Radio Shack TRS-80™ Microcomputer NEWS

March, 1981

P.O. Box 2910, Fort Worth, Texas 76101

THE MICROCOMPUTER NEWSLETTER PUBLISHED FOR TRS-80 OWNERS

Volume 3, Issue 3

Fort Worth Scene

TANDY CENTER

Mr. Dave McGlumphy of Red Bank, TN. sent us a letter in Mid-December (and you are reading about it in middle March if everything remains on schedule, which illustrates our time delays). The letter had three basic parts, two of which seem to reflect the views of many of our readers, and one of which was a helpful piece of information for Model I users. The information for Model I users can be found on page 4 of the newsletter with other Model I/III information.

(Continued on page 2)

MORE COMPUTER CLUBS

TRS-80 CLUBS

Central Alabama Microcomputer Society
c/o Lewis E. Garrison
6375 Pinebrook Drive
Montgomery, AL 36117
1/205-272-8462

Dal-Clf Computer Club—TRS-80
c/o R. E. Smith
3716 Shady Hollow Ln.
Dallas, Tx. 75233
1/214-331-2665

Naperville Central High School
Computer Club
440 West Aurora Avenue
Naperville, Illinois 60540

PROCOMP
844 Vernon St.
Manchester, CT 06040

Texhoma Microcomputer Enthusiasts
Wichita Valley TRS-80 Users Group
P.O. Box 4391
Wichita Falls, Texas 76308

JOHNS HOPKINS LAUNCHES NATIONAL SEARCH—PERSONAL COMPUTING TO AID THE HANDICAPPED

The first national search for ideas and inventions through which the full spectrum of personal computing technology can be harnessed to assist the handicapped was announced recently by The Johns Hopkins University.

To be conducted by the Applied Physics Laboratory of the University, and with the National Science Foundation and Radio Shack, a division of the Tandy Corporation, as cosponsors, the effort will be highlighted by a national competition for ideas, devices, methods, and computer programs to help handicapped people overcome difficulties in learnings, working, and successfully adapting to home and community settings. Categories that may be addressed include computer-based aids for the blind, deaf and mentally retarded; for individuals with learning disabilities, neurological or neuromuscular conditions; and the orthopedically handicapped.

(Continued on page 2)

NEWSLETTER INDEX IN THIS ISSUE...

Johns Hopkins (from page 1)

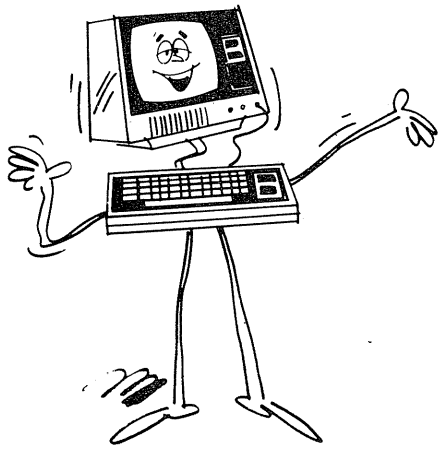
One hundred awards will be made, including a $10,000 grand prize, personal computer equipment, other cash prizes, computer training, and certificates of merit. Entries will be sought from computer specialists, full-time high school and college students, and from interested people generally, including those with handicaps. Regional and national awards will be made in all categories. National awards will be presented at a banquet in the fall of 1981 in the Washington D.C. area.

Paul L. Hazan, director of the Personal Computing to Aid the Handicapped project, said the competition is a challenge to the American people to use their conceptual skills in bringing forth practical aids based on computer technology that will help an individual or group of people with a handicap. "Just as important will be the opportunity provided the inventors and developers to make contact and form partnerships with the handicapped in a way that can lead to wide acceptance and use of the new computing technology," Hazan stressed.

Orientation meetings are being scheduled at major rehabilitation centers throughout the United States to bring together potential "inventors," handicapped people and professionals in habilitation-rehabilitation fields. Special presentations also will be made nationwide at chapter meetings of the Association for Computing Machinery (ACM), Institute of Electrical and Electronics Engineers (IEEE), and personal computer clubs.

Contestants will have from November 25, 1980 until June 30, 1981 to prepare and submit their entries.

To obtain additional information including a descriptive flyer and contest application, write to Personal Computing to Aid the Handicapped, Johns Hopkins University, Post Office Box 670, Laurel, Maryland 20810.



Ft. Worth (from page 1)

The first part of his letter dealt with the new subscription based newsletter. Here are his comments:

"The November Newsletter (which I received on December 2) says that you want $12/year for the Newsletter. I am somewhat grudgingly sending it, but I'm afraid that you won't send $12 worth of information compared to publications such as 80-Microcomputing. I hope that I'll be pleasantly surprised, After all, I'd feel really dumb paying you $12 to read your advertising. The Newsletter has a lot of potential. I hope it pays off, and I hope I can contribute to its value . . ."

Mr. McGlumphy's last paragraph pointed out an apparent discrepancy between our statement in the November Newsletter that ". . . our Model I factory . . . is running at full speed.", and other sources, including our own customer service people, at about the same time. When the article was written, the statement was true. Unfortunately, the long time between when we write our material, and when you actually read it, sometimes makes it appear that we are giving you bad information. We can assure you that it is our intention to always give you the best information we have AT THE TIME we write our material.

Hopefully, most of you who are reading this issue of the Newsletter have also seen the December-February issues. Over these last four issues, we have worked very hard to reduce the amount of "hard" advertising to virtually nothing. In the months before December, we had the "Product News" which was eight pages of hard advertising. As you know from this and other recent issues, we have eliminated this section. In its place we have the Product Line Manager's pages. We hope that you find these pages informative and useful.

We are not and will never be 80-Microcomputing or any of the other magazines. What we are is a source of direct and "reasonably" valid information which is intended to help you, our customers. I say "reasonably" because we do make mistakes, and plans do change. A prime example of this is the method of mailing the Newsletter. We told you in November that the Newsletter would be mailed first class beginning in January. We believed this and planned on it. As the January issue went to press, with the needed changes for first class mailing, we had to change and continue mailing third class. As a result, we will continue to mail free newsletters by third class. Paid subscriptions are being mailed first class. The information we gave you in November was the best available at that time.

For some of you who have been long time owners of TRS-80s this may be your last issue. If you have decided to not subscribe, tell us why! Wrong information? Not enough of a particular type of information? Help us serve you and others better by communicating with us.

We will say in advance, that there are certain things which we know you want: Advanced Product information, information on non-Radio Shack products, etc. These are topics which, in general, we simply can not address.

We are trying to give you a informative and usable newsletter about Radio Shack computers and computer-related merchandise. We do not accept outside advertising, and as of January we do not even carry our own ads. If you like what we are doing tell us. If you don't like it, tell us that too, and then go on to suggest ways that we can change to give you what you need. A simple "I don't like it." really doesn't help at all.

Notes on Previous Newsletters

Mr. Lewis E. Garrison of Montgomery, Alabama sent us this note on Budget Management:

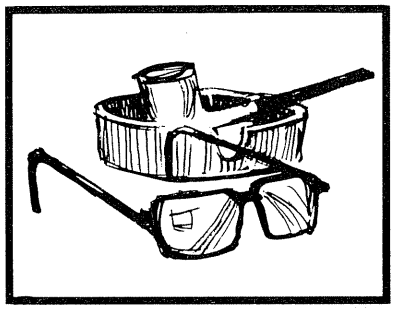
"In the November issue of the "TRS-80 Microcomputer News," you gave instructions on how to edit the Budget Management (26-1603) program for use on an 80 column printer. These instructions were easy to follow and worked fine. However the title and date now need to be centered. Here are the additional changes to the REPORTS program that need to be made to center the title and date of the reports:

"1. Edit line 3530. Change the "40" after the TAB to "26." Change the "25" in the STRING$ statement to 22."

"2. Edit line 3540. Change the "40" after the first TAB to "26." Change the "45" after the second TAB to "31."

"With these changes made, you will now have a professional looking output product from your 80 column printer . . ."

We would like to thank Mr. Garrison for pointing out this oversight on our part, and for providing the needed changes to make your printouts more attractive.



View From the 7th Floor

by Jon Shirley, Vice President Computer Division

I have received a lot of letters about our dropping the Model I and about the FCC, so let's revisit that issue one last time. The FCC regulations require that "home computers" (and they defined the term), manufactured after 1/1/1981 meet certain new standards for interference with TV sets, radios etc. (You Model II owners can skip this part as the Model II is not covered by these regulations but by another set that also covers big and huge computers and does not take effect for a while.) The key word is manufactured — computers made before January 1 can be sold and, with the normal inventory situation, I doubt that very many FCC certified computers have been actually purchased by now.

The Color Computer had to have specific certification, to a different standard, because it attaches to a Color TV, before we could sell any so it has always been certified. When we finally got the new regulations, and it took a long time to get them and then figure out what they really meant, we started to see what would have to be done to each product to meet the requirements. To make a long story short we discovered that it took many, many engineering man hours, outside consultants, special very expensive test equipment, an outside test site that is usually in the middle of a farm and many, many design changes to the computer. It took new circuit board layouts, metal shields, added capacitors, shielded ribbon cable (very expensive) plus a little black magic to meet the requirements. Shortly after I wrote my November column it became obvious that there was no way to make the Model I meet the requirements without a total redesign. Since the Model III was, in effect, a totally redesigned Model I, our decision became obvious also.

In December we received FCC certification for all versions of the Model III and started producing the FCC certified version in January. Now in case you think I am trying to make the entire FCC deal look like a bigger problem than it was, the trade press reported that Apple® requested 3 months additional time to meet the requirements for the Apple II® and Heath® (Zenith®) asked for 6 months and they are in the TV business! The FCC gave them both 3 months to get certified or stop production. So for those of you who have asked for some sort of retrofit kit because your Model I is interfering with your TV set, all I can suggest is that you turn off the TV or the Model I. Seriously, if you do have that problem the best solution is to increase the TV set signal. Put up a good new outside antenna with quality lead-in. If your set is old, especially if it is a tube set, have it serviced and aligned.

A lot of your letters asked about the future of the Model I peripherals. The FCC granted us the right to continue to sell the expansion interface and disk drives for the Model I to support the current Model I owners. Those, and almost all other Model I items will continue as long as there is demand. Software? Virtually all the new software in the works for the Model III will be Model I compatible so you will see lots more software including, finally, a good low cost disk editor/assembler.

Another subject that generated a lot of mail was our decision to go subscription for the newsletter. All such letters got the same reply. In fact we had so many we used a form letter on Scripsit and a Model II. Here are the reasons:

1. About 0.8% of the U.S. population moves every month. Or almost 10% per year. People also die, sell their TRS-80 or just don't want the Newsletter. Subscriptions give us a way to keep the mailing list clean and not just forever growing larger and larger.

2. By charging for the subscription we will be able to mail subscribers by first class. This will insure that those who want the Newsletter will get it and quickly. The first free year will *not* be by first class but if that is important you may subscribe early.

3. As you have noticed, the Newsletter is no longer an advertising piece. We will announce new-product availability that we feel is of interest, but that will not be the reason for the Newsletter.

By the way, our largest two competitors in small computers do the following: one has no newsletter and the other charges from day 1. So you get ours free for one year and have plenty of time to decide if you like it. If you don't, don't subscribe; it's still a free world, at least in the USA.

For all you Model II owners who have, by now, been totally confused by 1.2a and 2.0a, many thanks for your patience. I would like to tell all the TRS-80 owners about what happened as it illustrates just how crazy the computer biz is. I'm sure most of you know that the microprocessor has a lot of logic in it to interpret the commands it receives. You may not know that a lot of other chips also have imbedded logic or microcode and are also processors of a type. For example, in the Model II, the keyboard, video, serial ports and the disk controller are all run by their own intelligent processor IC's which are big chips called LSI devices. These are very complex devices, some much more complex and expensive than the Z80A microprocessor.

As with all complex IC's, there is always the chance that some of the code is incorrect and that the IC will not, under some circumstances, do what the manual says it will do. If you start to get the idea that microcomputers are designed by the semiconductor manufacturers, you are not far wrong. Anyway it seems that the disk controller IC in the Model II had a very obscure fault. By very obscure I mean that none of our software ever caused the fault to be found in over a year of Model II production, nor did any other users of the same IC find the fault. Enter Model II Accounts Receivable (26-4504), our first COBOL program. During many hours of testing it always worked here but, once it got to you, all of a sudden the Model II started producing an error message "sector not found." What that means is that the computer can't find the data it wrote out to the disk. Result here, PANIC! To compound the problem our engineers and software people, who worked a lot of overtime on this one, found that the data was on the disk but the controller for some reason could not find it.

After several days we had to conclude that TRSDOS was not at fault, nor was our COBOL, it was the Model II itself and most likely the disk controller IC. At this point our supplier of the controller IC sent a crew down to Fort Worth with orders not to return until the problem was solved. Well they found the problem and announced that, sure, they could remask the chip and make new ones — only it would take about 6 months. They then sat down with our system software people and worked out a software fix for the hardware problem and we released TRSDOS versions 1.2a and 2.0a. The final decision on what to do and how to do it occurred December 24, late in the after-

(Continued on page 4)

View 7th (from page 3)

noon and long after we all had promised to be home. Well no one ever said Murphy's law respected Christmas.

