(006) 8501

VOLUME NO. 2 ISSUE 12 JANUARY 1985

Bayou 99 Users Group, P.O. Box 921, Lake Charles, La. 70602

BAYOU BYTE

BAYOU 99 USERS GROUP

Price One Dollar Distribution: 150 copies

MEETING NOTICE

The January meeting of the Bayou 99 Users' Group will be at 7:00 P.M. on January 10th at the Nelson Elementary School. Anyone interested in learning to use the capabilities of the 99/4A is invited.

****

CONTENTS

****

MINUTES

The December meeting consisted of discussions of various software packages which have recently become available. Notice was made of reduced prices for TI software due to TI's efforts to avoid returns of unsold products.

Vice-President Richard Mitchell gave an impressive demonstration of the rapid transfer of memory image screens. The setup used by Richard utilized the Cor-Comp Disk Controller Card.

A few copies of Tom Nicosia's "Death of a Computer" as published originally in Texas Monthly and reprinted in Infoworld were available to members at the meeting. Two command modules were awarded as door prizes.

DUES

Beginning January 1, 1985 all members whose dues expire will be requested to add $1.00 per month additional for each month between their anniversary date and July 1, 1986. For members renewing in January, the requested $18.00 will cover membership dues for 18 months. Those due in July will pay the usual $12.00 and each succeeding month's dues will be one dollar less until December 1985 when the dues will be $6.00 to cover the six months until July. After December 1985 all membership dues will be payable in July each year.

This arrangement will simplify the present difficulties experienced by the Treasurer in collecting and crediting dues payments.

Your Newsletter Editor will also benefit greatly from this resolution adopted by the Officers of your Users Group.

According to our membership records, the following members' dues are payable January 1, 1985: William F. Campbell - Glenn Delahoussaye - Ken & Colleen Jordan - Andrew McGowan - and - Melvin Schmidt.

These members are requested to remit their dues in the amount of $18.00 for an 18 month membership until July 1, 1986.

The following members will be requested to remit $17.00 for membership from February 1 to July 1, 1986: L. M. Lowery - Mark Hammon - Mark Wilson - and - Bruce Wyman.

PRE-SCAN

BB Staff

An often neglected feature of extended BASIC is the ability of a programmer to decrease the length of the delay between the time RUN is entered and when the program starts. This delay is the time required by the CPU to pre-scan the program and allot space for arrays, data, and variables. The time required for this pre-scan of the program increases with the complexity and length of the program. If the delay time becomes bothersome, Extended BASIC programmers can utilize the pre-scan commands !@P- and !@P+ to turn the pre-scan step off and on.

When RUN is entered pre-scan is on and continues until the computer encounters a !@P- which will turn the pre-scan function off. The pre-scan remains off until a !@P+ is encountered. It is good practice to have the pre-scan "switch" isolated by itself on a program line. If a multiple statement line contains either of the "switch" commands !@P- or !@P+, the swtich command must be the last statement on the line.

If the program is working successfully, enter the off switch, !@P-, after the OPTION BASE and DIM statements and when a new variable, CALL,DEF,SUB, or SUBEND statement occurs, precede the statement with a program line containing the "on" switch, !@P+. By turning pre-scan on and off as required to include the first occurrence of a statement requiring memory allocation, the pre-scan time can be significantly reduced.

Remember, if the pre-scan were turned off when a space allocation in memory was needed, a syntax error will occur when your program reaches the statement which should have been included in pre-scan. The easiest way to prevent this kind of error is by using a multiple statement line (or lines) to list all the variables, CALL, DIM and DEF statements on program lines by-passed by a GOTO instruction when the program executes. These lines should appear in the first part of the program. If DATA statements at the beginning of a program offend you, the pre-scan can be turned on again prior to DATA statements at the end of your program.

An example of how the pre-scan switches could be used follows:

```
100 CALL CLEAR :: OPTION BASE 1 :: DIM AR$(10,6)
110 GOTO 160 :: CALL KEY :: CALL HCHAR :: CALL VCHAR :: CALL COLOR
120 CALL SCREEN :: CALL SOUND :: CALL SUBP
130 ALPHA, A, B, R, J, K, X, Y, Z, NAME$, ST$, CT$, ZIP$, STATE$, NO$
140 DEF F(W) = X^2 + Y^2
150 !@P-
160 REM
 .
 .      Program
 .
1790 REM
1800 !@P+
1810 DATA NAME, STREET, CITY,STATE, ZIP, PHONE NO.
1820 END
```

Using these "switch" commands in a program will require careful planning the first few times they are used. The use does, however, become easier with practice and the rewards are worth the effort. The frustrations of sitting in front of a computer that is giving no indication it is doing anything increase rapidly in just a few seconds. You now can significantly shorten a delay that occurs with every program.

