knows, they may be nice enough to let owners know of any errors that crop up. (In their first version I didn't come across any, except for an error in the manual.) TI has washed its hands off TI FORTH, by placing it in the public domain, and will not support it in any way.

Which to choose? No doubt the bottom line will be price and availability. One may have to consider whether one has or is willing to buy the necessary module (possibly even a disk drive!).
You will have to decide for yourself. TI FORTH should be available to members of user groups. (It is also available from certain dealers.) Wycove Forth can be ordered from Wycove Systems Limited, P.O.Box 499, Dartmouth, Nova Scotia, B2Y 3Y8, or it may perhaps be available from some dealers (I don't know). If you can afford it, why not get the benefits of both? By the way, although I have no personal financial or social stake in Wycove Forth, I request you please not copy it (such copying is very easy). They are asking a very reasonable price. Without sales support, the excellence of version 2 would not have been possible. I do have a vested interest in seeing this company survive and produce still more material! (Don't we wish TI's Home Computer Division had survived?...)

To sum up, both FORTHs are excellent languages to add on to your TI system. If you haven't yet got FORTH, and would like to have more speed, and other capabilies from your machine than Basic will offer you in your programming, do by all means acquire it. To learn about Forth, Leo Brodie's book, Starting Forth, is to be recommended. Both TI and Wycove manuals have a chapter explaining differences between their Forths and the one described by Brodie. Good Luck.

FROM A LOCAL BBS :

Message #769 is dated 0:46 AM. 4-Jun-84
To: ALL TI99 USERS. from THE SYSOP
Subject: Software for trade

I received a call from Doug Howe of Winnipeg Manitoba. He requested that I leave his address for any interested TI99 users. He is the TI99 Club librarian there, and has ALOT of public domain software for trade. If you wish to get in contact with him regarding trading, here is his address.

Doug Howe
1058 Louelda Street
Winnipeg, Manitoba
R2K 3Z4

EDMONTON USERS GROUP
P.O. BOX 11983
EDMONTON
ALTA. T5J 3L1