THE NATIONAL NINETY-NINER

VOL II - NO. 10 - NOVEMBER-DECEMBER, 1985

THE 99ER'S ASSOCIATION
3535 SO.H ST., #26
BAKERSFIELD, CALIF. 93304
(805) 397-4361
DON VEITH - EDITOR

$1.50

CREATED FOR TI 99/4A HOME COMPUTER OWNERS

TABLE OF CONTENTS

THE NATIONAL NINETY-NINER

VOL II - NO. 10 - NOVEMBER-DECEMBER, 1985

ANNOUNCEMENTS

A LETTER ON "FREEWARE"

The letter below was received by our organization on October 1, 1985. It has some interesting news about the use of a term "FREEWARE" by TI software authors.

Dear Don,

I was somewhat concerned to see the term FREEWARE used so loosely in the article by Danny Michael. FREEWARE is a registered trademark of Headlands Press, Tiburon, California. I understand Headlands Press has not been pleased with others using their trademark but this is not first hand knowledge.

I am a proponent of shared software. I know that others have used the term SHAREWARE and I certainly did not coin it. Shareware does not contain "FREE" and if that was widely adopted in place of the trademark FREEWARE the feeling that such software was free of charge might begin to change and even disappear.

Danny Michael is quite right about shareware not being free. I know people who are very faithful in paying the author for their works. But I probably know more people who do not make a contribution even though they make substantial use of a particular program. What may be even worse, they distribute the software in its modified form without providing the original unaltered version.

One of the major reasons that the TI-99/4A has not died is the high quality of shareware which has been available for the last year. Since the commercial software firms, with a few exceptions, are generally ignoring the 99, the flow of shareware is critical. If people fail to support the shareware authors with contributions, feedback, and suggestions, then that source of software will cease and the 99 will truly be an orphan.

Feel free to publish this letter to the editor. I strongly urge you to acknowledge the Trademark Status of FREEWARE to avoid possible conflict with Headlands Press.

Sincerely,

John L. Pearce

WE NEED A NEW NAME FOR FREEWARE ???

By Don Veith, Editor

FREEWARE is the registered trademark of of Headlands Press of Tiburon, California. A registered trademark held by any firm or individual may not be used without their consent. Please refrain from using the term FREEWARE in any 99/4A related publication.

The IBM PC user community created the concept of a program author sharing software with other computer owners on a trial basis. The user is allowed to try the software by simply forwarding the author a disk , mailer, and sufficient postage for its return. If the software is useful to the individual, they are then requested to forward a donation to the software author. The idea was to obtain high quality software from individuals who like to program but are really not interested in selling their wares through a commercial firm. The term **SHAREWARE** was coined by the IBM community to identify the concept of software sharing.

The problem that both the IBM and TI-99/4A communities are experiencing is identical. Too many people, who think the software is **FREE**, are obtaining copies and **NOT FORWARDING** a donation to the author. Ceasing to use the registered trademark **FREEWARE** will alleviate some of the confusion that the programs are not really **FREE**.

A new term is needed to represent software created by non commercial programmers in the TI-99/4A community. **TRIWARE** is a suggested new term to denote software previously identified as FREEWARE. The first three (3) letters TRI possess a double meaning. An I was substituted for a Y in the word TRY. It was selected to specifically remind people that the software was available for them to try before sending funds to the author. You are simply requested to forward a minimal donation if the software meets your requirements **AND** will be used on a daily basis.

The second meaning of TRI stands for the three (3) phases of this unique software sharing idea. The three phases are:

(1) A programmer creates the software.

(2) Distribution is allowed to interested members of the computer community.

(3) Receipt of donations from regular users of the software encourages the creation of additional work.

Our primary dilemma is that too many persons are stopping at step two (2) while continuing to use the software on a daily basis. There have been no visible effects on the supply of software yet!!! Conversations with software authors tend to make me believe the authors will be very reluctant to release programs in the future if this trend is not reversed. Comments to the effect, "I mailed out 600+ copies and received payment for 118 copies."

I do not wish to continue on this distasteful subject any longer. The continued supply of software is in the hands of every TI-99/4A owner. We all must pay something for software of useful value or no more will appear. It really is that simple!

POPULAR COMPUTING ran an editorial on the piracy of commercial software in their October, 1985 issue. One paragraph covered software piracy better than any other I had read. The paragraph is typed below in its entirety.

"Software is clearly a commodity, a product created by an individual or group of people. It takes time and money to write and publish a program, and those who invest their time and money want to be compensated when the programs are used. People deny compensation to the software authors whenever an illegal copy is made. Most software piracy takes place in the thief's own home. Their opinion (the software pirate(s) or THIEF(s)) is that whatever they do in the privacy of their home is nobody's business, and ethics and the law be damned."

TRIWARE is not a registered tradeamark of THE NATIONAL NINETY-NINER or THE 99'ERS ASSOCIATION. The term may be freely used to represent software distributed by authors under the SHAREWARE, FREEWARE, FAIRWARE concepts. You ARE ENCOURAGED to notify other members of the TI-99/4A community and urge them to adopt this term. After all, the word does have TI within it to denote software for our orphan computer even though the letters are not adjacent!!!!

A NEW BBS 94 TELEPHONE NUMBER

By Don Veith, Editor

We are saddened to announce that the Shoals 99'ers Users Group decided to cease the operation of their Bulletin Board System. The BBS had been shared by this fine group of people with our organization. The group still exists and will continue to operate. We wish them the best of luck in all their new endeavors. Luci and I have gained some very fine friends whom we shall cherish and continue to remain in contact with in the future.

A replacement BBS to represent our organization in this same area of the country has been located. We have had the priviledge of Roger Hickerson's acquaintance for the past year. Please give his board a call and continue to download articles, announcements, and other items of interest for our publication. The telephone number and other pertinent data are outlined below:

BBS #	TELEPHONE #	OPERATOR	LOCATION
4	(318) 474-6141	ROGER HICKERSON	LAKE CHARLES, LA

ODDS_'N_ENDS

HOW TO KILL AN ORGANIZATION

1. Do not attend meetings; if you do, arrive late.

2. Be sure to leave before the meeting is over.

3. Never offer your opinion at a meeting; wait until you get outside.

4. When at meetings, vote to do everything, then go home and do nothing.

5. The next day, find fault with your officers and fellow members.

6. Take no part in your organization's affairs.

7. Sit in the back and start up your own meeting with one or more members during discussion periods; if you keep it low no one will notice.

8. Get all the organization can give and give nothing in return.

9. Talk cooperation but never cooperate.

10. Never ask anyone to join the organization.

11. Threaten to resign at every opportunity ; especially when things are not going your way.

12. If I asked to help, always promise to do so but be busy when called upon.

13. Never read anything pertaining to the organization in case you learn something on your own.

14. Never accept an office; better to criticize than be criticized.

15. If in a moment of weakness you find you have gotten yourself on a committee; apply all of the above rules and let the chairman do all of the work.

16. Do not do anything more than you have to, and when others give freely and willingly of their time and talents to help the cause, be the first to leap to your feet to remind everyone: WHAT'S WRONG WITH THIS GROUP IS THAT IT'S BEING RUN BY A CLIQUE!!!!!!!!

Article copied from the Portland Users of Ninty-Nines (PUNN) newsletter.