PASSING VARIABLES

Passing variables from one part of the program to another presents no problem to the programmer of the TI-99/4A in any of the programming languages available; BASIC, Extended BASIC, FORTH, Pascal and Assembly Language reserve an address in memory for each variable defined in your program. The syntax varies depending on the program language, but each variable will have its own address and the value stored in that address changes according to the values generated by the program. Most operating systems require variables to be initialized. An HP computer program must have an initial value for each variable. That is to say, if you intend to use a variable named Z, then you need to have a statement, Z=Ø in the program. The 99/4A and the IBM-PC initialize all numeric variables to zero and all string variables to null before the program runs.

The TI-FORTH and TI Assembly Language, the variables are initialized at the start of the program. In FORTH you would enter Ø variable Z; in Assembly Language you would use Z DATA 0. Of course, the initial value could be non-zero and we would use 3 VARIABLE Z (FORTH), Z DATA 3 or Z TEXT "3" (Assembly) or VAR Z: INTEGER (PASCAL).

Extended BASIC permits passing of specified variables to a subprogram by use of CALL (subprogram name) and SUB PROGRAM NAME (var list) statements. If a subprogram has been named TEST, then CALL TEST will cause the program to branch to SUB TEST if the statement was SUB TEST (Z,A), then the values of Z and A at the time the branching took place would be transferred to the subprogram. In either BASIC the program may be interrupted with FCTN 4(Break) or (Clear) and the value of any variable can be obtained by entering PRINT Z or any other program variable. The fact that all variables have an initial value can be shown by a PRINT command for a variable not used in the program. If you typed PRINT DECEMBER and DECEMBER had not been assigned a value or had not been mentioned before, the value Ø would be printed on the screen.

As stated above the value of a variable is always present in memory and if that variable is called by the program or a command, the value of the variable is retrieved from the variable's address. However, if a RUN, OLD or MERGE command is entered, all variable addresses are cleared of any values or characters they may have contained.

The first time this caused a problem for me was in writing an accounting program that used more memory than was available. The program was then broken down to separate the one large program into four smaller programs and in this way keep the memory requirements to sizes the 99/4A could handle. My disk now had a short LOAD program, a journal ENTRY program, a SORT program, and a POST program. The main menu was contained in the ENTRY program and selection of the sort and post option in the menu branched to RUN statements which loaded the required supporting program. These supporting programs were concluded with a RUN statement to reload the ENTRY program with its menu.

Since the ENTRY program contained security routines and instructions that did not need to be repeated each time a chained supporting program concluded, a way was needed to set a flag that would cause the program to bypass password and instruction steps and start with the main menu. All variables and there were many written to a file prior to any RUN statements. The supporting programs then opened the appropriate file and read the values to be used. The supporting programs then wrote the new values to a new file. For example, daily transactions were entered from the ENTRY program and written to a random access disk file. If a sort was requested, the SORT program read the entries from the first file, performed the required sort and wrote the results to a second file. This offered a way to set a flag in the main program. A file could be opened and a value read which an IF...THEN statement could use to bypass the program steps which preceded the display of the main menu. This could cause a lot of problems since the file was permanent and DELETE cannot be used in a TI BASIC statement.

A solution was discovered by accident. While writing a program which used every character code available for a graphics character, I needed to run another program. After running the second program, I returned to the first and found that the values assigned to certain character codes were unchanged. Further checking confirmed a pattern identifier assigned to character codes 127-143 with a CALL CHAR statement remained unchanged until a new value was assigned or the computer was turned off.

Changes were then made to include a CALL CHAR statement in each supporting program which used one of the character codes (127-143) and a pattern identifier (a 16 place string can be used). When the supporting program loaded and ran the ENTRY program, the ENTRY program immediately checked for a particular string with a CALL CHARPAT statement. If the proper string was present, the program immediately jumped to the main menu solving the delay problem.

The utility of this CALL CHAR statement can be extended to pass variable values to a program that has been chained to another with the RUN statement.

