

094
8706

50¢



VAST
USERS
GROUP
*



VALLEY OF THE SUN
TI 99 USERS GROUP

newsletter

VOL. 3

JUNE 13, 1987

NO. 6

CONTENTS

VAST 99 INFO.....	2
EDITORS PAGE.....	3
LIB. HIGHLIGHTS.....	5
WHEREFORTHS.....	6
COMPUTER TUTOR.....	8
HINTS AND TIPS.....	9

VAST 99 BBS 437-4335

-----JUNE, 1987-----

VAST 99 INFORMATION

The VAST 99 USERS' GROUP is a support group for TI 99 Home Computer users. Our regular meetings are on the second Saturday of the month. This month's meeting is being held in the Arizona Title Building at 111 W. Monroe in downtown Phoenix. It will be in a meeting room on the 10th floor. The meetings start at 10:00 AM and continue until 11:00 AM with socializing starting at 9:00 AM. The yearly membership fee is \$6.00.

All meetings are open and anyone may attend. Only dues paying members may vote in elections and obtain programs from the Users' Group library.

The current officers are:
 President

Vice-President
 Stu Olson.....846-7624

Secretary
 Bob Nixon.....838-4088

Treasurer
 Ike Van Kampen.....934-5164

User Group Librarian
 Earl Bonneau.....269-3802

Newsletter Editor/BBS SysOp
 Jim Ely.....437-1796

A FORTH Tutorial is being conducted by Rene' LeBlanc in this newsletter. It consists of a continuing series of articles relating to his version of FORTH which is available from the User Group Library. For more information, please contact him at (602) 991-1403.

The Users' Group's BBS is now in operation 24 hours a day. Contact it at (602) 437-4335. There is a lot of interesting conversation and information available here so give it a try.

Deadline for submission of articles or advertising for the Newsletter is the last Saturday of every month. Articles may be submitted in any form, however, the preferred method is by phone transfer directly to the Editor.

Advertising rates are as follows:

Commercial:

Full Page \$10.00
 Half Page \$ 7.00
 Quarter Page \$4.00

Personal:

Four lines,
 30 Characters/line
 \$1.00
 \$.20 per line
 over four.

All rates are for **ONE** issue only!

Programs are available from the USERS' GROUP LIBRARY at the following rates:

SS/SD Disk \$2.00
 DS/SD Disk \$4.00

If copying of documentation is required, it will be at the rate of \$.10 per page. If the User Group supplies the disk, please add \$1.00 to the above charges. An exchange program for free programs is also in effect. Please contact the librarian for further information. A complete list of what is in the library is available on 2 disks free of charge if you supply the disks or for \$1.00 per disk if the User Group supplies the disks.

 * VALley of the Sun TI99 Users Group *

-----JUNE, 1987-----

From the Editors Desk

MINUTES

For

MAY 9, 1987

The May meeting of VAST 99 was held on Saturday May 9, 1987, at the Los Olivos Hotel on East McDowell in Phoenix. Stu Olson conducted the meeting as Mike Marfisi was unable to accept the presidency of the group.

The meeting began with the much acclaimed Swap Meet with many good deals available for TI users, and even some items for other "Foreign" computers. The secretary picked up a TI LOGO for just \$5.00 - what a deal. Many other great deals were available as well, i.e., 1/2 height drive for \$40.00.

Stu called a short business meeting to order at 10:20 a.m. A roll call of members currently owing dues, or coming due next month was read. Stu also announced that we would be required to find another meeting place starting next month as Los Olivos was beginning a remodeling project for the hotel (greatly needed). If anyone has recommendations, they were encouraged to get in touch with a board member.

A plan was discussed to notify members of the next meeting place. Stu indicated that when a place was discovered he would notify Gerry Kennedy who would then call all the members. Gerry felt that plan was somewhat one-sided and suggested that each member call a board member just prior to the meeting to discover the location. That suggestion was approved.

A proposal from the floor to appoint a committee to locate a meeting place died without a second.

The next discussion centered around the need to find a new president for the group. Volunteers were solicited with negative results. The matter was tabled until next month. Stu was "coerced" to take the position but felt his overtime work schedule would not allow for effective leadership.

The business portion of the meeting adjourned at 10:35 a.m., with all happily returning to the Swap Meet.