A REVIEW OF BASIC RULES

By Don Veith

The Home computer era has passed into memory for many people who purchased their computer to evaluate what the media hype over this device was all about. A few diehard nuts refuse to allow our "Orphaned Computer" to quietly fade away into the sunset like a Hollywood B movie cowboy hero. Two full years have passed since Texas Instruments dumped equipment onto the Christmas market place at fire sale prices. Some of the battles to obtain a particular cartridge at what we thought were bargain prices would qualify the survivor for combat pay. Paratroops have earned their nicknaame as "Those Baggy Pants Devils From The Sky". Even one of these well trained "Devils" would have paled at the combat present to "get my hands on an extra Black/Silver console, What I would give for a P-Code Card, They have how many Expansion Systems left?"

Yes, we all remember those days in the trenches two (2) years ago locating our own hoard of goodies while trying to aid the uneducated masses. It was not a good time to be President of a Users Group! The phone calls for assistance came at all times of the day. Questions, answered a thousand times before, were answered a thousand times once more. Remember, we were the experts who had all the answers for the masses. Oh yes, do any of you who were officers remember the scout teams we had out in the field to determine how much inventory each local dealer had left and where the best prices were. Yes, we maintained a sheet of inventory lists based upon quantity and price. Callers either simply WANTED the software/equipment, forget the price; or wanted the best price and did not worry about selection. How many of you thought prices for Logo, TI-Writer, and Multiplan at $74.95 were fantastic bargains. A newsletter I read this week had featured an advertisement for each piece of software for $19.95. Cast aspersions upon your inability to wait and truly pick up a bargain in computing power.

My purpose here was not to ramble on incessantly but to share a list of items from a "Computer Education" article in the October 2, 1984 issue of WOMAN'S DAY. Read each item on the two lists and actually check if you have done any of the listed items with your computer or applied the advice prior to purchasing a new piece of equipment. The results will prove intriguing. Boy did we all tell some tall whoppers when we justified that new $249.95 gizmo that runs twice as fast with less on line expense. Who the devil is kidding whom ??

12 THINGS A COMPUTER CAN DO FOR YOU

1. EDUCATE YOU - From drill and practice to tutorials and simulations, there are educational programs to enrich every member of a family.

2. ENTERTAIN YOU - There are thousands of arcade-style games and intellectual simulations available for all ages.

3. MAILING LIST AND BIRTHDAY REMINDER - Never miss another birthday or anniversary. Mailing lists are available in every form conceivable. Most also print labels or lists of their contents.

4. WRITE LETTERS-HOMEWORK - Any type of correspondence may be written in a quick and efficient manner. It may even be save for reuse at a later date.

5. BURGLAR PROOF YOUR HOME - Control lights and sensor units with the computer when you are both home or away.

6. BALANCE YOUR BUDGET - All your personal financial information may be stored in one place. Balances may be calculated for tax purposes as they are incurred.

7. MAKE SMARTER INVESTMENTS - Money management programs aid in investment selection and keep track of profits.

8. BANK FROM HOME - Plug into your bank's computer via your home telephone and a modem to handle all your banking needs.

9. COMPARISON SHOP - Shop from home and compare prices on your monitor.

10. PLAN MEALS - Plan your meals a week or two in advance. Convert recipe ingredients to fit the number of persons you are feeding.

11. RUN A SMALL BUSINESS - Home based businesses lend themselves well to computer related activities. Control your inventory, maintain customer records, and mail billings on accounts receivable.

12. MAKE YOUR OWN PROGRAMS - If you cannot locate that piece of software that exactly fits your needs, create an original program or modify another program to suit your specific demand.

8 QUESTIONS TO ASK THE COMPUTER SALESPERSON BEFORE YOU BUY

1. What equipment is included in the price of the computer?
Determine exactly what you will receive for the quoted price. Take notes on all items offered in the basic price by the salesman. If items are noted as discounted from list price, request the manufacturers SRP (Suggested Retail Price). Take the time to verify this information in a catalog, magazine, of another dealer. Many times the salesperson is working on a sales commission basis. The greater dollar volume progressed , the larger his level of renumeration on payday.

2. If the price does not include a disk drive, how much more will that cost me?
Once more, verify the costs involved by checking with other sources of disk drives.

3. What is the personal Computer's memory capability?
Computers on the market today should possess a minimum of 64K memory. Purchase prices vary between $300-$500.

4. If the cost of a monitor is not included, can I use the computer easily with my regular television screen?
Most computers can be connected to a television set. Check the price and quality of the monitor in case you desire to add it later.

5. How much do compatible printers cost?
Request a demonstration of a letter-quality printer (like typewritten letters) and a dot-matrix printer (computer letters formed from dots). You will pay a premium price for an extra wide 15" carriage on a printer.

6. What is the computer's word processing like?
Try the keyboard for size. How many letters or functions are entered through special key combinations. Does the monitor have an 80 or 40 character screen. You will have to make adjustments on a 40 column screen to do word processing.

7. What kinds of software are available for this computer?
The programs available for the computer are as important as the computer itself. If you are viewing a demonstration, ask about the software being ran on the computer. You may like the software better than the computer.

8. Does your store offer any help with installment or setting up of the computer.
Request to review the computer manuals that explain installation and use---and decide whether you can follow the directions or not. If things appear complicated, inquire whether setup assistance is available from the salesperson or another staff member.

Some very interesting suggestions are contained within the article. Many of us purchased our computers in discount department stores. We definitely had more knowledge about the 99/4A than many of the salespersons we encountered. A few exceptions did exist and we all gained additional information from a knowledgeable person. Good luck on your next foray into the market to purchase some new goodies for your system.

ARTICLES

The TI Forth Dimension
By Jeff Stanford

Welcome again. As I promised last time, this article will be devoted to a series of short graphics examples which everyone will enjoy and learn from. Years ago, when purchasing my first TI computer for three hundred dollars, I marveled at the excellent graphics available in the cartridge modules. I was also disappointed that it was not possible to match the quality or the speed of graphics in TI Basic. TI Extended Basic (EB) helped a little by increasing the speed and providing access to the TMS9918A Video Display Processor's (VDP) Sprites. It had routines to make the sprites move by themselves. With the EB module games could be created which had smooth moving graphics. I had to limit my work to programs in which the sprites moved relatively slow (as compared to a module's speed). Even though Extended Basic was faster than the console Basic, it was still too slow. Balls would pass through walls and paddles, and bullets through enemy planes because programs could not run fast enough to allow accurate coincidence checking.

Later I saved up my pennies and purchased the hardware to expand my 'system': disk, memory and peripheral box (or hook, line and sinker) and I also bought the Editor/Assembler (E/A) package. Then I thought 'Now I can write those fast programs with great graphics I always thought about, then I started reading the E/A manual. And I read and read and read and finally was able to write my first assembly language program to move a sprite in response to joystick inputs. The only thing I think that keep me sane during this period of learning was my prior experiences with assembly language (Motorola 6800). I feel pity for anyone how tries to learn assembly for the first time only from the E/A manual. The E/A manual is a store house of knowledge for an experienced programmer but, it is not a good learning guide. Well now I could write those fast games but writing in assembly takes time and planning. Working a full time job and studying part time for a masters degree did not leave much time for writing assembly code. Time passed and then I meet a friend who had gotten a copy of the fig-Forth model for the TMS9900 and had coded it in for his TI. I asked for and got a copy of his endeavors to help him finish it by typing in an editor. We got the system operational but the disk interface was cumbersome and we lacked necessary routines to make our system really useful.

