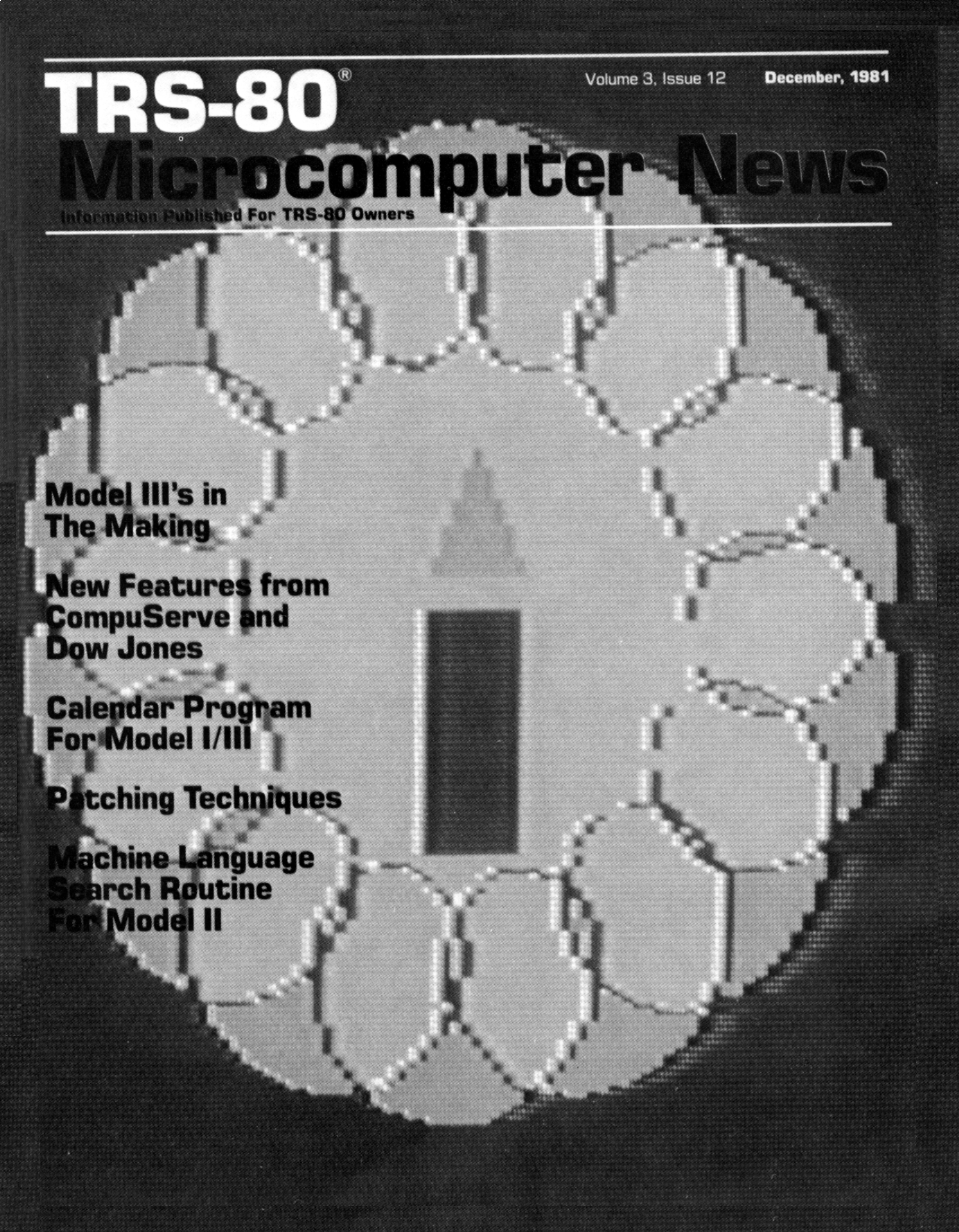
Volume 3, Issue 12 December, 1981

TRS-80® Microcomputer News

Information Published For TRS-80 Owners



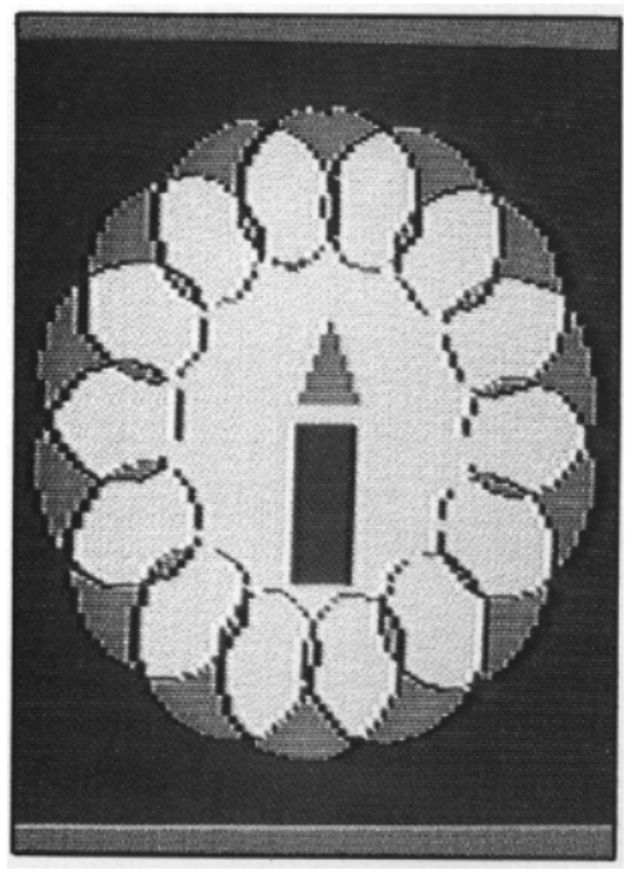
Model III's in The Making

New Features from CompuServe and Dow Jones

Calendar Program For Model I/III

Patching Techniques

Machine Language Search Routine For Model II



The cover photo was taken from a screen created by "A Christmas Wreath and Song" written by Mr. Gerald Plueard of Seattle, Washington. Mr. Plueard's program is listed on page 33 of this issue.

TRS-80® Microcomputer News

Volume 3, Issue 12 **December, 1981**

Information Published For TRS-80 Owners

Contents

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, Canada 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 76113-2910.

Retail Prices in this newsletter may vary at individual stores and dealers. The company cannot be liable for pictorial and typographical inaccuracies.

Back issues of Microcomputer News prior to January, 1981 are available through your local Radio Shack stores as stock number 26-2115 (Suggested Retail Price $4.95 for the set). Back issues of 1981 copies 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.

Material may be submitted by mail to P.O. Box 2910, Fort Worth, Texas 76113-2910, or through CompuServe. The Microcomputer News' CompuServe user ID number is 70000,535.

CompuServe™	CompuServe, Inc.
Dow Jones™	Dow Jones & Co.
ReformaTTer™	Microtech Exports
Time Manager™	Image Producers
VisiCalc™	Personal Software, Inc.
Program Pak™	Tandy Corporation
SCRIPSIT™	Tandy Corporation
TRSDOS™	Tandy Corporation
TRS-80®	Tandy Corporation

View From the Seventh Floor

by Jon Shirley,
Vice President Computer Merchandising

I have been reading magazines again and for those of you with a Color Computer and a very technical mind there is one called '68' MICRO JOURNAL. It has a lot of information on the more complex 68xx processor based micros that use the FLEX operating system but, since we use the 6809 in the Color Computer they have been giving it some space and there are some ads for machine language programmers aides. The address is PO Box 849, Hinson, TN 37343.

In the October issue there is a column on the Color Computer that chides us for providing "meager" information and not providing a listing of the ROM code. For those of you who want to play with machine code we have expanded the "meager" information with our new technical reference manual for the CC, it's $14.95 and should be available as you read this. It contains full information on the 3 LSI chips that accomplish all the neat things the CC can do, the 6809, the 6847 video generator and the "SAM" chip that connects the others together and handles the memory. This is *not* for the faint at heart, and does not concern itself with BASIC. As far as the ROM code goes, and the article calls it "classified-highest possible top secret", just as it says on the screen, that code belongs to Microsoft and they choose not to let us or any other of their customers publish the fruits of their labor. Enough said.

It seems that some of you do a better job of keeping up with our competition than we do. To all of those who wrote protesting a very unfair comparison of the Model III with the Heath Z-89 in the fall Heathkit catalog, many thanks. Your letters resulted in a letter from me to Heath Co. which in turn resulted in a very nice apology from the president of Heath, Mr. William Johnson, who stated that the offending comparison chart would be destroyed and replaced with a new one. Just as we try very hard to avoid unfair comparisons I certainly believe they do too.

One item that should be in the Computer Centers by now that is worth a look is our new 6 color plotter. This is a fun item that I have been playing with lately but it is also a serious business tool. It can be fitted with two types of pens, one for plotting on ordinary paper and the other a felt pen that will write on a transparency that is great for overhead projectors. It is also a bargain compared to those offered by some of the famous names in business graphics. We have no charge software available for Models I/II/III (disk) that produces bar and pie charts automatically. It is also very easy to program in BASIC.

If you are a Color Computer owner all our stores have a bunch of great new games just in time for Christmas. They include Space Assault, Polaris and Project Nebula and they are all super games. For the kids and adults, Art Gallery lets you do great graphics without programming but if you want to learn how to do graphics with programming try the Computer Learning Lab. And for you handyman types we have a program appropriately named Handyman that is really useful. If you don't find what you want there is lots more coming for the Color Computer including Spectaculator and Color Scripsit which should be in stock by late this month.

The Color Computer has also given birth to a disk system and we started shipping them in October. Disk software will follow next year, there are a lot of packages in the works.

I got a letter recently saying I should entitle this column "view from the basement." Seems the customer had some problems with service. Well I will be the first to admit that we are not perfect, (and neither is anybody else), but we really do care about the quality of our service and every such letter gets personal attention. We follow up with the repair center to get the problem solved as fast as possible. If you feel you have a repair problem that is not getting solved ask the store manager to call the repair coordinator for the area. He is the boss of several repair centers and he knows that our policy is get the customer up and running in as short a time as possible. On the other hand you must understand that occasionally we do have a parts availability problem or a sick computer technician so delays will occur but TRS-80's do get priority in our repair system as we know how important they are to many of you.

For several years now we have been asked why we do not use our computers in our stores. It is a good question and the real answer is that we have had a master plan to do that for some time but it took both availability of equipment, a lot of software and many changes to our mainframe software to make it possible. It all started to come together last spring when the first test installations were made in selected stores. The test has proven the system is great and during the first half of this year (1982) we will install a system in every one of our 4185 USA stores.

This system will consist of a Model III with 3 disks and 48K, a Line Printer V and our new auto answer modem. The store will send us orders, payroll information and daily sales report information and we will send them back acknowledgements of orders and other information. The transmittal will be done automatically at night. The computer will be left on with the disks in it and our mainframe will call up every store and download the information. What makes it really work well is that we are using a nation wide network that takes in the information from a local call at 300 baud, stores it up then sends it at a high baud rate over special lines to Fort Worth. All the software is in machine code to make the programs run as fast as possible.

Sometime over the next five to six months the stores in your area should get this system. It will be located where you can see it and I hope a side benefit is that it will help all our store personnel to become more expert on our computers. Another benefit we see is better inventory control on all our products so we will have the item you want in stock when you want it. And, by the way, while this project may have taken a long time to implement don't get the idea that we do not use our computers internally. There are literally hundreds in our offices and factories in Fort Worth running both our regular software like Scripsit, Profile and VisiCalc and a lot of software created by our data processing department for specific jobs.

Until next month...

Model III's in the Making

The first time that I saw a Model III, it was perched in the middle of a long conference table. It was so new that no one in the room knew how to run it. The CASS? prompt proved to be mind boggling and sent us all scurrying to find an operator's manual. As the Model III sat there glimmering in the late afternoon sun, all snug in it's attractive case, sporting internal disk drives, I knew that I had to have one of my own.

My Model III sits unobtrusively wherever I choose to put it, waiting patiently and quietly for my husband, my nine year old daughter or me to come along and tell it what to do. It is no respecter of persons and treats us all equally. If we are wrong it tells us, and if we are right it rewards us with obedience. It never tries to overwhelm us with it's vast capabilities, which still far exceed our expertise, but just waits quietly for instructions.

Whether word processing, games, graphics, business applications, or a multiplication table drill, it is ready and willing to do our bidding. It is so uncomplaining that I was more than a little surprised at what a grueling life it had before we took it in.

Recently, I visited the former home of my computer, the Model III plant in Fort Worth. There are other Model III plants but this one produces Model III's with disk drives. I couldn't help but be a little awed by the enormity of the plant facility—approximately 96,000 square feet. Could my computer be happy in the—by comparison—humble home in which it now found itself? Once inside I felt reassured because though the building is very clean and spacious, it lacked some of the amenities such as carpeted rooms, quiet, and most of all privacy. There were several hundred people employed at the plant. I had the feeling that I had somehow landed in the middle of a giant ant hill, and everywhere there was intense activity.

I knew when I met the Plant Manager or P.M. that it was a steady hand that had steered my computer's course. Though he had a kind face and a pleasing manner, I soon learned that he was a man of iron discipline—no easy task master here. Those computers had to do their stuff and do it right every time if they wanted to make it in the real world.

In one of the areas that I visited, the disk drives were undergoing rigorous alignment and functional testing. The alignment testing is to make sure that the mechanics of the drive are working properly. The alignment, tracking ability, azimuth of the head, write and write protect features of the drive are all tested. Additionally, the door, pulley, appearance, and other items are checked.

The functional test, a procedure which requires about 15 minutes, tests the marriage of the electronics of the drive to the electronics of the CPU. In the functional test complete Backups are made from Drive 0 to Drive 1 and from Drive 1 to Drive 0; the write protect feature is tested; the motor speed is tested; the read speed is checked by writing a certain track to check for read, write, and seek errors. If a drive fails here it is sent to be repaired and when it returns it must complete the whole process again.

The various parts of the computer are sent to areas of subassembly, progressing from one stage to another until all the components come together at one of the four main assembly lines. Some parts go through extensive testing before ever reaching the line.

The CPU, or mother board, which is a Tandy product, probably undergoes the most extensive testing. After arriving at the plant, it is "stuffed" with the Z-80 microprocessor, RAM, ROM, video RAM, and the character generator. After each step in the stuffing process the individual component is tested against a known good board to be certain that it is functioning correctly. When this testing is successfully completed, the CPU board goes to the burn-in area.

Burn-in, though it sounds like shades of the Inquisition, is simply a stress test for the board. All of memory is written to; the content of memory is verified for accuracy; and it is written to again. This goes on over and over for a full twenty-four hour period. If a chip is going to fail it will likely happen at this point.

If the board survives the burn-in (and only a very small percentage do not) it goes into the system. By now the mother board has completed two pretty extensive tests — the actual board comparison test and the burn-in. As a side note I noticed that in the many areas where computers were used to do the testing on the Model III's, the testing computers were Model III's.

The board goes onto the main assembly line where assembly commences. At the first of the line the base of the computer is put on a tray that has wheels which will be with it until it is boxed to go out the door. This helps to prevent it from receiving damage from lifting or unnecessary moving about.

Finally, all the various parts of the computer all come together. The CPU board, floppy disk controller board (FDC), and power supplies are installed in the chassis along with the mother board. The power supply, keyboard, and ribbon cables are also added.

When the boards are installed, the computer goes through another functional test like the one given to the drives. After the monitor is hooked up, still another test is run to see if the drives and boards will work with the video attached. The logo is put on the case, the case is set on the base, and a test for video alignment is run.

Now, with all its components together, the computer undergoes its first full system test. It is given a functional test where the drives are tested; the cassette and disk loading and saving are tested; the video and keyboard are tested. Then it goes for a burn-in (similar to the one on the mother board but this time it's done with a full system). After burn-in, the computer goes to a screw down station where the case is screwed together and the labels are added. Subsequently, the Model III completes a high voltage test to ascertain that nothing in it will short out.

Next it goes to production quality control where it goes through another complete system test. After this it undergoes a cosmetic inspection for any rough places in the exterior that need to be sanded or painted. Any corrections or touch ups are made here.

At this point the Model III is checked by someone from National Quality Control (a quality assurance group that is part of Radio Shack but works independently of the plant) on the assembly line where it undergoes — if you can believe it — another full system test. After this the computer heads for packing where it is given another visual inspection, packed into heavy cardboard boxes with the necessary manuals, and loaded onto Radio Shack trucks to be delivered to Regional Warehouses around the country.

When the Model III arrives at the Regional Warehouse, it is pulled out of the box where it is put through another full system test. If the computer survives this system test it then goes for another twenty-four hour burn-in. Is there no end?

Finally after all this it is boxed up and sent to the stores so that it can eventually find it's way into our homes and our hearts — a well disciplined, obedient, intelligent member of the family. I wonder if the P.M. would consider taking on a nine year old?

TRS-80 Computers and Serial Printers

This month, the Customer Service Article comes from the Hardware group. First I would like to explain what the hardware group does. We are responsible for questions on the hardware for all of the TRS-80 computers and computer related Radio Shack hardware such as line printers, disk drives, telephone modems, etc. We also handle Radio Shack communications software. Now that you know what we do, let me get on with the article.

First we'll deal with connecting serial printers to our computers.

MODEL I

The Model I is one of the easiest of our computers to connect to a serial printer. You simply plug in the printer and run the DecWriter program which comes in the RS-232C manual. Remember though, that this program is an example only, and it may need to be modified in order to work with your printer. Although we would like to help you modify the program to work with your Brand X printer, it is impossible for us to be familiar with every make of serial printer and all the finer points of the operation of each. However, there are some changes to the DecWriter program which we can give you that may be helpful. The DecWriter program is written for a 16K machine. In order to have it operate with the maximum memory available in your computer make the following changes:

For 32K Computers:

Memory Size? 48895 (Answer the Memory Size prompt with 48895)

Change Line 20 to:

```
2Ø POKE 16421,2:POKE 16422,Ø:POKE 16423,191
```

Change Line 40 to:

```
4Ø FOR X=-16639 TO -16567
```

DATA Lines:

Change all occurrences of 127 to 191.

For 48K Computers:

Memory Size? 65279 (Answer the Memory Size prompt with 65279)

Change Line 20 to:

```
2Ø POKE 16421,2:POKE 16422,Ø:POKE 16423,255
```

Change Line 40 to:

```
4Ø FOR X=-255 TO -183
```

DATA Lines:

Change all occurrences of 127 to 255.