Bob Nixon
Secretary

*****)+(*****

IN THIS MONTH'S ISSUE...

we have part 5 of our Assembly Tutorial by Steve Royce. This month he starts a discussion on writing files to disk. It starts on page 9. Computer Tutor this month is a "guest" column from the "CALL SAY" newsletter of the Grand Rapids Area 99er Computer User's Group. It is on using the "PRINT USING" command with your printer and is on page 8. Page 6 continues Rene's article on a Disk Copy Utility written in FORTH. And page 5 describes the Forth language disks that are available from the user group library. That's this issue....

VIEWS...

The main topic of discussion this month and probably the next several months (summer meetings seem to run a little tight with people on vacation etc....) will

-----JUNE, 1987-----

Editors Desk Continues. . .

be the direction the Group needs to take to keep the members we have and try to increase membership. The "movers and shakers" in the group are moving on to "other" computers as the TI is no longer filling their needs. They are suggesting that the group expand to include other computers besides the TI. They still want to be a part of the group, however the discussions of the TI don't necessarily interest (or pertain) to them any more. Since we don't right now have a very strong leadership, the question is WHAT TO DO? We need your input! We want to know what you think! If you don't speak up, you lose. What direction do you think the group should take? If we should change to a "generic" computer group, what should we cover and how? Do we have separate meetings for the different computers covered? Or do we have one general meeting and then break off into smaller groups and discuss the particulars of that specific machine? Your INPUT is vital! If we don't hear from you, we can't best serve your needs. So, you better speak up. If you get this newsletter and can't attend a meeting this summer, call one of the board members listed on the INFO page of the newsletter. Or, this is a hot topic on the BBS. Give it a call and add your \$.02 worth there. But, we need to hear from you one way or another. That is enough from the Soap Box. On to other things.

VAST 99 BBS...

I just can't seem to leave well enough alone. I have made a few more changes to the software and added a new function from the Main Menu. If you select [S]tatus from the Main Menu, you

will see your current status on the BBS, which includes your level, times on the board, number of Uploads and downloads and the last time you called. It also tells you if you have selected eXpert mode or not. It is basically the same screen you see shortly after you log on to the system. I am also going to add a line that tells you whether or not the SysOp is available for Chat. Something else I am working on is a reply function in the message bases. Keep an eye out for that.

By the way, this month marks the first year of operation of the group's BBS (June 22). And this weekend will mark the 5000th caller to the board. Quite an accomplishment, huh?

THINGS I WISH.....

I wish I had a REVIEW of CSGD III from someone in our group...

I wish I had a REVIEW of PRINTERS APPRENTICE by someone in our group...

I wish someone would tell me how to use TI-Writer.....

I wish more people would take an ACTIVE part in this group....

I wish Gerry would tell Stu he is spelling his last name wrong (OLSEN not OLSON).....

I wish Tom Moran would come to all the meetings he says he is going to come to.....

I think you get the idea.....

'til next month

Jim Ely, Newsletter Editor,
BBS SysOp

-----JUNE, 1987-----

LIBRARY HIGHLIGHTS

THE FORTH LIBRARY DISKS

We'll start with TI FORTH, the software version that TI put into the public domain. When this first appeared locally, the price was twenty dollars for the program disk and the manual. After the initial offering was made, and demand diminished, the resulting stock was sold and the manuals were never reprinted. Consequently, the cost of copying the manual took expense, effort and time, if one could find a manual to copy. Some dedicated person took the time and effort to type the manual, and it is now available on four single sided, single density (sssd) disks.

People then began to enhance or modify the original TI program, resulting in an Extended Basic version. This version is a two sssd disk set labeled FORTHXLD1 & FORTHXLD2.

As with any other language, someone always has trouble working with a new language. George L. Smythe was one of these people. He wrote a series of FORTH tutorials to help people overcome the problems that he encountered. The tutorials also contain some short programs to be typed in. His tutorials are on four sssd disks labeled 4TH-DOCS1 thru 4TH-DOCS4. The library has another disk containing miscellaneous information on FORTH in display variable 80 format (DV80). These tutorials can be printed thru TI-WRITER, BA-WRITER, etc. The TI-WRITER formatter should not be used as some Forth commands double as printer commands when using the formatter.

Our resident Forth expert, Rene' LeBlanc, has an ongoing tutorial in this newsletter which uses the Extended Basic version of Forth.

Once you have mastered the loading

and running of FORTH, there are four sssd disks that are screens that can only be viewed thru the FORTH program. These screens contain utility programs, games, etc.