A short demo program will illustrate the use of CALL CHAR and CALL CHARPAT to pass a value.

```
100 !*********************
110 !+                   *
120 !*   DEMO PROGRAM    *
130 !*       ==++==      *
140 !*PASSING  VARIABLES*
150 !* BETWEEN PROGRAMS *
160 !*        by         *
170 !* Roger  Hickerson *
180 !*   Bayou 99 U.G.   *
190 !*                   *
200 !*********************
210 CALL CLEAR
220 CALL CHARSET
230 PRINT "ENTER THE VALUE TO BE PASSED USING A 'D' FOR THE DECIMAL POIN
T."
240 PRINT "FLOATING POINT EXPONENTS AREENTERED NORMALLY PRECEDED  BY AN
'E'."
250 PRINT "THE FINAL CHARACTER FOR ANY NUMERIC MUST BE AN 'F'.      VALUE
S ARE PA
SSED ON IN THE FORM OF A STRING VARIABLE."
260 PRINT "IT WILL BE NECESSARY FOR THESTRING TO BE CONVERTED BY A VAL S
TATEMENT
 BEFORE USE IN ANY COMPUTATIONS."
270 INPUT S$ !INPUT NUMBER AS A STRING VARIABLE
280 IF POS(S$,"F",1)=0 THEN PRINT "YOU MUST HAVE AN 'F' AS THE FINAL CHA
RACTER."
 :: GOTO 270 ! TO MAKE SURE YOU READ THE DIRECTIONS
290 CALL CHAR(127,S$)!NUMBER STRING USED AS PATTERN IDENTIFIER
300 RUN 310 ! CLEARS ALL VARIABLE MEMORY LOCATIONS
310 CALL CHARPAT(127,G$)! RECALLS PATTERN IDENTIFIER
320 PRINT G$
330 D=POS(G$,"D",1)! LOCATE DECIMAL POINT
340 E=POS(G$,"E",1)! LOCATE E FOE EXPONENTIALS
350 F=POS(G$,"F",1)! LOCATE END OF STRING
360 IF D=0 THEN 420
370 I$=SEG$(G$,1,D-1)!SEPARATE INTEGER
380 IF E=0 THEN F$=SEG$(G$,D+1,F-D-1):: GOTO 490
390 F$=SEG$(G$,D+1,E-D-1)!SEPARATES FRACTION
400 E$=SEG$(G$,E,F-E)!SEPARATES EXPONENT
410 GOTO 490
420 IF E=0 THEN 470
430 I$=SEG$(G$,1,E-1)
440 F$=""
450 E$=SEG$(G$,E,F-E)
460 GOTO 490
470 I$=SEG$(G$,1,F-1)
480 E$=""
490 IF D=0 THEN P$="" ELSE P$="."
500 PRINT I$&P$&F$&E$
510 N$=I$&P$&F$&E$
520 PRINT N$
530 N=VAL(N$)! CONVERT STRING TO NUMBER
540 PRINT N*3+N/6 !PERFORM COMPUTATION FOR PROOF
550 END
```

A reader's challenge is to add to the program so that both positive and negative numbers and exponents can be handled.

[illegible]

ARRAYS

Arrays can be one of the most useful tools available to a programmer. One of the most used applications is to input data to a file and to read it back from a file. One dictionary definition states, quite simply, that an array is "a regular grouping or arrangement." TI's BASIC Users Reference Guide says "an array is a collection of variables arranged in a way that allows you to use them easily in a computer program." The reference guide goes on to say the most common way of grouping variables is a list which is called a one-dimensional array.

Arrays may be one-, two - three - or up to seven-dimensional when using the TI-99/4A and each is a collection of variables. Each value in an array is an element of the array or as some sources describe array elements, a subscripted variable. The subscript is in reality a pointer which is shown in parenthesis following the array name. Let's back up momentarily and find out what we need to know about array names. First, an array name must be consistent with and identify the types of variables in the array. Arrays of numbers must have numeric variable names; i.e., X, AA, A2, LIST, etc. Stirng arrays must likewise have string variable names; i.e., X$, AA$, GROC_LIST$, etc. and an array name must not duplicate a variable name used elsewhere in the same program. For example, the variable X and an array element X(1) cannot both be used in the same program. The use of the numeric variable X does not prevent the use of an array named X$ however.

How do we use an array? Again referring back to the BASIC Reference Guide, a one-dimensional array is simply a list, and so for a list, we would use a one-dimensional array. If the list will contain no more than 10 items, we can assign values to A(0), A(1), A(2)...A(9) without any problems. If, however, the array will contain more than 10 elements, the array must be dimensioned.