Well, both my friend and I got tied up with work and had to put our project aside. Later, I heard about the release of TI-Forth by Tex-Soft and immediately purchased one of the first copies available locally. When I got it home and read the manual enough to learn how to bring up the system and how the editors worked, I then delved into the graphics section and after a few initial false starts wrote my first program in TI-Forth. It drew a square in bit-map graphics of any size and in any location on the screen I desired. Finally, a language that had the power and speed I desired. A high level language that was relatively easy to learn and utilize. I must admit that I haven't written a program in basic or assembly since I bought TI-Forth because I like it that much.

This article covers a collection of short examples I wrote for a demonstration on Forth graphics demonstrated last year for my local users group. This demonstration later generated sufficient interest to start a Forth Special Interest Group (SIG) within our club. I hope these programs are documented well enough to allow even a beginning Forth programmer to understand and use them in their own programs. If not, I will answer any questions you may have if a self addressed stamped envelope (SASE) included with your question(s). Mail your questions to me in care of THE NATIONAL NINETY-NINER.

For reasons of brevity I will not present any programs for TEXT mode or GRAPHICS mode, but will only describe them briefly. TEXT mode implies exactly what the name means. In this mode you get 24 lines of 40 characters each. Though you can define new character fonts, there is only one foreground and one background for the whole screen and sprites are not available. This mode is used by TI-WRITER and other text editor programs. To access this mode in Forth you type 'TEXT' and press enter.

GRAPHICS mode has twenty-four lines of thirty-two characters each. New character fonts can be defined and Each of the 32 character sets (a set here means a group of eight characters - the 9918A convention) can have their own foreground and background colors. Full sprite capabilities are available in GRAPHICS mode. To access this mode of graphics you type 'GRAPHICS' and press enter. This mode is the same one used in both of the TI Basics. More character sets are available and each may have their own character set. You do not have to trade characters for sprite shapes. Forth has subroutines similar to Basic. Both bear the same names and have similar inputs although Forth's RPN convention puts the arguments first followed by the subroutine name. The subroutines for GRAPHICS mode supported by Forth are SCREEN, COLOR, HCHAR, GCHAR, and CHAR.

This first demonstration shows off the MULTI color mode which is called LOW RES mode by other computers. You get 48 rows of 64 blocks which consist of 4X4 (16) pixels. Each pixel can be any of the sixteen colors available on the TI. The only other time I have seen the MULTI-color mode implemented is in the VIDEO-GRAPHS cartridge's MOSIAC mode. To call up MULTI-color mode you type 'MULTI', space, 'MINIT' and press enter. You draw in MULTI-color mode with the Forth word MCHAR which has syntax: 'row column color MCHAR' where row = 0-47, column = 0-63, and color = 0-15. Also included along with this demo is a sprite example called BUG. After loading this demo type 'MULTIDEMO' to start it.

```
SCREEN #310
 0 ( MULTICOLOR AND SPRITE DEMO J. STANFORD AUG 1985 ) BASE->R HEX
 1 : BUG (   ---   ) ( A SPRITE DEMO USING DOUBLE SIZE SPRITES    )
 2      2000 SSDT  2 MAGNIFY ( DEFINE SPRITE TABLE AND SPRITE SIZE)
 3      0004  0281  8041  3A3F 80 SPCHAR ( THIS DEFINES THE FOUR  )
 4      3F3A  4180  8102  0400 81 SPCHAR ( CHARACTERS NEEDED FOR  )
 5      2022  2428 08070 0FCFC 82 SPCHAR ( THE BODY OF THE BUG.   )
 6     0FCFC  7080  2824  2220 83 SPCHAR ( THE COLOR WILL BE BLACK)
 7
 8      0000  0000  0001  0303 84 SPCHAR ( THIS DEFINE THE FOUR   )
 9      0303  0100  0000  0000 85 SPCHAR ( CHARACTERS NEEDED FOR  )
10      0000  0000 0BCFE  76FF 86 SPCHAR ( THE SHELL OF THE BUG.  )
11     0FF76 0FEDC  0000  0000 87 SPCHAR ( THE COLOR WILL BE RED  )
12
13      0FB 60 06 84 0 SPRITE -1 1 0 MOTION ( DEF SHELL, DIRECTION)
14      0FB 60 01 80 1 SPRITE -1 1 1 MOTION ( DEF BODY, DIRECTION )
15      2 #MOTION  ;   R->BASE --> ( START MOVEMENT OF SPRITES    )

SCREEN #311
 0 ( MULTICOLOR AND SPRITE DEMO J. STANFORD AUG 1985 ) BASE->R HEX
 1 : MULTIDEMO  ( --- )   ( A MULTI-COLOR DEMO )
 2      MULTI MINIT RANDOMIZE    ( INIT MULTICOLOR MODE AND RANDOM#)
 3      300 0 DO 15 RND          ( GET RANDOM NUMBER FOR: COLOR    )
 4               40 RND          (                      : ROW      )
 5               31 RND MCHAR    (                      : COLUMN   )
 6            LOOP
 7    2 0 DO                                ( LOOP TWO TIMES  )
 8      30 0 DO                             ( FOR EACH ROW    )
 9             0B RND 2+                 ( SELECT A RANDOM COLOR )
10             40 0 DO DUP I J MCHAR LOOP ( AND FILL ONE ROW     )
11             DROP                      ( DROP UNNEED NUMBER    )
12           LOOP
13           BUG                         ( NOW START THE SPRITES )
14        LOOP  0 #MOTION DELALL TEXT ;   ( CLEAN UP AND GOTO TEXT)
15                                                  R->BASE
```

The next demonstration may seem complex but when you break it up into different parts it is not too hard to understand. First, it is a demonstration of the TI's bitmap mode with routines to draw simple squares and circles. Second, it demonstrates fixed point mathematics with an integer implementation of SIN and COS functions which are used in the circle routine.

Bitmap mode and it's two other variants for split-screen modes allow the user to turn individual pixels (192 X 256 dots) on and off and in any of the sixteen TI colors. There is one limitation to the use of colors in bitmap mode, his being groups of eight pixels which share the same foreground and background colors. This is why bitmap drawings tend to 'bleed' beyond the lines drawn for a drawing. (note: this is not associated with the escape of colors when filling in figures, which is based on the whether the shape is closed or not. It is associated to how the 9918A maps the video screen. See Editor/Assembler manual for complete details.)