For those of you who do not own Model II's, you might be interested to know that we mailed every owner a new Accounts Receivable disk and made the "a" versions available, free, to every owner. And those of you who had to wait for your new disk drive Model III (and I hope you have it by now), might be interested to know that the wait was partly caused by another version of the same disk controller that had its own problem and it did require that the chip be replaced.

So if any of you want to go into the microcomputer business, do it with your eyes open; there are a lot of mine fields out there. Until next month.

PRINT@ Problem

Mr. Dave McGlumphy sent us this information about a problem he had running BASIC programs on two different Model I's. Here is that portion of his letter:

"... Several months ago, I 'upgraded' my Model I 16K Level II to the 'NEW' ROMs from the 'OLD' ROMs. I subsequently wrote a couple of BASIC programs which ran fine on my machine but wouldn't fly on machines with the 'OLD' ROMs because of a syntax error on PRINT@ statements which occurred when I used a SHIFT @ accidentally. SHIFT @ was OK on the 'NEW' ROMs but not acceptable on the 'OLD' ROMs. I could have gone through the program and manually fixed each syntax error as it occurred, but since this particular program had quite a number of occurrences, I used the following statement to fix all the errors at once:

```
FOR J=17129 TO
    PEEK(16633)+
    256* PEEK(16634):
    IF PEEK(J)= 178 AND
    PEEK(J+1)= 96 THEN
    POKE J+1,64: NEXT
    ELSE NEXT
```

"Notice that there is no statement number because this one-statement program is to be executed immediately just one time. 17129 is the address of the beginning of the BASIC program with the errors. 16633 and 16634 contain a pointer to the end of the BASIC program, so the FOR/NEXT loop starts PEEKing at the beginning of the BASIC program and continues to the end of the BASIC program. 178 is the token for 'PRINT,' and 96 is the decimal value of the ASCII @ when it is shifted. 64 is the decimal value of the unshifted @, and since that is what I really wanted in my program, that is what I POKEd into memory..."

Double Precision Square Root

Here is a letter from George R. Mabry about a double precision square root routine. I tested the routine and found that I consistently got 15 digits of accuracy.

"When taking the double precision square root of non-perfect square numbers with my TRS-80 Model I system, it was discovered that the result was accurate to only about 7 or 8 decimal places (Note: SQR is a SINGLE precision function. Ed.) The enclosed subroutine, however, will yield a double precision square root that has no more error than 1 part in 16 decimal places.

"The subroutine is based on the following equation:

(1) $(B\# + X\#)^2 = A\#$ where $B\# = SQR(A\#)$

Expanding equation (1) gives:

(2) $((X\#)^2 + (2* B\#* X\#) + ((B\#)^2 - A\#)) = 0$

The application of the quadratic equation to (2) gives the following:

(3) $X\# = (SQR(A\#) - B\#)$

The computer solution of (3), however, yields a value of 0 for X# since the two terms on the right side of (3) are identical. If equation (2) is rewritten as follows:

(4) $X\# = (((A\# - (B\#* B\#))/ (2* B\#)) - ((X\#* X\#)/ (2* B\#)))$

then X# can be computed by successive approximation. The first approximation is:

(5) $X\# = ((A\# - (B\#* B\#))/ (2* B\#))$

The final approximation is:

(6) $X\# = (X\# - ((X\#* X\#)/ (2* B\#)))$

The final square root is:

(7) $B\# = (B\# + X\#)$

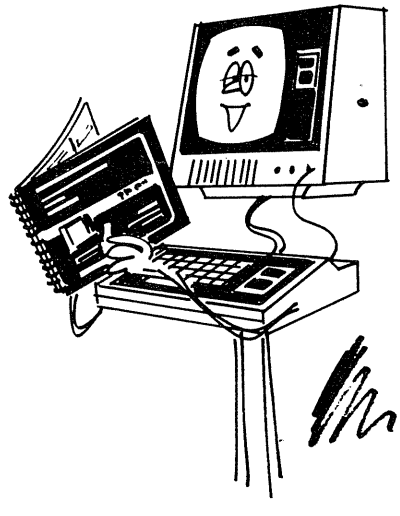
Here is the program:

```
1000 REM* DOUBLE PRECISION SQUARE ROOT *
1010 REM* INPUT A# , OUTPUT B# *
1020 REM* SCRATCH PAD VARIABLE - X# *
1030 IF A#<>0 THEN 1050
1040 B#= 0: RETURN
1050 B#= SQR(A#): X#= ((A#- (B#* B#))/ (2* B#)):
     X#= (X#- ((X#* X#)/ (2* B#))): B#= (B#+ X#)
1060 RETURN
```

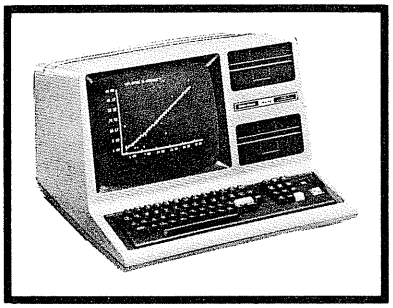
Model I/III Graphics Routine

Here is a Graphics routine for a Line Printer II, sent to us by Alan Jeffery Joseph of Fairhaven, MA:

```
10 CLEAR 100
20 FOR Y=0 TO 43
30 FOR X=0 TO 79
40 IF POINT (X,Y) THEN
   B$= B$+ CHR$(127) ELSE
   B$= B$+ " "
50 NEXT X
60 LPRINT B$
70 B$=""
80 NEXT Y
```



Mr. Joseph comments that "With this program all graphics in the first 80 spaces of each line go to the LINE PRINTER 2, ..."



Model I/III

Product Line Manager's News

The crew in the back room who designed the enhancements to Model III TRSDOS and Disk Basic didn't take our "biggest name in little computers" trademark too seriously. They keep adding "big computer" (synonym is costly) features that make our little computers think big (but cost less).

Radio Shack®——The biggest name in little computers™

These new commands can be used to design a business application that is "friendly" for an operator. For example, they can help with error situations (ERROR), protect disks and programs from incorrect commands (WP), offer assistance (HELP) and more. Whether you are writing a program for entertaining friends, for sale in the market place, for your business or someone else's you will appreciate these tools to help "bomb proof" your programs from even the most novice of users. Some of the newest additions are great tools to make programming faster, and more fun.

Let's finish last month's discussion of the additions to DISK BASIC first, and then begin with TRSDOS. CMD"A" is new and is similar to CMD"S" which returns to TRSDOS from BASIC, however it also displays an "OPERATION ABORTED" message to let the user know what is going on. CMD"B" let's you protect a program from being interrupted by the BREAK key. The BREAK key is ignored unless cassette, printer or serial I/O is underway. Use CMD"C" to compress your program and save disk space by deleting your choice of all remarks, all spaces or both. Try CMD"D" to examine the directory for any specified drive without exiting BASIC. It works for unprotected, non-invisible files. If a TRSDOS error condition occurs use CMD"E" to display the last error and remain in BASIC. CMD"P" returns a "STATUS" byte whose value depends on the type of printer in use and its status at the moment. Use CMD"P" to test for a printer that is busy, or out of paper and either stop the program or tell the operator to regain control with the BREAK key.

Here is an example of Model III's LIB display of TRSDOS commands. . . . . .

APPEND	ATTRIB	AUTO	BACKUP
BUILD	CLEAR	CLOCK	CLS
COPY	CREATE	DATE	DEBUG
DIR	DO	DUAL	DUMP
ERROR	FORMS	FORMAT	FREE
HELP	KILL	LIB	LIST
LOAD	MASTER	PATCH	PAUSE
PROT	PURGE	RELO	RENAME
ROUTE	SETCOM	TAPE	TIME
WP			

ERROR — You can pay extra for a "big" computer and big reference documents to look up the meaning of error codes or with our big "little" computers simply type in ERROR xx where xx is any of the 41 TRSDOS error codes for an explanation.

ROUTE — We talked last month about initiating I/O routing from MODEL III BASIC. Use it from TRSDOS also, for example, to route keyboard input directly to a printer.

CREATE — Increase the speed of an application that is dynamically allocating disk file space by using this command to first create and then pre-allocate a file for the size you need. Specify the logical record length based on the number of characters and the number of records needed.

FORMS — Set printer parameters for line width to provide for continued printing on the next line when a print line exceeds the width of your printer, and set the lines per page counter. The lines per page and number of lines printed information can be used by your application to handle Top-of-Form, etc.

MASTER — If your application consistently accesses a drive other than drive 0 then assign that drive as the Master Read or Write drive to speed up file searching. Then all file searches will begin with the drive you have specified.

SETCOM — For RS-232C serial communications use this command to initialize the RS-232 to the word length, baud rate, number of stop bits, and parity for your communications needs. We have included the listing of a simple terminal program in the manual for you to get started with or you can purchase our RS-232 Communications Software Package (26-1149) or Videotex (26-2220).

PATCH — There is no need to reassemble your machine language programs when you want to make a change, this command lets you alter disk programs or data files directly. If you are programming in machine language you will wear this one out. Otherwise chances are you won't use it unless we release modifications to TRSDOS, add a driver routine or make available some other routine. If we publish a change for use with the PATCH command then there will be no need to obtain another media and possibly transfer files to take advantage of that particular modification. In these cases we will provide instructions on how to implement the change.

TAPE — If you have added a disk to your system and want to use the software you have or you are still going to be working with tape on your Model III Disk System then use this utility to transfer machine language program files from Disk to Tape, Tape to Disk, or Tape to RAM. Simply specify the source and destination choices. As an example TAPE (S=T,D=D) would transfer a program from tape to disk.

HELP — This automated Model III TRSDOS reference card responds with the syntax formats, definition of the TRSDOS commands, and explanation of the abbreviations. If you can't remember the command you need then HELP will default to a list of all available TRSDOS commands.

WP — Can't find a write-protect tab? Have your application start by executing a DO file (I will cover this capability later) to call on this command first off. The drive you specify will be software write-protected. This works for any one drive and will not leave to chance your directions to a user to protect his program or data files with a write-protect tab.

CLEAR — Depending on the options you select, this command will:

- Zero user memory
- Clear the display
- Un-protect all user memory
- Reset the stack
- Reset I/O drivers

PURGE — Quick and accurate deletion of multiple files is easy with PURGE. Filenames will be displayed one at a time with a (Y/N/Q) ? option.

DUAL — This is great for debugging software, creating an application audit trail, or documenting your programs. All operator dialog is captured by copying video output to the printer. If you ever find an application bug (won't ever happen) use it to document what is happening so we can better understand your problem.

RELO — Loading a program into a different memory address is as simple as specifying the file and the hexadecimal number referring to the new address.

CLS — What can I say? It clears the display. It would be used for example to make sure that someone doesn't see a password you have just entered.

(Continued on page 6)

Model I/III (from page 5)

There are some new TRSDOS commands left to cover including the DO, and BUILD capability, other TRSDOS and Disk BASIC information, lots of news, software applications and languages to tell you about. Most affect both Model I and Model III users so keep in touch, I will start on these topics next month.

Model I/III Bugs, Errors and Fixes

In-Memory Information (26-1508)

In version 3.0 (for Model I and III), when entering data (cards) and approaching the end of available memory, all data will be lost if another card is entered and the "BYTES FREE" is less than TWICE the card length.

The program will accept the information, but then display the number of bytes free for about three seconds, indicate "OUT OF MEMORY" and execute a total re-start, as if beginning the program from scratch.

On page 7 of the documentation, there is a note that states the following:

Note: If the BYTES FREE value drops below the CARD LENGTH value you can't add any more records.

Please change this line to read:

Note: If the BYTES FREE value drops below two times the CARD LENGTH value, you can't add any more records; to do so will result in loss of all data (execution of (R)ESTART).