To dimension an array a DIM statement followed by the array name and number of elements (items) to be included on the array. For example, if we want to list the items on our grocery list which we named LIST$, the dimension statement would be DIM LIST$(25) which would provide a variable for 26 items on a grocery list. Did I say 26? Yes, since the subscript for the array variable start with zero. If you want the first element to be LIST(1) instead of LIST$(0), you need only to include the statement OPTION BASE 1 in your program. Both OPTION BASE and DIM statements should appear early in the program with OPTION BASE ahead of DIM.

These one-dimensional arrays are easy to use and often so convenient that programmers sometimes use several one-dimensional arrays when a two-dimensional array would better serve our purpose. If a list of the members of your bowling team were needed and their handicaps were also needed, an array NAME$(10) (we have some extras on the team) and an array HCAP(10) could be kept where NAME$(5) was Charles Smith and Charles's handicap was in HCAP(5). Examples of programming with single and multi-dimensional arrays will be given in a later issue.

LAGNIAPPE

* SCL Tech plans to market a 64K RAM disc card for the 99/4A expansion box. The card will be expandable up to 256K in increments of 64K. The card will be a disk emulator with an information transfer rate over 10 times faster than with conventional disk drives.

* In the "Smart Programmer" from Millers Graphics we have learned that Millers Graphics has a contract to work on the firmware for the RAM Disk Card. In the same article was the information that an external power supply option will be available. The date of release or price has not been released.

* In Tips from the Tigercup #16, Jim Peterson offered a challenge. (1) "How can you store a hundred or more values of any size, positive or negative, inter or non-interger, even in exponential notation, without dimensioning an array or opening a file?" And, (2) "How can you link programs by a RUN statement,thereby losing all data, and recover those values?" Saving them on the screen is one way, but the challenge is to find a better way. How about working on these challenges! See the article "Passing Variables" in this issue.

* The Exceltech Extended BASIC modules can be purchased for $79.95 in quantities of five or more. If you need this module (everyone does) and you don't have one yet, let one of the officers or the Library Committee members know. A few members have already said they will buy and we need a couple more orders to get the full discount.

* The Bayou Byte now has a TI Bulletin Board (TIBBS(tm)) program. Roger Hickerson purchased the program which is now set up and operating at his house. The telephone number is 474-6144 and the normal operating hours are 8:30 P.M. to 10:30 P.M. Roger would like someone to take it over so that it can be available several more hours per day and all day on weekends. A double disk drive, Hayes 300 (or 300/1200) Smart Modem and 32K (or 128) memory expansion is required equipment. If you have the time and equipment, give Roger a call at 477-3687

* We have a copy of Tigercub Software's Catalog #5 in the Library. Take some time at the next meeting to look this catalog over. There are entertainment and educational programs for all ages. The listening, learning and writing music programs range from excellent to exceptional. The price is only $3.00 each for cassette or disk, plus a few cents postage. These titles will never be available in the B99UG Library, but the Library will send your order in if you ask them.

* Thanks to Mike Kelley, 4013 Honeycutt Street, San Diego, CA 921D9, TIBBS phone #619/276-3173, we have a tip for Terminal Emulator II users. If you are tired of the TE II screen colors, the next time you are ready to go on-line, enter all the default values and have your modem on, then type CTRL., SHIFT G, FCTN V, CTRL., Shift 9, Shift = and then choose a foreground and a background color with:

!	Black	'	Cyan	$	Dk. Green
"	Med. Green	(	Med. Red	-	Magenta
#	Lt. Green	)	Lt. Red	.	Grey
$	Dk. Blue	*	Dk. Yellow	/	White
%	Dk. Red	+	Lt. Yellow		

Finally, type CTRL. and Shift 10.

The screen changes color after the second (background) color is entered, but you must complete the entry to gain control of the keyboard to print to the screen.

When the commands are complete, the color change occurs on both the originators and receivers screens.

****

FOR SALE

New 300/1200 Baud Hayes Modem - Vernon Hill
Call 478-3817 After 7:00 P.M.