The coordinate system used for the bitmap mode starts with (0,0) in the upper lefthand corner. Rows number from 0 to 191 and columns numbered for 0 to 255. To enter the bitmap modes type one of the following 'GRAPHICS2', 'SPLIT', or 'SPLIT2' and press enter. GRAPHICS2 gives you the full screen in bitmap mode, while SPLIT and SPLIT2 gives you a combination of bitmap mode and graphics text mode. These split screen modes are very useful because you can have the text field for command input and the bitmap field for the graphics. I recommend the used of the split screen modes for the following demo to allow you to see the commands you type in. Support for bitmap mode consists of words to control the drawing modes (consisting of DRAW to set the mode to draw graphics, UNDRAW to set the mode to erase graphics, DTOG to set the mode to toggle the graphics on the screen and DCOLOR to control the color the graphics will be drawn in) and words to control the actual drawing. (which are DOT and LINE which draws dots and lines respectively) The graphics commands added by my bitmap demo are: SQUARE whose syntax is 'x-dot y-dot size SQUARE' where (x-dot,x-dot) is the coordinate of the upper lefthand corner of the square and size is the length of a side, and CIRCLE whose syntax is 'x-dot y-dot size CIRCLE'. This draws a circle centered at (x-dot,y-dot) and radius of length 'size'. (NOTE: SQUARE makes use of LINE while CIRCLE uses DOT.)

The SIN and COS, as I said before, are examples of using fixed point mathematics. Thus complex calculations are allowed without resorting to the use of floating point notation. This implementation is a simple table lookup for SIN from 0.00 to 1.57 radians. COS is done by adding PI/2 (90 degrees) to the angle and then computing the SIN of the augmented angle. The input for both of these routines is the angle in radians multiplied by one hundred. This gives a two place accuracy for the input. The output is returned as a number multiplied by ten thousand which provides four places of accuracy. The syntax is the same for both functions. For example, if I wanted the SIN of PI/4 (45 degrees), I would type '79 SIN' which would return '7104' which is 0.7071 * 10000. Well I believe I have said enough, here is the demo type BM-DEMO to start it.

```
SCREEN #312
 0 ( BIT-MAP DEMONSTRATIONS                    J. STANFORD AUG 1985)
 1 ( GENERAL USE VARIABLES AND ROUTINES                            )
 2   0 VARIABLE X              ( VARIABLE TO STORE X COORDINATE  )
 3   0 VARIABLE Y              ( VARIABLE TO STORE Y COORDINATE  )
 4   0 VARIABLE SIZE           ( VARIABLE TO STORE SIZE OF SHAPE )
 5   0 VARIABLE X-DEL          ( VARIABLE TO STORE A X DELTA     )
 6   0 VARIABLE Y-DEL          ( VARIABLE TO STORE A Y DELTA     )
 7
 8 : M/RND ( D N1 --- N2) ( MIXED MODE DIVISION WITH TRUE ROUNDING)
 9       >R R M/ SWAP R 2 / + R) / + ;
10
11 : C-DEL (  ---  )        ( CALCULATE DELTAS FOR SQUARE ROUTINE )
12       X @  SIZE @  +  X-DEL !   ( CALCULATE X DELTA X-DEL=     )
13       Y @  SIZE @  +  Y-DEL ! ; ( CALCULATE Y DELTA             )
14
15                                                          -->

SCREEN #313
 0 ( BIT-MAP DEMONSTRATIONS                    J. STANFORD AUG 1985)
 1
 2 : SQUARE (  X  Y  SIZE  --- ) ( DRAW SQUARE UPPER LEFT = (X,Y )
 3       SIZE !  Y !  X !        ( SAVE THREE PARAMETERS          )
 4       C-DEL                   ( CALCULATE THE NEEDED DELTAS    )
 5           X @     Y @ X-DEL @     Y @ LINE   ( DRAW FIRST SIDE )
 6       X-DEL @     Y @ X-DEL @ Y-DEL @ LINE   (      SECOND     )
 7       X-DEL @ Y-DEL @     X @ Y-DEL @ LINE   (      THIRD      )
 8           X @ Y-DEL @     X @     Y @ LINE ; (      FOURTH     )
 9
10
11 : TABLE  ( #1 #2 ... #N N --- ) ( BUILD A LOOK-UP TABLE SIZE N )
12        <BUILDS 0 DO , LOOP DOES> SWAP 2 * + @ ;
13
14
15                                                          -->

SCREEN #314
 0 ( BIT-MAP DEMONSTRATIONS J. STANFORD AUG 1985) BASE->R DECIMAL
 1 ( TRIG LOOKUP TABLE  SINE IN RADIANS 0.00 -> 1.57 STEP 0.01    )
 2   10000 9999 9998 9995 9992 9987 9982 9975 9967 9959 9949 9939
 3    9927 9915 9901 9887 9871 9854 9837 9819 9799 9779 9757 9735
 4    9711 9687 9662 9636 9608 9580 9551 9521 9490 9458 9425 9391
 5    9356 9320 9284 9246 9208 9168 9128 9086 9044 9001 8957 8912
 6    8866 8820 8772 8724 8674 8624 8573 8521 8468 8415 8360 8305
 7    8249 8192 8134 8076 8016 7956 7895 7833 7771 7707 7643 7578
 8    7513 7446 7379 7311 7243 7174 7104 7033 6961 6889 6816 6743
 9    6669 6594 6518 6442 6365 6288 6210 6131 6052 5972 5891 5810
10    5729 5646 5564 5480 5396 5312 5227 5141 5055 4969 4882 4794
11    4706 4618 4529 4439 4350 4259 4169 4078 3986 3894 3802 3709
12    3616 3523 3429 3335 3240 3146 3051 2955 2860 2764 2667 2571
13    2474 2377 2280 2182 2085 1987 1889 1790 1692 1593 1494 1395
14    1296 1197 1098 0998 0899 0799 0699 0600 0500 0400 0300 0200
15    0100 0000     158 TABLE SINTABLE ( 158 ELEMENTS ) R->BASE -->

SCREEN #315
 0 ( BIT-MAP DEMONSTRATIONS                    J. STANFORD AUG 1985)
 1 BASE->R DECIMAL
 2 : S(PI  ( RADIANS*100 --- SIN*10000   RETURNS SIN 0->PI      )
 3       DUP 157 >               ( IF GREATER THAN 1.57 RADIANS )
 4       IF 314 SWAP - ENDIF     ( SUBTRACT FROM 3.14 RADIANS   )
 5       SINTABLE  ;
 6 : SIN   ( RADIANS*100 --- SIN*10000  RETURNS SIN OF ANY ANGLE )
 7       628 MOD                 ( BRING WITHIN +/- 2*PI        )
 8       DUP 0< IF 628 + ENDIF   ( IF NEGATIVE ADD 2*PI         )
 9       DUP 314 >               ( IF GREATER THAN PI THEN      )
10       IF 314 - S(PI MINUS     ( SUBTRACT PI AND NEGATE SINE  )
11       ELSE S(PI ENDIF ;       ( OTHERWISE JUST CALCULATE SINE)
12 : COS  ( RADIANS*100 --- COS*10000   RETURNS COS OF ANY ANGLE )
13       628 MOD                 ( NORMALIZE ANGLE, ADD PI/2 AND)
14       157 + SIN ;             ( CALCULATE SINE AS USUAL      )
15  R->BASE -->

SCREEN #316
 0 ( BIT-MAP DEMONSTRATIONS                    J. STANFORD AUG 1985)
 1 BASE->R DECIMAL
 2 : PLOT8 (  ---   )  ( PLOTS THE EIGHT POINTS OF A CIRCLE BASED )
 3                     ( MIRROR AND TRANSPOSE IMAGES              )
 4       X-DEL @ X @ +   Y-DEL @ Y @ +   DOT
 5       X-DEL @ X @ +   Y-DEL @ Y @ -   DOT
```

```
 6       X-DEL @ X @ -   Y-DEL @ Y @ +   DOT
 7       X-DEL @ X @ -   Y-DEL @ Y @ -   DOT
 8       X-DEL @ Y @ +   Y-DEL @ X @ +   DOT
 9       X-DEL @ Y @ +   Y-DEL @ X @ -   DOT
10       X-DEL @ Y @ -   Y-DEL @ X @ +   DOT
11       X-DEL @ Y @ -   Y-DEL @ X @ -   DOT ;
12 : CALX ( RADIUS RADIANS$100 --- X COORD FOR A CIRCLE X=R*COS(A)
13       COS 10000 */ ;
14 : CALY ( RADIUS RADIANS$100 --- Y COORD FOR A CIRCLE Y=R*SIN(A)
15       SIN 10000 */ ;                                  R->BASE --)

SCREEN #317
 0 ( BIT-MAP DEMONSTRATIONS                  J. STANFORD AUG 1985)
 1 BASE->R DECIMAL
 2 : CIRCLE ( X-CENTER Y-CENTER RADIUS --  ) ( BITMAP CIRCLE      )
 3       -DUP 0= IF DOT                 ( TEST FOR ZERO RADIUS     )
 4       ELSE
 5        SIZE ! Y-DEL ! X-DEL !        ( SAVE INPUTS FOR ALTER USE )
 6        159 0 DO                      ( INTERATE 1/8 OF CIRCLE   )
 7             SIZE @ I 2 /  CALX X ! ( CALCULATE X COORDINATE   )
 8             SIZE @ I 2 /  CALY Y ! ( CALCULATE Y CORRDINATE   )
 9             PLOT8                  ( PLOT EIGHT PARTS OF CIRCLE)
10             157 SIZE @ / +LOOP
11       ENDIF ;
12 : BM-DEMO SPLIT 6 ! DO
13                    20  10 I 20 * SQUARE
14                    182 64 I 10 * CIRCLE
15                    LOOP ."  TA DA!!!" ;          R->BASE ;S
```