MODEL II

This is the easiest of our computers to connect to a serial printer. All that is required is to:

1. Connect the printer to Channel B
2. Terminate Channel A (with a terminator plug)
3. Set up the communication parameters with SETCOM
4. Do a FORMS S.

That's all there is to it. Now, we have had some calls with people getting PRINTER NOT READY after a FORMS S. This is usually because the printer is not responding to Pin 5 on the RS-232 (clear to send). To correct this, check the pin-outs of your device to see if it responds to pins 4, 5, 6, & 8. If any of these pins are not responded to, tie them to pin 20 on the RS-232 (usually tying pin 5 is enough).

MODEL III

Connecting a serial printer to the Model III is more difficult because you will have to write your own driver since there is not a driver program for the Model III currently available. It is possible, to use the Model I DecWriter program for a starting point. (This program is contained in the Operator's Manual for the RS-232C, part no. MU2601145 available from National Parts for $7.50.) Model I output to the line printer is memory mapped so all your printer driver has to do is put the current byte to be sent to the printer in memory location 14312 and the information will be printed. In the Model III the line printer is port mapped (Port F8 for parallel, use E8-EB for serial — see your service manual, stock #MS2601061/62/63 for bit allocations). Because of this difference you may want to set up your serial printer driver to use the Model III ROM calls as described in the Model III Reference manual, pages 75-77.

SMART OR DUMB TERMINALS

We have had many questions on communications packages for the different computers, and if they are smart or dumb. The new Communications Package (26-1149) is capable of making your Model I or III function as a HOST or as a Terminal, but supports transfer of files only between Models I and III (any combination). The Model II TERMINAL Utility (supplied by TRSDOS) incorporates a RAM buffer which allows you to send data to or receive data from a Host computer (and store it on disk for later use if you want). Both of these programs are relatively "smart" in that they can send and receive files of information (not just what you type at the keyboard, but programs and data, too). The Videotex programs can be considered "dumb" since they only send to the Host what you type and display what is sent back to you on the screen (Model I/III and II versions can send output to the printer, too). All of these packages require:

For the Model I:	16K Level II 0K to 32K Expansion Interface RS-232C Telephone Interface II or DC Modem I or DC Modem II.
For the Model II:	32K or 64K Mod II Telephone Interface I or DC Modem I or DC Modem II.
For the Model III:	Level III RS-232C (Included with 26-1066) Telephone Interface II or DC Modem I or DC Modem II.

As you'll notice all of these selections require an RS-232C. Cassette Com (26-1139) which runs on a Model I with a DC Modem will work without the RS-232C. Information on all of this hardware and software is in the RSC-6 Computer Catalog.

Model I/III Host utility comes with Communications Package (26-1149) and Model II Host utility comes on the 2.0a operating system. Host programs are not available at the time of this writing for our other computers.

MODEL I LEVEL II ROM

Another subject we have had many questions on is the old and new Mod I Level II ROM's. This was handled in an earlier newsletter (July 1980, page 2) but we shall go over it again. The new ROM's can be distinguished from the old by the power up messages.

OLD ROM:	NEW ROM:
MEMORY SIZE? RADIO SHACK LEVEL II BASIC	MEM SIZE? R/S LII BASIC

Another difference is that a PRINT MEM at power up on the old ROM 16K unit will return 15572 whereas the new ROM will return 15570. New ROM's also cannot CLOAD from 2 cassettes, but can output control codes, and already has the KBFIX program built in.

The only problem we've had with the new ROM has been that some machine language tapes will not load. This is caused by the timing of the material on the tape. If you get a tape which does not load, or which loads with the old ROM, but not the new one, you can either have someone make a copy from an old-ROM or request a replacement. If the tape is made by another Mod I Level II, old or new ROM it should work fine. So if you have one of our machine language tapes, and it does not load, then have your nearest Radio Shack store exchange the tape for a new ROM version. An alternate solution would be to copy the program from an old ROM Mod I using T-BUG or DEBUG on tape. If you have programs from other companies either contact the manufacturer or use the second method described above.

FREQUENTLY ASKED QUESTIONS:

Question 1: I have a Radio Shack computer and am moving to Europe. Will it work overseas?

Answer: In general, if the destination country has 60 hz power, you can use a voltage converter to obtain 110-117 volts. Since most countries outside of the United States are 50 hz this procedure will not work. For the Model I only, you can purchase new power supplies overseas for the Keyboard/CPU and the Expansion Interface so they will work. You must purchase a new video and desired peripherals overseas to obtain 50 hz operation. No 50 hz equipment of any kind is available for sale in the U.S.

Question 2: I would like to have a schematic of my computer, where can I get one?

Answer: You can purchase a Service Manual for any of our computer devices using the stock number of the device preceded with an MS and the 26 followed by a zero. For example, the Service Manual for the video on the Mod I, which has a stock number of 26-1201, would be MS260-1201. The Mod III is MS260-1061/62/63. (The Mod III has the trailing numbers because there are three different kinds of Mod III's.) These Manuals have complete schematics, diagrams, port descriptions with bit allocations, etc. Essentially, these Manuals will contain as much information on the device as you can get in documented form. Additional information is contained in the owner's manual which comes with the device.

Question 3: What kind of protocol is required to communicate with serial printer, host computer, or any other device?

Answer: Initially you must determine if the device being communicated with is RS-232C, ASCII format, and has a TRS-80 compatible baud rate (usually from 110 to 9600). Once you have made this check read the above article to see what configuration is right for you.

Question 4: What is a smart or dumb terminal?

Answer: Actually there is no real definition of smart or dumb. I have defined "smart" as having the capability to store received data from memory to tape or disk, or to send data from tape or disk to a Host computer. Anything less than this we consider "dumb." This definition is not absolute, and others could define "smart" as being able to output received data to a printer or as being able to send control codes.

Question 5: I would like to get a listing of the ROM for the Mod I or III, where do I go?

Answer: This question is asked of us quite often and our reply is always the same. "We have no listing of the ROM because it is not ours. We lease it from Microsoft and they do not provide us with a listing of it." You may ask, "What about the ROM calls you have provided?" The ROM calls in the manual were provided to us by Microsoft. They are the only documented ROM calls which we intend not to change. All other addresses are subject to change which is part of the reason we don't publish them. Therefore, the published addresses are the only ones which we can support.

We have no secrets up our sleeves as most of you think. For help in this area, contact any club or user group in your area. They would probably be the best source of un-documented ROM calls.

COMPUTER CUSTOMER SERVICES ADDRESS AND PHONE NUMBERS

8 AM to 5 PM Central Time

Computer Customer Services
400 Atrium, One Tandy Center
Fort Worth, Texas 76102

Model I/III Business Software

Outside Texas	1-800-433-5641
In Texas	1-800-772-5973

Model II Business Software

Outside Texas	1-800-433-5640
In Texas	1-800-772-5972

Education Software

Outside Texas	1-800-433-1679
In Texas	1-800-772-5914

All Other Calls Related to Computers

Outside Texas	1-800-433-1679
In Texas	1-800-772-5914

Switchboard — 1-817-390-3583

Space War, Quick Quote and Value Line II

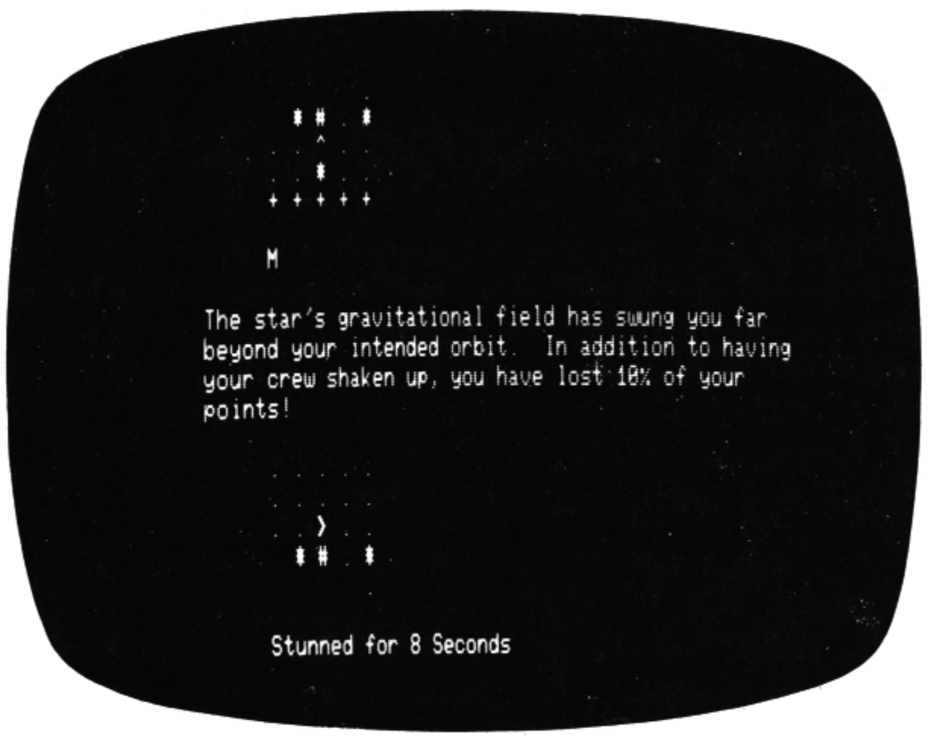
Editor's Note: The CompuServe Information Service is one of the largest information and entertainment services available to owners of personal computers and computer terminals. With each issue of TRS-80 Microcomputer NEWS, various features of CompuServe will be discussed. The CompuServe Information Service is sold at Radio Shack stores nationwide and in Canada.

SPACE WAR: CONQUERING THE UNIVERSE HAS NEVER BEEN SO MUCH FUN

Darth captains his spaceship, the Indicator, through the mysterious, murky and ever-surprising universe. It is a universe quite unlike any the world has known before, filled with enemy spaceships, crippling barriers and crystalized clouds. Every second, Darth must search his universe for the danger that lurks within.

The Indicator must go on the attack, lest the enemies gain any advantage. Darth, as captain, must maneuver with the speed and finesse of a veteran commander . . . moving, moving, then firing the Indicator's rockets at enemy positions. It is a cunning game of cat-and-mouse, a deadly game of hide-and-seek. Only the best will survive.

Darth checks his computer and finds that the Indicator's reactor power is running dangerously low. He zooms back to his starbase, the only home the noble warrior has ever known. There to refuel, he may remain at the starbase for microseconds only. Any longer, and the universe gods may transport him away from the starbase to any place of their choosing. Perhaps a galaxy far, far away . . .



It is not a pleasant prospect, so Darth times his refueling stop precisely and goes forth to the Space War once again. Ah, but our young commander is too hasty in his piloting decisions, and the Indicator smashes into a star. The collision results in disaster for Darth: "The star's gravitational field has swung you far beyond your intended orbit . . . your crew is badly shaken up, and you have lost 10 percent of your points," reads the message on Darth's transmission screen.

Darth shudders; in this lifelong battle against his enemies in a treacherous universe, his points are the only thing that make life worth living. Darth quickly recovers, however . . . darting here, darting there, ever the elusive target. The Indicator's radar suddenly detects a nearby cloud. Darth deftly slips inside, there to rest and watch and wait. His enemies are outside, searching for him . . . but Darth will bide his time, savoring his cloud cover and waiting for his enemies to venture past his hiding post. Fire! Fire! Move! Move! Fire!

* * * *

It is 6:45 p.m. and "Darth" signs off for the night. But Space War continues . . . and the players include Yoda and Luke and, well, use your imagination. That latter quality is perhaps the most important one in playing this computer game. Imagination . . . and lightning quick reflexes.

Space War is accessed under main menu item 3, Entertainment, on the CompuServe Information Service.

The game is played on a display grid, "the universe," measuring some eight points across and eight points down. The points may be open, permitting players to move about freely on them, or they may be occupied by a certain symbol, letter or number, each indicating some other presence. A "+" symbol indicates a barrier, an "*" means a star, while a "B" indicates a starbase, and "C" a cloud. Other players — up to eight individuals may play on separate terminals — are indicated by an assigned number (1, 2, and so on.)

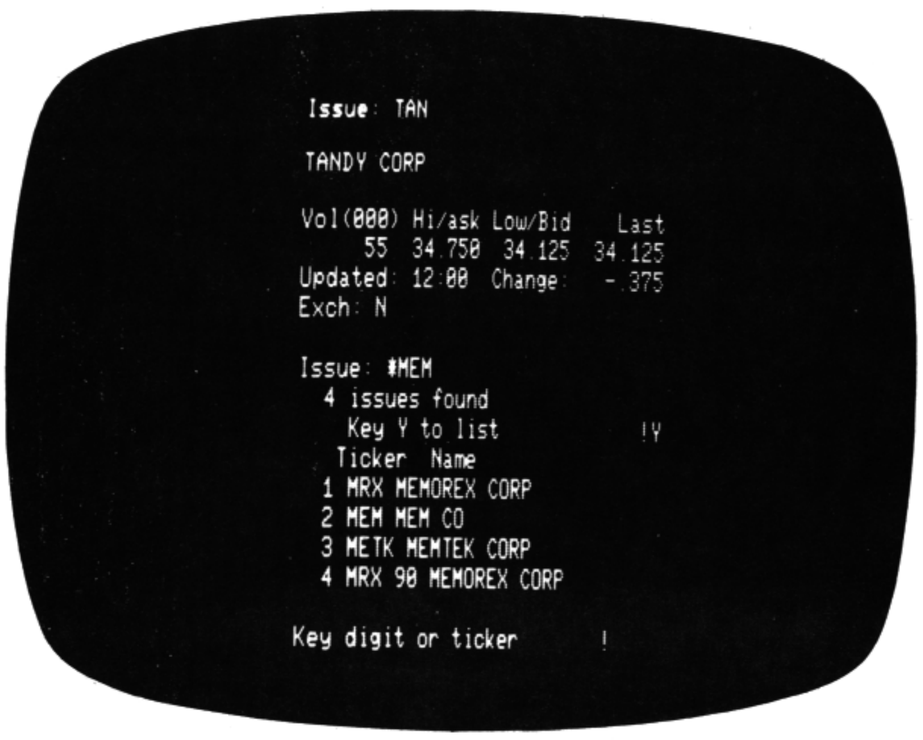
The object of Space War is to score as many points as possible by firing "hits" against individual opponents, while avoiding their lines of fire. One moves his spaceship about the universe by giving it directional signals via the keyboard: "R" to point right, "L" to point left, and "M" to move forward. During the course of Space War, ramming — actually moving onto the same point as an opponent — is heavily penalized.

Speed is of the essence in Space War, as is quick decision-making. Game players who crave continuous action find that Space War is just the ticket, because each player determines how much or how little his spaceship will move. Players who wish to "slow down" temporarily may do so by returning to the starbase; there, the player's spaceship will be "stunned" for 10 seconds, after which it must move on.

Space War aficionados note that the game is the only one of its kind, in that it allows multiple players to participate. These players, hailing from all parts of the U.S., generate true "interaction" on the computer. Space War also provides a vehicle for communication; you may talk to your opponents—to ask questions, to comment on a particularly worthy move by a player, to trade "fleet commander" instructions. A word of caution, however. He who talks at length during Space War play can easily lose track of the action, and be "fired" right out of the universe!

QUICK QUOTE: INFORMATION UPDATED THROUGHOUT THE TRADING DAY

If you follow the stock market, you are undoubtedly accustomed to turning to the financial pages of a newspaper for the latest figures.



The trouble is, those figures are sometimes not the very latest. Information in the newspapers is typically a day old. How to avoid the delay? Use the CompuServe Information Service Quick Quote system.

Quick Quote provides high, low, closing, volume and net change figures on over 9,000 securities traded on the New York and American stock exchanges and over-the-counter. The information is updated periodically during each trading day.

CIS users can thus receive current information on securities of particular interest at a cost of 2 cents for each current price extracted from the database.

The process is also easier than searching through one or more newspaper stock tables. If you know the ticker symbol or CUSIP number for a security, you simply input it to Quick Quote.

Suppose, however, that you don't know the ticker symbol, are not sure of the exact name and do not know if the security is traded on the NYSE, AMEX, or OTC. You can direct Quick Quote to do a "name search" in which you type in the first few letters of a name. Quick Quote will tell you how many companies have names that start with those letters and will ask you if you want to see a list of them. You can then pick the desired company from the list.

Once you have found the appropriate security, Quick Quote will display the name of the issue, where the stock is traded (NYSE, AMEX, or OTC), the volume, high, low, and last trading price for the day. If there is no current quotation for a security in the database, Quick Quote will display the most recent information and its date and time.

MicroQuote users on CompuServe can access Quick Quote to obtain current day prices in addition to the historical pricing information available in MicroQuote.

Quick Quote's timely pricing information is also useful in conjunction with data from the Standard & Poor's General Information File and/or the Value Line Database II, both of which are also available on the CompuServe Information Service.

VALUE LINE DATABASE II: KEEPING TABS ON TRW, TANDY AND OTHERS

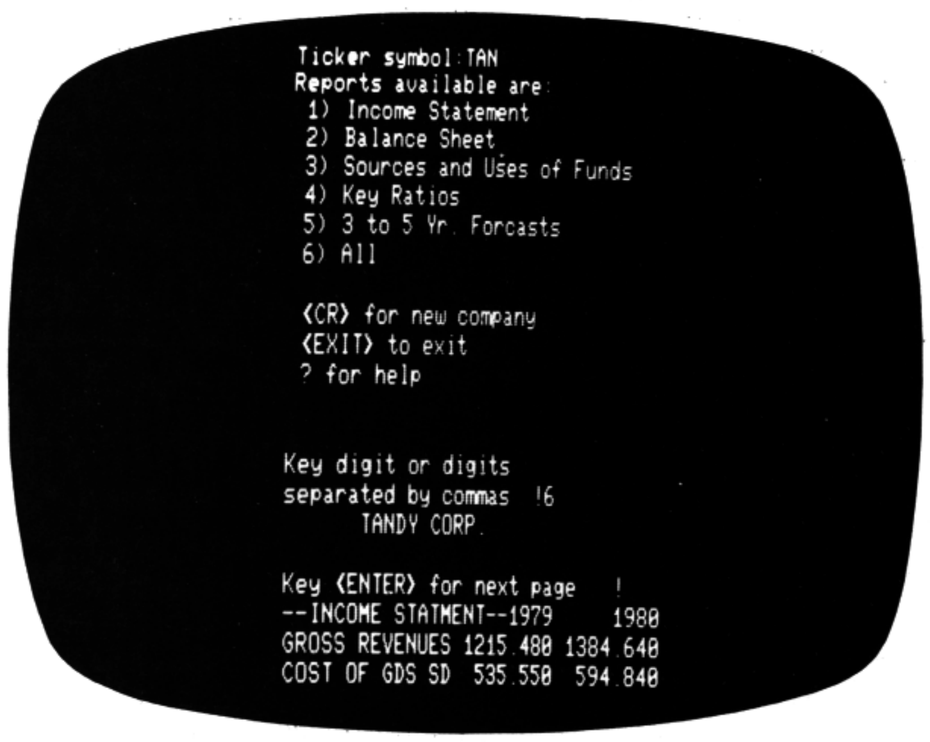
Imagine having at your fingertips detailed financial data on over 1,600 companies. Imagine having access to over 400 different items of information on each of those companies. Imagine the Value Line Database II, available today on the CompuServe Information Service.