Accounts Payable (26-1554)

During End-of-Period processing (in VERSION 3.0 only), an error code 5 may occur in line 45. To prevent this, change lines 45 and 151 of the "PROCESS" program to read:

```
45 W#=ABS(N#)*100:W#= INT(W#* 100D0+
   .5D0)/ 100D0: V$= "": X=W#/ D1#:
   W#=W#- X* D1#: V$= V$+ CHR$(X- (N#<0
   )* 128): X=W#/ D2#: W#=W#- X* D2#:
   V$=V$+ CHR$(X): X=W#/ D3#: W#=W#- X*
   D3#: V$=V$+ CHR$(X)+ CHR$(W#): RETURN
151 PRINT: PRINT CHR$(30); "UNEXPECTED
    ERROR CODE"; ERR/ 2+ 1; "IN LINE";
    ERL: GOSUB 65: END
```

Accounts Receivable (26-1555)

If you are using the 3.0 VERSION of accounts receivable under the three drive option, you will encounter a BAD FILE MODE error in line 628. To prevent this error from occurring, the following changes need to be made to the "SETUP" program:

Change lines 220 and 880 to read:

```
220 FL=1: GOSUB 280: R$= IN$: IF CF<>0
    THEN 220 ELSE IF R$<>"Y" AND R$<>
    "N" THEN PRINT CHR$(8);: GOTO 220
880 PD=2: PC=500: PT=2500: IF Q$="M"
    THEN ON ERROR GOTO 895: KILL PT$:
    PT$= LEFT$(PT$,LEN(PT$)-1) +"2":
    CLS: PRINT@458, "INSERT DATA DISK IN
    DRIVE 2 AND PRESS <ENTER>": ELSE
    GOTO 890
```

Notice that we removed the end of line 220, and added a statement to the end of line 880. You also need to add the following line:

```
225 IF R$="N" THEN GOSUB 560: IF Q$="I"
    THEN ZX=0: GOTO 80
```

Also in Version 3.0, there is an extra quote mark (") at the end of line 1590 in the "ARS" program. This extra quote can cause an error code 2 to occur during posting. If you will eliminate the last quote mark in line 1590, you will eliminate this error.

Business Mailing List 26-1558

The following procedure will allow you to recover and re-index Mailing List data, in versions prior to 3.0 only, which has been lost due to a system failure or an abnormal program exit. Note: this procedure will also "recover" items which were deleted if the space was not re-used by the program.

1) Enter the following program module:

```
4999 'RECOVERY
5000 CLS: N=0: TN=1: F=0
5005 PRINTTAB(20) "** FILE RECOVERY **"
5010 FOR Q=0 TO CP: V(Q,0)=0: V(Q,1)=0:
     NEXT
5015 PRINT@320,"** RECORD NUMBER :"N
5020 N=N+1: J=N: GOSUB 1840: GOSUB 1900:
     GOSUB 1910
5030 IF E1$< CHR$(32) OR E0$> CHR$(127)
     THEN 5100
5040 NS=1: GOSUB 3210: TN=TN+1: EL$=E1$
5060 IF N<CP THEN 5015
5100 GOSUB 1790
5110 PRINT: PRINT"> RECOVERY COMPLETE -
     PRESS <ENTER> ";
5120 GOSUB 1460: GOTO 210
```

2) Save the module using the following BASIC command:

```
SAVE"RECOVER/ASC",A (ENTER)
```

3) Type: `LOAD"MLS"` (ENTER)

4) Type: `MERGE"RECOVER/ASC"` (ENTER)

5) Type: `RUN` (ENTER)

6) When the Mailing List Menu appears, press the (BREAK) key.

7) When the screen shows "READY", type: `GOTO 5000` (ENTER)

8) The system will examine your data files and retrieve, resort and index all the valid data found. When this process is complete, the screen will show: `"RECOVERY COMPLETE - PRESS ENTER "`, Press (ENTER)

9) The Mailing List Menu will appear. Press the @ key to exit the program immediately.

10) Make BACKUP copies of your "recovered" disks. Examine the data carefully before copying onto your original disks. If the data is not correct, try the operation again.

Tiny Pascal (26-2009)

You should be aware that in mathematical calculations the largest value you can work with is 32767. There are two corrections that need to be made on page 9 of the Tiny Pascal manual:

OUTP(a,x) Outputs the value x to port a. This is the opposite of what the manual says.

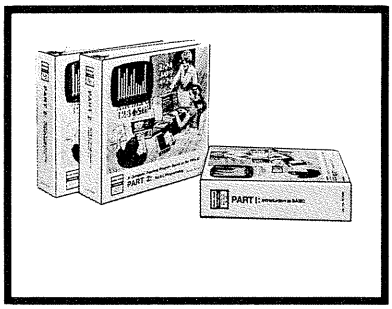
SQR(exp) returns the square of exp, not the square root.

FORTRAN—Model I (26-2201) and Model II (26-4701)

The FORTRAN User's Manual (Page 19 for Model I, Page 9 for Model II) indicates that the default logical record length (LRL) for a CALL OPEN statement is 128 bytes. This is incorrect.

If a programmer does not specify a LRL in the CALL to the OPEN subroutine, no error message is generated by the compiler, linker, or run-time package, and the LRL which is produced is unreliable and may vary from run to run. The format of the records written to files opened with no specified LRL is unreliable.

Solution: ALWAYS specify a LRL when using the CALL OPEN statement.



Education

Educational Product's News

This month's article for the Educator's page is written by Dr. Lee Droegemueller, Superintendent of Independent School District 196 in Rosemount, Minnesota, a suburb of Minneapolis. Dr. Droegemueller has been an advocate of microcomputers for computer literacy for all students as well as administrative applications. Under his leadership ISD 196 has been a frontrunner in developing and implementing new applications of microcomputers in education.

ENROLLMENT REPORTING USING VISICALC™

DR. LEE DROEGEMUELLER
SUPT. OF SCHOOLS
ROSEMOUNT, MINNESOTA

The Rosemount School District covers approximately 110 square miles of south suburban Minneapolis and St. Paul, Minnesota. The district has a total of ten elementary schools, three middle schools and two high schools. It is located in a fast growing area, with student enrollment increasing from 8,500 in 1975 to 12,500 in 1980. This growth has caused the opening of six new school buildings since 1976. The growth and additional schools resulted in attendance boundaries being changed each year. Classroom enrollment also increased during the year, with the result that rooms had to be divided at midyear.

One of the major problems caused by this continuing growth has been that the pupil/teacher ratio increases significantly during the school year. At the beginning of the school year, the administration allocated staff positions for these increases, and when the pupil/teacher ratio dictated, these new staff positions were to be filled. To reassure the Board of Education, the public, and staff that overcrowding would be promptly detected and alleviated, a timely, concise, accurate report had to be developed to allow us to monitor the pupil/teacher ratio.

The information needed at the elementary level was different than that needed at the secondary level. At the elementary level statistical data was needed by grade which included pupil/teacher ratios, change data, and summary information for each building. Only summary data and changes by building were needed and reported at the secondary level. Summary data on student enrollment for the district was accumulated by individual grade. It was further consolidated by elementary, middle and high school. A special pupil weighting formula was produced which related to budget reports.

Attempts at doing this report with a calculator, paper, pencil and typewriter were frustrating. There were 600 combined entries and calculations. To further complicate the task, kindergarten children could only be counted as half because they attended school only half days. Grades seven through twelve had to be given a factor of 1.4 for state aid revenue projections. With these and other factors involved, the probability of a typing or calculating error was always quite high. Many hours were spent each month preparing this report and the question of accuracy always remained.

The overview of the VISICALC program from Radio Shack stated that it addressed problems which needed paper, pencil and the calculator to solve. Due to the matrix configuration and the previously stated conditions, VISICALC appeared to be a natural solution to the problem. The ease of use of the manual, and the computer data displayed on the monitor so that it could be seen just as it would be printed, were a great help. Three aspects of VISICALC which made solving this problem easy were the ability to set row or column recalculation, the automatic or manual recalculation option, and the order of recalculation. The problem was reworked several times so that more information could be reported and memory use could be reduced.

The new report provided information on elementary school enrollment, including the grade, number of students, number of teachers, average pupil/teacher ratio by class, monthly change of the number of students by grade and summary data by building. Similar detail was required for the secondary schools. District summaries included changes in enrollment by grade, the calculation of weighted pupil units for budget projections analysis and total enrollment summaries. Sample output is shown in Figures 1, 2 and 3.

Figure 1.

*	CEDAR PARK			
*GRADE	STDS	STAFF	AVE	CHG
*				
* KDG	85	1.5	28.33	6
* ONE	84	3	28.00	0
* TWO	81	3	27.00	2
* THREE	98	3.5	28.00	6
* FOUR	122	4.5	27.11	-1
* FIVE	113	4	28.25	6
*				
* TOTAL	583	19.5	27.72	19
* SEP80	550	19.5	26.23	33

Figure 2.

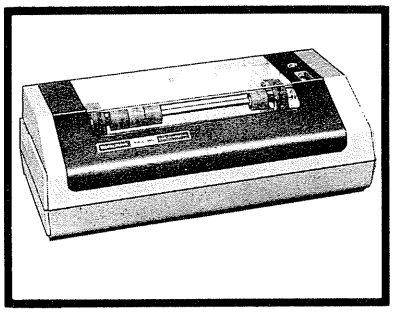
SUMMARY ONE YEAR			
GRADE	1980-81	SEP 79	CHG
*			
* KDG	978	922	56
* ONE	951	900	51
* TWO	973	948	25
*THREE	980	1108	-128
* FOUR	1169	1150	19
* FIVE	1192	1017	175
*			
*TOTAL	6243	6045	198
PUPIL UNITS	(PU)		
GRADE	1980-81	SEP 79	CHG
*			
* KDG	489	461	28
* ONE	951	900	51
* TWO	973	948	25
*THREE	980	1108	-128
* FOUR	1169	1150	19
* FIVE	1192	1017	175
*			
*TOTAL	5754	5584	170

There is a great potential for additional applications of VISICALC in school reporting. In this particular situation the elimination of several hours of work, the increase in reports generated, and the neatness of formatted printer output all underscore the convenience of VISICALC. The VISICALC enrollment report has reduced the number of questions at Board of Education meetings, made comparisons easier for the public, helped principals answer questions on class size, and improved administrative decision making.

Figure 3.

DISTRICT 196 SUMMARY

*	1980-81	SEP 79	CHG	80-81PU	SEP 79PU	CHG	*
* TOTAL K-5	6243	6045	198	5754	5584	170	*
* TOTAL MIDDLE	2952	2682	270	3718	3374.4	343.6	*
* TOTAL SENIOR	3267	3196	71	4573.8	4474.4	99.4	*
* GRAND TOTAL	12462	11923	539	14045.8	13432.8	613	*



Peripherals

Product Line Manager's News

This month I will keep the promise I made last month but with a little hedging. So much is happening that there will be room this month to outline only a part of the promised new comprehensive printer code standard (What I will show you is a portion of the Graphics control standard). I'll start, in great excitement, with something entirely new. We will be announcing a new printer soon: Line Printer VII. If the name isn't surprising, the printer is! This machine is Radio Shack's first dot-matrix machine to offer full graphics capabilities.

Designed with the color computer in mind, the unit includes the standard parallel interface for use with Models I, II, and III and features, in addition, a four pin DIN connector for use with the serial port of the Color Computer. The price of this little gem (26-1167) is only $399.00!

This is a full performance printer; no narrow or strange paper for this fellow. It prints 8 inch, 80 column lines on regular 9½" tractor paper (26-1423 — $7.95). Here is a run-down on its features:

Graphic or alpha-numeric characters can be intermixed on the same line

5 x 7 matrix characters or 7 bit dot addressable graphics patterns

Switch selectable input — parallel or serial (7 or 8 bit — 8 bit required for graphics)

Adjustable tractor 4½ to 9½ inches

6 or 9 (graphic mode) lines per inch

30 CPS speed

The printer uses a unique single hammer system. The carriage movement, the motion of the hammer, and the revolution of a ridged platen accomplish all the printing wonders.

Line Printer VII will produce all the ASCII standard alphanumeric characters. Once set in the graphic mode, however, any code except those for special control and graphic information will be ignored. Special commands move the carriage to any of the 480 dot columns available in the 8" line. The impression produced by the printer in any dot column is determined by sending an 8 bit binary code. The 8th bit (most significant) signifies graphic information. The other seven bits (set to "1" or "0") correspond to the column pattern placed on the paper. Another command allows repetition of patterns for code economy.

In this manner, the programmer can control all the dots produced on the page to a density of 3780 dots per square inch!