Well as my favorite cartoon character says, 'That's all folks'. Hope you enjoy these little graphics examples and can use them for yourselves. Until later.

ASSEMBLY LANGUAGE PRINT ROUTINES

By Edgar Bohmann -- JSC User's Group (JUG)

Here are some general print routines which I have developed for use in assembly language programs. These routines are set up as BLWP subroutines to isolate their register usage from your main assembly language program which calls them. Included in the listings is a PAB definition for my printer ("RS232.BA=600"). Substitute your printer's description in PNAME and be sure to change the value of PNAML to reflect the length in bytes of your printer's description.

The PAB locations used here are >1F00 for the PAB description and >1F40 for the line buffer to be printed. You may use other areas of VDP for your PAB and buffer if you like, but make sure they are not being used by the computer for something else.

```
      REF  VSBW,VNBW,DSRLNK
*
PABLOC EQU  >1F00   VDP LOCATION OF PAB
DATLOC EQU  >1F40   VDP LOCATION OF LINEP
PABPNT EQU  >8356   POINTER TO PAB
*
DATBUF BSS  80      80-BYTE LINE BUFFER
PRTWSP BSS  32      WORKSPACE FOR ROUTINES
*
*                   **PAB DEFINITION**
PPAB   DATA >0012   OPEN CODE & FLAGS
       DATA DATLOC  LOCATION OF BUFFER
       DATA >5050   RECORD LENGTH
       DATA 0
PNAML  DATA 12      LENGTH OF PRINTER NAME
PNAME  TEXT 'RS232.BA=600'   PRINTER NAME
PPABE  EQU  $       END OF PAB DEFINITION
*
PCLS   BYTE 1       CLOSE CODE
PWRT   BYTE 3       WRITE CODE
*
POPEN  DATA PRTWSP  BLWP VECTOR FOR OPEN
       DATA POPN
PCLOS  DATA PRTWSP  BLWP VECTOR FOR CLOSE
       DATA PCLO
POUTP  DATA PRTWSP  BLWP VECTOR FOR OUTPUT
       DATA POUT
*
POPN   LI   R0,PABLOC   GET VDP ADDRESS
       LI   R1,PPAB     POINT TO PAB DEF
       LI   R2,PPABE-PPAB  LENGTH OF PAB
       BLWP @VNBW       MOVE PAB TO VDP
       LI   R6,PABLOC+9 ADDRESS TO SAVE
```

```
      MOV  R6,@PABPNT    IN PAB POINTER
      BLWP @DSRLNK       OPEN PRINTER
      DATA 8
      LI   R0,PABLOC     GET VDP ADDRESS
      MOVB @PWRT,R1      SET FOR WRITE
      BLWP @VSBW           IN PAB
      RTWP
*
PCLO  LI   R0,PABLOC     GET VDP ADDRESS
      MOV  @PCLS,R1      SET FOR CLOSE
      BLWP @VSBW           IN PAB
      LI   R6,PABLOC+9   ADDRESS TO SAVE
      MOV  R6,@PABPNT    IN PAB POINTER
      BLWP @DSRLNK       CLOSE PRINTER
      DATA 8
      RTWP
*
POUT  LI   R0,DATLOC     VDP ADDR OF BUFFER
      LI   R1,DATBUF     POINT TO BUFFER
      LI   R2,80         80-BYTE LINE BUFFER
      BLWP @VMBW         MOVE LINE TO VDP
      LI   R6,PABLOC+9   ADDRESS TO SAVE
      MOV  R6,@PABPNT    IN PAB POINTER
      BLWP @DSRLNK       WRITE A LINE
      DATA 8
      RTWP
*
```

Here is a program that can be used to test the print routines given above. The DEF statement will cause the program to be included in the REF/DEF table when it is loaded. The assembled object code can be loaded by either the LOAD AND RUN option of the Editor/Assembler or by a CALL LOAD from Basic and Extended Basic. If either Basic is used, a CALL LINK will have to follow the load to execute the program.

The test program given here will print two lines over and over until you reset the computer. For convenience, two additional routines are included with the program: PCLEAR will clear the line buffer in RAM and MOVMSG will copy a message into the line buffer to prepare it for printing.

```
*                      **ROUTINE FOR TESTING**
      DEF  TEST
TSTWSP BSS  32           MY WORKSPACE
TEST  LWPI TSTWSP
      BLWP @POPEN        OPEN PRINTER
      BL   @PCLEAR       CLEAR BUFFER
      LI   R0,MES61          MESSAGE TO PRINT
      LI   R1,MES61E-MES61 LENGTH OF MESSAGE
      BL   @MOVMSG           MOVE TO LINE BUFF
      BLWP @POUTP        PRINT MESSAGE
      BL   @PCLEAR       CLEAR BUFFER
      LI   R0,MES62      NEXT MESSAGE
      LI   R1,MES62E-MES62  LENGTH
      BL   @MOVMSG            MOVE IT
      BLWP @POUTP             PRINT IT
      BLWP @PCLOS        CLOSE PRINTER
      JMP  TEST          **LOOP BACK**
*
MES61  TEXT 'TEST MESSAGE'
MES61E EQU  $
MES62  TEXT 'ANOTHER MESSAGE'
MES62E EQU  $
*
PCLEAR LI   R0,>2020     LOAD 2 BLANKS
       LI   R1,40        80 BYTES = 40 WORDS
       LI   R2,DATBUF    LOCATION OF BUFFER
PCLR1  MOV  R0,*R2+      BLANK 2 BYTES
       DEC  R1           DONE 40 WORDS YET?
       JNE  PCLR1        LOOP TIL DONE
       RT
*
MOVMSG LI   R2,DATBUF    LOCATION OF BUFFER
MOVM1  MOVB *R0+,*R2+    MOVE A BYTE
       DEC  R1           MESSAGE MOVED?
       JNE  MOVM1        LOOP TIL DONE
       RT
*
```

As I mentioned above, you can load the program with either Basic or Extended Basic. However, as you may know, Extended Basic does not include a DSRLNK to allow programs like this to access peripheral devices. Fortunately there are several versions of DSRLNK floating around which you can include in your program if you have access to them. John Phillips, John Clulow, and I have each provided versions of DSRLNK to User's Groups through THE 99'ERS ASSOCIATION.