With Value Line information, you can analyze the performance of more than 1,600 major industrial, transportation, utility, retail, banking, and insurance companies.

There are also more than 100 industry composites to assist you in analyzing the relative strength or weakness of a particular company's performance.

The Value Line Database II contains detailed data from historical annual income statements, balance sheets, sources and uses of funds, as well as reported quarterly results.

You can also obtain Value Line's earnings estimates, target price forecasts and risk measures (BETA), sales and earnings by business line and precomputed commonly used ratios and per share figures.



The Value Line Database II is a product of Arnold Bernhard & Co., who is perhaps best known for its weekly Value Line Investment Survey (a different product).

Arnold Bernhard & Co. maintains a staff of security analysts and statisticians who collect and develop information from a wide variety of sources including annual and quarterly reports, SEC filings and pricing information suppliers.

Over 30 updates per year keep the Value Line Database II current with frequent updates during the early part of the year when much financial information becomes available.

To access the Value Line Database II, you must first sign a special separate agreement with Arnold Bernhard & Co. and there is an 8-cents-per-item data charge for each financial data item retrieved.

If you plan to access large amounts of data, quarterly and annual subscriptions are available for $1,750 and $5,500 respectively in which case there are no per item fees.

Questions and comments about the CompuServe Information Service can be sent to Richard A. Baker, Editorial Director, CompuServe Information Service, 5000 Arlington Centre Blvd., P.O. Box 20212, Columbus, Ohio 43220 or through Feedback, main menu item 5, CompuServe User Information.

Menu System Introduced

**"APPETIZERS, ENTREES, DESSERTS . . .
OR THE CASE OF THE MYSTERIOUS
MENU MUNCHER"**

Innovation is essential in this fast-paced, electronic world of ours. And Dow Jones Information Services is no exception. We strive not only to provide quality data bases, but also to provide enhancements to our service that will facilitate its use for our subscribers.

Recently we incorporated a "master-menu" system to provide easy access to all our data bases, although not without initial "growing pains." Admittedly, most new systems often encounter problems or glitches; however, our bug was of a different species. It came to be called the "mysterious menu muncher."

It started months ago, when we first began testing the menu. Each day we would put it on line and work with it, and we were quite satisfied with the results. However, the following day when we would continue our testing, some part would be missing. In the beginning it was just a word here and there. Possibly the results of a programmer's tired fingers. But the situation grew worse. Entire categories would disappear. One evening Current Quotes was gone, the next — Media General. Soon, the entire menu would disappear overnight.

At last, one evening we caught our creature in the act. He was a non-descript monster, of variable size and color, and he was slithering in and out of our computer banks, apparently enjoying the feast provided by our menu. When cornered, he admitted his guilt, but replied, "I thought a menu offered food — I was just hungry." We reached an equitable solution and left a dish of milk outside the door each night.

Our menu has since remained intact, and has recently been introduced to the service. To avoid any misunderstanding in the future with potential menu munchers, I would like to clarify that this menu provides "food for thought." It lists the available data bases of Dow Jones Information Services, and provides an easy port of entry into the data base you wish to access.

INTRODUCTION

For those of you who have recently logged on to the service you may have noticed that our copyright screen will now ask you to type //MENU or enter query:

The computer will still accept all direct requests using the symbols and codes as described in the User's Guide, but if you type //MENU you will see an array of choices to select. Initially, all these services may be a bit overwhelming; however, the menu will provide quick and easy access to the data bases. Rather than memorizing data base identifying codes (period, comma, semi-colon, etc.) the service and its mnemonic key are right in front of you.

Below is a sample of what the menu (it is in two sections) looks like.

SCREEN 1

```
DOW JONES NEWS/RETRIEVAL
(C) 1981, DOW JONES & CO. INC
TYPE //MENU AND PRESS
RETURN FOR LIST OF SERVICES OR
ENTER QUERY.
```

SCREEN 2

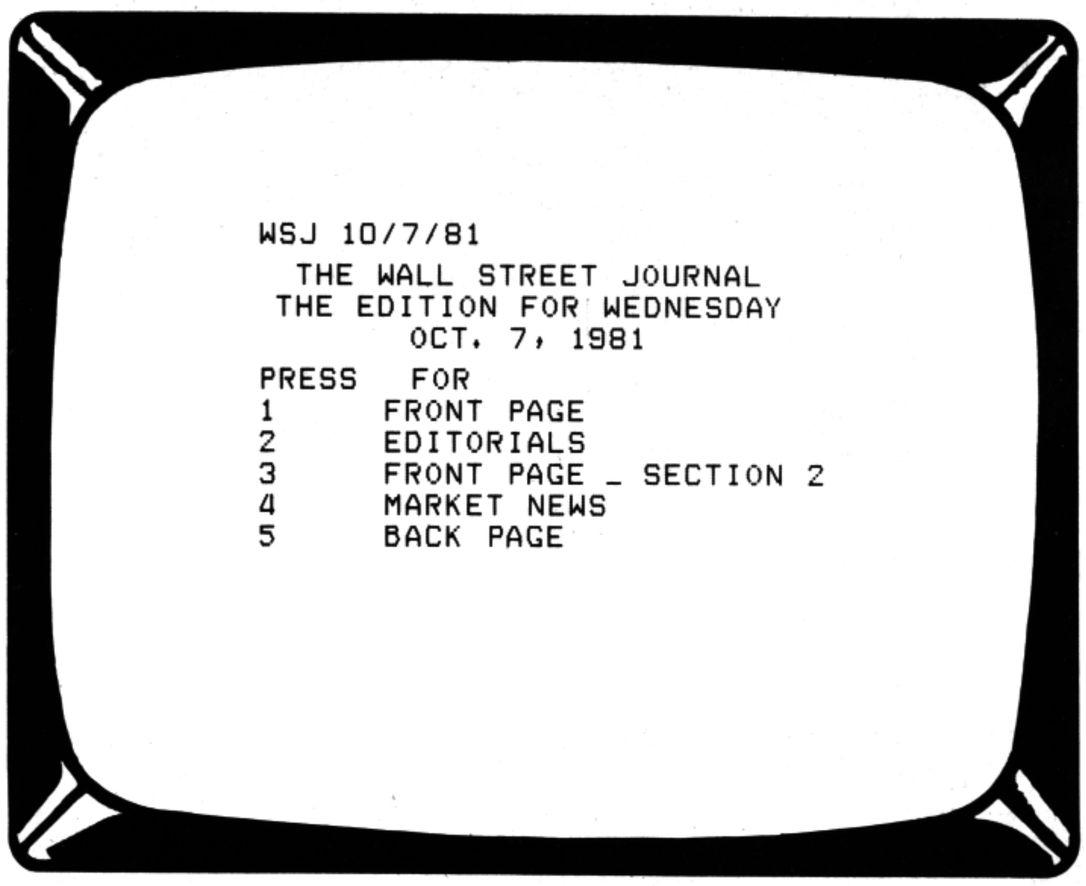
```
   CODE        SERVICE
//DJHOW    USER INSTRUCTIONS
//DJNEWS   DOW JONES NEWS SERVICE
//CQ       CURRENT QUOTES
//HQ       HISTORICAL QUOTES
//UPDATE   ECONOMIC UPDATE
//WSJ      WALL STREET JOURNAL HIGHLIGHTS
           ONLINE
//DSCLO    DISCLOSURE ONLINE
//EARN     CORPORATE EARNINGS ESTIMATOR
TYPE CODE AND RETURN FOR DESIRED
SERVICE OR HIT RETURN FOR MORE
CHOICES
```

SCREEN 3

```
   CODE        SERVICE
//FTS      FREE TEXT SEARCHES
//MEDGEN   MEDIA GENERAL
//MMS      MONEY MARKET SERVICES
//SHOP     COMP-U-STAR
//SPORTS   SPORTS
//WTHR     WEATHER
//WSW      WALL $TREET WEEK
TYPE CODE AND RETURN FOR DESIRED
SERVICE
```

EXAMPLES OF USE

Suppose you want to scan today's *Wall Street Journal.* You type //WSJ (RETURN) and you will be in the Wall Street Journal Highlights Online data base. This new data base is itself menu driven, providing you with a series of choices to further define your particular area of interest.



Perhaps you are interested in current quotes. When you type //CQ you will be greeted by a welcome page to the data base asking you to input the symbol for the quote you want, or to check "DJHOW" for further instructions. Some of the data bases (news, current and historical quotes in particular) still need to use direct access identifiers with the data base. For example, news requests need to be prefaced by a period, and for headlines you need to add (spacebar) 01 to the company or industry symbol. Historical quotes requests are to be entered as before but minus the semicolon (;). Similarly, stock issues other than common and preferred need their special identifiers as outlined in the User's Guide.

DATA BASE NAVIGATION

In general, however, the menu will get you where you want to go, with minimum delay. In fact, if you find the mnemonic code easy to remember, you can go directly to the data base you want and bypass the menu. (After reviewing the menu a few times you may find you only use it to check out new data base additions.) At the initial enter query statement you can input //(code) for direct access to the data base. For example, to retrieve Media General, you may enter //MEDGEN, or you can use the identifier system and enter $(company code)/P for access to this data base.

Navigation from data base to data base can be accomplished by entering //(code) for whatever service you wish to retrieve. For example, you have just checked the current and historical prices of a certain company, and now you want to see what the Corporate Earnings Estimator has to say. Just enter //EARN, and you will go directly to this data base. If you cannot remember the mnemonic code, type //MENU to receive the menu, and refresh your memory. It could not be easier.

If you have any questions on using the menu, please call our Customer Service Hotline at 1-800-257-5114 and in New Jersey and Canada call 609-452-1511.

If you have purchased a TRS-80 Videotex package and would like to try our service but have not yet called Dow Jones for your free password—give us a call at the above number. You'll also receive a free hour's usage during non-prime hours to try out our new menu and the service.

Season's Greetings!

How do you like our fancy new style? It will take us some time to get used to working with color, so bear with us. We are very excited by the freedom that our new size and format allow. It is now possible for us to carry larger articles than in the past without dominating the entire issue.

This issue is the result of several months' planning and work. Because of the efforts of many individuals, not only are we able to provide you with more (and we hope better) information, but we are also getting it to you more nearly on schedule. We can't (and won't) promise you 48 pages every month, but 48 pages is the monthly target. I would like to extend sincere thanks to the Production, Art, Photography, Copy, Distribution, and Data Processing departments here at Radio Shack for the many hours of work that are done monthly in order to get the Microcomputer News to you.

More thanks must be given to the many of you who have taken the time to send us programs and article ideas. Information which comes from you shapes the content of Microcomputer News. Our November issue had a new section called Level I, which resulted from material sent in by Level I users. That section should be back in January, and will appear whenever we have enough material to justify it. So, Level I users, and others, if you want information on a particular topic, you have to do two things:

1. Tell us what you want.
2. Send us material.

How do you submit information to the Microcomputer News? There are three general forms in which information can be submitted:

1. Letter
2. Magnetic Media (Tape or Disk)
3. CompuServe

Please send us only material to be published. Questions about particular problems you are having are best sent directly to Computer Customer Services.

If you have questions on technique (How do I . . .) which relate to CompuServe, Dow Jones, Profile, or VisiCalc these can also be forwarded to the Microcomputer News. Letters and magnetic media should be sent to:

Microcomputer News
P.O. Box 2910
Fort Worth, Texas 76113-2910

Material submitted through CompuServe should be directed to user number 70000, 535.

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.

When Linda Miller joined the Microcomputer News staff several months ago, one of her first assignments was to prepare the material for a "special" December, 1981 newsletter. That issue was to have been 24 pages, and consist primarily of user programs. Well, since that original assignment the entire nature of the December issue has changed. Linda still did a lion's share of the work and she has my thanks and gratitude.

How did we get to 48 pages? During June and July the decision was made to expand from the 24 pages and I mentioned a target of 48 pages. Shortly 48 pages and a color cover were approved. We weren't very sure what was going into those extra pages, but we knew we needed to get more information directly to you.

(Continued on page 46)

Those of you who have visited a Radio Shack Computer Center lately should have found out the good news: warehouse inventories of the Daisy Wheel II word processing printer are finally catching up to orders. Deliveries are mostly within 30 days. Thanks to tremendous air shipments you should be able to pick up one these popular printers "off the shelf" in one of the 180 plus Computer Centers by early spring.

Tractors are also improving in deliveries and will soon (if not already) catchup to printer deliveries. Please note that there are no more 26-1446 tractors left. We have been delivering 26-1447 since around September. This new device provides bi-directional operation at the same price of the old uni-directional tractor ($289.98). You can depend upon Radio Shack to continuously provide the best value and most advanced product available.

The DW II has proven to be one our most reliable products. The thing is built like a tank. We have had trouble gathering together field failure reports for the design engineers. A recent visit to the factory provided one reason for this. Almost all of the critical motors and parts are produced "in house" by the vendor. One part on the assembly line is welded together with a LASER to tolerances measured in MICRONS!

If you have have read this far you have noticed something different about the typographical design of the column this month. The entire text is reproduced full size from copy printed by the DW II and the multi-strike ribbon and Model II SCRIPSIT 2.0. Some of you may have noticed the many program listings and other charts and lists throughout the past several issues have employed the same process, sometimes slightly reduced in size. Some people have wondered about the ability of the multi-strike (as opposed to a single strike carbon ribbon) to produce copy suitable for photo reproduction. If the proof is in the pudding, here's the pudding.

The principle reason for this madness is to illustrate the variety of type styles which will be available for DW II beginning in April, 1982 (two - Scientific 10 and Bold PS will be available in August, 1982). So far you have been looking at LETTER GOTHIC 12. This font emulates the style available on one of "big brother's" LASER printers. One of the advantages of a 12 pitch wheel is that you can obtain a 132 column report printed on a standard sheet of paper placed in the "landscape" or sideways position.

Another new wheel is Scientific A/N 10. It looks like this. This wheel gives you the familiar upper and lower case alphabet (A-Z, a-z), space and ! % * ( ) _ - + = { } : . < > ? , ' / , all being available from the keyboard as usual. You also have many new characters, some of which replace keyboard characters and others which can be accessed through Scripsit 2.0's Print Utility. These characters are:

NAME	SYMBOL	NAME	SYMBOL	NAME	SYMBOL
Exp. 1	^{1}	Exp. 2	^{2}	Exp. 3	^{3}
Exp. 4	^{4}	Exp. 5	^{5}	Exp. 6	^{6}
Exp. 7	^{7}	Exp. 8	^{8}	Exp. 9	^{9}
Exp. 0	^{0}	Alpha	α	Beta	β
Gamma	γ	Cap. Delta	Δ	delta	δ
Epsilon	ϵ	Eta	η	Lambda	λ
Mu	μ	Xi	ξ	Pi	π
Rho	ρ	Sigma	σ	Tau	τ
Cap. Omega	Ω	Omega	ω	Approximately	$\sim$
Varies as	$\propto$	Infinity	∞	Left Arrow	$\leftarrow$
Right Arrow	$\rightarrow$	Radical	$\surd$	Reverse Diag.	\
Partial deriv.	∂	Sm. integral	$\int$		

There are also several characters which can be used together to form large symbols:

Summation		Large Integral	
Lg. Summ. up	⎲	Lg. Intg. up	⌠
Lg. Summ. down	⎳	Lg. Intg. down	⌡

Open Bracket		Close Bracket	
Lft bracket up	⎡	Rt. bracket up	⎤
Vertical line	⎢	Vertical line	⎥
Lft bracket dn	⎣	Rgt bracket dn	⎦

Get this wheel, SCRIPSIT 2.0 for Model II and get into the term paper typing business!

This font is known as OCR-B. It is recognized by many large page scanners. Print your news releases in this font and the big newspapers can "read" it directly in their computer. No more mis-prints and misquotes! The OCR-B wheel also contains several interesting characters:

¿ ¡ Ñ Å å ù ` É Ø ø ¤ ij IJ Æ æ § ¥ £ ¢ ß Ä Ö ä ö ü

One font not well illustrated here is this new CUBIC 15 style. We do not have the software available, at the time of this writing, to allow 15 cpi operation of the DW II. Please note that patches and supplements to Scripsit 2.0, and other improved word processing software to be released, will support this new wheel. Here we have used the standard 12 pitch, producing a slightly wider spacing than is normal.

This attractive font is called Bold PS. It is a proportionally spaced wheel ideally suited to newsletters and other text printing. By the time this wheel becomes available next August, we should be able to support full justification in the PS mode, producing excellent, sharp, camera-ready copy!

Don't order these wheels now. They will probably be in stock in June, 1982. Note that all DW II wheels carry 124 printable characters and are rated for 40 million characters. Don't compare those 7.95 plastic wheels commonly available (for other printers) with these long lasting wheels.

NOTES FLOATING TO THE TOP OF MY DESK

There seems to be some confusion about the true significance of the "word processing mode" of Line Printer VIII (26-1168). Please note that this mode alters only the response of line feed control codes, nothing else. If you word processing software does not use super- or sub-scripts or other forms of special line feeds (software like Scripsit for Model I or III) IT DOES NOT MATTER which mode you place the printer in!

Here's another VERY IMPORTANT note. Model II Scripsit 1.0 is not compatible with the advanced features of the L.P. VIII. That is to say, if you intend to use underline, bold, superscript, etc. L.P. VIII will not work. This software was developed before the development of the L.P. VIII and the software standards which were implemented in it.

However, because of these new standards, reported in this column last MAY, all new software will "always" be compatible with all new printers. (Never say ALWAYS when there are programmers around.)

The newly released SCRIPSIT 2.0 for Model II is compatible. It contains the architecture to allow the ever changing and varied printer parameters to be inserted in the code.