****

PEEKS and LOADS

"The Original"
Atlanta #1 TIBBS

**CALL LOAD(-31806,16)..DISABLES FCTN QUIT KEY
**CALL LOAD(-31806,64)..KILLS SPRITES
**CALL LOAD(-31806,32)..DISABLES AUTO SOUND PROCESSING
**CALL LOAD(-31806,128).DISABLES FCTN QUIT, SOUND AND SPRITES
**CALL LOAD(-31806,0)...RESTORE ANY OR ALL OF THE ABOVE
**CALL PEEK(-31974,A,B). in the command mode. Then..PRINT A*256+B-1776. This is roughly the equivalent to the SIZE command in XB. The 1776 figure is the approx. overhead in TI BASIC. XB has slightly more. If you have ever had a very, very long program and are unable to run it with your disk drives, this is for you. It is much easier with MINI-MEM, and that explanation follows.
**CALL LOAD(-31888,63,255)::NEW..frees memory/disables disks.
**CALL LOAD(-31888,55,215)..then, RUN, NEW or EDIT to restore.
This is equivalent to CALL FILES(0) in XB (which of course you can't do) and has the effect of completely disabling the disk drives, and freeing up the memory allocated to the disks. Any calls to the drives, once the LOAD has been involked, will FREEZE THE COMPUTER, and you will have to turn it off to restore. Involking this command prior to loading your long program via cassette, will negate your having to turn your PES on and off again.

MINI-MEM....

With the mini-mem installed, it's even neater, and you can save your very long programs on disk and use them again, WITHOUT having to turn your PES on and off. here's how.

1. Use the call load command above.
2. Load your long program via cassette. Then save EXPMEM2.
3. Restore your disk by typing CALL FILES(1).....NEW....then OLD EXPMEM2.
4. Save to DSK1. under whatever name you desire.
5. When you wish to use the long program, merely CALL FILES(1), OLD DSK1. PROGRAM, SAVE EXPMEM2, CALL LOAD(-31888,63,255), NEW, OLD EXPMEM2.
6. Run your program.
7. If you still get a MEMORY FULL message at that point....sorry, I can't offer any more than that. To restore the DRIVES without turning the PES off and on, CALL LOAD(-31888,55,215)::NEW or RUN or EDIT.

TIPS FROM THE TIGERCUB

#17

TIGERCUB SOFTWARE
156 Collingwood Ave.,
Columbus OH 43213

My new catalog #5 is now available for $1.00, which is deductable from your first order. It contains over 130 programs in Basic and Extended Basic at only $3.00 each (plus $1.50 per order for casette, packing and postage, or $3.00 for diskette, PP&M).

The entire contents of Tips from the Tigercub Nos. 1 through 14, with more added, are now available as a full disk of 50 programs, routines and files for only $15.00 postpaid.

Nuts & Bolts is a diskfull of 100 (that's right, 100!) XBasic utility subprograms in MERGE format, ready for you to merge into your own programs. Contents include 13 type fonts, 14 text display routines, 12 sorts and shuffles, 9 data saving and reading routines, 9 wipes, 8 pauses, 5 music, 2 protection, etc., etc., all for just $19.95 postpaid!

And if you send an order before 31 December 1984 and mention your user group, you may take a 10% discount.

My 28-Column Converter, published in Tips #15, has a bug which causes a line to disappear if the wrap-around causes it to begin with a period and you are using the formatter option. Here is the fix -

Change line 300 to read: 300 FOR W=1 TO 5 :: READ CH$,R$

Change line 280 to read:

280 DATA @,{,&,},^,~,*,!,.,\ In other words, your DATA items will be the "at" sign above the 2, the left brace on the front of the F key, the ampersand on the 7 key, the right brace on the front of the G, the carat sign above the 6, the tilde on the front of the W, the asterisk above the 8, the whatsit? on the front of the A, the period, and the backslash on the front of the Z.

A couple of other changes will automatically turn off the automatic fill and adjust, and turn it back on. At the end of line 180, add :: PRINT #2:".NF" and change line 270 to NEXT J :: PRINT #2:".FI:AD;"
:: CLOSE #2 :: CLOSE #1 :: END

Now, as long as the text strings in your program don't contain those oddball characters, all should be well. However, the program has one more bug which is common to all 28-column converter programs, and for which I can find no really good fix. If a program line is exactly 80 characters long, the next program line will follow immediately after it instead of starting on the next line. So, load the file in the Editor mode and scan it before you print it. If any of you whiz kids (or whiz grandpas) can figure out a way to program around that problem, please let me know!

A challenge in Tips #9 was to write a 1-line XBasic program which would take only 70 seconds to scramble the numbers from 1 to 255 into a completely random sequence without duplication. Richard Mitchell, the editor of Super 99 Monthly, came up with an algorithm which is shorter than mine and runs about 10 seconds faster - but it sure does chew up a lot of memory!

```
1 DIM A(255),C(254):: RANDOM
IZE :: CALL PEEK(-31808,B)::
IF B=0 OR A(B)=B THEN 1 ELS
E C(D)=B :: A(B)=B :: D=D+1
:: IF D=255 THEN END ELSE 1
```

And if you're not subscribing to Super 99 Monthly, you should be! It's only $12 a year, and full of very useful programs, routines and tips. The address is Bytemaster Computer Services, 171 Mustang Street, Sulphur LA 70663.