Another alternative is to use a pseudo-DSRLNK routine like the one below. This is a "stripped down" version of DSRLNK which is only good for one peripheral. The standard DSRLNK searches through all DSR ROMs until it finds a device name which matches the one specified in your PAB. This version only searches one ROM and is set up here to check the 1st RS232 card (CRU address of >1300).

This routine is intended for calling with a BL so the BLWP @DSRLNK calls in the printer routines above should be replaced with BL @DSRLK calls. Also the DATA 8 instructions following the BLWP calls must be deleted. The advantage of this shortened version is that it is less than half the size of the standard DSRLNK so it is easier to type, takes up less memory, loads faster, and executes faster. One other change that must be made is to delete the REF statement for VSBW, VMBW, and DSRLNK. The Extended Basic loader does not resolve REFerences and the routines VSBW and VMBW must be explicitly EQUated to their X-Basic values as follows:

```
VSBW EQU  >2020
VMBW EQU  >2024
```

The value of PNAMP must be matched to the name length of your printer but must only reflect the characters up to the first period of the name. For a printer description of PIO.LF set PNAMP to 3.

```
*          **PSEUDO DSRLNK**
PNAMP  EQU  5       LENGTH OF 'RS232'
*
DSRLK  LWPI >83E0     GPL WORKSPACE
       LI   R0,PNAMP  GET NAME LENGTH
       MOV  R0,@>8354 SAVE FOR DSR USE
       INC  R0        ADJUST FOR .
       A    R0,@>8356 ADJUST PAB POINTER
       LI   R12,>1300 CRU FOR 1ST RS232
       LI   R1,1      DSR VERSION #
       SBO  0         TURN ON DSR
       LI   R2,>4008  STD ADDR FOR DSR LINKAGE
       JMP  SG02
SG0    MOV  R3,R2     TRY NEXT DEVICE
SG02   MOV  *R2,R2    GET NEXT LINK ADDR
       JEQ  NOROM     EXIT IF NO MORE
       MOV  R2,R3     SAVE LINKAGE
       INCT R2        POINT TO DSR FOR DEVICE
       MOV  *R2+,R9   SAVE IT IN CASE WE NEED IT
       LI   R5,PNAMP*>100 NAME LENGTH IN HI BYTE
       CB   R5,*R2+   SEE IF LENGTH MATCHES
       JNE  SG0       NO
       SRL  R5,8      YES
       LI   R6,PNAME  GET ADDRESS OF NAME
NAME1  CB   *R6+,*R2+ SEE IF NAMES MATCH
       JNE  SG0       NO
       DEC  R5        YES
       JNE  NAME1
       BL   *R9       NAME MATCHES
       NOP            NEED AN ERROR RETURN SPOT
       SBZ  0         TURN OFF DSR
NOROM  LWPI PRTWSP    PREPARE TO RETURN
       RT
*
```

One last point to mention is the fact that the CLOSE operation is not required for the RS232 peripheral. The PCLOS subroutine is included here mainly for completeness. Basic programs require the CLOSE operation to reclaim the VDP buffer space that was allocated when the "file" for the RS232 card was opened. However this is not necessary in assembly language and the DSR itself takes no action in response to a CLOSE command. CLOSE commands are required for real file oriented devices like disks because this causes the sector buffer in memory to be written to the disk (on write operations) and also causes the file directory (which is kept in memory while the file is open) to be written to the disk. Such activities are not necessary for devices like the RS232 card.

CP/M - Part VI.

By Leonard Lanigan

My apologies for missing last month's column-I have been having some trouble with my computer, and have found it necessary to steal my son's console to continue.

I have received the first four disks from the FOG library, in the Osborne 1 format. The disks consist of several files of general information about the library, and an auto-expense accounting template for Personal Pearl. As promised, the Osborne 1 format disks work just fine on the MSS CP/M card, though the system tracks must be copied onto the disks before they will boot up. Unfortunately, all of the documentation for the disks is on the disks in the form of Wordstar files. Given that Wordstar was provided with the Osborne computer, and is thus available to all Osborne owners, this is logical. Alas, I do not have Wordstar available for the MSS card, and I can't print out the documentation, so everything has been done by trial and error.

I have managed to run the Pearl templates, and they seem to function as they should. While the DOC files are unreadable to me, Pearl is able to find the files it needs and once they are loaded, we get along fine.

There are many volumes of program disks available from FOG, in a variety of categories, and I strongly recommend membership

in FOG for any CP/M user, regardless of the machine used. You should be aware that nearly all of the programs available are designed as enhancements to some more general program such as Personal Pearl, dBase II, Wordstar, or Microsoft BASIC. These packages were provided with the Osborne computers, and represent the most common software in the Osborne world. Given that I have only Personal Pearl available, I am limited in the number of packages I can evaluate at this time. I hope to evaluate other packages as I am able.

If anyone has questions, comments suggestions, etc. regarding the Morning Star Software CP/M card, CP/M software, the phases of the moon, 8085 assembly code or any other aspect of this column, please let me know. I am beginning to know how several of the staff writers feel about writing in a vacuum, with no feedback. I may be the only one who reads this!

CUSTOMIZING SUPERBUG II

By Edgar Dohmann

I appreciate all the interest that has been shown in SUPERBUG II. I spent a lot of time working on this program and the manual for it and it is quite rewarding when people write or call me to comment on the program or make suggestions for improvement. It is also nice to receive a few dollars from people who think enough of the program to help compensate for some of my time and effort. Thanks to all of you and also thanks to THE NATIONAL NINETY NINER and MICROpendium for carrying articles and announcements about SUPERBUG II.

The purpose of this article is to explain how you can customize a couple of SUPERBUG II features to suit your own needs or desires. I had intended to include this information in the manual that is on the disk but I used up all available sectors before I got this far. The most likely portions of SUPERBUG II which you might like to customize are the default output device specification and the screen colors which you can toggle with the J command. You will need a Disk Patch program like the Disk Fixer, Advanced Diagnostics, Disk + Aid, or Ed's Disk Analyzer to perform these modifications. Be sure to make a backup copy of the SUPERBUG II distribution disk you have received before attempting these conversions. This serves two purposes: you can thus preserve a copy of the original authorized distribution disk to pass on to others, and second you will have a backup in case something goes wrong in the disk patching process.