Here is a chart which may help you compare our various Daisy Wheel II print wheels:

Your computer center or CMR department should have these wheels in stock now:

Courier 10 (26-1420)
Now is the time for all computers to come to the aid of their owners. Now
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*

Madeleine PS (26-1422)
Now is the time for all computers to come to the aid of their owners. Now is the time for
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*()_+-={

Cubic PS (26-1425)
Now is the time for all computers to come to the aid of their owners. Now is the time for
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*()_+-={

Prestige Elite 12 (26-1421)
Now is the time for all computers to come to the aid of their owners. Now is the time f
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*()_+-={}[];:'"

Title Italic 12 (26-1426)
Now is the time for all computers to come to the aid of their owners. Now is the time f
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&() +-={}[];:'"*

The following print wheels should be available about April, 1982:

OCR-B 10 (26-1484)
Now is the time for all computers to come to the aid of their owners. For
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*

Letter Gothic 12 (26-1485)
Now is the time for all computers to come to the aid of their owners. Now is the time f
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*()_+-={}[];:'"

Cubic 15 (26-1487 - printed using 12 pitch)
Now is the time for all computers to come to the aid of their owners. Now is the time f
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*()_+-={}[];:'"

Scientific A/N 10 (Available August, 1982)
Now is the time for all computers to come to the aid of their owners. For
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*

Bold PS (Available August, 1982)
Now is the time for all computers to come to the aid of their owners. Now is the time for
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 !@#$%^&*()_+-={

Hopefully, next month we can outline in detail how to customize Scripsit 2.0 for Line Printer VIII. With that vary rash promise I'd better get out of here! Happy printing!

small Computer Company
P.O. Box 2910, Ft. Worth, TX 76113-2910

Editor's Note: This month we must report a bug in the existing version of the SELECTOR/EFC program (Select Records function) of the PROFILE II/PROFILE+ system.

DEBUGGING A PRINT PROGRAM

Mr. Larry Kawanatzck of Birmingham, MI, describes the following problem:

During Inquire/Update/Add, I am able to look at all 500 records in my file. However, when I go to print a report or labels, nothing after Record 320 is printed. I have listed the file under TRSDOS and there does not seem to be any problem in it.

Answer: What has happened here may be the result of an I/O error during a BACKUP. If the BACKUP program is unable to read a sector, it substitutes a record filled with nulls (ASCII value 0). The PROFILE report programs treat these records as end-of-file situations.

Running the following BASIC program should solve the problem. It cannot hurt a file.

```
1Ø INPUT "ENTER PROFILE FILE NAME"; N$
2Ø N$=LEFT$(N$+"ØØØØØØØØ",8)
3Ø OPEN "R", 1, N$+"/KEY"
4Ø FIELD 1, 85 AS A$(1), 85 AS A$(2), 85 AS
     A$(3)
5Ø FOR X=LOF(1)-1 TO 1 STEP -1
55 GET 1, X
6Ø FOR Y=1 TO 3
7Ø PRINT X; Y;
8Ø IF LEFT$(A$(Y),2)=CHR$(Ø)+CHR$(Ø) THEN LSET
     A$(Y)=""
     : PUT 1,X
     : PRINT "FIXED" ELSE PRINT
9Ø NEXT Y
1ØØ NEXT X
11Ø CLOSE
     : END
```

SPEEDING UP PROFILE OPERATIONS

PROFILE uses a large number of different files in each data base. Because of this, using an alternate directory slows the program down drastically. We suggest that diskettes to be used with PROFILE be formatted with no alternate directory. Under TRSDOS, issue the command:

```
FORMAT :d ALT=ØØ
```

where "d" is the number of the drive you wish to format.

We also recommend using PROFILE with "VERIFY DETECT" (not VERIFY) turned off. If you do this, you must be VERY CAREFUL not to switch diskettes without performing an "I" command. Under TRSDOS, execute the command:

```
AUTO VERIFY DETECT OFF
```

This will cause the computer to turn the VERIFY DETECT feature off each time it is rebooted. For a complete discussion of VERIFY DETECT see page 148 of the Model II operators manual.

KILL YOUR DATA BUT SAVE YOUR FILE

As you know, PROFILE II will let you delete an entire file, all its formats and data alike. You can do this by pressing 'K' at the Main Menu (see page 47 of the PROFILE II manual).

This is fine for a file you'll never use again — every trace of that file will be removed from your diskettes. But what if you want to keep the file description, screens, and reports intact, and just begin again with new data?

PROFILE II generates four data segments on disk when you begin to create the file; thereafter, PROFILE II requires that all segments, whether used or unused, be present. To save the formats but get rid of the data, follow these steps carefully.

If you're using PROFILE+, or if you've patched PROFILE II to permit variable length segments, you'll first need to check the record lengths of Segments 2 through 4. (With an unmodified PROFILE II, these segments always have a record length of 256.)

If you're using PROFILE+, the Define Data Formats printout shows the length of each segment. With PROFILE II, the easiest way to get the record lengths is to look in the disk directory. At TRSDOS READY, type:

```
DIR */DAT:d
```

where d is the drive number of the remaining two segments, execute a DIR */DA2:d and DIR */DA3:d.

Now that you have a written record of all segment lengths, you're ready to delete the existing data. Let's say your filename is XYZ. At TRSDOS READY, issue these four KILL commands:

```
KILL XYZØØØØØ/KEY
KILL XYZØØØØØ/DAT
KILL XYZØØØØØ/DA2
KILL XYZØØØØØ/DA3
```

For each command, you'll be asked to confirm by pressing 'Y'.

Now you must recreate the four data segments you've just killed. To replace Segment 1, type this command:

```
CREATE XYZØØØØØ/KEY:d {NRECS=Ø,LRL=256}
```

Again, d is the desired drive number. If Segments 2 through 4 were of fixed length, the CREATE command is similar for the /DAT, /DA2, and /DA3 segments. If you've used variable length segments, replace the "256" in the command with the correct number.

Let's say you want Segments 1 and 2 on Drive 1 and Segments 3 and 4 on Drive 2, and suppose Segments 2 through 4 have record lengths of 100, 150, and 35, respectively. The full sequence of commands will go like this:

```
CREATE XYZØØØØØ/KEY:1 {NRECS=Ø,LRL=256}
CREATE XYZØØØØØ/DAT:1 {NRECS=Ø,LRL=1ØØ}
CREATE XYZØØØØØ/DA2:2 {NRECS=Ø,LRL=15Ø}
CREATE XYZØØØØØ/DA3:2 {NRECS=Ø,LRL=35}
```

That's all there is to it! Now run the Expand Files program from the Main Menu, exactly as you would with a brand new file.

BUG IN SELECT RECORDS PROGRAM

The following bug has been reported by Yeshiva University (New York City) in the Select Records function of Model II PROFILE II/PROFILE+: If Record 1 has been deleted from a file, the SELECTOR/EFC program does not perform record selection properly.

To correct this problem, please be sure that the first record in a file is neither blank nor deleted. A more permanent solution will be published as soon as we work out the proper patch.

* * *

On behalf of all of us at The small Computer Company, the editors and writers of the Profile column would like to wish you a Merry Christmas and a very Happy New Year. See you in '82.

small Computer Company

Time Manager

How are you doing on your Christmas shopping list? I hope that your shopping is complete before Christmas eve night. I am swamped with my Christmas shopping list, a "honey do" list at home, a car maintenance list, a "hot" projects list at work and more. There is also a "followup" file, a "get it done today" list, and a "didn't get it done" list. Setting priorities and juggling items on those "lists" becomes another chore in itself.

This Newsletter article was finished because every morning my Model III has been confronting me with a "You Didn't Get It Done" list which included:

```
>*<. N          WRITE DEC NEWSLETTER!
```

Every morning I start work with a cup of coffee and a glimpse at today's Time Manager™ screen. Here is an example of a typical day level summary.

```
10/19/81          MONDAY, OCTOBER 19, 1981
=================================================================
)#(. N WRITE DEC NEWSLETTER!
 #. T RESERVE NEW YORK TICKETS
 #. T JONES ABOUT CONTRACT
 1. M 3 PM - PRODUCTION
 1. M REVIEWED SPECS FOR NEWPROJA
 2. A REVIEW MEDIA PLAN
 2. E PLAN TRADE SHOW
 2. I FINALIZE PLAN
 3. B BUSINESS LUNCH $8.95 AE
 3. G CARDS WITH JONES AT 9 PM
 3. P OFFICE EQUIPMENT
=========================(4 MORE)================================
SELECTED=ALL                                                  L0
[CLEAR] (SHIFT-ARROW) (ARROW) # [ENTER] [/] ?_                 #
```

Every task shown above that is marked with an * is a priority item that has carried over to today's screen. It only takes a few seconds each morning to add today's projects and assign them by the priorities of the moment as *, 1, 2, 3, etc. Since Time Manager also allows me to keep track of total time spent on categories like Advertising, Telephone, Meetings, or more I have assigned a category to every item on the screen. For example in the screen above you see the entry for a needed phone call. I have assigned it a category T for telephone.

As I enter new tasks I'll watch Time Manager instantly schedule my day as it sorts first by priority and then by category.

Let's look at how I put Time Manager to work in the office. It's a busy morning already here in Ft. Worth . . .

Time Manager's alarm buzzer just went off. I had set it for 9:15 A.M. and the Archer Mini Amplifier (277-1008) that I have attached to my Model III with the cassette cable serves as the buzzer. It's time to leave for a meeting:

```
10/26/81  MONDAY, OCTOBER 26, 1981
=================================================================
   >1<. M 9:30 . . DISCUSS 500MB-SINGLE DENSITY 2-1/4" FLOPPIE
```

The entry had automatically appeared because of my original entry made last week when we set the meeting.

10 A.M. Ft. Worth — I am in the midst of a phone conversation:

"Yes let's meet, say December 15? . . . just a minute and I'll see."

Easy . . . I'll move from Time Managers Day level Screen to the Month level and look at December's agenda. There is not a # noted for December 15 (the # would indicate something on the schedule) so I'll confirm our meeting.

```
10/19/81                DECEMBER 1981 #
      SUN   MON   TUE   WED   THU   FRI   SAT

                  1#    2     3     4     5

       6     7   ( 8 )   9    10    11    12

      13    14    15    16    17    18    19

      20    21#   22    23    24#   25#   26

      27    28    29    30#   31#

[CLEAR] (SHIFT-ARROW) (ARROW) # [ENTER] [/] ?_                 #
```

As we talk I'll record an entry for December 15 with a category of Meeting:

```
1 . M < VISIT WITH FAST CODE, INC.>
```

"Yes I have made a note on your visit. Here is what I want to cover . . ."

Now . . . I'll do a quick keyword and Category scan (NEWPROJA, T for telephone) next. (/KNEWPROJA,/CT, /S◆)

```
10/19/81          MONDAY, OCTOBER 19, 1981
=================================================================
)1(. M REVIEWED SPECS FOR NEWPROJA




=================================================================
SELECTED=M:MEETINGS          KEY='NEWPROJA'                   L0
[CLEAR] (SHIFT-ARROW) (ARROW) # [ENTER] [/] ?_                 #
```

O.K. . . . Now let's do a quick keyword and category scan (NEWPROJA, M for meeting). I'll scan the last few weeks. (/CM, /S◆)

```
10/19/81          MONDAY, OCTOBER 26, 1981
=================================================================
)1(. T OK'D START ON NEWPROJA TO SPECS




=================================================================
SELECTED=T:TELEPHONE         KEY='NEWPROJA'                   L0
[CLEAR] (SHIFT-ARROW) (ARROW) # [ENTER] [/] ?_                 #
```

On the 19th we gave you our comments in a meeting. We then agreed by phone on the 26th to get started . . ."

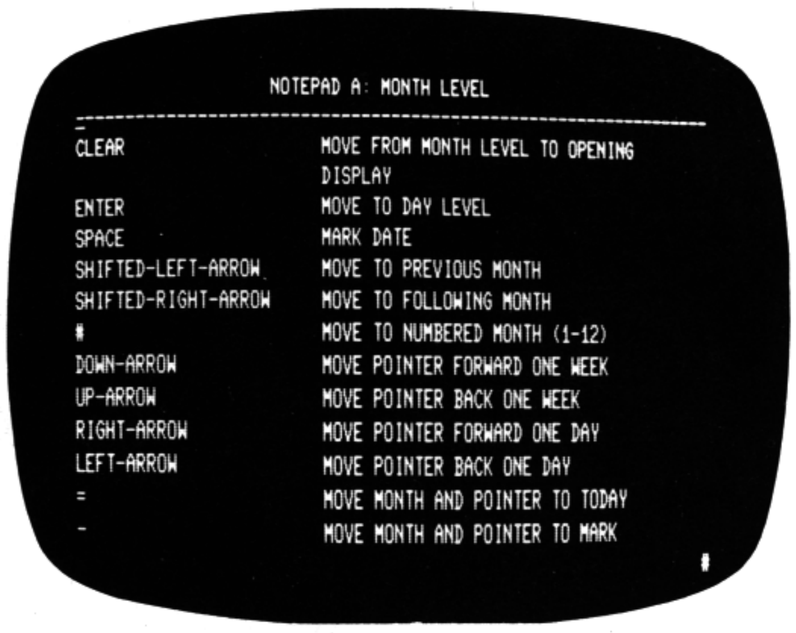
"So you think you can produce 100,000 lines of code by tomorrow? O.K., if you say so. Fully Tested? O.K."

I better make a note of that . . . I'll access Notepad A and update my file on NEWPROJA. Let's see I will use the notepad editor and jot down a record of this . . .

Actually there are eight Notepads for your use, see the screen below. They come prepared with "quick reference" detail for Time Manager's commands including the editor. Or you can use some of them as I do, to record important phone numbers, phone logs, or often needed notes on formulas, procedures, etc.

Well maybe you have a feel now for something that I am very excited about. Time Manager for both Model I and III will soon be available. Now I have to admit that you could buy a lot of notepaper for jotting down your "to do" lists for the $99.95 price of our Time Manager program (26-1582). But while some of you are going from notepad to notepad, looking in files, phone logs, meeting records and more, some happy owners of Time Manager (and several Radio Shack Product Line Managers) will be using this amazing tool for increasing personal productivity. It has many more features that we will talk about in future articles including the accounting and totalling features. It makes Time Manager great for developing tax records, expense account information, job costing and more. One note — If you have a very early Model III, which included a slip of paper discussing an optional modification to your Model III to generate certain control codes, then you will need that modification to use some features of Time Manager. If you find a unique application for Time Manager be sure and send it in to the Newsletter attention:

Newsletter Editor
P.O. Box 2910
Ft. Worth, Texas 76113-2910.



Model I/III Bugs, Errors, and Fixes

GENERAL LEDGER (26-1552)

Model III General Ledger version 1.1 is not printing properly when used with a Line Printer VI and the LPC printer driver. The problem may be corrected by following these steps:

1. BACKUP the diskette(s) and make the changes to the backup copy of the program.
2. In BASIC, load the program by typing:

```
LOAD "GLTXPOST" [ENTER]
```

3. Make the following change:

```
Old line: 5ØØØ N=N+1:L=Ø
New line: 5ØØØ N=N+1:L=12
```

4. Type: `SAVE "GLTXPOST" [ENTER]`
   to save the changes in the program.
5. Return to TRSDOS and make new backups of the corrected version.

ACCOUNTS PAYABLE (26-1554)

The Model I Version 3.0 Accounts Payable program will not accept a numeric character in the password after setup.

The problem is corrected by following the steps listed below:

1. BACKUP the diskette(s) and make the changes to the backup copy of the program.
2. In BASIC, load the program by typing:

```
LOAD "SETUP" [ENTER]
```

3. Make the following change:

Old line:
```
93 PE$=INKEY$:IFPE$=""THEN93ELSEIFPE$=CHR$(13)
THENX=8:PE$=""ELSEIFASC(PE$)<65ORASC(PE$)>9Ø
THEN93
```

New line:
```
93 PE$=INKEY$:IFPE$=""THEN93ELSEIFPE$=CHR$(13)
ΓHENX=8:PE$=""ELSEIFASC(PE$)<48ORASC(PE$)>9Ø
OR(ASC(PE$)>57ANDASC(PE$)<65)THEN93
```

4. Type: `SAVE "SETUP" [ENTER]`
   to save the changes in the program.
5. Return to TRSDOS and make new backups of the corrected version.

PROFILE (26-1562)

When printing some reports with Model I version 3.0 of Profile, an extraneous line is being printed at the end of the page. The problem may be corrected by following these steps:

1. BACKUP the diskette(s) and make the changes to the backup copy of the program.
2. In BASIC, key in the following program:

```
1Ø REM "PROFIX"
2Ø CLEAR 1ØØØ
3Ø OPEN "R", 1, "INIT"
4Ø FIELD 1, 1ØØ AS FD$
5Ø GET 1, 1
6Ø CH$=FD$
7Ø MID$(CH$, 76, 1)=CHR$(66)
8Ø MID$(CH$, 81, 1)=CHR$(Ø)
9Ø LSET FD$=CH$
1ØØ PUT 1, 1
11Ø CLOSE
```

Save this program on your TRSDOS diskette (for later use) with the name "PROFIX." Remove the TRSDOS diskette and place the PROFILE program diskette in drive 0. Run the program. The BASIC program will change the PROFILE to correct the printer problem. RESET the Model I and run the PROFILE program as instructed in the manual.

BUDGET MANAGEMENT (26-1603)

In Model III version 3.0 of Budget Management, when you edit accounts the computer does not accept a negative value for the checkbook balance.

The problem is corrected by following the steps listed below:

1. CLOAD the EDIT tape.
2. Make the following corrections:

Old line:
```
1125 Z#=ABS(CDBL(VAL(B$))):A#=Z#-A3#(KK):
B#=ØDØ: GOSUB 4ØØ:A3#(KK)=Z#
```

New line:
```
1125 Z#=CDBL(VAL(B$)):A#=Z#-A3#(KK):
B#=ØDØ:GOSUB 4ØØ:A3#(KK)=Z#
```

3. Type: `CSAVE "E" [ENTER]`
   to save the changes on a fresh cassette tape.

If you are using Disk, follow these steps:

1. Make a BACKUP of your program disk.
2. Using the backup, in BASIC type:

```
LOAD "EDIT" [ENTER]
```

3. Make the change shown in line 2 above.

4. After the change has been made, type:

   `SAVE "EDIT" [ENTER]`

   to save the change.