Also be sure to get the National Ninety-Niner from the 99ers Users Group Association (3535 So. H St. #93, Bakersfield CA 93304), also only $12 a year. Their roster of writers is beginning to look like the Who's Who of the TI world.

Danny Michael has written an assembly language program which will dump a graphics screen to a dot matrix printer (Epson or Gemini, and probably others) in less than 50 seconds - and he's giving it away. Just send him an initialized disk in a diskette mailer with an address label back to you and enough return postage. His address is Route 9, Box 460, Florence AL 35630.

Please, can ANYONE tell me where can buy diskette mailers at a decent price? The cheapest I have found are $0.65 each for an 11" x 9" piece of cardboard!

Somebody said they liked my Alphabet Song in the last Tips, and somebody else wanted some more routines for the speech synthesizer, so I put it all together and here's what I came up with. If you can type the alphabet without a mistake, you get an encore.

```
100 CALL CLEAR
110 PRINT "       ALPHABET S
ONG"
120 FOR J=1 TO 20
130 PRINT
140 NEXT J
150 PRINT "              by Ji
m Peterson": :"Wait, please"
;
160 OPEN #1:"SPEECH",OUTPUT
170 DIM T$(26,2)
180 DATA 12,12,4,4,1,1,4,7,7
,8,8,10,10,10,10,12,4,4,7,8,
6,10,4,8,8,10
190 FOR J=1 TO 26
200 READ X
210 T$(J,1)="//"&STR$(X)&" "
&STR$(X/10*32)
220 T$(J,2)=CHR$(J+64)
230 NEXT J
240 T$(23,2)="DOUBLE"&"!"&"!
"&"U"
250 CALL CLEAR
260 PRINT "READY - TYPE THE
```

```
ALPHABET"
270 T=0
280 K2=64
290 CALL KEY(3,K,ST)
300 IF (ST<1)+(K<65)+(K>90)T
HEN 290
310 IF K<>K2+1 THEN 330
320 T=T+1
330 PRINT #1:T$(K-64,1):T$(K
-64,2)
340 CALL HCHAR(12,17,K)
350 K2=K
360 IF K<>90 THEN 290
370 IF T=26 THEN 390
380 GOTO 270
390 FOR K=65 TO 90
400 CALL HCHAR(12,17,K)
410 PRINT #1:T$(K-64,1):T$(K
-64,2)
420 NEXT K
430 PRINT #1:T$(1,1):"NOW IV
E":T$(3,1):"SAID MY":T$(5,1)
:"A B":T$(3,1):"SEEZ"
440 PRINT #1:T$(8,1):"WONT Y
OU":T$(10,1):"COME AND":T$(1
2,1):"PLAY WITH":T$(1,1):"ME
"
450 GOTO 270
```

Terry Atkinson's routine to redefine the cursor has aroused some interest, so I fiddled around and came up with this version to change the cursor automatically to whatever character, normal or redefined, that you input.

```
100 !CURSOR CHANGER by Jim P
eterson
110 INPUT A$ :: A=ASC(A$)::
CALL CHARPAT(A,A$):: FOR J=1
 TO 16 STEP 2 :: H$=SEG$(A$,
J,2):: CALL HEX_DEC(H$,D)::
T=T+1 :: H(T)=D :: NEXT J ::
120 CALL INIT :: CALL LOAD(8
196,63,248)
130 CALL LOAD(16376,67,85,82
,83,79,82,48,8)
140 CALL LOAD(12288,H(1),H(2
),H(3),H(4),H(5),H(6),H(7),H
(8))
150 CALL LOAD(12296,2,0,3,24
0,2,1,48,0,2,2,0,8,4,32,32,3
6,4,91)
  CALL LINK("CURSOR")!THAN
KS TO TERRY ATKINSON
170 SUB HEX_DEC(H$,D):: N=1
:: DEC=0
180 FOR J=1 TO LEN(H$):: A$=
SEG$(H$,LEN(H$)-J+1,1):: IF
ASC(A$)>58 THEN HT=ASC(A$)-5
5 ELSE HT=VAL(A$)
190 DEC=DEC+N*HT :: N=N*16 :
: NEXT J
200 IF DEC<>32768 THEN D=DEC
 ELSE D=-(65536-DEC)
210 SUBEND
```

And of course you can always color the cursor with CALL COLOR(0,5,11) or whatever colors you like.