Locate the file you wish to patch on the disk: SBU6, SBU66, or SBU60. SBU6 and SBU66 are in memory image (Program) format while SBU60 is in compressed object format. As a result, the areas you will want to patch will be different for each file. Hopefully, your disk patch program has hints on how to search, interpret, and patch each of these types of files.

The default output device specification is near the front of the file. Start searching about 500 bytes or so into the program for the character string:

```
RS232.BA=9600
```

This string will be followed by 7 spaces so you can replace this string of 20 characters with your own default output specification. Be sure not to exceed 20 characters or you will overlay area that is used by the program which will mess up any name longer than 20 characters. Also be sure to change the byte immediately preceding the printer name from >0B to the length of your new device name.

The five color codes are located near the end of the program. Start searching about 100 bytes from the end of the program for the following sequence of data:

```
F4F4
1B1B
1313
1E1E
4E4E
```

These are the 5 color combinations which you toggle through with the <J> command. Change them to whatever combinations you wish to use. It is relatively easy to change these 5 codes to other values but it would take major surgery to add more colors to the list.

If you wish to add more choices, to the list, the code for toggling through the list immediately follows the list and if you dis-assemble it you may be able to figure out how it works and decide how you might like to change it. Again, be sure you have a backup copy of the original distribution before you try this.

TIPS #26 FROM THE TIGERCUB

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

>>>>>>NOW AVAILABLE<<<<<<<<

Tips from the Tigercub VOLUME 2 The entire contents of Tips Nos. 15 through 24, with 64 routines and files, also $15.00 postpaid. Or both for $27 postpaid.

>>>>ALSO NOW AVAILABLE<<<<<<

New Catalog #6, for $1 which is deductable from your first order. Describes 140 original programs for only $3 each (plus

$1.50 per order for casette or disk, package and postage). If you have my previous catalog, the following are now available in Extended Basic versions - Fast Addition Practice, Submarine Hunt, Rithmatik, Wawaland (also now available in Basic with Speech), Long Division Cryptograms, Miss Spell, Scrambulation, Bargraffer, Squinch, Dry Gulch, Name That Tune, Scrum, Midnight Trail, Nimbo, Kindertimes, Optical Illusion, Bazoo, Synonymy, Speeder Reader, Changeroo, Glunk, Fraction Math, Three Buckets Puzzle, Roman Numbers, Match A Patch, Kinderminus, I E Spelling, Casting Out Nines, Haunted Graveyard, Spalling Teecher, Homonymy, Antonymy, Old -Timer Puzzle, Ten Thousand Sights, Mechanical Aptitude Test, Junior Speeder Reader, and Bars and Balls.

Due to reduced prices for disks and mailers, the PPM charge is now $1.50 for either disk or casette - BUT PLEASE BE SURE TO SPECIFY WHICH! And my best seller - NUTS BOLTS, a full disk of 100 (yes, I said 100) utility subprograms in MERGE format, ready for you to merge into your own programs. 13 type fonts, 14 text display routines, 9 wipes, 8 pauses, 3 programming aids, 9 data saving and reading routines, 5 graphics routines, 4 time and date, 6 music, 12 sorts and shuffles, 2 printer aids, 4 key and joystick, 4 math, 2 protection and 7 miscellaneous, plus a tutorial on subprograms. With documentation, example of using each subprogram. All for only $19.95 postpaid.

Now for the old business - I was mortified to find an error in the Unprintable Unkeyable Program in Tips #22. The last line should end with ELSE 180, not ELSE 130. In the Grocery Shopping program in Tips #21, your wife will never get to the zucchini unless you delete line 140 and change line 200 to - 200 IF EOF(1)<>1 THEN 130. Sorry about that. And the update to the Menu Loader in Tips #22 will not list all listable files, just D/V80 files. I now have a version to really list all listable files, I think, plus show protection, dump the catalog to the printer, rescan, etc., but am not sure all the bugs are out so will publish it next month. Folks have been asking why their orders for TI-WRITER COMPANION, mentioned in Tips #22, were being returned unopened, so I called Bill Browning. He said he found he was going broke selling it for $2.50, but he is now prepared to supply it for $6.50. Still a bargain, in my opinion.

Barry Ensley warns that when FCTN V is used for a blank in a filename, as mentioned in Tips #25, it is not recognized by the Disk Manager. In Tips #21, I said that the special characters available on the Gemini printer could not be accessed from TI-Writer. I have since learned that Star Micronics hid a valuable feature of their printer in a paragraph of gobbledegook computerese in the manual. See "Other Function Codes", ESC ">", ESC "=" and ESC "#". In plain English, you can access these codes by CTRL U, FCTN R, CTRL U, SHIFT >, then type the character with an ASCII 128 less than the character you want. In other words, if you want CHR$(160), hit the space bar (ASCII 32), etc. To get back to the normal character mode, use CTRL U, FCTN R, CTRL U, SHIFT #. Many thanks to David Aragon (San Antonio Area 99ers newsletter, Aug. 1985), who described how to do the same by transliteration. In Tips #25, I said that a program which had been converted to I/V 254 format by adding REM lines could be converted back to program format by deleting the REM lines and reSAVing. Well, it usually can - but not always!

I have been receiving inquiries as to whether my programs published in the Tips are public domain programs which can be placed in user group libraries and on BBS's. Well, the copyright notice on this newsletter is really only intended to keep anyone from reprinting it for personal profit. I have always thought that programs published for the purpose of being keyed in should be OK to copy, and I don't intend to claim that "you must own the magazine"! However, a peculiar situation has developed. The short programs which I wrote to give away to promote my other programs, have become the bread and butter of my business! If it was not for the sales of the Tips disk and the Nuts Bolts disk, I would long ago have gone out of business. So, I would appreciate it if you would exercise some restraint in putting my Tips programs in your libraries or in downloadable form on your BBS. And I do consider my two Tips disks, as complete collections of programs, to be copyrighted material which should not be placed in libraries for copying.

In the Automatic Mouse Maze in Tips #23, you can improve the maze by adding these lines -

```
475 IF (C>20)$(X<10)THEN 500
515 X=X+1
555 X=X+1
595 X=1+1
1325 X=0
```

And the last word - I think - on the challenge to quickly scramble the numbers 1 to 255. Ian Swales sent me, from Belgium, two routines which beat everyone else - and then sent me two more which beat his first ones! His PEEK version -

```
100 DIM A(255),C(255):: FOR K=255 TO 1 STEP -1 :: RANDOMIZE :: CALL PEEK(-31808,B)::
J=INT(B$K/256+1):: C(K)=MAX(J,A(J)):: A(J)=MAX(K,A(K)):: NEXT K
```

And see if you can unravel the logic of this truly elegant bit of code!

```
100 DIM A(255):: RANDOMIZE :: FOR K=255 TO 1 STEP -1 :: J=INT(RND$K+1):: T=MAX(J,A(J))):: A(J)=MAX(K,A(K)):: A(K)=T :: NEXT K
```

So, on to new business -

>>>>>>>>>>ANNOUNCING<<<<<<<<<<

The TI-99/4A TRAVelER

a magazine-on-disk!