GRAPHICAL ANALYSIS OF EXPERIMENTAL DATA (26-1722)

Under "Function Selection" in the option list, if you choose LOG and the Base E contains a decimal, the first time through the analysis the decimal is recognized. The second time, however, the decimal is ignored unless you re-RUN the program. To correct the disk version of the program, follow these steps:

1. BACKUP the diskette(s) and make the changes to the backup copy of the program.
2. In BASIC, load the program by typing:

   `LOAD "EXPDATA" [ENTER]`

3. Make the following correction:

   Old line 70:
   ```
   7Ø CU=32:CP=PEEK(16416)+PEEK(16417)
   *256:X$=""
   ```

   New line 70:
   ```
   7Ø CU=32:CP=PEEK(16416)+PEEK(16417)
   *256:X$="":PC=Ø
   ```

4. Type:

   `SAVE "EXPDATA" [ENTER]`

   to save the changes in the program.
5. At TRSDOS Ready, make a backup copy of the corrected diskette.

To fix the cassette version of the program, follow these steps:

1. Load the program from tape using the command:

   `CLOAD [ENTER]`

2. Make the line change shown in 3 above.
3. Save a copy of the corrected program onto tape using the command:

   `CSAVE"filename" [ENTER]`

   where "filename" is the name you give to the program.

SERIES I EDITOR/ASSEMBLER (26-2011/26-2013)

The Series I Editor/Assembler assembles the OTDR instruction improperly. OTDR should assemble as "EDBB" not "ED8B."

If you have the 26-2011 tape version of this program, return the original program cassette (along with proof of purchase) to your local Radio Shack Store. Ask the store personnel to send the program back to Fort Worth for warranty exchange. The stores have a procedure for doing this.

If you have the 26-2013 Disk version of the program, you can correct the problem by applying the following patch to your EDTASM program:

```
PATCH EDTASM/CMD (ADD=6DE2,FIND=8B,CHG=BB) [ENTER]
```

You may also wish to correct the comment line in the SAMPLE/SRC listing from: ;ED8B to ;EDBB.

Christmas Tree

With Snow on the Line Printer

by Todd Knowlton
Lytle, TX

Enclosed is a program that I would like you to consider for your December issue of the Microcomputer News. The program produces a Christmas tree on a lineprinter. The program was written on a Model I but I see no reason why it would not work on a Model III.* Only a computer and a lineprinter are necessary to run it. The program listing is in two parts — the tree program and the snow program. You may combine them for printing.

```
1Ø ' CHRISTMAS TREE
2Ø ' TODD KNOWLTON
3Ø T=3Ø
     :CLS
     :PRINTTAB(12)"*----*----*-- CHRISTMAS TREE
     --*----*----*"
4Ø PRINT
     :PRINTTAB(14);"PRESS ENTER WHEN LINEPRINTER
     IS READY"
5Ø INPUT A$: IF PEEK (14312) <> 63 THEN PRINT
     TAB(2Ø); "LINE PRINTER NOT READY"
     :GOTO 4Ø
6Ø LPRINT CHR$(138)
7Ø LPRINT TAB(23) "CHRISTMAS TREE"
8Ø LPRINT" "
     :LPRINT" "
9Ø LPRINT TAB(T)"*"
     :LPRINT TAB(T-1);"# #"
1ØØ FOR T1=2 TO 5
     :LPRINT TAB(T-T1);"#";TAB(T+T1)"#"
     :NEXT T1
11Ø LPRINT TAB(T-6)"####    ####"
12Ø FOR T1=3 TO 11
     :LPRINT TAB(T-T1)"#";TAB(T+T1)"#"
     :NEXT T1
13Ø LPRINT TAB(T-12)"######            ######"
14Ø FOR T1=7 TO 18
     :LPRINT TAB(T-T1)"#";TAB(T+T1)"#"
     :NEXT T1
15Ø LPRINT TAB(T-19); STRING$(39,35)
16Ø FOR N=1 TO 5: LPRINT TAB(29)"HHH"
     :NEXT N
2ØØ INPUT"DO YOU WANT SNOW (Y/N)";SN$
21Ø IF SN$="Y" THEN 23Ø
22Ø IF SN$="N" THEN END
23Ø CLS
     :PRINT"RETURN PRINTHEAD TO TOP OF TREE"
     :FOR X=1 TO 1ØØØ
     :NEXT
24Ø INPUT"PRESS ENTER WHEN READY";RE$
25Ø LPRINT TAB(2Ø)"*"
     :LPRINT TAB(39)"*";TAB(53)"*"
26Ø GOSUB 34Ø
     :LPRINT TAB(25)"*";TAB(49)"*"
     :LPRINT TAB(3Ø)"*"
27Ø GOSUB 34Ø
     :LPRINT TAB(31)"*";TAB(43)"*"
     :LPRINT TAB(19)"*"
28Ø GOSUB 34Ø
     :LPRINT TAB(25)"*";TAB(33)"*"
29Ø GOSUB 34Ø
     :LPRINT TAB(14)"*";TAB(39)"*"
3ØØ GOSUB 34Ø
     :LPRINT TAB(17)"*";TAB(29)"*";TAB(44)"*"
31Ø GOSUB 34Ø
     :LPRINT TAB(22)"*";TAB(33)"*";TAB(47)"*"
32Ø GOSUB 34Ø
     :LPRINT TAB(25)"*";TAB(38)"*";TAB(41)"*"
33Ø GOSUB 34Ø
     :LPRINT TAB(21)"*";TAB(31)"*";TAB(29)"*"
     :GOTO 35Ø
34Ø LPRINT" "
     :LPRINT" "
     :RETURN
35Ø LPRINT" "
     :LPRINT" "
```

Keep Order Sort

by Richard Stern, Ph.D.
School of Engineering/Applied Science
UCLA
Los Angeles, CA

Recent articles in this publication have appeared illustrating the use of BASIC to sort a one-dimensional array (March/April, 1980) and the use of machine language to sort a one-dimensional array (July, 1980) and a two-dimensional array (March, 1981). These excellent programs have concentrated on "speed of sorting" as the most desirable factor for sorting strings and thus have utilized either "bubble sort" or "shell sort" algorithms. However, neither of these methods are "order preserving sorts," often a very desirable characteristic.

To illustrate this shortcoming, consider the following array of 6 sets and 3 fields for a "ledger" entry. The first field represents the "cash" date of a money exchange (the day the amount changes hands) while the second field represents the "accrual" date (the date the obligation incurred), the third field is the amount.

A$ (i,j) =

Field	Cash 1	Accrual 2	Amount 3
Set			
1	02/03	01/01	$100
2	01/04	01/01	$200
3	02/03	01/02	$150
4	01/04	01/02	$100
5	03/01	03/01	$ 50
6	04/01	04/01	$ 75

ORIGINAL ENTRIES

In the original array sets 1 & 3 have the same cash date and the accrual dates are in the order 01/01, then 01/02. The same is true for sets 2 & 4 with a cash date of 01/04. A "shell sort" of field 1 immediately compares set 1 to set 4 and, because of disorder, swaps these sets producing the following array. Since field 1 is now correctly sorted, the array would be left untouched.

A$ (i,j) =

Field	Cash 1	Accrual 2	Amount 3
Set			
1	01/04	01/02	$100
2	01/04	01/01	$200
3	02/03	01/02	$150
4	02/03	01/01	$100
5	03/01	03/01	$ 50
6	04/01	04/01	$ 75

SHELL SORT

It can be seen that the accrual dates (field 2) of sets 1 and 2, 3 and 4 are reversed i.e., the original order of 01/01 coming before 01/02 has been reversed. Of course the content of the sets are unchanged. Indeed, if the machine language sort of William Terrell is used on one field, the ordering of the other fields is changed drastically whenever the value of field being sorted has a constant value. If a bubble sort is used, that is where set 1 is compared with all other sets and the smallest number is brought to the top followed by set 2 and so forth, the following array results—

A$ (i,j) =

Field	Cash 1	Accrual 2	Amount 3
Set			
1	01/04	01/01	$200
2	01/04	01/02	$100
3	02/03	01/01	$100
4	02/03	01/02	$150
5	03/01	03/01	$ 50
6	04/01	04/01	$ 75

BUBBLE SORT

Again sets 3 and 4 in field 2 are reversed from their original order. The basic problem is that both the bubble sort and the shell sort compare sets which are separated (such as 1 & 4). Not so with a ripple sort where sets 1 & 2 are compared, 2 & 3, 3 & 4 and so forth. Any swaps never move a set more than one position away from its current location. Thus a ripple sort of field 1 will not change the order of field 2 and final array is—

A$ (i,j) =

Field	Cash 1	Accrual 2	Amount 3
Set			
1	01/04	01/01	$200
2	01/04	01/02	$100
3	02/03	01/01	$100
4	02/03	01/02	$150
5	03/01	03/01	$ 50
6	04/01	04/01	$ 75

RIPPLE SORT

Using much of the format of Terrell, I have written a BASIC-machine language program, for 16K tape based Model Is or IIIs, which performs a ripple sort on a two-dimensional array. This sort will preserve order within the array fields. It is possible to sort fields starting with the least important field and work towards most important.

The machine language sort starts at 7F00H (32512) and entry to the program begins at 7F10H (32528). Memory must be saved at 32511 and the entire machine sort program uses only 234 bytes of data.

To relocate the program alter the machine language data elements in the BASIC program from line 620 to the end by changing the '127' to '191' (for 32K) or to '255' (for 48K). The same number must replace the '127' in line 180. In line 110 the program should load from -16640 to -16407 for 32K and -256 to -23 for 48K. Be sure to save memory at 48895 for 32K or at 65279 for 48K. Line 150 must also be changed to reflect the new "data error check."

In order to demonstrate the power of the ripple sort the BASIC program generates an array which only uses the letters A, B and C of string length 1, 2 or 3 letters. Try running the program with 100 sets of 5 fields. It will automatically sort the data starting with the highest numbered field and work towards the lowest. (Change the "PRINT" in line 520—560 to "LPRINT" if you have a line printer).

When the sorted array is displayed, note that whenever the first field has a number of sets with the same value, the second field is still sorted (left over from the previous sort of that field). If the values are the same for both the first field and the second field, the third field is also sorted and so forth.

If it is desirable to sort only certain fields, the "FOR-NEXT" statements in lines 390 and 470 need to be changed. "ZC" is the number of the field sorted for each pass through lines 410-460. If strings with variable names other than "A$(i,j)" are to be sorted the "A$" should be changed to the name of the variable in lines 410-460. For further information on sort programs, please read the articles cited in the first paragraph. They apply in general to this ripple sort program as well.

I would like to thank the authors of those articles for showing me how sorts could be accomplished and I hope that readers will enjoy using the program.

```
Ø '***** RIPPLESORT FOR TWO DIMENSIONAL ARRAYS
     *****
1Ø ' 2D ARRAY SORT BASED ON ALLEN EMERT PROGRAM
     FROM JUL 8Ø
2Ø ' TRS-8Ø MICROCOMPUTER NEWSLETTER, MODIFIED BY
     WILLIAM
3Ø ' TERRELL OCT 8Ø FOR ARRAYS OF THE FORM
     A$(I,J)
4Ø ' REMODIFIED BY RICHARD STERN JULY 81
5Ø ' POKE MACHINE LANGUAGE PROGRAM INTO HIGH
     MEMORY
6Ø ' PROTECT MEMORY AT 32511
7Ø CLS
8Ø POKE 16553, 255
9Ø PRINT "LOADING MACHINE LANGUAGE PROGRAM"
1ØØ K=Ø
11Ø FOR I=32512 TO 32745
     : READ J
12Ø K=K+J
13Ø POKE I, J
     : NEXT I
14Ø ' CHECK FOR LOADING ERROR
15Ø IF K <> 22528 THEN PRINT "DATA ERROR"
     : END
16Ø PRINT "DONE"
17Ø ' SET UP CALLING ADDRESS OF MACHINE LANGUAGE
     PROGRAM
18Ø POKE 16526, 16
     : POKE 16527, 127
19Ø ' INPUT NUMBER OF SETS, S AND FIELDS, F
2ØØ ' GENERATE AND DISPLAY ARRAY TO BE SORTED
21Ø CLEAR 1ØØØØ
     : DEFINT A-Z
     : Z=Ø
22Ø INPUT "NUMBER OF DATA SETS (MAX = 1ØØ)"; S
23Ø INPUT "NUMBER OF FIELDS (MAX = 5)"; F
24Ø DIM A$(S,F), Z(4)
25Ø FOR I=Ø TO S-1
     : FOR J=Ø TO F-1
     : A$(I,J)=""
     : NEXT J
     : NEXT I
26Ø FOR I=Ø TO S-1
27Ø ZA=5
     : PRINT I;
28Ø FOR J=Ø TO F-1
29Ø A$(I,J)=STRING$(RND(3),RND(3)+64)
3ØØ PRINT TAB(ZA); A$(I,J);
     : ZA=ZA+12
     : NEXT J
31Ø PRINT
     : NEXT I
32Ø ' SORT FIELDS STARTING WITH HIGHEST FIELD,
33Ø ' F AND WORKING TOWARDS LOWEST FIELD, Ø
34Ø ' Z(Ø)=NUMBER OF DATA SETS,S
35Ø ' Z(1)=POINTER TO FIELD BEING SORTED
36Ø ' Z(2)=NUMBER OF DATA FIELDS, F
37Ø ' Z(3)=POINTER DELTA - SORTED FIELD TO ZERO
     FIELD
38Ø ' Z(4)=POINTER DELTA - FIELD TO FIELD
39Ø FOR I=F-1 TO Ø STEP-1
     : ZC=I
4ØØ PRINT "FIELD "; I; " BEING SORTED"
41Ø Z(Ø)=S
42Ø Z(1)=VARPTR(A$(Ø,ZC))
43Ø Z(2)=F
44Ø IF Z(1)<Ø AND VARFTR(A$(Ø,Ø))>Ø THEN
     Z(3)=65536+Z(1)- VARPTR(A$(Ø,Ø))
     ELSE Z(3)=ABS(VARPTR(A$(Ø,ZC))-
     VARPTR(A$(Ø,Ø)))
45Ø Z(4)=VARPTR(A$(Ø,1))-VARPTR(A$(Ø,Ø))
46Ø Z=USR(VARPTR(Z(Ø)))
47Ø NEXT I
48Ø PRINT "SORT DONE"
49Ø ' DISPLAY SORTED FIELD
5ØØ INPUT "PRESS ENTER FOR SORTED ARRAY"; ZZ
51Ø FOR I=Ø TO S-1
52Ø K=5
     : PRINT I;
53Ø FOR J=Ø TO F-1
54Ø PRINT TAB(K); A$(I,J);
     : K=K+12
55Ø NEXT J
56Ø PRINT
     : NEXT I
57Ø END
58Ø 'MACHINE LANGUAGE DATA
59Ø DATA Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø
6ØØ DATA Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, 2Ø5, 127
61Ø DATA 1Ø, 94, 35, 86, 27, 27, 237, 83, Ø, 127
62Ø DATA 35, 94, 35, 86, 237, 83, 8, 127, 35, 94
63Ø DATA 35, 86, 237, 83, 1Ø, 127, 35, 94, 35, 86
64Ø DATA 237, 83, 12, 127, 35, 94, 35, 86, 237,
     83
65Ø DATA 14, 127, 42, 8, 127, 17, Ø, Ø, 237, 83
66Ø DATA 6, 127, 84, 93, 237, 83, 2, 127, 35, 35
67Ø DATA 35, 34, 4, 127, 229, 213, 14, Ø, 126, 71
68Ø DATA 26, 184, 32, 2, 2Ø3, 193, 48, 3, 2Ø3,
     193
69Ø DATA 71, 175, 176, 4Ø, 25, 197, 19, 35, 78,
     35
7ØØ DATA 7Ø, 197, 225, 235, 78, 35, 7Ø, 197, 225,
     193
71Ø DATA 26, 15Ø, 56, 1Ø, 32, 69, 19, 35, 16, 246
72Ø DATA 2Ø3, 65, 32, 61, 237, 75, 12, 127, 175,
     225
73Ø DATA 237, 66, 235, 225, 237, 66, 58, 1Ø, 127,
     71
74Ø DATA 197, 229, 213, 24, 11, 197, 237, 75, 14,
     127
75Ø DATA 9, 235, 9, 235, 229, 213, 221, 225, 253,
     225
76Ø DATA 6, 3, 253, 78, Ø, 221, 126, Ø, 221, 113
77Ø DATA Ø, 253, 119, Ø, 221, 35, 253, 35, 16,
     238
78Ø DATA 193, 16, 218, 24, 2, 2Ø9, 225, 42, Ø,
     127
79Ø DATA 237, 75, 6, 127, 175, 237, 66, 4Ø, 11, 3
8ØØ DATA 237, 67, 6, 127, 42, 4, 127, 195, 7Ø,
     127
81Ø DATA 42, Ø, 127, 124, 69, 176, 4Ø, 7, 43, 34
82Ø DATA Ø, 127, 195, 6Ø, 127, 2Ø1
```