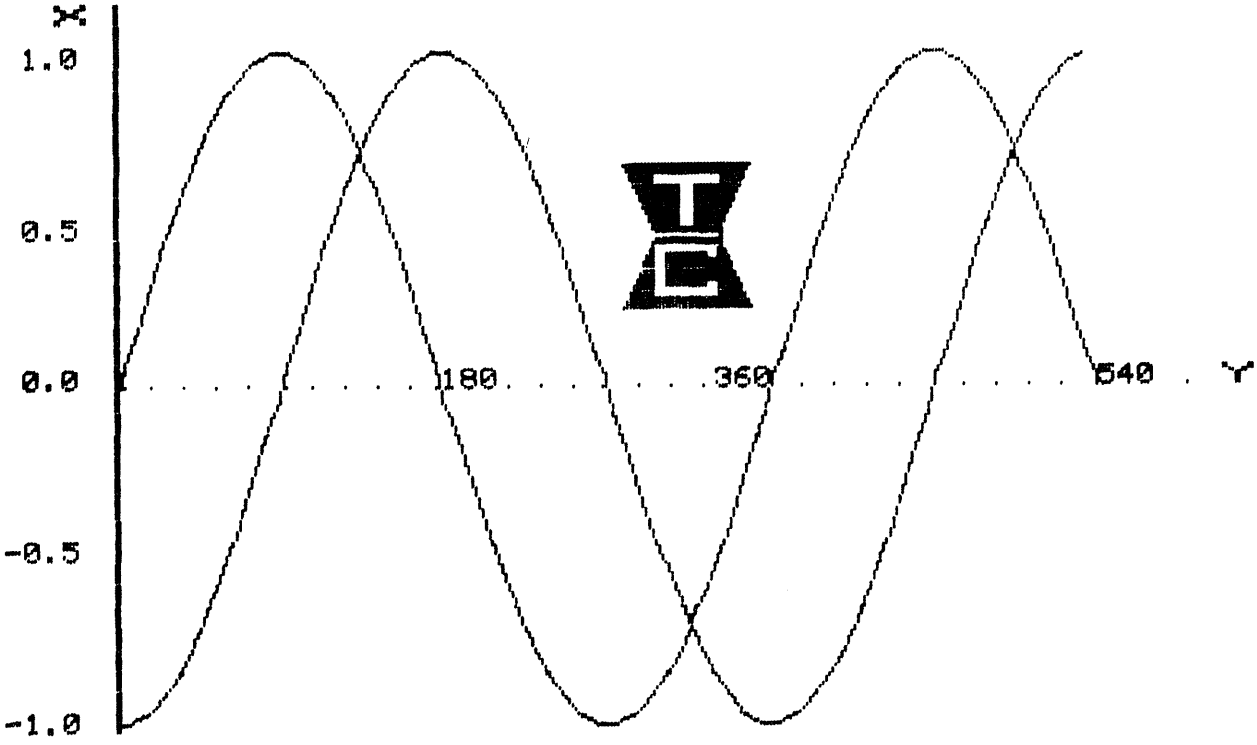
By the way, the current version of the Color Computer produces 7 bit output. Radio Shack will provide a special driver program which will allow the Color Computer to send the 8 bit characters, required for graphics, to the printer (700-2013).

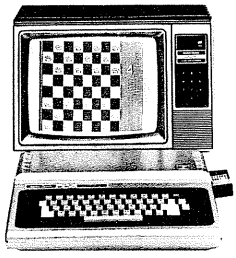
Here are the Line Printer VII graphics codes in the Radio Shack Standard:

DEC	HEX	FUNCTION
18	12	Set Graphic Mode
10 13	0A } 0D }	Normal Line Feed (CR plus LF) prints buffer and generates new line.
26	1A	C/R — Prints buffer and issues CR but no LF.
28,N	1C,N	Repeat the next dot pattern N times.
27,16,BB	1B,10,BB	Move carriage to dot column BB (9 bit value).

(Continued on 18)

Full size sample of LP VII graphics

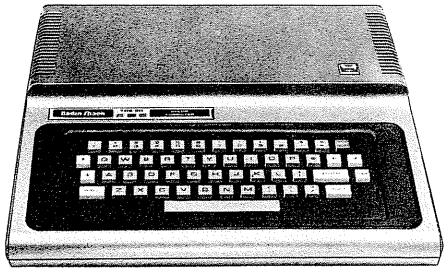




Color Computer

Product Line Manager's News

By now, I should hope that you have read, seen, heard, or talked about the "new" kid on the block. That's right, the TRS-80 Extended Basic Color Computer (catalog no. 260-3002 $599.00). (Looks just like the standard color computer, doesn't it. Well, what you can't see is what makes it so fantastic!) If you haven't, I'm going to tell you a little about it now....



As is the format of this column, I want to clear up some of the errata before we proceed. The TRS-80 Computer Catalog No. RSC-4 was a little misleading (... of course our new RSC-5 is available now...) in the description of the Extended Color BASIC (available in the 26-3002 or as a kit (26-3018 $99.00) which requires 16K RAM (26-3015 $119.00) and installation, etc.). To quote from the catalog "You can save an image from the screen, display a predefined image, "zoom" the graphic image "in" or "out" ... rotate the image, move the image from one area of the screen to another,". That's what it says... now here is what you can really do with Extended Color BASIC: "You can save an image from the screen,..."; What you can do is save the data which generates the graphics (if your software is set up to use data statements) so they can be read back in whenever you wish to recreate the graphics. "... display a predefined image, "zoom" the image "in" or "out"": What you have the capabilities of doing with the Color Computer's Extended BASIC is choosing from up to eight screens. Think of a large note-pad with only ⅛ of it showing through a "window." If you move the window, you see a different area, or if you move the window farther away from the pad, you see more sections of it. Get the picture? Well, the Extended BASIC Color Computer works in a similar way; you write or draw on one section not being displayed, then you display it. Displaying a predefined image is true. However, "zooming" is not. It is possible to do, but not easily. What you have to do is redraw the image over and over again. changing the location of the image on one of the screens not being currently displayed. "... rotate the image..."; this would be accomplished much the same as "zooming" would be handled. "... move the image from one area... to another...": What you are really doing is drawing the graphics on one of the screens and changing the window through which you are looking at the graphic.

Confused? Well, read on and maybe things will get clearer....

Let's get a little common ground between us before we get into the thick of things: The Extended BASIC version of the Color Computer contains all of the features of the standard machine with some very interesting additions. First off, the computer has 16K of RAM of which a maximum of about 14.5K is usable for programming (the balance is saved and used for video, BASIC, and graphics). You can allocate more memory for graphics if your creations need it or if you're going into higher resolution (more on that later). All of the accessories for the standard Color Computer will work on the Extended BASIC machine, including the Joysticks and all the program paks.™ Some of the program paks will even appear to be smoother in operation (Quasar Commander especially) of the graphics because of the additional memory. Besides all that, you get EDITing, which means not having to retype that line when you make a mistake; multi-character string names (first two characters significant) and easy access to higher resolution (there's that word again) graphics. It's not just the fact that the Extended version offers higher resolution graphics, it is HOW the higher resolution graphics are offered. What this version of BASIC has are one line instructions to DRAW a line from point A (x,y) to point B (x,y), make it either the foreground or background color (yes, you can set what foreground/background colors will be), make that line the diagonal of a box and even fill that box with either the foreground or background color. Just ONE statement does it all! The program statement is ... **LINE**. There is another one line statement which will draw a CIRCLE. All you need to tell it is where the center is (x,y), the radius, the color, aspect (optional), starting and ending points (optional—in case you don't want a full circle, only a partial one). Again, just ONE statement does all that! That program statement is... you guessed it... **CIRCLE**. There is a **PAGE** function which lets you look at one of eight pages of graphic memory (remember that note-pad and "window"). Depending on which graphics resolution mode you are in (there are 4 to choose from), you will see all or part of that page (maybe that's where they got the "zoom" idea). Speaking of graphics modes, you can select from low resolution (at last, an explanation, the number of distinct points on the screen) of 2048 points (64x32) up to 49152 points (192x256, even though the RSC-4 catalog states 196x256 p.29). High resolution means you can "turn on" 100 distinct points on the screen and those points won't take up more than a ½ inch square on a 13" monitor. Do you remember the program in the back of the Level I, Model I BASIC manual, the one about termites? The one that takes about 15 minutes to completely clear the screen? Well, if you tried something like that on the Extended BASIC Color machine, you might be there awhile, waiting for those "termites" to eat up the screen. It would almost be like having a pest controller on your team!

Now, a few more brief definitions of some of the other commands you'll find in Extended Color BASIC:

PCLEAR x: allocates x (1-8) number of pages of memory for graphics.

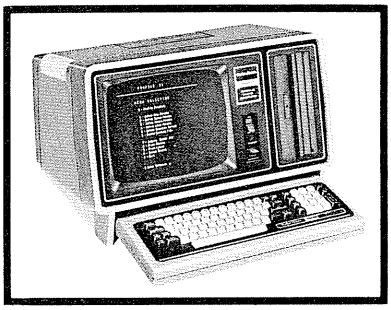
PMODE x,y: where x (1-4) is the graphics mode and y (1-8) is the starting page of memory you wish to display. The graphics mode and the memory starting page are dependent on each other and the number of pages you have PCLEARed.

SCREEN x,y: where x (0-1) is either 0 for text display or 1 for graphics display and y (0-1) is the choice between two sets of four colors. For y = 0: green, red, yellow, blue; for y = 1: buff, cyan, magenta, orange. These colors can also be adjusted by using the color and tint controls on the TV set.

One quick note here: higher resolution graphics and text cannot be shown simultaneously on the screen.

PCLS(x): used after you have selected your starting page for graphics, it clears the screen, using the value color of x (1-8).

(Continued on page 17)



Model II

Product Line Manager's News

Another hint for PROFILE-II users. Would you like to create your own custom MENU? Would you like to pass the program some parameters and get you or your secretary away from answering the FILENAME and SCREEN NUMBER question everytime (sometimes getting the wrong screen or report format)?

A simple custom MENU with a short description of each of the screen or report formats that you use will eliminate trying to remember which format is which.

Try this routine on a BACKUP of your PROFILE disk...

First, you must still have BASIC on the disk. Second, rename the current MENU to X by typing `RENAME M TO X` and pressing (ENTER).

You must create a machine language program with DEBUG. From TRSDOS, type `CLEAR` and press (ENTER), now type `DEBUG ON,` press (ENTER), type `DEBUG` and (ENTER). Press (M) and enter `E000` for the address. Press the (F1) key and the cursor should be in the upper left hand corner of the screen at the first set of double zeros. Enter the following codes exactly. You do not have to press the space bar.

```
21 09 E0 06 0A 3E 26 CF
C9 42 41 53 49 43 20 4D
45 4E 55
```

Double check the entries you made, using the arrow keys to position the cursor and make any corrections.

Press the (F2) key, then the LETTER (O). You should be back in TRSDOS. Now you must save this program, type

```
DUMP M {START=E000 END=E012}
```

and press (ENTER) Build a DO file by typing `BUILD USER` (ENTER). In the DO file, type `BASIC MENU` and press (ENTER) three times. This will store the DO file with a filename of USER.

When you return to TRSDOS READY, type `BASIC` (ENTER) and you are ready to create the MENU. I will give you a sample menu using "DJS" for the example file names (lines 200 thru 230). In all cases, use the file name you normally enter when using PROFILE but followed with enough 0's to make it 8 characters long.

```
5 GOTO 10
7 SAVE "MENU":END
10 CLS
20 PRINT@(3,25),"1) ADD / UPDATE
   RECORDS"
30 PRINT@(5,25),"2) PRINT FILE (LONG
   FORM)"
40 PRINT@(7,25),"3) PRINT FILE (SHORT
   FORM)"
50 PRINT@(9,25),"4) PRINT LABELS"
60 PRINT@(11,25),"5) PROFILE DIRECTORY"
70 PRINT@(13,25),"6) EXPAND FILE"
80 PRINT@(15,25),"7) PROFILE MENU"
90 PRINT@(17,25),"8) EXIT TO TRSDOS"
100 PRINT@(19,28),"SELECTION = ";
110 A$=INKEY$: IF A$="" THEN 110 ELSE
    A=VAL(A$): CLS
120 ON A GOTO 200, 210, 220, 230, 240,
    250, 260, 270
130 GOTO 10
200 SYSTEM "CLERK/EFC {DJS00000,1,
    HEADING FOR TOP OF SCREEN}"
210 SYSTEM "PRINT/EFC {DJS00000,1,
    INSTRUCTIONS FOR SORTING}"
220 SYSTEM "PRINT/EFC {DJS00000,2, OR
    SELECTING RECORDS}"
230 SYSTEM "LABEL/EFC {DJS00000,1, LABEL
    INSTRUCTIONS}"
240 SYSTEM "DIR/EFC"
250 SYSTEM "EXPAND/EFC"
260 SYSTEM "X"
270 SYSTEM
```

If you have more than 9 items in your menu, replace line 110 with the one below but you will have to press (ENTER) after your menu selection.

```
110 INPUT A: CLS
```

Notice the format number in lines 210 and 220. This answers the question for you when the program asks "Enter Format Number (1-5)."

Also notice that there is no space before the format number in 200, 210, 220, or 230.

Now type `GOTO 7`, press (ENTER) and the program will be stored on disk with a file name of MENU.

To use this new menu, type (M) from `TRSDOS READY`, just as you did before.

A few points about lines 120 and 200 thru 270. For each item you add in lines 20 thru 90, there must be a corresponding line (lines 200 thru 270) to execute the command. Each line, 200 and on, contains the PROFILE program you wish to use.

CLERK/EFC is the normal screen program used to enter, add or update the data. The braces are required, followed by the FILENAME filled to 8 places with zeros. The number (1-5) is the screen or print format you wish to use.

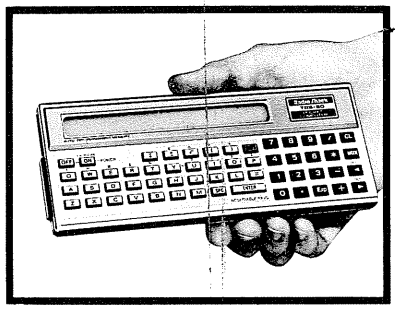
The statement after the number will be placed at the top of the screen and can be any instruction or title you wish the operator to see when using the program. Most of the PROFILE modules can be passed parameters with this method and, another advantage, when you press (ENTER) to exit the module, you will be returned directly to the MENU.