700 sectors of articles and programs in each issue (SS/SD or DS/DD)! with contributions by Mack McCormick, Ron Albright, and many others! Special pre-publication prices - $30 for 6 issues; $7 for sample issue (first issue will be Sept 85) Send your check now to:

Barry A. Traver, Editor
835 Green Valley Drive
Philadelphia ,PA 19128
phone (215) 483-1379

To give you an idea of Barry Traver's knowledge of our computer, try this one. I've figured out the why, but I'll have to

ask Barry to explain the why of the why!

```
100 ! LINPUT PUZZLE/BUG by
      B.A. Traver
110 ! QUESTIONS? Send SASE
      to Barry Traver
120 ! 552 Seville St.
      Phila. PA 19128
130 CALL CLEAR :: PRINT "LIN
PUT PUZZLE/BUG":"BY BARRY TR
AVER"
140 PRINT "Can you figure ou
t why your computer will not
 obey?"
150 PRINT "Why won't it stop
 when you tell it to?": : :
160 LINPUT "Want me to stop?
 (YES/NO)":N$
170 IF N$="YES" THEN STOP EL
SE 160
180 END
```

It seems that many of you still haven't heard of Super 99 Monthly, published monthly (and on time!) by Bytemaster Computer Services, 171 Mustang Street, Sulphur, LA 70663, for $12 per year. The May issue contained a Word Processor Dump, to dump a graphics/text screen into a D/V80 file which can be printed out of the TI-Writer Formatter - that program alone is worth the annual subscription price!

I've said it before, there is more than one way to skin that poor cat. This is my routine to alternate between the #1 and #2 joysticks. Z=Z+1+(Z=2)*2 :: CALL (JOYST (Z,X,Y) Compact, isn't it? Now, the Reading-Berks 99ers publish a newsletter called "A Byte of Info", which is hardly more than a byte long, but the August byte was a mouthful! Check this -

```
100 Z=2 110 Z=1/Z^2 :: CALL JOYST(Z, X,Y)
```

And this! Elegant!

```
Z=Z=0 :: CALL JOYST(Z+2,X,Y)
```

Here is another of those programs that write a program. This one will read a screen of graphics and/or text and convert it into a RUNable program of DISPLAY AT statements which will recreate the screen. First, we need a file of the hex codes of all the normal characters, to check against to see if any have been redefined. Rather than key in all 95 of the 16-digit codes, let's write a program to write a program of them -

```
110 OPEN #1:"DSK1.HEXCODES",
VARIABLE 163 :: LN=30000 ::
FOR D=32 TO 124 STEP 8 :: FO
R CH=D TO D+7 :: CALL CHARPA
T(CH,CH$)
120 D$=D$\CHR$(179)\CHR$(200
)\CHR$(16)\CH$ :: NEXT CH
130 PRINT #1:CHR$(INT(LN/256
))\CHR$(LN-256*INT(LN/256))\
CHR$(147)\SEG$(D$,2,LEN(D$))
\CHR$(0):: LN=LN+1 :: D$=""
:: NEXT D
140 PRINT #1:CHR$(255)\CHR$(
255):: CLOSE #1 :: END
```

RUN that to create a MERGE format program of DATA statements. Now, key in the GRAFWRITER program -

```
31000 SUB GRAFWRITER
31001 OPEN #1:"DSK1.PG",OUTP
UT,DISPLAY ,VARIABLE 163
31002 RESTORE 30000 :: L=300
00 :: GOSUB 31018
31003 FOR CH=32 TO 127 :: CA
LL CHARPAT(CH,CH$):: READ A$
 :: IF CH$=A$ THEN 31004 ELS
E GOSUB 31019 :: GOSUB 31018
31004 NEXT CH
31005 FOR CH=128 TO 143 :: C
ALL CHARPAT(CH,CH$):: IF CH$
=RPT$("0",16)THEN 31006 ELSE
 GOSUB 31019 :: GOSUB 31018
31006 NEXT CH
31007 PRINT #1:L$\CHR$(157)\
CHR$(200)\CHR$(5)\"CLEAR"\CH
R$(0):: GOSUB 31018
31008 FOR R=1 TO 24
31009 M$=L$\CHR$(162)\CHR$(2
40)\CHR$(183)\CHR$(200)\CHR$
(LEN(STR$(R)))\STR$(R)\CHR$(
179)
31010 FOR C=3 TO 30 :: CALL
GCHAR(R,C,G):: CALL HCHAR(R,
C,42):: IF F=0 AND G=32 THEN
 31013
31011 F=1 :: IF FF=1 THEN 31
012 ELSE CC=C-2 :: FF=1
31012 A$=A$\CHR$(G)
31013 NEXT C :: IF CC=0 THEN
 CC=1 :: A$=" "
 31014 PRINT #1:M$\CHR$(200)
CHR$(LEN(STR$(CC)))\STR$(CC)
\CHR$(182)\CHR$(181)\CHR$(19
9)\CHR$(LEN(A$))\A$\CHR$(0)
31015 L=L+10 :: F,FF,CC=0 ::
 M$,A$="" :: GOSUB 31018 ::
NEXT R
31016 PRINT #1:L$\CHR$(134)\
CHR$(201)\L$\CHR$(0):: GOSUB
 31018
31017 PRINT #1:CHR$(255)\CHR
$(255):: CLOSE #1 :: SUBEXIT
31018 L1=INT(L/256):: L2=L-2
56*L1 :: L$=CHR$(L1)\CHR$(L2
):: L=L+10 :: RETURN
31019 PRINT #1:L$\CHR$(157)\
CHR$(200)\CHR$(4)\"CHAR"\CHR
$(183)\CHR$(200)\CHR$(LEN(ST
R$(CH)))\STR$(CH)\CHR$(179)\
CHR$(199)\CHR$(16)\CH$\CHR$(
182)\CHR$(0):: RETURN
31020 SUBEND
```

Next, Enter MERGE DSK1. HEXCODES to merge in those DATA statements. Then save the program by SAVE DSK1.GRAFWRITER,MERGE Now, load any program which has a screen you would like to copy. Run the program to the point where the screen display is ready, then break it with FCTN 4. Put in a temporary line going to itself, such as 1001 GOTO 1001, and run the program again to be sure you found the right place. Then replace that temporary line with CALL GRAFWRITER :: STOP Put in the disk containing the Grafwriter program and enter MERGE DSK1.GRAFWRITER. Then RUN the program. When it stops, type NEW, then MERGE DSK1.PG and then RUN!

MEMORY FULL! - Jim Peterson

THAT'S ALL FOR THIS MONTH FOLKS !!!

MR. BILL PECHNIK
1467 CARMI DR.
PENTICTON BC CANADA V2A 4R9

SUBSCRIPTION FORM
FOR "THE NATIONAL NINTY-NINER"
PUBLISHED BY THE 99'ERS ASSOCIATION

NAME: _____ DATE: _____

ADDRESS: _____

CITY: _____ STATE _____ ZIP _____

SUBSCRIPTION TYPE	AMOUNT	CHOICE
CLASS - BULK RATE	$12.00	_____
CLASS - US AND CANADA	$17.00	_____
CLASS - OVERSEAS	$22.00	_____

E MAIL CK./M.O. FOR SUBSCRIPTION CHOICE SELECTED ABOVE TO:

THE 99'ERS ASSOCIATION
ATTN: [illegible] KEITH
3533 SO. H ST., #93
BAKERSFIELD, CALIF. 93304