Here is the Assembly Language listing for the 16K version:

```
                        ØØ1ØØ            ORG     7FØØH
7FØØ    ØØØØ            ØØ2ØØ  SETS:     DEFW
7FØ2    ØØØØ            ØØ3ØØ  ADDR1:    DEFW
7FØ4    ØØØØ            ØØ4ØØ  ADDR2:    DEFW
7FØ6    ØØØØ            ØØ5ØØ  K:        DEFW
7FØ8    ØØØØ            ØØ6ØØ  SRTFLD:   DEFW
7FØA    ØØØØ            ØØ7ØØ  FIELDS:   DEFW
7FØC    ØØØØ            ØØ8ØØ  PTRDEL:   DEFW
7FØE    ØØØØ            ØØ9ØØ  COLDEL:   DEFW
7F1Ø    CD ØA7F         Ø1ØØØ  RPLSRT::  CALL    ØA7FH
7F13    5E              Ø11ØØ            LD      E,(HL)
7F14    23              Ø12ØØ            INC     HL
7F15    56              Ø13ØØ            LD      D,(HL)
7F16    1B              Ø14ØØ            DEC     DE
7F17    1B              Ø15ØØ            DEC     DE
7F18    ED 53 7FØØ      Ø16ØØ            LD      (SETS),DE
7F1C    23              Ø17ØØ            INC     HL
7F1D    5E              Ø18ØØ            LD      E,(HL)
7F1E    23              Ø19ØØ            INC     HL
7F1F    56              Ø2ØØØ            LD      D,(HL)
7F2Ø    ED 53 7FØ8      Ø21ØØ            LD      (SRTFLD),DE
7F24    23              Ø22ØØ            INC     HL
7F25    5E              Ø23ØØ            LD      E,(HL)
7F26    23              Ø24ØØ            INC     HL
```

```
7F27   56           Ø25ØØ          LD     D,(HL)
7F28   ED 53 7FØA   Ø26ØØ          LD     (FIELDS),DE
7F2C   23           Ø27ØØ          INC    HL
7F2D   5E           Ø28ØØ          LD     E,(HL)
7F2E   23           Ø29ØØ          INC    HL
7F2F   56           Ø3ØØØ          LD     D,(HL)
7F3Ø   ED 53 7FØC   Ø31ØØ          LD     (PTRDEL),DE
7F34   23           Ø32ØØ          INC    HL
7F35   5E           Ø33ØØ          LD     E,(HL)
7F36   23           Ø34ØØ          INC    HL
7F37   56           Ø35ØØ          LD     D,(HL)
7F38   ED 53 7FØE   Ø36ØØ          LD     (COLDEL),DE
7F3C   2A 7FØ8      Ø37ØØ  STNXT1: LD     HL,(SRTFLD)
7F3F   11 ØØØØ      Ø38ØØ          LD     DE,ØØØØH
7F42   ED 53 7FØ6   Ø39ØØ          LD     (K),DE
7F46   54           Ø4ØØØ  INCRS1: LD     D,H
7F47   5D           Ø41ØØ          LD     E,L
7F48   ED 53 7FØ2   Ø42ØØ          LD     (ADDR1),DE
7FAC   23           Ø43ØØ          INC    HL
7F4D   23           Ø44ØØ          INC    HL
7F4E   23           Ø45ØØ          INC    HL
7F4F   22 7FØ4      Ø46ØØ          LD     (ADDR2),HL
7F52   E5           Ø47ØØ          PUSH   HL
7F53   D5           Ø48ØØ          PUSH   DE
7F54   ØE ØØ        Ø49ØØ  COMP:   LD     C,ØØH
7F56   7E           Ø5ØØØ          LD     A,(HL)
7F57   47           Ø51ØØ          LD     B,A
7F58   1A           Ø52ØØ          LD     A,(DE)
7F59   B8           Ø53ØØ          CP     B
7F5A   2Ø Ø2        Ø54ØØ          JR     NZ,LTEST
7F5C   CB C1        Ø55ØØ          SET    ØØH,C
7F5E   3Ø Ø3        Ø56ØØ  LTEST:  JR     NC,SHRTST
7F6Ø   CB C1        Ø57ØØ          SET    ØØH,C
7F62   47           Ø58ØØ          LD     B,A
7F63   AF           Ø59ØØ  SHRTST: XOR    A
7F64   BØ           Ø6ØØØ          OR     B
7F65   28 19        Ø61ØØ          JR     Z,CKLGTH
7F67   C5           Ø62ØØ          PUSH   BC
7F68   13           Ø63ØØ          INC    DE
7F69   23           Ø64ØØ          INC    HL
7F6A   4E           Ø65ØØ          LD     C,(HL)
7F6B   23           Ø66ØØ          INC    HL
7F6C   46           Ø67ØØ          LD     B,(HL)
7F6D   C5           Ø68ØØ          PUSH   BC
7F6E   E1           Ø69ØØ          POP    HL
7F6F   EB           Ø7ØØØ          EX     DE,HL
7F7Ø   4E           Ø71ØØ          LD     C,(HL)
7F71   23           Ø72ØØ          INC    HL
7F72   46           Ø73ØØ          LD     B,(HL)
7F73   C5           Ø74ØØ          PUSH   BC
7F74   E1           Ø75ØØ          POP    HL
7F75   C1           Ø76ØØ          POP    BC
7F76   1A           Ø77ØØ  NXTLTR: LD     A,(DE)
7F77   96           Ø78ØØ          SUB    (HL)
7F78   38 ØA        Ø79ØØ          JR     C,SWPRP1
7F7A   2Ø 45        Ø8ØØØ          JR     NZ,STNEXT
7F7C   13           Ø81ØØ          INC    DE
7F7D   23           Ø82ØØ          INC    HL
7F7E   1Ø F6        Ø83ØØ          DJNZ   NXTLTR
7F8Ø   CB 41        Ø84ØØ  CKLGTH: BIT    ØØH,C
7F82   2Ø 3D        Ø85ØØ          JR     NZ,STNEXT
7F84   ED 4B 7FØC   Ø86ØØ  SWPRP1: LD     BC,(PTRDEL)
7F88   AF           Ø87ØØ          XOR    A
7F89   E1           Ø88ØØ          POP    HL
7F8A   ED 42        Ø89ØØ          SBC    HL,BC
7F8C   EB           Ø9ØØØ          EX     DE,HL
7F8D   E1           Ø91ØØ          POP    HL
7F8E   ED 42        Ø92ØØ          SBC    HL,BC
7F9Ø   3A 7FØA      Ø93ØØ          LD     A,(FIELDS)
7F93   47           Ø94ØØ          LD     B,A
7F94   C5           Ø95ØØ          PUSH   BC
7F95   E5           Ø96ØØ          PUSH   HL
7F96   D5           Ø97ØØ          PUSH   DE
7F97   18 ØB        Ø98ØØ          JR     SWPRP2
7F99   C5           Ø99ØØ  SWAP:   PUSH   BC
7F9A   ED 4B 7FØE   1ØØØØ          LD     BC,(COLDEL)
7F9E   Ø9           1Ø1ØØ          ADD    HL,BC
7F9F   EB           1Ø2ØØ          EX     DE,HL
7FAØ   Ø9           1Ø3ØØ          ADD    HL,BC
7FA1   EB           1Ø4ØØ          EX     DE,HL
7FA2   E5           1Ø5ØØ          PUSH   HL
7FA3   D5           1Ø6ØØ          PUSH   DE
7FA4   DD E1        1Ø7ØØ  SWPRP2: POP    IX
7FA6   FD E1        1Ø8ØØ          POP    IY
7FA8   Ø6 Ø3        1Ø9ØØ          LD     B,Ø3H
7FAA   FD 4E ØØ     11ØØØ  SWITCH: LD     C,(IY)
7FAD   DD 7E ØØ     111ØØ          LD     A,(IX)
7FBØ   DD 71 ØØ     112ØØ          LD     (IX),C
7FB3   FD 77 ØØ     113ØØ          LD     (IY),A
7FB6   DD 23        114ØØ          INC    IX
7FB8   FD 23        115ØØ          INC    IY
7FBA   1Ø EE        116ØØ          DJNZ   SWITCH
7FBC   C1           117ØØ  COLADV: POP    BC
7FBD   1Ø DA        118ØØ          DJNZ   SWAP
7FBF   18 Ø2        119ØØ          JR     SETNXT
7FC1   D1           12ØØØ  STNEXT: POP    DE
7FC2   E1           121ØØ          POP    HL
7FC3   2A 7FØØ      122ØØ  SETNXT: LD     HL,(SETS)
7FC6   ED 4B 7FØ6   123ØØ          LD     BC,(K)
7FCA   AF           124ØØ          XOR    A
7FCB   ED 42        125ØØ          SBC    HL,BC
7FCD   28 ØB        126ØØ          JR     Z,INCRS2
7FCF   Ø3           127ØØ          INC    BC
7FDØ   ED 43 7FØ6   128ØØ          LD     (K),BC
7FD4   2A 7FØ4      129ØØ          LD     HL,(ADDR2)
7FD7   C3 7F46      13ØØØ          JP     INCRS1
7FDA   2A 7FØØ      131ØØ  INCRS2: LD     HL,(SETS)
7FDD   7C           132ØØ          LD     A,H
7FDE   45           133ØØ          LD     B,L
7FDF   BØ           134ØØ          OR     B
7FEØ   28 Ø7        135ØØ          JR     Z,DONE
7FE2   2B           136ØØ          DEC    HL
7FE3   22 7FØØ      137ØØ          LD     (SETS),HL
7FE6   C3 7F3C      138ØØ          JP     STNXT1
7FE9   C9           139ØØ  DONE:   RET
                    14ØØØ          END    RPLSRT
```

SYMBOLS:

```
ADDR1   7FØ2    ADDR2   7FØ4    CKLGTH  7F8Ø    COLADV  7FBC
COLDEL  7FØE    COMP    7F54    DONE    7FE9    FIELDS  7FØA
INCRS1  7F46    INCRS2  7FDA    K       7FØ6    LTEST   7F5E
NXTLTR  7F76    PTRDEL  7FØC    RPLSRT  7F1ØI   SETNXT  7FC3
SETS    7FØØ    SHRTST  7F63    SRTFLD  7FØ8    STNEXT  7FC1
STNXT1  7F3C    SWAP    7F99    SWITCH  7FAA    SWPRP1  7F84
SWPRP2  7FA4
```

NO FATAL ERROR(S)

Twinkling Tree

For the Model I/III

by Richard Pelletier
4226 Chamoune Avenue
San Diego, CA 92115

This is one of those programs that you probably have too many of this time of year — yes, you guessed it — it is a CHRISTMAS TREE PROGRAM.

It is a very simple program. It POKEs a series of stars into a triangular (Christmas tree) shape. Then it POKEs a stump and PRINTs at the bottom — just simply — MERRY CHRISTMAS as the stars begin to randomly flash; this program does nothing more — visually — but it is nice to look at.

I asked my 12 year old sister to type this program into my TRS-80. She did, and it ran successfully. My sister knows nothing about computers, so I figure, if she can type this in, anyone can!

This program demonstrates the use of PEEK and POKE for graphics, a one dimensional array for string information that would be needed randomly, and the use of a simple mathematical property of multiplication that states for all non-zero numbers, a negative times a negative equals a positive; a negative times a positive equals a negative.

Example:

```
10 X = -1        This example program
20 PRINT X       will produce a forever
30 X = X* -1     loop PRINTing -1 and 1
40 GOTO 20       until you hit the BREAK key.
```

I love mathematics! I love my TRS-80! — excuse me . . . I think I am taking too much of your time in describing this short program (THE CHRISTMAS TREE).

This is a fairly simple program — 12 lines short. At first glance, one may not know exactly what this program does — until he looks at line 80 and then he may only get an idea.

I will tell you what: type in this program, line for line, onto your TRS-80 Model I Level II or your Model III BASIC TRS-80. Sorry to those of you with other machines — but I am sure that there are other similar programs in this issue for you.

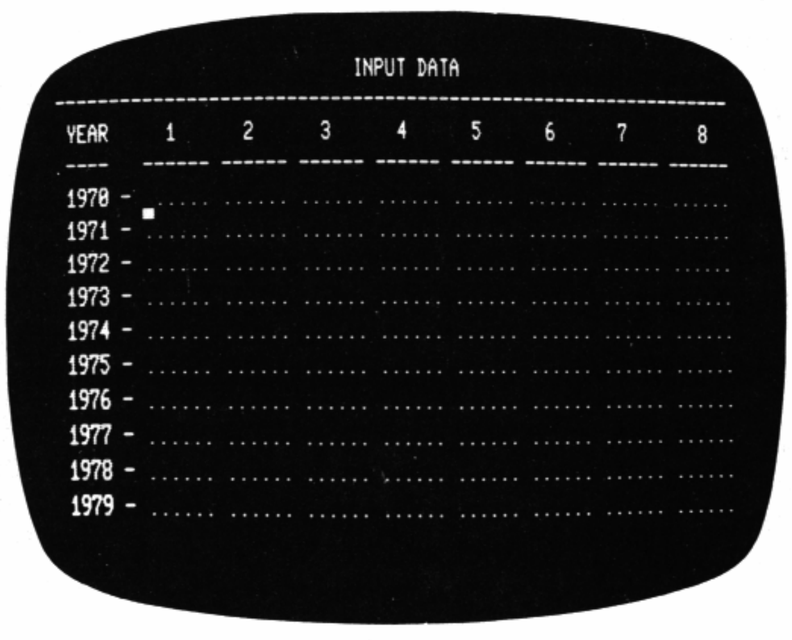
Note: Beginners may find some interesting techniques in this program if they take some time to examine it carefully.

```
1Ø DIM S(1ØØ)
     : CLS
     : X= -3
2Ø FOR L=1 TO 1Ø
     : X = X + 2
3Ø FOR W=15392 + (64*L) - ((X/2) + .5)
     TO 15392 + (64*L) + ((X/2) + .5)
4Ø POKE W, 42
     : Y=Y + 1
     : S(Y)=W
5Ø NEXT W
     : NEXT L
6Ø X= -1
     : FOR Y = 16Ø94 TO 16Ø98
7Ø POKE Y, 191
     : NEXT Y
8Ø PRINT @ 914, "M E R R Y  C H R I S T M A S";
9Ø RANDOM
1ØØ Y=RND(99) + 1
     : IF PEEK(S(Y)) = 42 THEN POKE S(Y), 43
     : ELSE POKE S(Y), 42
11Ø X=X * -1
     : IF X=1 THEN POKE S(1), 42
     : ELSE POKE S(1), 43
12Ø GOTO 1ØØ
```

Matrix Entry

For the Model III

by Albert F. Sargent
Hot Springs, Ark.



This data entry subroutine is useful for entering matrices and has the advantage that any element of data may be changed by moving the cursor using SHIFT arrows (left, right, up, or down. Note: Model III users should use SHIFT ← Z instead of SHIFT ←.) and then typing over the incorrect data. ENTER causes the cursor to jump from one data field to the next. The final ENTER (at element 8,10) causes entry of all data displayed. When the program is run the display looks like the above display.

To use this subroutine independently of another program delete line 440 and add line 100 (100 CLEAR 100).

```
2ØØ CLS
     : PRINT TAB(27) "INPUT DATA"
     : PRINT STRING$(63, "-")
     : PRINT
23Ø PRINT"YEAR      1      2      3      4
     5      6      7      8"
24Ø PRINT" ----    ------ ------ ------ ------
     ------ ------ ------ ------"
25Ø FOR I=197Ø TO 1979
26Ø PRINT USING " #### - ..... ..... .....
     ..... ..... ..... ..... .....";I
27Ø NEXT I
28Ø FOR J=5 TO 14
29Ø FOR I=8 TO 57 STEP 7
3ØØ POKE 16417, CINT(J/4)+6Ø
     : POKE16416, I+64*J-256*CINT(J/4)
     : PRINT CHR$(14);
32Ø A$=INKEY$
     : IF A$<>CHR$(13) THEN PRINT A$;
     : GOTO 32Ø ELSE PRINT CHR$(15);
34Ø NEXT I, J
35Ø FOR I=1 TO 1Ø
36Ø FOR J=1 TO 8
37Ø K=15361+(I+4)*64+7*J
     : Z$=""
39Ø FOR L=K TO K+5
4ØØ Z$=Z$+CHR$(PEEK(L))
41Ø NEXT L
42Ø M(I, J)=VAL(Z$)
43Ø NEXT J, I
44Ø RETURN
```

Line 200 clears the screen, tabs 27 spaces to print the heading INPUT DATA, prints a string of 64 hyphens under the INPUT DATA heading and prints a blank line.

Lines 230-240 print column headings and a broken line made up of hyphens under each heading.

Lines 250 and 270 set up a FOR/NEXT loop where variable "I" will be equal to the numbers 1970 to 1979.

Line 260 — Each time through the loop, "I" is printed according to the format defined in the PRINTUSING statement along with eight strings of six dots each separated by a space.

Line 300 — Address 16416 is the Least Significant Byte (LSB) and address 16417 is the Most Significant Byte (MSB) of the cursor address. These two addresses point to the position of the cursor. CHR$(14) turns the cursor on.

Line 320 If A$ does not equal ENTER (CHR$13) then A$ is "printed." "Printed" because if A$ equals one the four SHIFT arrow keys, the cursor moves to different positions on the video. For example, if A$ equals CHR$(27) which is the code for the SHIFT UP arrow then the cursor moves up one line. The entry isn't terminated until A$ equals CHR$(13). When A$ equals CHR$(13) then the LSB and MSB are incremented to point to the next field position on the video. CHR$(15) turns the cursor off.

Lines 350-430 set up a routine to PEEK video memory and store the values stored in the matrix displayed on the video in an array called M.

A BASIC Program for Determining the Median

Also Determines Percentile Values

by John C. Gallagher, M.D.
Treasure Island, FL

This program is to determine the median in a set of whole integer numbers. It also determines the percentile value for each of the entered numbers. I have not seen any published BASIC program for determining the median, and this program is presented to meet this need. As written, the program requires a printer for output.

The median is a useful measurement of central tendency of a set of data. The arithmetic mean is easier to calculate, but it can be greatly affected by a single extreme value. The median is not so affected. The median is usually determined by arranging the data numbers in ascending or descending order and finding the value of the middle one.

For example, if there are 9 numbers, the median will be the value of the 5th one. If the following are the data: 6, 32, 45, 8, 3, 1, 50, 12 and 25, they would be arranged in ascending order as: 1, 3, 6, 8, 12, 25, 32, 45 and 50. The middle number (5th number), which is 12, is the median. In a series of N numbers, if N is odd, the median will be the (N + 1)/2th number.

If the series consists of an even number of data items, the median is assumed to be midway between the N/2th number and the (N/2 + 1)th number. If N = 10, the median would be the average of the 5th and 6th numbers.

This program for calculating the median has to take into account the following conditions: (1) the data will not be arranged in ascending or descending order, (2) the number of entries may be either odd or even, and (3) some numbers will be repeated. Also, this program is designed only for whole (or integer) numbers.

The program consists of four parts: (1) counting the number of entries, (2) determining the lowest entry, (3) determining the highest entry, and (4) calculating the median and the percentile of each entry.

The first part of the program is for counting the entries in the data line(s). This part of the program is on lines 50-100. The ON ERROR line (line 50) is to prevent the OUT OF DATA error that would occur after the last entry was read.

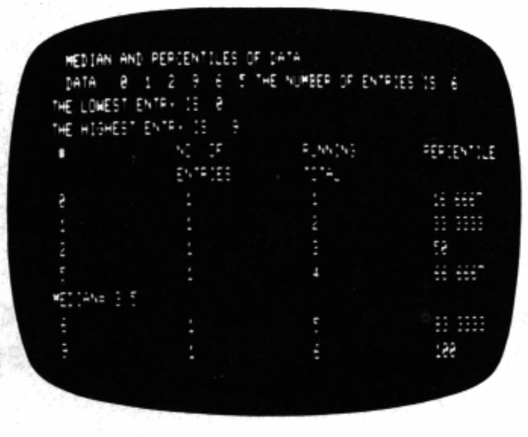
The second and third parts of the program are for determining the lowest (lines 110-200) and the highest (lines 210-300) entries. These two results are used in line 360 for sorting the data.

