99/4A OWNER/USERS GROUP

MONTHLY NEWSLETTER

Not affiliated with Texas Instruments, Inc.

FEBRUARY 1984 VOLUME 2: ISSUE 2

COMPUTER ALPHABETIZING (By: Brett Pierce)

Here's a sorting routine that you may find a use for in a program. It's a sorter that puts a list of string variables into alphabetical order. To see it work: key it in, add these two lines, and type RUN:

```
10 FOR X=1 TO 6 :: INPUT RAND$(X) :: NEXT X :: LAST=0
1000 FOR X=1 TO 5 :: PRINT ALPH$(X) :: NEXT X
```

You will be asked to type in any six strings which will be put into the array RAND$ (for random). Then the routine will resequence the strings into alphabetical order and put them into the array ALPH$ (for alphabetic) which is printed out.

When the routine is used with a larger program, you may find that you want to add randomly organized strings to a list of strings that has already been alphabetized. This could be done by putting all of the strings into the array RAND$ and then using the routine, but it is much faster if the computer only has to go through the new words and not the entire list. To do this, make sure the already alphabetized list is in order in the array ALPH$ and LAST equals the number of the strings in the list (the routine must know how many strings to scan). Then execute the routine with the new strings in the array RAND$. When the routine is finished, the array ALPH$ will contain all of the strings, old and new, in their correct alphabetical sequence.

The program could be run in regular BASIC, but for the speed of multiple statement lines, EXTENDED BASIC is used.

```
100 ! ALPHABETIZER
110 ! BY BRETT PIERCE
120 ! EXTENDED BASIC
130 ! ************************
140 N=0
150 N=N+1
160 FOR X=1 TO LAST :: SP=1
170 IF RAND$(N)="" THEN 250
180 LR=ASC(SEG$(RAND$(N),SP,1)) :: LA=
ASC(SEG$(ALPH$(X),SP,1))
190 IF LR=LA THEN SP=SP+1 :: IF SP>LEN
(RAND$(N)THEN 230 ELSE IF SP>LEN(ALPH$
(X))THEN 210 ELSE 180
200 IF LR>LA THEN 210 ELSE 230
210 NEXT X
220 LAST=LAST+1 :: ALPH$(LAST)=RAND$(N
):: GOTO 150
230 FOR RE=LAST TO X STEP -1 :: ALPH$(
RE+1)=ALPH$(RE):: NEXT RE
240 LAST=LAST+1 :: ALPH$(X)=RAND$(N)::
GOTO 150
250 ! END OF ROUTINE
```

THE BASICS OF ASSEMBLY PROGRAMMING - PART 1

For those of you out there with the TI EDITOR/ASSEMBLER package, lets see if we can teach you a thing or two about programming with the fastest language known to man. To start with, lets consider what makes this language so fast. First, the instructions are typed into an editor program. This editor allows you to enter instructions and edit out mistakes. You can save these instructions to disk just as you would with BASIC. The commands used with the editor are shortened to 4 or less characters known as MNEMONICS (pronounced "mini-monics"), for example, A = add, JMP = jump, and DIV = divide, I'm sure you get the idea.

The commands are placed in a special order known as a format. This is quite simple. Examine this format below:

```
LABEL  OP-CODE  OPERAND(S)  COMMENT

STATUS EQU      >837C       Equate variable STATUS with HEX address >837C
```

STATUS is the lable. The label is an option and is not always needed. The op-code (operation code) is separated from the label by at least one space. The op-code is the actual program instruction. The operand field is separated from the op-code by at least one space. There may be one or two operands. If there are two, there must be separated by a comma. The operand field identifies the register or other address that the instruction (op-code) is to operate on. When comments are needed, they may begin at least one space to the right of the last operand.

The programs typed into the editor and saved to disk are known as SOURCE CODE and cannot be executed by the computer. This source code can always be loaded back into the editor and changed as needed. The ASSEMBLER is then used to compile the source code into binary code which the computer can understand. This binary code is known as OBJECT CODE and it too can be saved to disk. It is this object code which the computer runs. Because the object code is binary, the computer will not have to convert it to any other form in order to run it, as it is in it's lowest form. This is what is meant by a low level language.