For those people who want to collect everything associated with Forth, also available is the Forth source code in editor assembler.

FORTHUTIL1 is loaded using Extended Basic. This disk will boot up to a Forth copy utility program. If this is not what you want, press clear <FCIN 4> and access to the two other utility programs is available: proprietary disk unprotect and disk initializing. FORTHUTIL2 is the Forth source code for the programs.

Finally, the last set of disks available is JPGRAPHICS ver 3.1, a program designed to draw color graphics in the TI-FORTH environment without having prior knowledge of TI-FORTH. It requires only single key strokes and LOGO commands. I had problems with this disk as it was double sided, single density and it wants to boot from DSK1, and my DSK1 drive is only sssd. Finally copied the program using Forth from DSK2 to DSK1 and deleted the 180 disk_hi reference. To run, first load Forth. From Forth type in "3 LOAD" without the quotation marks. The demo program is quite good in performing various colored graphics and musical songs. Available as 3 sssd disks or 1 dssd disk.

Earl Bonneau,
Group Librarian

**YOUR PERSONAL
AD
COULD GO HERE!**

-----JUNE, 1987-----

WHEREFORTH'S OF FORTH

As your faithful WHEREFORTH'S author (with feet of clay) was preparing for another article on the COPY-DISK program, he noticed several anomalies which showed up when copying disks with sizes other than the one he first tested the program with. Also, he noticed that AFTER testing the first version, he made a couple "minor improvements" in the listing in WHEREFORTH'S #14 and didn't actually retest the program.

With the rush of hundreds of my faithful followers immediately running home from the VAST meetings to enter their new Forth programs found in the WHEREFORTH'S article and try them out, I am AMAZED that all of you apparently were able to get your programs working just fine in spite of the bugs I put

in them. You have all been so polite to have suffered through these errors and finding how to correct them on your own without even wanting to bother me with a phone call to help you straighten out the puzzle I created. You really needn't be so shy. I don't mind hundreds of irate phone calls telling me I screwed up. It would do me some good.

Anyway, I have corrected the problems and also made a minor improvement or two. These changes occur in screen #s 1, 5, 6, 7, 8, and 9. I thought it would be simpler if I just relisted the entire program in this issue for you, then next month we can start evaluating again with a program that is actually correct to begin with.

```

\ Disk Copy Program SCR#1
TEXT
: INSTRG ( -- )
  PAD 1+ 40 EXPECT PAD 40 1 DO DUP I + C@ 0=
  IF 1 1- SWAP C! BL PAD COUNT + C! LEAVE THEN LOOP ;

: (C,R)OK? ( ? -- f )
  DEPTH 2 < IF 0 ELSE 2DUP 24 < SWAP 39 < AND THEN ;

: ACCEPT_AT ( col row -- ) (C,R)OK? IF GOTOXY THEN INSTRG ;

: DISPLAY_AT ( $addr col row -- ) GOTOXY COUNT TYPE ;

-->

\ Disk Copy program SCR#2
: !$ELEM ASCII", WORD HERE C@ 1+ ALLOT ;

: <ER1> ." Index too large" ;

: $ARRAY ( n -- )
  <BUILDS DUP C, 0 DO !$ELEM LOOP DOES>
  DUP C@ ROT DUP >R >
  IF 1+ R> -DUP IF 0 DO COUNT + LOOP THEN
  ELSE R> DROP DROP OR [ ' <ER1> 2+ ] LITERAL THEN ;

7 $ARRAY $msg Reading Block: ,Writing Block: ,Please insert SO
URCE disk and,Please insert TARGET disk and,hit any key when rea
dy,Disk Copy Complete!,PLEASE ENTER # SECTORS ON DISK: ,
-->

\ Disk Copy program SCR#3
: .RR 0 $msg 2 10 DISPLAY_AT ; ; .Rblk 17 10 GOTOXY . ;
: .WB 1 $msg 2 12 DISPLAY_AT ; ; .Wblk 17 12 GOTOXY . ;
: ANYKEY 4 $msg 2 8 DISPLAY_AT KEY DROP ;
: .SD 2 $msg 2 7 DISPLAY_AT ANYKEY ;
: .TD 3 $msg 2 7 DISPLAY_AT ANYKEY ;
: .COMPL 5 $msg 2 20 DISPLAY_AT ;
: .ASK 6 $msg 2 23 DISPLAY_AT ;

-->