The fourth part of the program is for calculating the median by sorting the data and counting the entries. As progressively higher numbers are counted, a running total is kept and the percentiles are calculated. Lines 410 and 420 are used if there are an even number of data entries, and line 430 is used if there are an odd number of entries.

The examples that follow can be used to test the program.

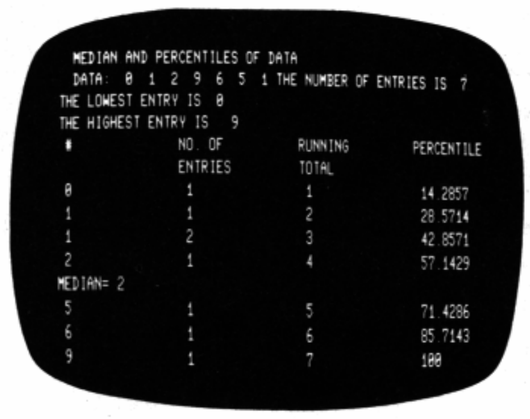
Example 1. On line 100 enter "DATA 0, 1, 2, 9, 6, 5, 1"

The results will be:



Example 2. On line 1000 enter "DATA 0, 1, 2, 9, 6, 5"

The results will be:



The Program:

```
1Ø REM MEDIAN AND PERCENTILES
11 REM PUT DATA ON DATA LINE AFTER LINE 5ØØ
12 REM USE WHOLE REAL NUMBERS ONLY
2Ø INPUT "WHAT IS TITLE OF DATA"; A$
3Ø LPRINT A$; "  MEDIAN AND PERCENTILES OF DATA"
4Ø LPRINT "  DATA: ";
5Ø ON ERROR GOTO 9Ø
6Ø READ X
    : LPRINT X;
7Ø N=N+1
8Ø GOTO 5Ø
9Ø LPRINT
    : RESUME 1ØØ
1ØØ RESTORE
11Ø REM  LO=LOWEST NUMBER IN SERIES
12Ø READ X
13Ø READ Y
14Ø IF X<Y THEN LO=X
15Ø IF Y<=X THEN LO=Y
16Ø FOR A=1 TO N-2
17Ø READ X
18Ø IF X<LO THEN LO=X
19Ø NEXT A
2ØØ RESTORE
21Ø REM  HI=HIGHEST NUMBER IN SERIES
22Ø READ X
23Ø READ Y
24Ø IF X>Y THEN HI=X
25Ø IF Y>=X THEN HI=Y
26Ø FOR A=1 TO N-2
27Ø READ X
28Ø IF X>HI THEN HI=X
29Ø NEXT A
3ØØ RESTORE
31Ø LPRINT "THE NUMBER OF ENTRIES IS "; N
32Ø LPRINT "THE LOWEST ENTRY IS "; LO
33Ø LPRINT "THE HIGHEST ENTRY IS  "; HI
34Ø LPRINT " # ", "NO. OF", "RUNNING",
    "PERCENTILE"
35Ø LPRINT " ", "ENTRIES", "TOTAL"
36Ø FOR C=LO TO HI STEP 1
37Ø J=Ø
38Ø FOR Q=1 TO N
39Ø READ X
4ØØ IF X=C THEN J=J+1
    : S=S+1
    : LPRINT X, J, S, S*1ØØ/N
41Ø IF S=N/2 AND P=Ø THEN W=X
    : P=1
42Ø IF S=N/2+1 AND P=1 THEN WW=X
    : LPRINT "MEDIAN="; (W+WW)/2
    : P=2
43Ø IF S=(N+1)/2 AND P=Ø THEN LPRINT "MEDIAN=";
    X
    : P=1
44Ø NEXT Q
45Ø RESTORE
46Ø NEXT C
47Ø PRINT "END OF PROGRAM"
48Ø END
```

Restore Data Lines

by Don Taylor

This BASIC subroutine allows you to set the BASIC pointer used to read DATA statements to the beginning of any line in the program without having to read any data that might be in the way. Variable "D" stores the line number where you want to start reading. The routine starts at the bottom, lowest line number, of BASIC text and line by line searches for a line number equal to "D." If one is not found, the program prints an error message and halts. If the line number is found, the READ pointer which is stored in RAM is modified for the new READ location. Because this is written in BASIC it will be somewhat slow if there are a great number of lines to search. To speed this up you should put your data statements toward the beginning of the program.

If you would like to know which line is currently being read there is an almost foolproof way to find out.

```
PRINT (PEEK(166Ø2) + 256 * PEEK(166Ø3))
```

This will print the line currently being read, IF at least one item has been read since the last RESTORE command.

Each READ command corrects the current line number, but RESTORE does not.

The Program:

```
1Ø ON ERROR GOTO 1ØØØØ
1ØØ FOR I=1 TO 1Ø
11Ø READ A
12Ø PRINT A
13Ø NEXT I
14Ø INPUT "RESTORE TO WHICH LINE "; D
15Ø GOSUB 1ØØØ
16Ø FOR I=1 TO 5
17Ø READ A
   : PRINT A
18Ø NEXT
19Ø GOTO 14Ø
2ØØ DATA 1
21Ø DATA 2
22Ø DATA 3
23Ø DATA 4
24Ø DATA 5
25Ø DATA 6
26Ø DATA 7
27Ø DATA 8
28Ø DATA 9
29Ø DATA 1Ø
3ØØ DATA 11
31Ø DATA 12
1ØØØ A=PEEK(16548) + 256 * PEEK(16549)
1Ø1Ø B=PEEK(A) + 256 * PEEK(A+1)
1Ø2Ø IF B=Ø THEN PRINT "ERROR, END OF PROGRAM"
     : END
1Ø3Ø C=PEEK(A+2) + 256 * PEEK(A+3)
1Ø4Ø IF C <> D THEN A=B
     : GOTO 1Ø1Ø
1Ø5Ø A=A-1
1Ø6Ø E=INT(A / 256)
1Ø7Ø F=INT(A - 256 * E)
1Ø8Ø POKE 16639, F
1Ø9Ø POKE 1664Ø, E
11ØØ G=INT(D / 256)
111Ø H=INT(D - 256 * G)
112Ø POKE 166Ø2, H
113Ø POKE 166Ø3, G
114Ø RETURN
1ØØØØ IF ERR / 2 + 1 = 4 THEN PRINT "OUT OF
      DATA"
      : RESUME 14Ø
1ØØ1Ø RESUME 1Ø
```

Lines 10, 10000, 10010 form an error trapping routine so if the error encountered (line 10) is Out of Data the message "OUT OF DATA" is printed and execution goes to line 140 (line 10000). If ERR/2+1 does not equal 4 then the program returns to line 10 (line 10010). This could result in an endless loop where ERROR/2+1 <> 4.

Lines 110-130 set up a loop to read the data in lines 200-310.

Line 140 asks for input of the line number that you want the data statements restored to. The line number is stored in variable "D."

Line 150 branches to line 1000.

Line 1000-1130 make up the subroutine which resets the pointers and allows you to read previously read DATA statements. Line 1000 contains two undocumented addresses (16548 and 16549) which could possibly change with future versions of the ROM.

Line 1000 'Get pointer to bottom of program
Line 1010 'Get pointer to next line of program
Line 1030 'Get line number of this line
Line 1040 'If wrong, up pointer, try again
Line 1050 'Back up pointer one byte
Line 1060 'Get top half of pointer
Line 1070 'Get bottom half of pointer
Line 1080 'Save bottom half of read pointer
Line 1090 'Save top half of read pointer
Line 1100 'Get top half of line number
Line 1120 'Save bottom half of line number
Line 1130 'Save top half of line number

Line 1140 RETURNs you to line 160 where the DATA statements can be read again starting at the line specified in line 140.

NOTE: This program runs on Model I and III under Disk BASIC as well as on the Model I Level II BASIC and Model III BASIC.

Learning Character Recognition

by Douglas Fuge
2427 Hoohoihoi Street
Pearl City, HI 97682

Having just purchased a Line Printer IV and SCRIPSIT, I am taking the opportunity to forward a program I wrote for my daughter, a pre-schooler. The idea came when I returned home from a long absence and was treated to the sight of my three-year-old typing her name on the screen of my TRS-80 in the command mode.

The attached program, which I named "Bracken" after my daughter, provides for use of words in the data statements or entry of 10 phrases. It was interesting how fast she learned to replicate the displayed words. In addition to learning character recognition, the exercise taught her the location of the letters of the alphabet on the keyboard and through repetition, she learned to read the words.

Note that lines 1 through 70 comprise the core program, the subroutine starting at line 100 is the "live keyboard" routine from the May 1979 Newsletter and the subroutine beginning at 1000 prints the happy face on the video display.

I would be pleased if you would consider the attached program for reprinting in the Microcomputer News. I hope some readers with pre-school children will be able to enjoy the happy face when the exercise is properly typed.

```
1 CLS
      : INPUT" 1= 1Ø PHRASES      2= 1Ø WORDS FROM
      MEMORY";O
      : ON O GOTO 2, 4
2 CLS
      : CLEAR 5ØØ
      : DIM H$(1Ø)
      : PRINT"ENTER 1Ø PHRASES"
      : FOR V=1 TO 1Ø
      : PRINT V;
      : INPUT H$(V)
      : NEXT V
3 GOTO 5
4 CLEAR 5ØØ
      : DIM H$(1Ø)
      : FOR V=1 TO 1Ø
      : READ H$(V)
      : NEXT V
      : CLS
5 FOR V=1 TO 1Ø
      : H$=H$(V)
1Ø FL=LEN(H$)
      : H=FL/2
      : CLS
      : PRINT@ 48Ø-H, H$
      : PRINT@ 544-H, "";
2Ø GOSUB 1ØØ
3Ø IF IN$=H$ THEN 4Ø ELSE 1Ø
4Ø GOSUB 1ØØØ
5Ø FOR Z=1 TO 2ØØØ
      : NEXT Z
6Ø NEXT V
7Ø GOTO 5
1ØØ IN$=""
      : W$=INKEY$
      : W=14
      : WD=Ø
      : WL%=WD
      : IF FL=WD THEN FL=1
1Ø5 PRINT STRING$(ABS(FL),136);
      STRING$(ABS(FL),24);
11Ø PRINT CHR$(W);
      : FOR W%=1 TO 25
      : W$=INKEY$
      : IF W$ <> "" THEN 115 ELSE NEXT
      : PRINT CHR$(15);
      : FOR W%=1 TO 25
      : W$=INKEY$
      : IF W$ <> "" THEN 115 ELSE NEXT
      : GOTO 11Ø
115 PRINT CHR$(W);
      : IF ABS(FL)=WL%
           THEN 125
           ELSE
             IF FL>Ø AND W$ >= " " AND W$ <= "z"
                THEN 17Ø
                ELSE
                  IF FL<Ø AND W$>"/" AND W$<":"
                     THEN 17Ø
117 IF W$="," THEN PRINT W$;
      : WL%=WL%+1
      : GOTO 175
12Ø IF W$="."
      AND WD=Ø THEN WD=1
      : GOTO 17Ø
123 IF (W$="-"OR W$="+") AND WS=Ø AND WL%=Ø
      THEN WS=1
      : GOTO 17Ø
125 IF W$ <> CHR$(8)
      THEN 15Ø
      ELSE
        IF WL%=Ø
           THEN 11Ø
           ELSE
             PRINT CHR$(24);
             : IF FL > Ø
                  THEN 135
                  ELSE
                    IF PEEK (16418) = 44
                       THEN 14Ø
13Ø IF PEEK (16418) = 46
      THEN WD = Ø
           : GOTO 135
      ELSE
        IF PEEK(16418)=43Ø OR PEEK(16418)=45
           THEN WS = Ø
135 IN$=LEFT$(IN$, LEN(IN$)-1)
14Ø WL%=WL%-1
      : POKE 16418, 136
      : GOTO 11Ø
15Ø IF W$ = CHR$(24)
      THEN PRINT STRING$ (WL%,CHR$(24));
           : GOTO 1ØØ
155 IF W$ <> CHR$(13) THEN 11Ø ELSE PRINT
     STRING$(ABS(FL)-WL%, 32);
16Ø PRINT CHR$(15);
      : W%=25
      : NEXT
      : RETURN
17Ø PRINT W$;
      : IN$=IN$+W$
      : WL%=WL%+1
175 IF ABS(FL) = 1 THEN 16Ø ELSE 11Ø
1ØØØ FOR Z=Ø TO 35
      : IF (1225-Z[2) < 1 THEN Y = Ø ELSE
      Y = .363636 * SQR(1225-Z[2)
      : SET(61+Z,24+Y)
      : SET(61+Z,24-Y)
      : SET(61-Z,24+Y)
      : SET(61-Z,24-Y)
      : NEXT Z
1Ø1Ø FOR Y = 21 TO 26
      : SET(96,Y)
      : SET(26,Y)
      : NEXT Y
1Ø2Ø FOR Z = Ø TO 2Ø
      : IF (4ØØ-Z[2) < 1 THEN Y = 1 ELSE
      Y = .363636 * SQR (4ØØ-Z[2)
      : SET(61+Z,24+Y)
      : SET(61-Z,24+Y)
      : NEXT Z
1Ø3Ø FOR Z = 4 TO 5
      : SET(61+Z,18)
      : SET(61-Z,18)
      : SET(61+Z,17)
      : SET(61-Z,17)
      : NEXT Z
1Ø4Ø RETURN
2ØØØ DATA BRACKEN, TINA, MICHELLE, GRANDPA,
      MALIA, CHEERIE, DAISY, TERI, JANEEN, BIG
      BIRD
9999 GOTO 9999
```

Word Alphabetizer

by Burt T. Jackson
South Holland, IL

This program alphabetizes a list of words. It asks for the number of words to be alphabetized, the input of those words and prints them out in the order that they were entered and then in alphabetical order. The program was designed to run on a Level II machine.

To alphabetize a list of ten zoo animals, a run of the program might look like this.

```
ENTER THE NUMBER OF WORDS TO BE ALPHABETIZED? 1Ø
ENTER WORD NUMBER 1    '# will be from 1 to 1Ø.
? ELEPHANT             'And nine other zoo animals
```

After the animals have been entered the list is printed in the same order as it was entered.

```
ELEPHANT        BEAR          MONKEY        ZEBRA
YAK             LLAMA         LION          BUFFALO
JAGUAR          TIGER
```

The list is sorted and printed again in alphabetical order.

```
BEAR            BUFFALO       ELEPHANT      JAGUAR
LION            LLAMA         MONKEY        TIGER
YAK             ZEBRA
```

The speed of the sort depends on the length of both the words and list.

The program:

```
5  CLEAR 1000
10 CLS
20 INPUT "ENTER THE NUMBER OF WORDS TO BE
     ALPHABETIZED"; N
30 DIM A$(N)
40 FOR X = 1 TO N
     : PRINT "ENTER WORD NUMBER"; X
     : INPUT A$(X)
50 CLS
     : NEXT X
60 FOR X = 1 TO N
     : PRINT A$(X),
     : NEXT X
     : PRINT
70 PRINT
     : PRINT
80 FOR I = 1 TO N
90 FOR J = 1 TO N - 1
100 IF A$(J) > A$(J+1) THEN 110 ELSE 140
110 T$ = A$(J)
120 A$(J) = A$(J+1)
130 A$(J+1) = T$
140 NEXT J, I
150 FOR X = 1 TO N
160 PRINT A$(X),
     : NEXT X
```

Monthly Calendar

by Dean Beazly
Mansfield, IL

This program allows you to make a calendar for any month from the year 1700 on.

```
5 LPRINT TIME$"  PROGRAM FOR MAKING A MONTHLY
    CALENDAR BY DEAN BEAZLY"
10 CLEAR
     : A=0
     : INPUT "# OF MONTH, # OF YEAR FOR THE
     CALENDAR";M,YR
20 Y=(YR)-1700
25 LD%=(Y+100)/400
30 LD%=(LD%+(Y/4))-INT(Y/100)
35 LD=LD%
36 PRINT"LEAP";LD
40 DA=LD+(Y*365)+5
41 IF Y/4=INT(Y/4)THEN FD=1
42 IF M<2 AND FD=1 THEN GOSUB 400
43 IF M=2 AND FD=1 THEN GOSUB 402
     : PRINT"FD=";FD
45 ON M GOSUB 210, 220, 230, 240, 250,
     260, 270, 280, 290, 295, 299, 300
49 LPRINT"SUN    MON    TUE        WED        THU
     FRI    SAT"
50 WK%=DA/7
52 DW%=DA-(WK%*7)
55 B=DW%
56 IF M=2 THEN E=E+FD
60 FOR L=0 TO B
     : A$(L)=""
     : NEXT
65 FOR W=L TO 7
66 IF W=1 THEN L=1
70 A=A+1
71 IF A>(E) THEN S=1
72 A$(L)=STR$(A)
73 IF S=1 THEN A$(L)=" "
75 L=L+1
79 NEXT
80 LPRINT USING"% %    % %     % %       % %
     % %     % %      % %  ";A$(L-7), A$(L-6),
     A$(L-5), A$(L-4), A$(L-3), A$(L-2), A$(L-1)
85 L=1
86 IF S=1 THEN LPRINT DA-5;"DAYS FROM
     1/1/1700"
     : GOTO 10
90 GOTO 65
210 DA=DA
     : E=31
     : LPRINT TAB(20)"JAN";YR
     : RETURN
220 DA=DA+31
     : E=28
     : LPRINT TAB(20)"FEB.";YR
     : RETURN
230 DA=DA+59
     : E=31
     : LPRINT TAB(20)"MARCH";YR
     : RETURN
240 DA=DA+90
     : E=30
     : LPRINT TAB(20)"APRIL";YR
     : RETURN
250 DA=DA+120
     : E=31
     : LPRINT TAB(20)"MAY";YR
     : RETURN
260 DA=DA+151
     : E=30
     : LPRINT TAB(20)"JUNE";YR
     : RETURN
270 DA=DA+181
     : E=31
     : LPRINT TAB(20)"JULY";YR
     : RETURN
280 DA=DA+212
     : E=31
     : LPRINT TAB(20)"AUG";YR
     : RETURN
290 DA=DA+243
     : E=30
     : LPRINT TAB(20)"AUG";YR
     : RETURN
295 DA=DA+273
     : E=31
     : LPRINT TAB(20)"OCT.";YR
     : RETURN
299 DA=DA+304
     : E=30
     : LPRINT TAB(20)"NOV.";YR
     : RETURN
300 DA=DA+334
     : E=31
     : LPRINT TAB(20)"DEC.";YR
     : RETURN
400 FD=0
401 LD=LD-1
402 DA=DA-1
     : RETURN
```

TRS-80 PILOT Plus is Here

RADIO SHACK'S NEW PILOT Plus AUTHOR LANGUAGE

by Dr. Dan Gibbs

Radio Shack's TRS-80 PILOT Plus authoring language was designed with one major purpose in mind: to encourage courseware development by classroom teachers and administrators, and to simplify that task.