PROFILE programs that you can execute from the BASIC MENU are:

CLERK/EFC = Video screens used to add or update your data files.
PRINT/EFC = Printer formats
LABEL/EFC = Label printing
EXPAND/EFC = Used to expand the number of records
DIR/EFC = PROFILE directory
SELECTOR/EFC = Selects records for merging with other programs

(Continued on page 12)

I found the bug!



Pocket Computer

Product Line Manager's News

Well, Spring is about to spring upon us here in Texas and the weather is just perfect for flying . . . which brings me neatly to the first subject of this month's column, our Pocket Computer Aviation software package called AVPACK I.

This software package became available in February in all of our stores, under Catalog number 26-3513 and priced at $24.95. The package consists of one cassette tape with a single program which fills the entire computer memory (with only 6 steps left!). AVPACK I is a departure from our other Pocket Computer software formats in that we have supplied a plastic overlay for the keyboard which shows the function of the 15 Reserve keys used in operation of the program, rather than have a menu which would consume too much memory and not allow the incorporation of all the functions into one load module.

First, let's have an overview of the features of this package. As I previously implied, the program has 15 functions initiated by a (SHFT) and then pressing one of the Reserve keys. These keys cause the computer to perform conversions or calculations according to the following:

CONVERT

C — Temperature from Celsius to Fahrenheit.
F — Temperature from Fahrenheit to Celsius.
M — Statute Miles to Nautical Miles and Kilometers.
N — Nautical Miles to Statute Miles and Kilometers.
K — Kilometers to Statute Miles and Nautical Miles.
G — U.S. Gallons to Imperial Gallons and Liters.
B — Imperial Gallons to U.S. Gallons and Liters.
L — Liters to U.S. Gallons and Imperial Gallons.

CALCULATE

A — True Altitude, using Pressure Altitude, Temperature and Reference Altitude.
V — True Airpseed, using Pressure Altitude, Temperature and Indicated Airspeed.
X — Mach Number, based on True Airspeed and Temperature.
Z — True Airpseed, based on Mach Number and Temperature.
D — Off-Course Correction and Drift Angles.
H — True Heading and Ground Speed with winds.
J — Enroute Winds.

Those of you who are private pilots or who have flown a light aircraft recently, will recognize that these functions will handle nearly all of the requirements of visual flight navigation, with the possible exception of Dead Reckoning (Rhumb Line) calculations. So, now that we've had our 'pre-flight,' let's 'take-off' and look at some specifics.

For those not familiar with the Pocket Computer, once you load the program, everything including your answers and data is stored in the computer even if power is turned off. Thus, once a calculation or conversion is completed, the answer is stored in one of 26 locations in the computers memory. These locations are listed in Appendix E of the AVPACK I Operators Manual. An answer or piece of data may be recalled at any time simply by typing its letter and pressing ENTER.

As an example of the ease of use of these conversions, let's take the following example:

You've flown your plane down to Puerto Vallarta for a little vacation and it's time to go have some fun, but you have to fuel up first. Mexico sells gasoline by the Imperial Gallon and you want to find out how many gallons (US) you used on the last leg of your flight. After topping-off your tanks, you find that you used 18 Imperial Gallons and your flight log shows 2 Hours and 50 Minutes flying time. Now for some Pocket Computer Power! Turn on the computer, type 18 and press (SHFT) (B). This converts to 21.6 U.S. Gallons or 81.8 Liters as shown on the display. Now press (G) (this is the memory location which stores 'Liquid Measure in U.S. Gallons') and press (ENTER). The display will now show 21.61728, the true equivalent in U.S. Gallons. So-far-so-good . . . but now for the tricky part! Your log shows 2 Hours, 50 Minutes but this 'time' must be in decimal format to allow figuring out how much fuel you used per hour. The Pocket Computer has a Function called DEG which we will put to use here. DEG will take an input of Degrees, Minutes and Seconds in the form 'DDD.MMSS' and convert it to decimal. Since this conforms to the notation for time, we will use it to convert our flying time to the required decimal equivalent. Type G/DEG2.50 and press (ENTER). The display will show 7.629628235 indicating the number of U.S. Gallons of gasoline burned per hour on the last leg of your flight. By now you may be saying 'Run that by again?' . . . OK, here's what happened. Register 'G' contains our fuel used in U.S. Gallons, the slash (/) causes division and the DEG2.50 converts the hours and minutes to 2.833333333 decimal hours before dividing into the 21.61728 Gallons stored in 'G.' Tricky, isn't it!

This is just one example of the powerful conversion capabilities of this program and the Pocket Computer. Rather than continue with another example of the Calculations, I am going to proceed to another topic which will take up the rest of my allotted space. So, if you are a pilot, or an aspiring pilot, I hope I have piqued your interest enough so that you will go down to your nearest Radio Shack and ask for a 'hands on' demonstration. And, if by chance they don't have this program package yet, please ask the store manager to order one for you to look at.

My next subject is SOLICITING . . . soliciting SOFTWARE that is! Yes, we ARE looking for good programs for the Pocket Computer! As you can well imagine, it would be impossible for Radio Shack to write programs to cover all or anywhere near all of the application possibilities of this incredible little (BIG) computer. So, I am asking you, our faithful readers and TRS-80 owners, not only for ideas for us to look at, but preferably for fully completed Pocket Computer software for us to evaluate and consider for marketing through our growing chain of Radio Shack stores and Computer Centers which now number around 7000! Toward this end, I would like to outline the following procedures (and facts-of-life of a big corporation) for your guidance in submitting your program or idea, to wit:

1) Submit a concise description of the program, what it does, who would use it and why you feel it would have mass appeal. State whether or not the program has been copyrighted and whether or not you've sold copies of it (this doesn't affect our interest, only the way we must handle it to protect your interests and ours). Obviously, the program must be your own work. Contractually, you will be required to "hold Radio Shack harmless" in the event of a copyright suit by someone else.

2) Give us time. We often move slowly. We will let you know if we're interested, and arrange to have you send a copy of the program for more serious evaluation. And above all, you must realize that there are many reasons why we might reject your program . . . poorly written or documented, limited market, poor error trapping, hard to use, similar to a program now in development,

(Continued on page 12)

Model II (from page 10)

Four of the modules that cannot accept parameters are EXPAND/EFC, SELECTOR/EFC, DIR/EFC and SELECT/BAS.

One warning and one suggestion.

THE WARNING: Do NOT try to execute either of the DO files for the LIMITED MENU or the UNLIMITED MENU, this will kill the "M" program that you created with DEBUG.

THE SUGGESTION: You should not try to execute any of the "CREATION" programs such as CREATE/EFC or LPFORM/EFC from this menu. The "CREATION" programs CAN be called but they CANNOT be passed any parameters and are not normally used on a day-to-day basis. They can still be executed from the original menu that you have now renamed "X."

Good luck with your new MENU. It should make using PROFILE-II a lot more enjoyable.

Model II Bugs, Errors and Fixes

Patch for TRSDOS 2.0

In the TERMINAL utility, the "G" (Get disk file into RAM) function, when used with a variable-length record file, will insert 1 extra byte before every logical record.

To prevent this from happening, apply the following two patches:

```
PATCH TERMINAL A=3723
      F=FE4628 C=D65620
PATCH TERMINAL A=372A
      F=06004E23AFB9C004C9
      C=474E230DC0E1C3F632
```

General Ledger (26-4501)

The Batch Total in a posting summary does not equal the document balance.

To correct this, Change line 730 of "Txpost" to read:

```
730 IN#=0: FORQ=0TO NE-1:
    IF AM#(Q) >0 THEN
    IN#=IN#+AM#(Q)
```

You also need to change line 6730 of the "Txentry" program. Change this line to read:

```
6730 IN#=0: FORQ=0TO NE -1:
     IF AM#(Q) >0 THEN IN#=IN#+
     AM#(Q)
```

Payroll (26-4503)

When preparing checks with either version 1.0 or 1.1, values are being rounded improperly. Some values are being rounded with an error of up to three cents.

To correct this problem, change line 820 of the "INPUT" to read:

```
820 R#(I)= INT(R#(I)* 100D0+ .5D0)/
    100D0: E#(I)= H#(I)* R#(I):
    E#(I)= INT(E#(I)* 100D0+ .5D0)/
    100D0: GOTO 840
```

Accounts Payable (26-4505)

There have been several customers who have encountered error code 5 in the invoice section of version 1.0 Accounts Payable. The same error is also encountered in End-of-period processing. To correct these problems, make the following changes:

For invoices, change line 350 of the "APINVCE/BAS" program to read:

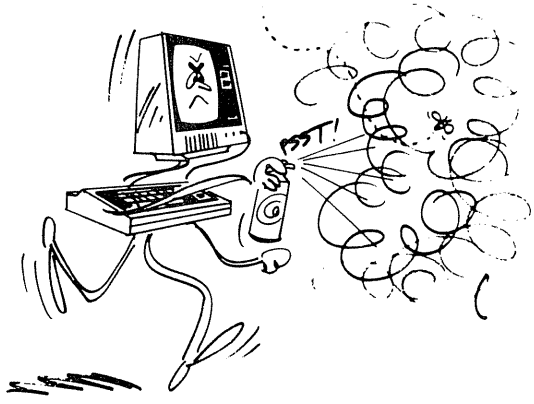
```
350 N#=VAL(IN$): W#= ABS(N#)* 100:
    W#= INT(W#* 100D0+ .5D0)/ 100D0:
    V$="": . . .
```

The rest of the line is unchanged. Be sure you save a copy of the corrected program.

For End-of-Period, change line 800 in the "APPROC/BAS" program to read:

```
800 W#= ABS(N#)* 100:W#= INT(W#* 100D0+
    .5D0)/ 100D0:V$="":...
```

The rest of this line is unchanged. Again, do not forget to save a copy.



Pocket (from page 11)

too far from our 'style,' etc. If we reject it, you will receive a form letter, not an individual reply... if you have a 'thin skin,' it might be better if you don't submit. Incidentally, we like to see programs which use the full power of the Pocket Computer in order to maximize the somewhat limited user memory, the previously mentioned AVPACKI program being an example.

3) Be prepared to wait UP TO a year before you see an accepted program on our shelves. All the testing, editing, debugging and manual writing just takes time!

4) Be prepared to accept the decision of our software evaluation team. Anyone else you write within Radio Shack will just refer you to them. Their decision is final.

5) If we like your program, we might offer you an outright purchase price, or a royalty on copies sold. Our royalties are much like some publishers offer... not big, but if the package sells, they can REALLY mount up fast.

Now, if you're still interested, send your letters to:

PC Software Evaluation
Radio Shack
700 One Tandy Center
Fort Worth, TX 76102

Before I close for this month, let me give you a few hints/ideas which might be helpful in your efforts to write 'mass-market' software for the Pocket Computer. We view the Pocket Computer as having tremendous application in a wide range of VERTICAL markets. By that I mean not only the engineering and financial disciplines, but areas such as medicine, pharmacology, insurance (all types), agriculture, photography, meteorology, oceanography to name but a few. I am sure that with over 100,000 readers of this newsletter, there must be at least one 'expert' in nearly every discipline I could name who could provide a good application program for the Pocket Computer. In closing then, I would urge you to give this some thought and if you are not familiar with the Pocket Computer, stop by one of our stores and get a demonstration! Until April then... more Pocket Power to you.

STOP THAT OUT-OF-SORTS FEELING

A fast 2D string array sort

By William Terrell

Many of you have probably developed programs which included routines to sort arrays of strings. Unfortunately, string sorting programs in BASIC are very slow. The March/April 1980 issue of the TRS-80 Microcomputer News covered a number of BASIC sort routines with the best taking 16 minutes to sort 500 strings. As a result, when the July 1980 issue provided a machine language sort which claimed cut the time to sort to seconds, it caught my interest.

I quickly keyed in the demonstration program—a machine language program poked into memory from BASIC and invoked by the USR command. It did everything they claimed. Arrays of 500 strings could be sorted in 4 seconds!

Now I had no excuse for not completing the many programs which I had abandoned as impractical due to excessive processing time. But wait! I soon discovered that this program was limited to sorting single dimension string arrays. Mailing lists, inventory programs, and the like required two-dimensional arrays. Sorting on only one dimension would completely scramble the data. I was still without a suitable string sorting program.

Perhaps I could modify this program to provide two-dimensional array string sorting capability. The first step was to figure out how the program worked. After I had poked the machine language program into memory, I loaded and ran my trust disassembler program. This converts the machine code into assembly language mnemonics, a set of commands which humans can understand.

After a couple of hours of study, the basic flow of this program was clear to me. The number of strings being sorted and the pointer to the first string were passed to the machine language program by the BASIC's USR command pretty much as the Level II manual describes. This data was accepted by the machine language program and put into storage locations. Next, the program selected and compared strings. Sorting was accomplished by swapping BASIC's variable pointers. I located the part of the program which did the actual swapping. Now all I had to do was modify that section to swap all related pointers at the same time.