```

JUNE, 1987

WHEREFORHS CONTINUES

```
\ Disk Copy Program SCR#4
BASE->R DECIMAL
0 VARIABLE #B \ Number of Blocks
0 VARIABLE BP \ Block Pointer
S0 @ PAD 40 + - B/BUF / CONSTANT #HB \ # Hi Bufs
LIMIT FIRST - B/BUF / CONSTANT #LB \ # Low Bufs
HEX 8370 @ 1400 - B/BUF / CONSTANT #VB \ # VDP Bufs
1400 CONSTANT VDPBUF \ Address of first VDP Buffer
R->BASE
-->
```

```
\ Disk Copy Program SCR#5
: stoi ( addr -- n ) 0. ROT DUP >R (NUMBER) R COUNT + = 0=
: IF R COUNT CR TYPE ." NOT A NUMBER" THEN R> DROP DROP ;
: GETN ( -- n ) ACCEPT AT PAD stoi ;
: !#B ( #sectors -- ) 4 / #B ! ;
: #B@ ( -- ) #B @ ;
: B@ ( -- n ) BP @ ;
: B! ( n -- ) BP ! ;
: B+ ( -- ) 1 BP + ! ;
: READIT ( addr blk# -- ) DUP .Rblk 1 R/W B+ ;
: WRITEIT ( addr blk# -- ) DUP .Wblk 0 R/W B+ ;
: ASK_SIZE .ASK GETN !#B ;
-->
```

```
\ Disk Copy Program SCR#6
: GET #B 0 BLOCK 10 + @ DUP
CASE 360 OF !#B ENDOF
720 OF !#B ENDOF
1280 OF !#B ENDOF
1440 OF !#B ENDOF
DROP ASK SIZE ( NON-STANDARD DISK HEADER )
ENDCASE 2 5 GOTOXY ." Your disk has " #B@ . ." blocks" ;
: LO(I) ( I -- addr ) B/BUF * FIRST + ;
: HI(I) ( I -- addr ) B/BUF * PAD 40 + + ;
: VB(I) ( I -- vaddr ) B/BUF * VDPBUF + ;
: 2DROP DROP DROP ;
-->
```

```
\ Disk Copy Program SCR#7
: GET-LO-BUFS ( -- ) #LB 1 ( Skip FIRST )
DO I LO(I) B@ #B@ OVER = IF 2DROP LEAVE ELSE READIT THEN
LOOP ;

: GET-HI-BUFS ( -- ) #HB 0
DO I HI(I) B@ #B@ OVER = IF 2DROP LEAVE ELSE READIT THEN
LOOP ;

: GET-VDP-BUFS ( -- ) #VB 0
DO B@ #B@ = IF LEAVE ELSE FIRST B@ READIT
FIRST I VB(I) B/BUF VMBW THEN
LOOP ;
-->
```

```
\ Disk Copy Program SCR#8
: PUT-LO-BUFS ( -- ) #LB 1 ( Skip FIRST )
DO I LO(I) B@ #B@ OVER = IF 2DROP LEAVE ELSE WRITEIT THEN
LOOP ;

: PUT-HI-BUFS ( -- ) #HB 0
DO I HI(I) B@ #B@ OVER = IF 3DROP LEAVE ELSE WRITEIT THEN
LOOP ;

: PUT-VDP-BUFS ( -- ) #VB 0
DO B@ #B@ = IF LEAVE ELSE I VB(I) FIRST B/BUF VMBR
FIRST B@ WRITEIT THEN
LOOP ;
-->
```

----- JUNE, 1987 -----

C

This month's Computer Tutor article is "borrowed" from the April, 1987, issue of the Grand Rapids Area 99er Computer Users' Group Newsletter, "CALL SAY." No author was given but credit will be given if the author is brought to our attention.

O

M

USING "PRINT USING" WITH YOUR PRINTER

P

One of the more obscure statements available with TI Extended Basic is one called "PRINT USING." Even more obscure is the fact that this statement can be used to format variables and constants that will be dumped to your printer. The Extended Basic manual, on page 150, shows several examples of how PRINT USING can be used to format data for screen display, but nary a word on how to do the same with open files. It can be done, and is much more powerful than you may realize.

U

t

E

P

Any discussion of PRINT USING will require an understanding of the IMAGE statement, so if you are not familiar with it, you better brush up on it first. The PRINT USING statement uses IMAGE in one of two ways: either with a string expression or a line number reference. I prefer the latter, as it allows more flexibility, but since these different methods are explained in the manual, I will limit this to a few simple examples that are not shown in the manual.