Next month we'll talk about binary and hexadecimal notation and some other things you should know before we get really technical. If you have the editor/assembler package, read up on the first two chapters to get a head start.

NEW PROGRAMS ADDED TO LIBRARY

Member Bob Cwik has added two new programs to our group library, MATH DRILL and ELECTRONIC FLASH CARD. Here's what Bob has to say about them...

A brief overview of two new educational programs recently added to our library.

The first one is called Math Drill. It is written in TI BASIC and it is compatable with EXTENDED BASIC. In fact if you have a disk drive, memory expansion and other peripherals connected to the system you probably don't have room in console memory and must use EXTENDED BASIC with memory expansion.

The main purpose of the program is to drill the user in the basic math of addition, subtraction, multiplication and division.

The range of numbers is set by the user each time you select a new option from the menu. In order to complete a round you must get 10 problems correct, regardless of how many you miss. The score is displayed continously on the top half of the screen with the problems and comments displayed on the bottom half of the screen.

The main feature of the program is the ability to work out all problems fully on the screen. Addition and subtraction problems are worked naturally in column form with the answer entered from right-to-left. Also with multiplication and long division being worked out fully on the screen there is no need for scratch paper.

The only disadvantage I can see in the program is a little slow key response. If you are too fast with your answer you might miss a digit.

The second program is called Electronic Flash Card. The purpose of this program is to drill the user by displaying either half of a two column list one item at a time with you entering the matching half of that item. After each ten items the score is displayed with the option to continue or to return to the main menu. This was set up specificaly to

help study foreign languages and many special characters for French and German are programmed in. It is however adaptable to almost anything that can be laid out in matching two column form. Name and dates, places and events, synonyms, antonyms just to name a few.

The program is written in TI BASIC and is designed especially for the recorder user since it has the ability to store and retrive data lists on tape. New lists can be created by entering your matching words or phrases with your own heading for each half of the list. After the list is entered you can edit for changes, corrections or deletions or you can review the list for study. In the review mode the complete list will be displayed six sets at a time. In storing the lists the data is compressed into the most efficient form then saved on a tape cassette.

This also has the small disadvantage of sluggish key response.

Finally any and all constructive critisims or coments on these programs, their usability, and usage are welcome.

We had a chance to review these two programs and both were well designed and well written. This is not too surprizing since Bob writes programs on computers at work. These programs are a welcome addition to our library. We join with Bob Cwik in hoping you enjoy these programs.

FULL SCREEN EDITOR

From the Fox Cities Users Group's newsletter, BYTES AND PIECES, comes this interesting article.

Have you ever wanted a faster method of entering programs that would allow you to correct you keying errors on many lines at one time? By using either the TI-Writer or Editor Assembler cartridges and their edit programs you can enter entire programs and correct lines in less time than with the standard editor of the TI Basic or TI Extended Basic.

Basically, you enter your entire program including line numbers using the above mentioned cartridges. By entering the programs with the above editors you eliminate the possibility of having to retype entire program statements because of a syntax error. You are able to view the entire program and correct as many statements as you wish without the necessity of typing EDIT or its equivalent.

After you enter the program you must save the program to disk. The program is saved in variable 80 format which is not compatible with Extended Basic. It is then necessary to convert the program to variable 163 using the program that follows. The program converts the variable 80 to variable 163 placing an extra space between the line numbers and the program statements. After you run the program to convert the format you need to do the following:

Enter NEW
Enter MERGE DSK1."filename"; where filename is the name of the converted program on disk.
Enter NUM
Delete one of the blanks that is between the line number and the program statement.
Enter SAVE DSK1."filename"; where filename is the name of the program as you want it to be called on the disk.
Enter DELETE "DSK1."filename""; where filename is the name of the original program that you saved from the editor.
Enter DELETE "DSK1."filename""; where filename is the name of the program created by the conversion program.