A few other details had to be addressed as well. The program needed to know how many related data fields

(Continued on page 14)

Machine Sort—Listing 1

```
7EDF                00110           ORG     7EDFH
                    00120           ;GET VARPTR FOR Z(0)
7EDF CD7F0A         00130 ENTRY     CALL    0A7FH
                    00140           ;LOAD NUMBER OF DATA
                                    SETS
7EE2 5E             00150           LD      E,(HL)
7EE3 23             00160           INC     HL
7EE4 56             00170           LD      D,(HL)
                    00180           ;STORE IT
7EE5 ED530A7F       00190           LD      (STORN+1),DE
                    00200           ;LOAD VARPTR TO SORTED
                                    DATA
7EE9 23             00210           INC     HL
7EEA 5E             00220           LD      E,(HL)
7EEB 23             00230           INC     HL
7EEC 56             00240           LD      D,(HL)
                    00250           ;STORE IT
7EED ED53F87F       00260           LD      (STORV),DE
                    00270           ;LOAD NUMBER OF FIELDS
7EF1 23             00280           INC     HL
7EF2 5E             00290           LD      E,(HL)
7EF3 23             00300           INC     HL
7EF4 56             00310           LD      D,(HL)
                    00320           ;STORE IT
7EF5 ED53FA7F       00330           LD      (DIMENS),DE
                    00340           ;LOAD POINTER DELTA
7EF9 23             00350           INC     HL
7EFA 5E             00360           LD      E,(HL)
7EFB 23             00370           INC     HL
7EFC 56             00380           LD      D,(HL)
                    00390           ;STORE IT
7EFD ED53FC7F       00400           LD      (PTRDEL),DE
                    00410           ;LOAD FIELD-FIELD DELTA
7F01 23             00420           INC     HL
7F02 5E             00430           LD      E,(HL)
7F03 23             00440           INC     HL
7F04 56             00450           LD      D,(HL)
                    00460           ;STORE IT
7F05 ED53FE7F       00470           LD      (COLDEL),DE
7F09 210000         00480 STORN     LD      HL,$-$
7F0C 22F67F         00490           LD      (STORDE),HL
                    00500 ;START SELECT & COMPARE
7F0F ED5BF67F       00510 CYCLE     LD      DE,(STORDE)
7F13 CB3B           00520           SRL     E
7F15 AF             00530           XOR     A
7F16 CB3A           00540           SRL     D
7F18 3002           00550           JR      NC,BYPASS
7F1A CBFB           00560           SET     7,E
7F1C ED53F67F       00570 BYPASS    LD      (STORDE),DE
7F20 7A             00580           LD      A,D
7F21 B3             00590           OR      E
7F22 C8             00600           RET     Z
7F23 2A0A7F         00610           LD      HL,(STORN+1)
7F26 ED52           00620           SBC     HL,DE
7F28 22F27F         00630           LD      (STORT3),HL
7F2B 210000         00640           LD      HL,$-$
7F2E 22F07F         00650           LD      (STORT2),HL
7F31 2AF07F         00660 AGAIN     LD      HL,(STORT2)
7F34 22EE7F         00670           LD      (STORT1),HL
7F37 2AEE7F         00680 REPEAT    LD      HL,(STORT1)
7F3A ED5BF67F       00690           LD      DE,(STORDE)
7F3E 19             00700           ADD     HL,DE
```

(Continued on page 14)

Out-of-Sorts (from page 13)

were being swapped and where they were. This information could be passed to the machine language program in the same manner as the original program. Now that I had the concept worked out, all I had to do was implement it.

Rather than start from scratch, I first copied the disassembled version of the original machine language program using the Radio Shack Editor/Assembler. Since I expected to have to relocate the machine code, I assigned a label to each absolute address. Next, provision was added to allow the program to accept the added inputs mentioned above. Finally, the swapping instructions were expanded to swap the related string fields at the same time. I was ready to test it!

For convenience, I loaded the new machine language program (see Listing 1) into high memory with the SYSTEM command. The original TRS-80 Microcomputer News BASIC demonstration program was then modified to interface with the new program. Of course, it didn't work the first time (or the second or third time either). After a few iterations with the Editor/Assembler, I had it sorting string arrays of 100 by 5 size. Sorting time did not seem to be longer than the original program. Now all that remained was to pretty it up and tell the world.

I chose to produce a demonstration program (see Listing 2) very similar to the original with lots of REMarks so that prospective users can understand how to adapt the program to their needs. In fact, I put so many REM's in it that they have to be deleted to allow the program to run in a 16K Level II machine. If you do duplicate the whole program for your records, CSAVE a similar program with the REM's deleted for running it.

Before loading this demonstration program, you must protect memory (in response to MEMORY SIZE?) at 32478. After CLOAD and RUN, the program will take a few seconds to poke the machine language program into the protected memory space. Then it will ask you:

"NUMBER OF DATA SETS (100 MAX) YOU WANT?"

As it is presently dimensioned, you may enter any number up to 100 to represent the 'rows' of data. Then it will ask you:

"HOW MANY FIELDS (5 MAX) IN EACH?"

You should enter any number up to 5 to represent the number of related 'fields' in each 'row.' The computer will now take from a few seconds up to a minute or so to generate a random, two-

(Continued on page 15)

Sort—Listing 1 (from page 13)

```
7F3F 22F47F      00710           LD      (STORHL),HL
7F42 EB          00720           EX      DE,HL
7F43 210000      00730           LD      HL,$-$
7F46 19          00740           ADD     HL,DE
7F47 19          00750           ADD     HL,DE
7F48 19          00760           ADD     HL,DE
7F49 E5          00770           PUSH    HL
7F4A ED5BEE7F    00780           LD      DE,(STORT1)
7F4E 210000      00790           LD      HL,$-$
7F51 19          00800           ADD     HL,DE
7F52 19          00810           ADD     HL,DE
7F53 19          00820           ADD     HL,DE
7F54 ED4BF87F    00830           LD      BC,(STORV)
7F58 09          00840           ADD     HL,BC
7F59 EB          00850           EX      DE,HL
7F5A E1          00860           POP     HL
7F5B 09          00870           ADD     HL,BC
7F5C E5          00880           PUSH    HL
7F5D D5          00890           PUSH    DE
7F5E 0E00        00900           LD      C,0
7F60 7E          00910           LD      A,(HL)
7F61 47          00920           LD      B,A
7F62 1A          00930           LD      A,(DE)
7F63 B8          00940           CP      B
7F64 3003        00950           JR      NC,CLEAR
7F66 0E01        00960           LD      C,1
7F68 47          00970           LD      B,A

7F69 AF          00980 CLEAR     XOR     A
7F6A B0          00990           OR      B
7F6B 2819        01000           JR      Z,SKIP
7F6D C5          01010           PUSH    BC
7F6E 13          01020           INC     DE
7F6F 23          01030           INC     HL
7F70 4E          01040           LD      C,(HL)
7F71 23          01050           INC     HL
7F72 46          01060           LD      B,(HL)
7F73 C5          01070           PUSH    BC
7F74 E1          01080           POP     HL
7F75 EB          01090           EX      DE,HL
7F76 4E          01100           LD      C,(HL)
7F77 23          01110           INC     HL
7F78 46          01120           LD      B,(HL)
7F79 C5          01130           PUSH    BC
7F7A E1          01140           POP     HL
7F7B C1          01150           POP     BC
7F7C 1A          01160 BACK      LD      A,(DE)
7F7D 96          01170           SUB     (HL)
7F7E 380A        01180           JR      C,POP
7F80 2053        01190           JR      NZ,FINIS
7F82 13          01200           INC     DE
7F83 23          01210           INC     HL
7F84 10F6        01220           DJNZ    BACK
7F86 CB41        01230 SKIP      BIT     0,C
7F88 204B        01240           JR      NZ,FINIS
                 01250 ;MOVE BACK TO ZERO COLUMN
7F8A ED4BFC7F    01260 POP       LD      BC,(PTRDEL)
7F8E AF          01270           XOR     A
7F8F E1          01280           POP     HL
7F90 ED42        01290           SBC     HL,BC
7F92 EB          01300           EX      DE,HL
7F93 E1          01310           POP     HL
7F94 ED42        01320           SBC     HL,BC
                 01330           ;SET FOR NO. OF COLUMNS
```

(Continued on page 15)

Out-of-Sorts (from page 14)

dimensional string array to your specification. This slow process is not part of the sorting routine, but rather a clumsy way to create some test data for the program. Next, the computer will ask you:

"SORT ON WHICH FIELD (ZERO IS 1ST)?"

You should respond with the number of the 'field' in the data 'row.' Remember that the first field is "0." We're now ready to start the data sort as soon as you respond to:

"PRESS 'ENTER' WHEN READY?"

This last prompting question was put in the program so you could start timing the sorting if you want. As soon as the sorted string array starts printing on the video, you know that the sort is complete. Finally, the computer will display:

"FOR ANOTHER SORT ON THE SAME DATA, ENTER '1'
FOR NEW TEST DATA, ENTER '2'
TO QUIT, ENTER '3'?"

You may take any of those three actions by entering the indicated number. Entering a '1' avoids the long delay to create a new string array.

Many of you have more than 16K memory in your computers. You may poke the machine language program into the top of your memory by changing the BASIC program line 190 to add 16384 (for 32K) or 32768 (for 48K) to both numbers in the line. Since this produces addresses above 32767, you will have to modify each number by the formula:
– 1*(65536 – desired address).

Don't forget to change the MEMORY SIZE accordingly. You also have to change the calling address in line 270 by replacing the '126' with 190 (for 32K) or with 254 (for 48K). Next, fix the machine language program absolute address data in lines 50000 and beyond by replacing '127' with 191 (for 32K) or with 255 (for 48K) everywhere but the first occurrence of '127.' Finally, fix the check in line 240 by adding the amount you added in the data substitutions. You are now able to size bigger string arrays in lines 290, 300, 350, and 360.

When adapting this technique to other programs, be sure that you define Z as an integer (after your last CLEAR command). Also, don't insert any other steps in between your equivalent of lines 1100 through 1160. The machine language program depends on this to find the data being passed to it by the USR command. Finally, if any of the VARPTR's in lines 1140 or 1150 are negative, they must be converted by adding 65536 to each neg-

(Continued on page 16)

Sort—Listing 1 (from page 14)

```
7F96 3AFA7F     01340           LD      A,(DIMENS)
7F99 47         01350           LD      B,A
7F9A C5         01360           PUSH    BC
                01370           ;BYPASS FOR 1ST SWAP
7F9B 180D       01380           JR      SWAP1
                01390 ;MOVE TO NEXT COLUMN
7F9D C5         01400 SWAP      PUSH    BC
7F9E ED4BFE7F   01410           LD      BC,(COLDEL)
7FA2 1B         01420           DEC     DE
7FA3 1B         01430           DEC     DE
7FA4 2B         01440           DEC     HL
7FA5 2B         01450           DEC     HL
7FA6 09         01460           ADD     HL,BC
7FA7 EB         01470           EX      DE,HL
7FA8 09         01480           ADD     HL,BC
7FA9 EB         01490           EX      DE,HL
                01500 ;START VARPTR SWAP
7FAA 4E         01510 SWAP1     LD      C,(HL)
7FAB EB         01520           EX      DE,HL
7FAC 7E         01530           LD      A,(HL)
7FAD 71         01540           LD      (HL),C
7FAE EB         01550           EX      DE,HL
7FAF 77         01560           LD      (HL),A
7FB0 23         01570           INC     HL
7FB1 13         01580           INC     DE
                01590           ;2ND BYTE SWAP
7FB2 4E         01600           LD      C,(HL)
7FB3 EB         01610           EX      DE,HL
7FB4 7E         01620           LD      A,(HL)
7FB5 71         01630           LD      (HL),C
7FB6 EB         01640           EX      DE,HL
7FB7 77         01650           LD      (HL),A
7FB8 23         01660           INC     HL
7FB9 13         01670           INC     DE
                01680           ;3RD BYTE SWAP
7FBA 4E         01690           LD      C,(HL)
7FBB EB         01700           EX      DE,HL
7FBC 7E         01710           LD      A,(HL)
7FBD 71         01720           LD      (HL),C
7FBE EB         01730           EX      DE,HL
7FBF 77         01740           LD      (HL),A
                01750           ;GET COLUMN CTR
7FC0 C1         01760           POP     BC
                01770           ;DO NEXT COLUMN
7FC1 10DA       01780           DJNZ    SWAP
7FC3 2AF67F     01790           LD      HL,(STORDE)
7FC6 EB         01800           EX      DE,HL
7FC7 2AEE7F     01810           LD      HL,(STORT1)
7FCA AF         01820           XOR     A
7FCB ED52       01830           SBC     HL,DE
7FCD 22EE7F     01840           LD      (STORT1),HL
7FD0 D2377F     01850           JP      NC,REPEAT
7FD3 1802       01860           JR      EXIT
7FD5 D1         01870 FINIS     POP     DE
7FD6 E1         01880           POP     HL
7FD7 2AF07F     01890 EXIT      LD      HL,(STORT2)
7FDA 110100     01900           LD      DE,1
7FDD AF         01910           XOR     A
7FDE 19         01920           ADD     HL,DE
7FDF 22F07F     01930           LD      (STORT2),HL
7FE2 ED5BF27F   01940           LD      DE,(STORT3)
7FE6 ED52       01950           SBC     HL,DE
7FE8 DA317F     01960           JP      C,AGAIN
7FEB C30F7F     01970           JP      CYCLE
0002            01980 STORT1    DEFS    2
```

(Continued on page 16)

Out-of-Sorts (from page 15)

ative VARPTR. (ED. We added line 1130 which handles negative VARPTR for line 1140, the most likely line to need correcting.) This may be done with an IF . . . THEN . . . statement in the same line. You need not modify lines 1110 or 1160.