T

U

t

O

```
100 TCOST=19.55
110 IMAGE "##.##"
120 OPEN #1:"PIO"
130 PRINT #1,USING 110:TCOST
140 CLOSE #1
```

P

Running this simple program will effectively show how the PRINT USING statement will work with an open file. Of course,

A

there are many other variations of IMAGE that can be used, so experiment with them and watch how it performs when line 130 dumps it to the printer. Shown below are a few more examples for use with an open file. Substitute these lines for the same line numbers in the above example.

U

t

b

```
110 IMAGE "##.##"    ##.##"
130 PRINT #1,USING 110:TCOST1
    ,TCOST2
```

This IMAGE statement will allow you to print two (or more) variables at a predetermined spot on the same line. The length of the string expression in the IMAGE statement can be as long as you wish, up to the limit of the Extended Basic line.

O

P

```
110 IMAGE "#####"    ##"
130 PRINT #1,USING 110:"TOTAL
    COST",TCOST
```

U

This version shows how you can format the printed line for string data as well as numerical data. A string variable could be used in place of the string constant, as below.

D

```
105 A$="TOTAL COST"
110 IMAGE "#####"    ##"
130 PRINT #1,USING 110:A$,TC
    OST
```

k

It is also possible to place the IMAGE statement inside the PRINT USING statement as shown below. First, delete line 110.

D

```
130 PRINT #1,USING "##.##":T
    COST
    or
130 PRINT #1,USING "#####
    ##"    ##.##":A$,TCOST
```

O

W

A few other points to remember include the fact the IMAGE and PRINT USING can be used to

D

JUNE, 1987

COMPUTER TUTOR CONTINUES.....

round off calculated variables. A single string expression, such as "#####.##" will round off and decimal align numbers as small as .01 up to 999999.99 and print the number at any designated location. This function could save many hours of algorithm development for accomplishing the same thing.

So, in the long run, the PRINT USING statement is one that any programmer should be very familiar with, and use as much as possible.

*
* V A S T 9 9
*

<<<<<<<>>>>>>
< T H E
< V A S T 9 9
< B B S
< 2 4 H O U R S
< A D A Y
< 7 D A Y S
< A W E E K
< F O R
< N E W S
< I N F O R M A T I O N
< A N D
< F U N !
< 6 0 2 - 4 3 7 - 4 3 3 5
<<<<<<<>>>>>>

WHEREFORTH'S OF FORTH CONTINUES...

```
\ Disk Copy Program SCR#9
: COPY-DISK TEXT SP! DR0 .SD GET_#B 0 B!
  BEGIN B@ #B@ <
  WHILE CR .RB B@ >R
    GET-LO-BUFS GET-HI-BUFS GET-VDP-BUFS .TD
    CR .WB R> B!
    PUT-LO-BUFS PUT-HI-BUFS PUT-VDP-BUFS .SD
  REPEAT
  CR .COMPL ;

SO @ PAD 40 + - B/BUF / ' #HB ! \ Set #HB to final value
```

Rene' LeBlanc

HUNTS AND TIPS

TMS 9900 ASSEMBLY LANGUAGE TUTORIAL
PART V
DSR's_and_PAB's

by STEVE ROYCE - WNY 99'ERS

File handling is not quite as easy in Assembly as it is in BASIC or EX-BASIC, but it really isn't too complicated. The main issues involved are setting up a Peripheral Access

----- JUNE, 1987 -----

HUNTS AND TIPS

CONTINUES...

Block (PAB) to establish the attributes of the file, writing the PAB to an address in VDP, manipulating the bytes in the PAB to accomplish specific functions, reading from and writing to a buffer in VDP and invoking the built-in Device Service Routine Link (DSRLNK) utility. These functions, in and of themselves, are a lot to cover and once we start trying to establish some order in the handling of files, it becomes even more lengthy. So, we are going to cover file handling in at least three articles, each one building on the previous articles until we finally arrive at an (in my opinion), efficient set of our own utilities which can be used to make file handling a lot easier.

This month and next, we will deal with data files to be stored on disk. Since I am a firm believer in the value of RELATIVE data files, even though they may take up more space on a disk, I will only deal with RELATIVE disk files in these articles. Once you understand RELATIVE files, SEQUENTIAL files are even easier. It's a lot like learning to drive a car with a standard transmission, then the automatic transmission is easy.