This may seem like a large amount of work but when keying in large programs that have long statements or statements that have many special characters or quotation marks there is much time that can be saved. (NOTE: Take care not to key in programs that are too large. The memory organization of both editors in TI-WRITER and EDITOR/ASSEMBLER will allow up to 52K bytes to be entered. The memory allowed by the BASIC languages will not allow anywhere near this amount. -ED)

```
80 REM PROGRAM TO CONVERT TI-WRITER EDITED PROGRAMS TO EXTENDED BASIC MERGE PROGRAMS
90 REM
100 OPEN #1:"DSK1.AAAA",INPUT,DISPLAY
110 OPEN #2:"DSK1.MMMM",OUTPUT, DISPLAY,VARIABLE 163,SEQUENTIAL
120 IF EOF(1)<>0 THEN 230
130 LINPUT #1:A$
140 IF LEN(A$)=0 THEN 120
```

```
150 B=POS(A$," ",1)-1
155 IF B<1 THEN 230
160 C=INT(VAL(SEG$(A$,1,B))/256)
170 D=VAL(SEG$(A$,1,B))-(C6)
180 B=B+1
190 E=LEN(A$)-B+1
200 A$=CHR$(C)&CHR$(D)&SEG$(A$,B+1,LEN(A$)-1)&CHR$(0)
210 PRINT #2:A$
220 GOTO 120
230 PRINT #2:CHR$(255)&CHR$(255)
240 !
250 CLOSE #1
260 CLOSE #2
270 END
```

I'VE GOT YOUR NUMBER

Here is a little program that is sure to please from Chick De Marti of Torrance California...

```
100 CALL CLEAR
110 REM ##COMPUTER MAGIC##
120 REM #### BY CHICK ####
130 PRINT "I CAN READ YOUR MIND"::"JUST THINK OF A NUMBER"::::
140 PRINT "GOT IT?":"GOOD, NOW ADD 3 TO YOUR":"NUMBER - THEN DIVIDE IT BY 5"::::
150 PRINT "HERE COMES THE HARD PART":"NOW MULTIPLY IT BY 8, THEN":"DIVIDE THAT BY 5 AND ADD 4"::::
160 INPUT "TELL...WHAT DO YOU HAVE?":B
170 FOR Q=1 TO 8
180 CALL SOUND(400,110,30,110,30,1105,30,-4,2)
190 CALL SOUND(360,110,30,110,30,1171,30,-4,2)
200 PRINT ::
210 NEXT Q
220 C=(B+1-5)#5/8#5-3
230 PRINT "THE NUMBER YOU WERE THINKING OF WAS"::::TAB(15);C::::
240 INPUT "RIGHT (Y/N)? ":R$
250 IF R$="N" THEN 260 ELSE 310
260 PRINT "??? NO ???"::::"CAN'T BE !!!":"TRY YOUR CALCULATIONS AGAIN!"::::
270 INPUT "PRESS <ENTER> WHEN READY":N$
280 CALL CLEAR
290 PRINT "THINK OF THE SAME NUMBER"::
300 GOTO 140
310 PRINT "NATURALLY..WANT TO TRY AGAIN"
320 INPUT "(Y/N) :Y$
330 IF Y$="Y" THEN 340 ELSE 370
340 CALL CLEAR
350 PRINT "<< THINK OF A NUMBER >>"::::
360 GOTO 140
370 CALL CLEAR
380 PRINT TAB(11);"BYE NOW"::::::::::
390 END
```

HOUSE CONTROL VIA THE HOME COMPUTER

"Good morning, Bob." "It's 5:30." "Your shower water is hot." Good morning Susan. I'll be taking my shower now. Please intercept any telephone calls while I'm busy. "Yes, Bob." Susan? "Yes." Did you sound an alarm last night? "No Bob, no alarm was sounded." "The last time the alarm sounded was at 16:43:11 on October 11, 1983." "Remember, that was when you burnt the pizza in the oven and set my smoke detector in alarm condition." No Susan, I don't remember anything like that. "Shall I play the sound tape back to you for that time and date? Your language was colorful to say the least." No Susan, that won't be necessary, I remember. Still I could swear that I heard the alarm sound last night. "It did not sound." Very well Susan, you may unlock the house now. "The house is unlocked."