Now, let's get to work finishing all those programs which require a fast sorting subroutine!

Editor's comments: We modified the program slightly (reduced the array from 150 x 5 to 100 x 5) so that it will run in either a 16K Level II Model I, or a 16K Model III BASIC Model III.

The only restriction on the array dimensions is the limits of your memory. You might wish to look at the Model II version of the demonstration program. In that version, we allow the response to the INPUTs to control the dimensioning of the array A$.

In general we left Mr. Terrell's work as we received it. We did renumber the lines to make entry a little easier. Next month we hope to present a second article from Mr. Terrell called "Get Your Data For Free."

Model III Level I Manual Error

The Model III Level I manual, page 66, gives you a sample program to use with your line printer.

Level I does not support the TAB function with LPRINT. Change lines 10, 30 and 80 to read:

```
10 LPRINT "TELEPHONE
   LIST"
30 LPRINT" ","NAME",,
   "TELEPHONE NUMBER"
80 LPRINT" ",A$,,B$
```

These changes will let you use this program so you can "play" with your line printer. Please ignore all references to the TAB function used with LPRINT.

Sort—Listing 1 (from page 15)

```
0002            01990 STORT2   DEFS    2
0002            02000 STORT3   DEFS    2
0002            02010 STORHL   DEFS    2
0002            02020 STORDE   DEFS    2
0002            02030 STORV    DEFS    2
0002            02040 DIMENS   DEFS    2
0002            02050 PTRDEL   DEFS    2
0002            02060 COLDEL   DEFS    2
0000            02070          END
00000 TOTAL ERRORS
```

Please note the addresses in parenthesis are for the Model II version of this program, ORGed at Hex F000.

AGAIN	7F31	(F052)	BACK	7F7C	(F09D)
BYPASS	7F1C	(F03D)	CLEAR	7F69	(F08A)
COLDEL	7FFE	(F11F)	CYCLE	7F0F	(F030)
DIMENS	7FFA	(F11B)	ENTRY	7EDF	(F000)
EXIT	7FD7	(F0F8)	FINIS	7FD5	(F0F6)
POP	7F8A	(F0AB)	PTRDEL	7FFC	(F11D)
REPEAT	7F37	(F058)	SKIP	7F86	(F0A7)
STORDE	7FF6	(F117)	STORHL	7FF4	(F115)
STORN	7F09	(F02A)	STORT1	7FEE	(F10F)
STORT2	7FF0	(F111)	STORT3	7FF2	(F113)
STORV	7FF8	(F119)	SWAP	7F9D	(F0BE)
SWAP1	7FAA	(F0CB)			
PART19	(F12F)	START	(F121)		

Machine Sort—Listing 2 Basic Program

```
100 ' *** 2D ARRAY SORT BASED ON ALLEN EMERT PROGRAM
110 '     FROM JULY 80 TRS-80 MICROCOMPUTER NEWSLETTER
120 '     MODIFIED BY WILLIAM TERRELL OCT 80
130 '     FOR ARRAYS OF THE FORM A$(I,J) ***
140 '
150 ' *** POKE MACHINE LANGUAGE PROGRAM INTO HIGH
          MEMORY ***
160 '     *** PROTECT MEMORY AT 32478 ***
170       POKE 16553, 255
180       K=0
190       FOR I= 32479 TO 32749: READ J
200       K= K+ J
210       POKE I, J: NEXT I
220 '
230 ' *** CHECK MACHINE LANGUAGE PROGRAM ***
240       IF K <> 32160 THEN PRINT"DATA ERROR": END
250 ' *** INITIALIZATION ***
260 ' *** SET UP CALLING ADDRESS OF MACHINE LANGUAGE
          PROG ***
270       POKE 16526, 223: POKE 16527, 126
280 ' *** SET UP DATA SPACE FOR ZR DATA SETS OF ZF
          FIELDS ***
290       CLEAR 10000: DEFINT I-K, Z: ZR= 100: ZF= 5
300       DIM A$(ZR,ZF), Z(4): Z=0
310 '
320       CLS
330 '
340 ' *** GENERATE AND DISPLAY TEST DATA FOR SORT
          PROGRAM ***
350       INPUT"NUMBER OF DATA SETS (100 MAX) YOU
          WANT"; ZR
360       INPUT"HOW MANY FIELDS (5 MAX) IN EACH"; ZF
370       FOR I= 0 TO ZR-1: FOR J= 0 TO ZF-1: A$(I,J)=
          "": NEXT: NEXT
380       FOR I= 0 TO ZR-1
390       ZA= 5: PRINT I;
400       FOR J= 0 TO ZF-1
```

(Continued on page 17)

Pocket Program

Measurement Conversion

Mr. A. R. Lewellen of Highland, MI. sent us this measurement conversion program. Input a measure in feet and decimal portions of feet, and the program will convert to feet, inches and 16ths of inches.

FT., IN., /16 CONVERSION

```
10:INPUT "ENTER DIMENSION
   ";A
20:B= INT (A/ 12)
30:C= A- (B* 12)
40:D= INT (C)
50:E= C- D
60:F= INT ((E* 16)+ .5)
70:X= F/ 2
80:Y= INT(F/ 2)
90:IF X= Y THEN 200
100:K= 16
110:PRINT B; " FT.,"; " ";
    D; " IN.,"; " "; F; "/
    "; K
120:GOTO 10
130:END
200:G= F/2: H= F/4:
    I= F/8: J= F/16
210:IF G= 0 THEN 110
220:IF G= 1 LET F= 1:
    K= 8: GOTO 110
230:IF G= 3 LET F= 3:
    K= 8: GOTO 110
240:IF G= 5 LET F= 5:
    K= 8: GOTO 110
250:IF G= 7 LET F= 7:
    K= 8: GOTO 110
260:IF H= 1 LET F= 1:
    K= 4: GOTO 110
270:IF K= 3 LET F= 3:
    K= 4: GOTO 110
280:IF I= 1 LET F= 1:
    K= 2: GOTO 110
290:IF J= 1 LET D= D+1:
    F= 0: K= 16: GOTO 110
300:END
```

Color (from page 9)

If you use these last 4 instructions in the sequence listed here, you will have set up the beginning of your program for graphics or text. All that is left is the input routines for the data and what you want to do with that data.

As you can see, Extended Color BASIC is a powerful language, but with simple commands that allow you to use it. Programming is relatively easy to learn and do (especially using the Extended Color BASIC manual provided with either the computer or the kit). For those of you who only need the instruction set, there is also a quick reference card included with the computer.

Sort—Listing 2 (from page 16)

```
410         FOR K=1 TO RND(10)
420         A$(I,J)= A$(I,J)+ CHR$(RND(26)+64)
430         NEXT K: PRINT TAB(ZA); A$(I,J);: ZA= ZA+ 12:
            NEXTJ
440         PRINT: NEXT I
450 '
1000 ' *** SORT ROUTINE ***
1010 ' *** VARIABLES PASSED TO MACHINE LANGUAGE
          PROGRAM ***
1020 ' Z(0) = NUMBER OF DATA SETS
1030 ' Z(1) = POINTER TO FIELD BEING SORTED
1040 ' Z(2) = NUMBER OF FIELDS IN DATA
1050 ' Z(3) = POINTER DELTA: SORTED FIELD TO ZERO
     FIELD
1060 ' Z(4) = POINTER DELTA: FIELD TO FIELD
1070 '
1080        INPUT"SORT ON WHICH FIELD (ZERO IS 1ST)";
            ZC
1090        INPUT"PRESS 'ENTER' WHEN READY"; ZZ
1100        Z(0)= ZR
1110        Z(1)= VARPTR(A$(0,ZC))
1120        Z(2)= ZF
1130        IF Z(1)<0 AND VARPTR(A$(0,0))>0 THEN
            Z(3)=65536+ Z(1) - VARPTR(A$(0,0)): GOTO
            1150
1140        Z(3)= ABS(VARPTR(A$(0,ZC))- VARPTR(A$(0,0
            )))
1150        Z(4)= VARPTR(A$(0,1))- VARPTR(A$(0,0))
1160        Z=USR(VARPTR(Z(0)))
1170 '
1180 ' *** DISPLAY DATA ***
1190        GOSUB 1290
1200 '
1210 ' *** BRANCH TO NEXT ACTION ***
1220        PRINT"FOR ANOTHER SORT ON SAME
            DATA, ENTER '1'"
1230        PRINT"FOR NEW TEST DATA, ENTER '2'"
1240        INPUT"TO QUIT, ENTER '3'"; ZZ
1250        ON ZZ GOTO 1080, 350, 1260
1260        END
1270 '
1280 ' *** DISPLAY ROUTINE ***
1290        FOR I= 0 TO ZR-1
1300        K= 5: PRINT I;
1310        FOR J= 0 TO ZF-1
1320        PRINT TAB(K); A$(I, J);
1330        K= K+ 12
1340        NEXT J: PRINT
1350        NEXT I: RETURN
1360 '
50000 ' * DATA TO POKE MACHINE LANGUAGE PROGRAM *
50010 DATA 205, 127, 10, 94, 35, 86, 237, 83, 10, 127
50020 DATA 35, 94, 35, 86, 237, 83, 248, 127, 35, 94
50030 DATA 35, 86, 237, 83, 250, 127, 35, 94, 35, 86
50040 DATA 237, 83, 252, 127, 35, 94, 35, 86, 237, 83
50050 DATA 254, 127, 33, 0, 0, 34, 246, 127, 237, 91
50060 DATA 246, 127, 203, 59, 175, 203, 58, 48, 2, 203
50070 DATA 251, 237, 83, 246, 127, 122, 179, 200, 42,
      10
50080 DATA 127, 237, 82, 34, 242, 127, 33, 0, 0, 34
50090 DATA 240, 127, 42, 240, 127, 34, 238, 127, 42,
      238
50100 DATA 127, 237, 91, 246, 127, 25, 34, 244, 127,
      235
```

(Continued on page 18)

Peripherals (from page 8)

I'm predicting this printer will be a real winner! It is an ideal mate for the low cost Color Computer. Its features and performance make it an excellent choice for many users of Model I and III machines as well. Included in this article is a sample of the graphic capabilities of Line Printer VII. This unit should be arriving in quantity at the warehouse in late March. Don't forget to order the proper cable. (Model I, III — 26-1401 $39.95; Color — 26-3020 $4.95; Model II — 26-4401 $39.95)

NOTES THAT FLOAT TO THE TOP OF MY DESK

Line Printers VI (26-1166) and VII (26-1167) will be certified Class B by the FCC, making them suitable for use with Model III and the Color Computer in home environments. (They produce less TV interference than some of our other printers.) To operate Model III to class B standards it will be necessary to make sure that your printer cable is shielded. The shielded version of the cable (26-1401A) will become the new standard. They will be available sometime this month. It's a lot more expensive cable but will be supplied at no increase in price. The use of the shielded version will actually be necessary only in some installations.

Some users have experienced problems with the ribbon used in LP III (and now the LP V). Our studies have shown that some of the troubles were caused by over-use of the ribbon. Don't forget, with a 2,000,000 character ribbon life, the LP III will use up a ribbon in a little over 5 hours of printing and LP V in a little over 4 hours. Don't over-use that ribbon. The new low price (26-1414 — $13.95) should help that problem. Additionally, all LP V's will come with a head cleaning kit designed to insure proper operation. Current owners can obtain one through your local Radio Shack store. Ask your manager to order 700-3010. It will be supplied at a suggested charge of $1.50.

Be sure to read the article in the April issue which will cover the saga of LPC. If you are a user of our business packages you may have run afoul of the great TOP OF FORM mystery. Let's hope we can lay it to rest.

Important — last month we told you that the new Modem I (26-1172) worked at both 300 and 600 baud. A last minute change has made the Modem I work at 300 baud ONLY.

Thanks for reading this month. Look for more news next month. Remember —You read it here first!

Model I/III Printer Routine

Here is a letter from Gary J. Himler of Granada Hills, CA:

"Thanks for publishing the hints, routines, and short programs in the NEWS. The Paging Routine by Richard Halloran in the December issue is most helpful. I have made some changes to make it easier to use (at least I think so) and have included the revised listing for your readers, if you would like to publish it.