Most folks don't seem to know, and some folks refuse to believe, that the Memory Expansion can't store strings. If you are one of the disbelievers, plug in your Memory Expansion and try this -

```
100 FOR J=1 TO 255 :: M$=M$&
CHR$(J):: NEXT J
110 DIM A$(100):: X=X+1 :: A
$(X)=M$ :: PRINT X :: GOTO 1
10
```

Now RUN that. On my console, I get MEMORY FULL when X=43 although the SIZE command shows I have 24399 bytes of program space free (in the Expansion) - but only 204 bytes of free stack (in the console). Without the Memory Expansion I can get X up to 51, and in Basic to 53.

This can be a serious handicap if you are running a program which reads in a large number of strings from DATA statements, or generates strings while running.

Of course, when the Memory Expansion is attached, the program and the numeric variables are stored in the Expansion, leaving all the console memory available for strings - but if you do not generate strings, the console memory remains unused, because numeric data cannot overflow into it!

If your program generates more numeric variables than the Memory Expansion can hold, you can however store them in the console by converting them to strings, using STR$, and convert them back to numbers with VAL. This will allow you store an additional 700 to 900 or more numbers. Try this -

```
100 DIM A(3040),A$(1000):: F
OR X=1 TO 3000 :: A(X)=99 ::
PRINT X :: NEXT X
110 Y=Y+1 :: A$(Y)=STR$(99)
:: PRINT Y :: GOTO 110
```

When you get MEMORY FULL, type SIZE.

Dave Renkenberger sent me a neat little routine, and I played around with it a bit. For you who are not football fans, I'd better explain that the Wave is performed at football stadiums when the cheerleaders get the fans to stand and cheer, one seating section at a time, across the stadium - and those drunks on the roof are usually out of sequence.

```
90 !THE WAVE by David Renken
berger/modified by Jim Peter
son
100 CALL CLEAR :: CALL SCREE
N(4)
110 A$="**the wave**"
120 DISPLAY AT(4,14-LEN(A$)/
2):A$
130 B$="press any key to sto
p"
140 DISPLAY AT(22,14-LEN(B$)
/2):B$
150 E$="995A3C3C3C3C2466"
160 A$="000018187EBD3C3C"
170 FOR CH=91 TO 118 :: CALL
 CHAR(CH,A$):: M$=M$&CHR$(CH
):: NEXT CH :: FOR R=8 TO 12
 :: DISPLAY AT(R,1):M$ :: NE
XT R
175 FOR T=1 TO 26 STEP 5 ::
DISPLAY AT(22,T):SEG$(M$,T,1
):: NEXT T
180 FOR CH=91 TO 123 :: CALL
 CHAR(CH,E$):: CALL CHAR(CH-
5,A$):: CALL SOUND(-999,-7,5
*RND):: CALL KEY(3,K,ST):: I
F ST<>0 THEN STOP
190 NEXT CH :: GOTO 180
```

MEMORY FULL

Happy hackin'

Jim Peterson

TO ALL MEMBERS:

The Official Ballot for the Election of Bayou 99 Users' Group Officers is printed below. Please mark your choice of the Nominating Committee's Candidates or write-in names of another candidate of your choice. The reverse side of this ballot has been prepared with the address of the B99UG and has a postage stamp affixed for your convenience. Remove this entire page and fold the ballot as indicated by the dashed lines with the ballot inside and the address out, then secure with scotch tape or a staple.

Please mark XXXX and mail your ballot immediately. We expect 100% return of these ballots. The final date for mailing is January 7th to assure that your vote will be counted.

BAYOU 99 USERS GROUP

>>

CUT CUT CUT

OFFICIAL BALLOT

The Nominating Committee presents the following slate of candidates for Officers of the Bayou 99 Users' Group for 1985. Members are requested to indicate their choice of one candidate for each office. Choices should be indicated by an X to the left of the Nominating Committee's candidate or a write-in candidate of your choice.

PRESIDENT

☐ Steve Manuel

☐ ______________(Write-in)

VICE-PRESIDENT

☐ Mark Wilson

☐ ______________(Write-in)

FOLD

TREASURER

☐ Robert Nordan

☐ ______________(Write-in)

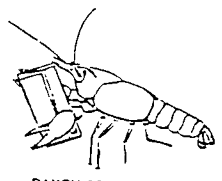
SECRETARY

☐ Bruce Wyman

☐ ______________(Write-in)

PLEASE FOLD, BALLOT IN - ADDRESS OUT - AND SECURE WITH TAPE OR STAPLE, THEN MAIL.