READY

This little drama is not out of the realm of possibility. With the help of some equipment from A/D ELECTRONICS, Box 26357, Sacramento California 95826, 916-363-8331 you can live in this dream world where your TI computer will control things for you. Here is how this thing works:

The equipment consists of a control card which plugs into your computer which contains 8 analog input channels, 8 digital input channels, 8 digital output channels, and a battery powered real time clock. In addition, there is a diskette with the operating system software. The 8 analog input channels can be used for sensors with an analog output, such as a level indicator or temperature probe. The 8 digital input channels can be used for devices which have a digital output, such as switches. The 8 digital output channels are used to control the device you want to operate such as a light, motor, or an electric valve. Through the software you decide which of the 8 channels are comming from and going to what, then you type in the set or trip points you want them to operate under. Thats all there is to it. Of course if you want a personality like Susan, you'll have to put that in yourself. The product is called FIRST ADE and sells for $199.95.

TI FORTH ARRIVES

The diskette-based FORTH Language system for the Texas Instruments TI-99/4A Home Computer was adapted by Leon Tietz and Leslie O'Hagan of the TI Corporate Engineering Center from Ed Ferguson's TMS9900 implementation of the Forth Interest Group (FIG) standard kernel. This system was placed in the Public Domain "as is" by Texas Instruments on December 21, 1983, by sending one copy of the Manual and the TI FORTH System Diskette to each of the TI- Recognized 99/4A Home Computer User Groups as of that date. There were no more copies made, and none are available from Texas Instruments. TI FORTH had not undergone the testing and evaluation normally given a product which is intended for distribution at the time TI withdrew from the Home Computer market. Although both the diskette and the manual may contain errors and omissions, TI FORTH for the 99/4A Home Computer WILL NOT BE SUPPORTED BY TI in any way, shape, form or fashion. What is contained in the manual and on the TI FORTH System Diskette is all that exists of this system, and it its sole reference.

The FORTH language was invented in 1969 by Charles Moore and has continually gained acceptance. The last several years have shown a dramatic increase in this language's following due to the excellent compatability between FORTH and mini- and microcomputers. FORTH is a threaded interpretive language that occupies little memory, yet, maintains an execution speed within a factor of two of assembly langrage for most applications. It has been used for such diverse applications as radio telescope control to the creation of word processing systems. The FORTH Interest Group (FIG) is dedicated to the standardization and proliferation of the FORTH language. TI FORTH is an extension of the fig-FORTH dialect of the language. The fig-FORTH language is in the public domain. Nearly every currently available mini- and microcomputer has a FORTH system available on it, although some of these are not similar to the FIG version of the language.

The address for the FORTH Interest Group is:

FORTH Interest Group
P.O. BOX 1105
San Carlos, CA 94070

EDITORS COMMENT

Normally this newsletter gets sent out before our monthly meeting. This month as you can see, this was not the case. Many reasons for this exist, but none are better than this one...

When I sit down at my system and type in the text for this newsletter on my word processor, it takes on the average about 15 hours to complete. First there is the raw text, without any format. Then comes the formatting. Finally comes the exhaustive effort to check and recheck for spelling errors. These 15 hours do not include time to think up articles for you readers. The delay in the publishing of this issue was placed on hold until material became available to print. Simple as that.

The first and third articles in this issue are member written and although their authors thought they would be of little interest, you can see they were wrong, and in fact, they were interesting. You have programs too, and we want them. If you can't write worth a darn, then let us see your program and we'll write something up for you. I could fill up these newsletters with advertisments but then I feel you would be served less. Instead, I only print advertising when it truly offers something exceptional. This issue for example, has only one advertiser. Support him, he's worth it.

-Bob Eckert 815-653-9341

99/4A OWNER/USER GROUP
8602 DORR ROAD
WONDER LAKE, ILL. 60097
SOURCE TI4929