"The following advantages are gained:

"If the last line of the program listing is "END" then line 90 will cause the LLISTing to be terminated and the printer to go to Top-of-Form. It is no longer necessary to know the number of lines in the program, or to enter them in line 30 of Mr. Halloran's program.

"Line 50 numbers each page of the program in the upper right hand corner.

"If you save the program you wish to list as "PROGRAM",A then it is not necessary to change the name of the file to be opened in line 20 for each program to be LLISTed.

Here is the procedure I recommend:

Keep the LLIST program on disk.

Save the program to be listed as "PROGRAM",A

Then enter RUN"LLIST"

"It is important that the number of characters in a program line not exceed the number of characters the printer can print per line (80 for Line Printer II), otherwise the pagination will get skewed."

```
10 ' LLIST USED TO PRINT PAGINATED LISTING
20 '
30 CLEAR 500
40 ON ERROR GOTO 150
50 CLS: PRINT@320,"LOADING 'PROGRAM' INTO MEMORY"
60 OPEN"I", 1, "PROGRAM"
70 POKE 16425, 1
80 PN= PN+ 1: LPRINT STRING$( 70," ") "PAGE " PN:
   LPRINT" ": LPRINT" "
90 IF PEEK(16425)= 50 THEN LPRINT CHR$(12): GOTO 80
100 LINE INPUT #1,R$
110 LPRINT R$
120 IF RIGHT$(R$,3)= "END" CLOSE ELSE 90
130 LPRINT CHR$(12)
140 GOTO 9999
150 CLS: PRINT@192, "TO PRINT A PAGINATED LISTING,
    SAVE THE PROGRAM TO BE LISTED"
160 PRINT: PRINT"USING THE COMMAND SAVE 'PROGRAM',A"
170 PRINT: PRINT"THEN RUN 'LLIST'"
180 PRINT: PRINT: PRINT
190 GOTO 9999
9999 END
```

Sort — Listing 2 (from page 17)

```
50110 DATA 33, 0, 0, 25, 25, 25, 229, 237, 91, 238
50120 DATA 127, 33, 0, 0, 25, 25, 25, 237, 75, 248
50130 DATA 127, 9, 235, 225, 9, 229, 213, 14, 0, 126
50140 DATA 71, 26, 184, 48, 3, 14, 1, 71, 175, 176
50150 DATA 40, 25, 197, 19, 35, 78, 35, 70, 197, 225
50160 DATA 235, 78, 35, 70, 197, 225, 193, 26, 150, 56
50170 DATA 10, 32, 83, 19, 35, 16, 246, 203, 65, 32
50180 DATA 75, 237, 75, 252, 127, 175, 225, 237, 66,
      235
50190 DATA 225, 237, 66, 58, 250, 127, 71, 197, 24, 13
50200 DATA 197, 237, 75, 254, 127, 27, 27, 43, 43, 9
50210 DATA 235, 9, 235, 78, 235, 126, 113, 235, 119,
      35
50220 DATA 19, 78, 235, 126, 113, 235, 119, 35, 19, 78
50230 DATA 235, 126, 113, 235, 119, 193, 16, 218, 42,
      246
50240 DATA 127, 235, 42, 238, 127, 175, 237, 82, 34,
      238
50250 DATA 127, 210, 55, 127, 24, 2, 209, 225, 42, 240
50260 DATA 127, 17, 1, 0, 175, 25, 34, 240, 127, 237
50270 DATA 91, 242, 127, 237, 82, 218, 49, 127, 195,
      15, 127
```

Model II Version Sort

The following information will allow Model II users to use this month's machine language sort routine. Please read the article under Model I/III for information about the sort itself. Here we will present the information you need to convert the machine language routine to Model II, as well as present you with a BASIC program which will demonstrate the sort.

If you wish to assemble the sort procedure, make the following changes to the assembly listing shown in the Model I/III section:

1. Change the origin (ORG) from 7EDFH to F000H in line 110. If you wish to assemble relocatable code, (using our 26-4702 Editor/Assembler), leave out the ORG statement and use this address (or your own) with the linking loader.

2. Delete the CALL 0A7FH statement in line 130. In its place put three (3) NOP statements.

3. At the end of the listing, delete the END statement and ADD the code shown in listing 1.

4. Assemble and debug the code. For the sample BASIC program, you should create a disk file with the filename "SORT/CIM"

If you want to simply enter the code into memory, and are not concerned with relocating it from F000H, Listing 2 gives the Hex values to enter using DEBUG. For details on using DEBUG, see your TRSDOS manual. The starting address is F000H.

After you have finished entering the Hex values using DEBUG, exit DEBUG using the O option. DUMP the code to disk using the command line shown in listing 3.

Now enter and save the BASIC program shown in listing 4. This is the sample program which demonstrates the method of using the sort routine.

When you load BASIC, set memory using BASIC -M:61439. In your programs, the lines equivalent to 260 - 320 should not be modified, nor should anything be added to them.

Have fun, and happy sorting! Our thanks again to Allen Emert who provided the original machine sort in our July 1980 issue, and to William Terrell who modified that program to produce this new version. We would also like to acknowledge Tom Mornini of Fair Oaks, CA who suggested we publish the assembly listing for Model II, and whose disassembled listing helped us in providing the above information.

Listing 1.

Add this code to the end of the assembly language version of the sort routine:

```
F121 2100F0        02070 START     LD      HL,ENTRY
F124 E5            02080           PUSH    HL
F125 2A0328        02090           LD      HL,(2803H)
F128 E9            02100           JP      (HL)
F129 FE4F          02110           CP      4FH
F12B 2802          02120           JR      Z,PART19
F12D CBC8          02130           SET     01H,B
F12F DD7E04        02140 PART19    LD      A,(IX+04H)
F121               02150           END     START
```

Refer to your Editor/Assembler Users Guide for particular techniques and punctuation requirements.

Listing 2.

Here is a Hex listing of the program if you are using DEBUG to enter the machine code. Note that the first four digits are the Hex beginning address for that line of entries:

```
F000  00 00 00 5E 23 56 ED 53  2B F0 23 5E 23 56 ED 53
F010  19 F1 23 5E 23 56 ED 53  1B F1 23 5E 23 56 ED 53
F020  1D F1 23 5E 23 56 ED 53  1F F1 21 00 00 22 17 F1
F030  ED 5B 17 F1 CB 3B AF CB  3A 30 02 CB FB ED 53 17
F040  F1 7A B3 C8 2A 2B F0 ED  52 22 13 F1 21 00 00 22
F050  11 F1 2A 11 F1 22 0F F1  2A 0F F1 ED 5B 17 F1 19
F060  22 15 F1 EB 21 00 00 19  19 19 E5 ED 5B 0F F1 21
F070  00 00 19 19 19 ED 4B 19  F1 09 EB E1 09 E5 D5 0E
F080  00 7E 47 1A B8 30 03 0E  01 47 AF B0 28 19 C5 13
F090  23 4E 23 46 C5 E1 EB 4E  23 46 C5 E1 C1 1A 96 38
F0A0  0A 20 53 13 23 10 F6 CB  41 20 4B ED 4B 1D F1 AF
F0B0  E1 ED 42 EB E1 ED 42 3A  1B F1 47 C5 18 0D C5 ED
F0C0  4B 1F F1 1B 1B 2B 2B 09  EB 09 EB 4E EB 7E 71 EB
F0D0  77 23 13 4E EB 7E 71 EB  77 23 13 4E EB 7E 71 EB
F0E0  77 C1 10 DA 2A 17 F1 EB  2A 0F F1 AF ED 52 22 0F
F0F0  F1 D2 58 F0 18 02 D1 E1  2A 11 F1 11 01 00 AF 19
F100  22 11 F1 ED 5B 13 F1 ED  52 DA 52 F0 C3 30 F0 FF
F110  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
F120  FF 21 00 F0 E5 2A 03 28  E9 FE 4F 28 02 CB C8 DD
F130  7E 04 FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
```

Listing 3.

Use this command line to DUMP a copy of the machine code to disk with the file name "SORT/CIM":

```
DUMP SORT/CIM START=F000, END=F140, TRA=F121, RORT=R
```

Listing 4.

```
10 CLS
20 T$="": T1$=""
30 SYSTEM "LOAD SORT/CIM"
40 CLEAR 23000, 61439
50 DEFUSR= &HF121
60 DEFINT A-Z
70 INPUT"NUMBER OF DATA SETS"; ZR
80 INPUT"NUMBER OF FIELDS IN EACH SET"; ZF
90 DIM A$(ZR,ZF), Z(4)
100 Z=0
110 FOR I=0 TO ZR-1: FOR J=0 TO ZF-1
120   A$(I,J)="": NEXT J,I
130 FOR I=0 TO ZR-1
140   ZA=5: PRINT I;
150   FOR J=0 TO ZF-1
```

(Continued on page 20)

Radio Shack®
MICROCOMPUTER NEWS
P.O. BOX 2910
FORT WORTH, TX 76101

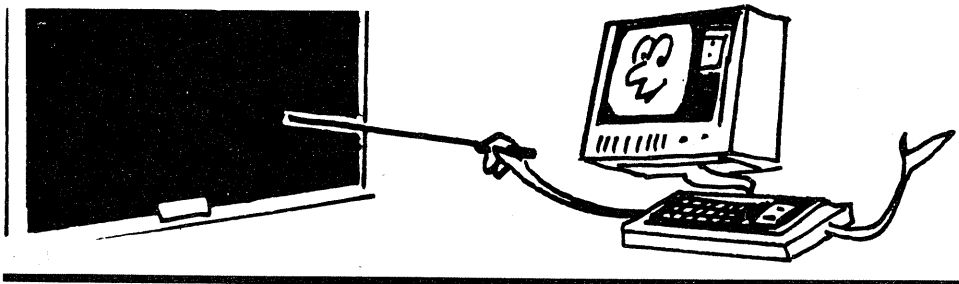
BULK RATE
U.S. POSTAGE
PAID
Radio Shack
A Div. of Tandy Corp.

ADDRESS CHANGE
☐ Remove from List
☐ Change as shown
Please detach address label and mail to address shown above.

IF UNDELIVERABLE DO NOT RETURN

Model II Sort Listings (from page 19)

```
160     FOR K=1 TO RND(10)
170       A$(I,J)= A$(I,J)+ CHR$(RND(26)+ 64)
180       NEXT K
190     PRINT TAB(ZA); A$(I,J);
200     ZA= ZA+ 12: IF ZA>70 THEN PRINT: ZA=5
210     NEXT J
220   PRINT: NEXT I
230 INPUT"SORT ON WHICH FIELD (ZERO IS 1ST)"; ZC
240 INPUT"PRESS ENTER WHEN READY"; ZZ
250 T1$= TIME$
260 Z(0)= ZR
270 Z(1)= VARPTR(A$(0,ZC))
280 Z(2)= ZF
290 IF Z(1)<0 AND VARPTR(A$(0,0))>0 THEN Z(3)= 65536+
    Z(1)-VARPTR(A$(0,0)): GOTO 310
300 Z(3)= ABS(VARPTR(A$(0,ZC))-VARPTR(A$(0,0)))
310 Z(4)= VARPTR(A$(0,1))-VARPTR(A$(0,0))
320 Z= USR0(VARPTR(Z(0)))
330 T$= TIME$
340 FOR I=0 TO ZR-1
350   PRINT I;
360   K=5
370   FOR J=0 TO ZF-1
380     PRINT TAB(K); A$(I,J);
390     K=K+12: IF K>70 THEN K=5:PRINT
400     NEXT J
410   PRINT
420   NEXT I
430 PRINT
440 PRINT T$, T1$
450 PRINT"FOR ANOTHER SORT ON THE SAME DATA, ENTER
    '1'"
460 PRINT"FOR A NEW DATA SET, ENTER '2'"
470 INPUT"TO QUIT, ENTER '3'";ZZ
480 ON ZZ GOTO 230, 40, 490
490 END
```



COMPUTER SERVICES ADDRESS
AND PHONE NUMBERS
8 AM to 7 PM Central Time
Computer Services
900 Two Tandy Center
Fort Worth, Texas 76102
1-800-433-1679 (WATS Except Texas)
1-800-772-5914 (WATS Inside Texas)
1-817-390-3583 (Switchboard)

TRS-80 Microcomputer News is published monthly by Radio Shack, a division of Tandy Corporation. A single one year subscription is available free to purchasers of new TRS-80 Microcomputer systems with addresses in the United States, Puerto Rico, and APO or FPO addresses. Subscriptions to other addresses are not available. The subscription rate for renewals and other interested persons is twelve dollars ($12.00) per year, check or money order in U.S. funds. All correspondence related to subscriptions should be sent to: Microcomputer News, P.O. Box 2910, Fort Worth, Texas 76101.

Back issues of Microcomputer News are not available.

The TRS-80 Newsletter welcomes the receipt of computer programs, or other material which you would like to make available to users of TRS-80 Microcomputer systems. In order for us to reprint your submission, you must specifically request that your material be considered for reprinting in the newsletter and provide no notice that you retain copyrights or other exclusive rights in the material. This assures that our readers may be permitted to recopy and use your material without creating any legal hassles.