BAYOU 99 USERS* GROUP
POST OFFICE BOX 921
LAKE CHARLES, LA 70602



"NOTICE"

BAYOU 99 USERS GROUP
P.O. BOX 921
LAKE CHARLES, LA. 70602

MEETING 2nd. THURSDAY EACH MONTH AT 7:00 P.M. A. A. NELSON ELEMENTARY, 1001 COUNTRY CLUB ROAD, WEST OF LAKE STREET ABOUT 2 BLOCKS ON SOUTH SIDE OF COUNTRY CLUB ROAD.

Learn More About Your Texas Instrument Computer

Join A Users Group Now

1985 MEETING DATES

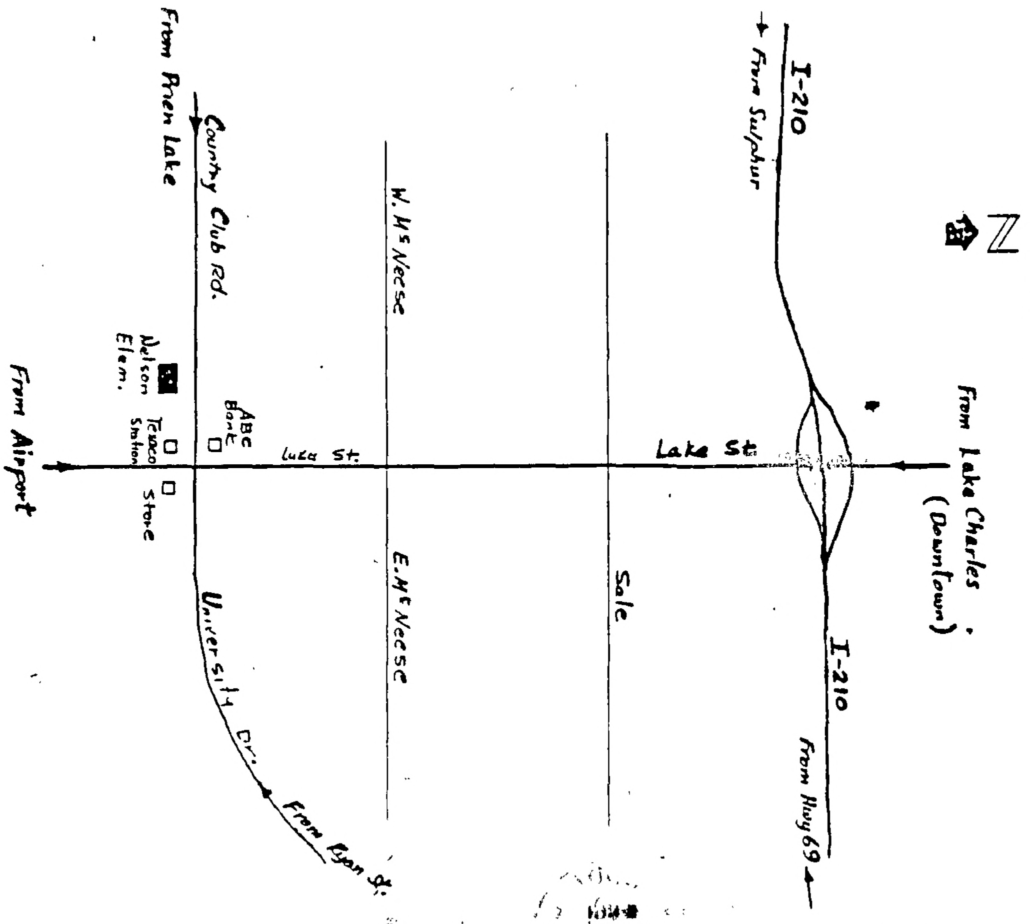
JAN 10	FEB 14	MAR 14	APRIL 11
MAY 9	JUNE 13	JULY 11	AUG 8
SEPT 12	OCT 10	NOV 14	DEC 12

BAYOU 99 USERS GROUP

MEETING LOCATION

A. A. NELSON ELEMENTARY

1001 COUNTRY CLUB ROAD



BAYOU 99 USERS' GROUP
POST OFFICE BOX 921
LAKE CHARLES, LA 70602

LAKE CHARLES, LA DEC 31 PM 1984 70601

EDMONTON USER'S GROUP
P.O. BOX 11983
EDMONTON-ALBERTA, CANADA T5J-3L1