In your BASIC and Ex-BASIC manuals, you will read that INTERNAL data format is more easily interpreted by your computer. My first attempt at creating a disk file in Assembly was therefore to use INTERNAL format. I tried for weeks to create a FIXED, RELATIVE, INTERNAL file, but, every time I attempted to simply open then close a file with these attributes, then switched to DISK MANAGER to see if it existed, it WASN'T THERE!!

I modified my PAB, changed buffer addresses, put in the Assembly equivalent of a "CALL KEY" to check my work at every step, used the Assembly

Super-Debugger--you name it, I tried it. It took quite a while for me to finally realize that, in Assembly, if you want INTERNAL format, you must do the conversion from DISPLAY to INTERNAL yourself. Sorry, but I'm not about to try that yet, so we will stick to DISPLAY format.

PAB BYTES

Your E/A manual, pages 293 to 299, does a reasonable job in explaining the function of each byte in a PAB, so I'm not going to cover that ground in this article. Review those pages and note that, since our file will be a DISPLAY, FIXED, RELATIVE file that our choices for byte 1 of the PAB are >01, >03 or >05.

STRUCTURING

BASIC, Ex-BASIC and many other languages are designed around a well structured set of standard routines and addresses which are dedicated to specific purposes. You would never know it from your BASIC manual, but the internal addresses for every function are defined and, for the most part, inflexible. Assembly defines some specific addresses such as the screen image table, some CPU addresses, the memory mapped devices, but leaves you to your own creativity for the balance of the program organization. What I hope to end up with in my Assembly programming is a well organized yet flexible set of routines which I may incorporate into any program that needs them. In this way, once my subroutines and addresses are established, a lot of custom structuring for a specific program becomes unnecessary. I think it's a wonderful idea.

Let's structure an area of VDP RAM for PAB's and a read/write buffer area. Remember last month we started our sound table at >1000. I have reserved a block of >500 bytes for my

JUNE, 1987

HINTS AND TIPS CONTINUED...

sound table, based on the fact that the song and sounds in TOMBSTONE CITY take >0300 bytes, and that I may want a few more sounds in my programs. So, my VDP area for PAB's will start at >1500, and I have reserved >0100 bytes for PAB's. That should be enough to set up eight PAB's, which should be enough for any program I can imagine. The next block of VDP, starting at >1600, will be used as a read write buffer area for the PAB. Since the maximum record length in a file is >FF bytes, I had initially set aside a block of >100 bytes. However, I have expanded that to >200 bytes. I'm not sure why yet, but I'll make use of that extra >100 bytes somehow. So, our read/write buffer area is from >1600 to >17FF.

Remember, all these areas are still flexible in use. If I don't use all >500 bytes of my sound table, I can allocate them for other use. But, I at least have the ability to use a structured and organized set of pre-defined areas of VDP.

Our structured VDP table so far looks like this:

- >0000 to >02FF SCREEN IMAGE TABLE
- >0300 to >037F SPRITE ATTRIBUTE LIST
- >0380 to >039F COLOR TABLE
- >03A0 to >03FF UNUSED
- >0400 to >07FF SPRITE PATTERN TABLE DEFAULT
- >0780 to >07FF SPRITE MOTION TABLE
- >0800 to >0F7F PATTERN DESCRIPTOR TABLE
- >0F80 to >0FFF UNDEFINABLE CHARACTERS >F0 TO >FF
- >1000 to >14FF SOUND TABLE
- >1500 to >15FF PAB DEFINITION AREA
- >1600 to >17FF READ/WRITE BUFFER AREA FOR PAB's

Before we get into the routine to open and close a file, let me offer the following suggestion. Edit and assemble the code, then make a copy of the object code using DISK MANAGER. Use that disk for the LOAD AND RUN. If anything should go wrong, you won't have destroyed the source code. After you LOAD AND RUN the object code, use DISK MANAGER again to see whether the file 'DSK1.FILE1' has been successfully created. Next month, we'll write a record to the file.

NEWSLETTER
VAST 99 USERS' GROUP
c/o 1425 E. Del Rio Dr.
Tempe, AZ 85282



FIRST CLASS MAIL

TO:

EDMONTON 99ERS COMPUTER SOC.
PO BOX 11983
EDMONTON, ALBERTA
CANADA T6C 3L1

FIRST CLASS MAIL

June, 1987