In an authoring language, more emphasis is placed upon instructional design and strategies than it is in the other "mainstream" computer languages. This emphasis is especially appropriate: the authoring language must be easy to use and yet flexible enough to allow educators to implement their instructional plans without having to sacrifice their design.

TRS-80 PILOT Plus is just such an authoring language. It is based on PILOT, the computer language which has been used in education for many years as a computer-assisted instruction (CAI) development language. The acronym PILOT is itself a capsulized description of the language's intent, that of Programmed Inquiry, Learning, Or Testing. The "Plus" in the TRS-80 version of PILOT refers to the graphics creation and file handling capabilities which have been added.

Attractive, well-planned screen displays are essential in effective CAI programs. All too often, the person actually authoring courseware at a computer is more caught up in the manipulation of program statements and instruction "code" than with the logical placement of text and graphics on the display. In the development of quality courseware, decisions must be made concerning the placement of such text items as positive and negative feedback messages, student key input, titles, captions, and continuation messages. The questions "What will the student see?" or "How will the student perceive this?" must be asked almost constantly during the development process.

A brief description and some examples of the commands used in TRS-80 PILOT Plus will show how this authoring language can meet the needs of educators interested in creating their own CAI materials.

PILOT Plus is a command-oriented language. All of the twenty-three program commands and two of the six graphics commands are single alphabetic characters. Each of these characters represents a key word which describes the function of each command. This one-on-one relationship between the command and its function makes review of all the commands as simple as reciting the alphabet! For example, A: means ACCEPT (student input), B: means BIG (characters), C: means COMPUTE (assign value to a variable), and so on. (In one of the sections on the PILOT Plus Reference Card which accompanies each PILOT Plus manual, all PILOT Plus commands are listed alphabetically and their functions are given.)

Some of PILOT's most-used commands are the T, A, M, Y, and N commands.

The T command displays Text on the screen. The statement,

```
T: HOW DO YOU FEEL?
```

displays "HOW DO YOU FEEL?" on the screen.

The A command Accepts a student response from the keyboard. This statement,

```
A:
```

causes the computer to wait until the student responds with a typed-in answer.

The M command Matches the student's response against possible responses that the teacher determines. The statement,

```
M:OK/OKAY/GOOD/FINE/WONDERFUL
```

matches student responses against those five words.

The Y modifier can be used with other commands to tell the computer what to do if a match is found. This statement,

```
TY: "I'M GLAD YOU'RE OKAY"
```

will display "I'M GLAD YOU'RE OKAY" if the student's response matches any of those in the M command.

The N modifier tells the computer what to do if no match is found. The statement,

```
TN: "I'M SORRY TO HEAR IT"
```

will display "I'M SORRY TO HEAR IT" if the students' response doesn't match any of the responses the teacher has specified in the M command.

This set of instructions,

```
T: HOW DO YOU FEEL?
A:
M: OK/OKAY/GOOD/FINE/WONDERFUL

TY: I'M GLAD YOU'RE OKAY
TN: OH, I'M SORRY TO HEAR IT
```

performs all those operations in sequence.

In PILOT Plus the author can maintain control over the placement of text with the PRINT AT instruction which is placed after a T: (TEXT) command. This instruction contains numeric coordinates from the TRS-80 PRINT AT grid to specify beginning text location. The PILOT Plus documentation emphasizes the use of the PRINT AT instruction because the location of the displayed text is just as much a part of the total graphics design as the drawn graphics figures. (A lacquered card packaged with each PILOT Plus manual contains the PRINT AT Grid on one side and the GRAPHICS Grid on the other. This card was designed to be written on with a grease pencil in the planning of screen displays.)

The GRAPHICS/PLT utility program which is included on each PILOT Plus diskette can be used to create screen displays which contain text and/or graphics figures. Text placement and graphics creation are much easier with this utility, and the results are immediately seen by the developer. Screen displays which are created with GRAPHICS/PLT are individually named and saved in separate "files" on the lesson diskette. These files are opened and displayed with the PILOT Plus file-handling commands, H: and K:. For example,

```
H:>"HELLO"
```

The file called HELLO is located on diskette and opened.

```
K:
```

Displays previously-opened file (HELLO) on screen, then closes it.

TRS-80 PILOT Plus graphics can also be created using graphics commands. The difference between graphics created with and displayed in the utility program and those created with the graphics commands and displayed under program control is not discernible to the student. The chief difference lies in the method of creation. With the utility program, the user is not working with number coordinates, but directly with the visual effect as he manipulates keys and draws on the screen. With the graphics commands, the user supplies numbers (coordinates) from the TRS-80 GRAPHICS Grid card.

The graphics commands are still as easy to use as other program commands. For example,

S:(64,24)	SETs a dot or point at screen location 64,24, on the graphics grid
X:(64,24)	EXTINGUISHes or erases the dot
G:S,20,30,100,30	SETs a GRAPHICS point at location 20,30, then draws a line from this point to end point at location 100,30
G:R,20,30,100,30	RESETs or erases Graphics point at 20,30 and the line between it and location 100,30
G:SB,20,35,100,45	SETs a BOX with one corner at 20,35 and the opposite corner at 100, 45
G:RB,20,35,100,45	RESETs or erases a BOX with one corner at 20,35 and the opposite corner at 100,45

The graphics utility program and the commands that control the text display, graphics, student input, and screen files are only a few of the commands available with TRS-80 PILOT PLUS. The other features are described in the manual. Also included in the package are a reference card and a program diskette. They are packaged in a durable binder for long use.

The TRS-80 PILOT Plus logo really says it all. This logo features the wheel and compass at a ship's helm. On an ocean vessel, control of the ship is maintained with the pilot's steering wheel; proper direction is possible because of the compass. In TRS-80 PILOT Plus, an author language especially for teachers, this same CONTROL and DIRECTION are possible in a different sphere — in the development of quality educational materials for today's students.

Express for Success

The Radio Shack Proposal Writing Guide

by Luanne Kelly

Radio Shack Education Consultant Dr. Norman T. Bell and Dallas Independent School District Administrator Dr. Frank Jackson have had years of experience in writing successful proposals that helped their schools obtain federal and private funding. Now the two educators have combined their proposal writing insights in the Radio Shack Proposal Writing Guide.

The guide presents a programmed, step-by-step approach to the writing of a successful proposal — a document which states all of the facts about the project that the proposal reader will need to know, and which does so clearly and concisely. The aim of the guide is to help the educator sell his or her project ideas by communicating them effectively.

Bell and Jackson base their book on the idea that a clear plan for a project and a clear statement of that plan go together. Most proposal readers assume that if you are well-organized in expressing your ideas, you will be well-organized in implementing them. The Proposal Writing Guide leads the educator through the process of developing each of the six components necessary to a good proposal.

Each of the book's six units follows this pattern:

EXPLANATION — The educator is given an EXPLANATION of the proposal component and its main elements.

EXAMPLE — The educator reads an EXAMPLE of the proposal element to be written. Examples are based upon a proposal to implement a computer-assisted reading program in a public school system.

PRACTICE — The educator writes a PRACTICE element similar to the example but based upon information relevant to his or her own proposal.

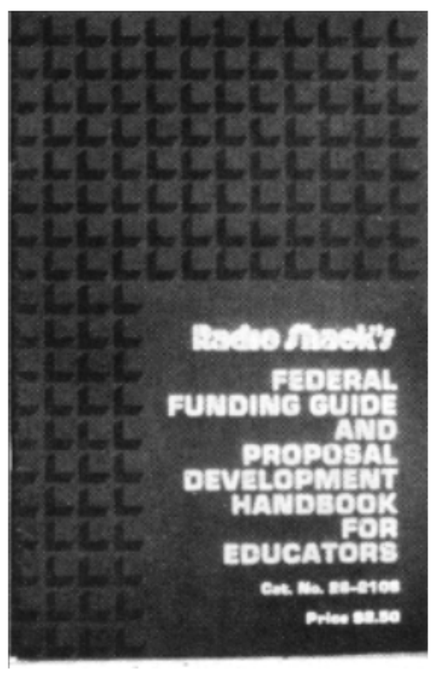
When you have worked your way through the practice exercises in the guide, you will be well on your way to submitting an effective proposal to the targeted funding agency.

Dr. Bell, who is Professor of Educational Psychology at Michigan State University, recently directed a federally funded faculty development program at Michigan State's college of medicine. Dr. Bell is also the author of the Radio Shack Computer Education Series, including Part 1: Introduction to BASIC, Part 2: BASIC Programming, and the forthcoming volume, Part 3: Advanced BASIC.

Dr. Frank Jackson is currently Administrator of Instructional Support Systems for the Dallas Independent School District. For six years, Dr. Jackson was in charge of Proposal Development for the Dallas schools. During that period, the amount of federal funding received annually by the school district was increased from approximately $5 million to more than $20 million.

The Proposal Writing Guide was designed as a companion piece to the Radio Shack Federal Funding Guide and Proposal Development Handbook for Educators, written by Dr. Jackson in 1980. The Federal Funding Guide identified the major federal funding sources and introduced the main parts of the effective proposal. Now the new Proposal Writing Guide provides valuable step-by-step direction for the educator who wishes to sharpen his or her proposal writing skills.

The Proposal Writing Guide will be available through your local Radio Shack store or dealer in January, 1982. The stock number for the guide is 26-2754 and the suggested retail price is $9.95.



Spelling Checker

The Model II SCRIPSIT Spelling & Hyphenation Dictionary is an invaluable aid in proofreading any document and putting it into finished form. The spelling checker finds spelling errors and allows you to correct them, while the hyphenator "evens up" ragged line endings by dividing words in the proper location.

Once you've installed the dictionary, it can be invoked as a SCRIPSIT command like any other. (You do need to have the dictionary diskette inserted, however.) From within your document, type ESC U S, and it will be checked for spelling errors at about 2,000 words per minute. SCRIPSIT will keep you informed of how many words it has run across that aren't in its dictionary, and what page it's currently looking at.

When the dictionary program is finished looking at your document, it will position the cursor at the first word it didn't find, highlight it, and ask you what you want to do about it. You can replace it with the correct spelling, add it to your user dictionary (including hyphenation information!), ignore it (uncommon words, names of obscure towns, etc.), flag it for more detailed examination later, or delete it entirely. All changes are made to the document as you specify them, and text will be automatically reformatted to take care of any insertions or deletions made during correction.

The SCRIPSIT Spelling & Hyphenation Dictionary has about 100,000 words in its dictionary. This is more than anyone else's. And it's easier to use, since it interfaces directly with SCRIPSIT. (The simple installation procedure makes it actually part of SCRIPSIT.) You can also have a user dictionary of up to 2,047 special words, names, or whatever.

When your document (letter, memo, or article, like this one!) has had all spelling errors eliminated, you can hyphenate it. Type ESC U H and just watch. This is totally automatic hyphenation! You will never be called upon to make a hyphenation decision, nor do you have to remember where any given word is divided. Instead, the hyphenator looks the word up in the dictionary to determine where to divide it, picks the best location (a word might have several), and hyphenates it there. It updates your document automatically while it does this, displaying each page briefly after it finishes with it, so you can see what it is doing. It won't hyphenate more than two lines in a row, any capitalized word, or the last word in a paragraph — just like your English teacher taught you back in high school. And you can protect parts of your text from hyphenation by "locking" individual paragraphs.

The document preparation process now includes two extra steps: it is now type, spell check, hyphenate, repaginate, print. These two extra steps (requiring little operator intervention) will virtually eliminate typographical errors (no more "recieve" for "receive," "hte" for "the"), and produce a more professional final appearance.

You will probably find, once you begin working with the Spelling & Hyphenation Dictionary, that you'll be wondering how you got along without it. Besides correcting for fingers that don't always behave in an orderly fashion, my documents are now hyphenated, evening up the right margin even when I don't justify them. It was always such a problem to hyphenate a document before that I rarely bothered. When I did take the time, I hardly ever got the hyphens in the right places the first time. Now, I just type ESC U H before I repaginate, and it's taken care of. I do it before I repaginate, rather than after, to keep the last line of the page from becoming only quarter- or half-length, as it might otherwise.

They're quite complex programs, but the complexity is all inside. You don't worry about it, any more than you worry about the rest of the functions of SCRIPSIT. And having the routine invoked as a SCRIPSIT command offers just about the ultimate in convenience — it's always there when you need it.

"Fine," you say, "but I have a Model III. What can I do?" Well, there is a Model I/III version of the spelling checker will soon be available, too. The Model I/III version won't automatically hyphenate, but it will check the spelling of any ASCII file. This includes files from SCRIPSIT, SuperSCRIPSIT, another word processor (is there one?), or even BASIC programs saved in ASCII! The Model III dictionary contains over 75,000 words, and the Model I version contains about 30,000 words. As with the Model II version, you can have a user dictionary of over 2,000 words.

The Model I/III doesn't have highlighting, so the word to be corrected (or whatever) is displayed blinking, in context (with the line of text above and below it). The format may not be what is requested for the particular word processor, but word-wrap will function correctly and carriage returns are interpreted properly. As with the Model II version, you can correct a word, flag it, add it to your user dictionary, ignore it, or delete it. The Model I/III program creates a new version of your file for you, leaving the original untouched, in case you suffer from a "spontaneous reboot" or "pernicious amnesia" (forgetting to pay the light bill). It's quite a versatile program!

Just type CHECK followed by the name of the file you wish to check. Or don't type a name, and you'll be asked to specify a file. SuperScripsit invokes the spelling checker from its main menu.

The Model I/III SCRIPSIT Spelling Dictionary will check your file at a speed of about 2,000 words per minute. It will also ignore SCRIPSIT print formatting command lines (such as centering on, or margin setting).

A spelling checker, no matter how many words it has in its dictionary, can't protect against certain types of errors. If I had typed the word "it's" instead of "its" in the preceding sentence (a common error), no spelling checker in the world could have detected the error. The same goes for certain pairs of words that are very close, such as "affect" and "effect." There is no way that a distinction of meaning alone can be detected by a spelling checker. Even with this restriction, however, a spelling checker is worth its weight in gold for people like me who type "spelling" as "spellling" sometimes.

The PATCH:

A Series of Guidelines to Help Remove the Headaches.

Or, Caveat Patchor: Let the Patchor Beware

by George Robertson

This article is written for those computer users who have, at least once, lost many hours of work because a "simple immediate fix" has caused more problems than it solved. It is a rare person who has never lost any data or programs when, upon applying a "quick fix," he discovers a new bug has appeared in a totally unexpected and unrelated area. I can tell you that it has happened to me any number of times.

I confess that I have lost data, programs, and my temper, on some of these occasions. "How could you lose anything if you are following your own procedure?," you might ask. It happened because I said to myself, "I'm really in a hurry . . . I don't have time to use the methodical procedure . . . This patch is so simple that it couldn't possibly affect anything else . . . This patch has been well tested by others so it can't be bad . . . This patch is "official," I got it straight from the programmer . . . My guru gave it to me . . . The home office told me it was good . . . etc."

If you are like me, you really get angry with yourself for not being more careful. Also, we want to try to blame someone else for our failure to be responsible for our own decision to continue with no thought of recovery.

Let's face it, there is no such thing as a perfect computer program. Most people who have ever used a computer program for a while have unexpectedly "discovered a bug." Some of these "bugs" are more serious and frustrating than others.

A vocabulary has surfaced which describes the discovery of the more serious types of bugs, such words as: Bomb, Blow Up, Silent Death, Hang, Crash, Limbo, Never-Never Land, and several unprintable epithets. From this choice of words, you can see what the state of mind is when a problem is suddenly discovered.

This state of mind is why patches are usually accepted for immediate use, no matter what the problem circumstances are. What we would like to point out is that patches may also contain (or cause) unexpected bugs. While patches are tested, they can not be tested under the same rigourous conditions as the original program. The result may be a patch which doesn't appear to cause any problems in general, but which causes major problems in some "obscure" portion of a program.

With this in mind, we would like to suggest that you not patch or correct a diskette if you are not having the stated problem with your diskette. (Note: As with most suggestions, there are exceptions to this one. Occasionally Radio Shack will find a problem which we know will occur, or which, if it occurs will cause serious damage. In these extreme cases we will state that the patch should be applied whether the noted problem is occurring or not.) For example, if we issue a patch which affects users of Line Printer I's, and you have a Line Printer IV, don't make the patch! Further, if we issue a patch that says some users are experiencing error code XX, and you are not getting error code XX, again—don't make the patch. This sounds like a trivial warning, but you would be surprised at the number of people who apply every patch they are able to get their hands on and then start complaining of problems in a program that "has always worked fine."

As the responsible manager of my computer I know that patching any program is an inexact science. Patches are developed to solve a bug in a particular section of a program. Even though the original creator of the program has developed the patch, the problem is that the final testing of the patch usually does not include the multitudes of situations that can exist in the many different user environments.

As the responsible user of my computer, I never assume that any patch given to me will work in my situation. Also, I am not a perfect typist, so I double (triple) check myself at each stage. Indeed, I also have a healthy disregard for the perfection of the patches developed by myself.

The following method was developed as a cure for all of this advanced paranoia, I am lazy. I don't like recovery and re-entry work. If you will use the following method EVERY TIME you install a patch, I will guarantee to you that you will somewhere save yourself many hours of non-productive recovery work, not to mention the peace of mind that comes from knowing that you can always go back to where you started.

There are no revelations in this procedure. It is simple, obvious, and straightforward. Unfortunately, I have found that almost no one uses this procedure exclusively. I think though, that it is only the ones who do not use good procedures that call me for help or describe to me their horrors.

1. The absolute first thing to do before anything else is done is to BACKUP your diskettes. Backup every diskette which has anything to do with the problem program you are attempting to fix. Backup both your program diskettes and your data diskettes. This will allow you to come back and start over if necessary. I can't stress enough about how important this first backup step is. Using these backup copies as your patched testing diskettes from this point forward allows you to retrace any number of steps that you may have taken by simply going back to the originals.
2. The second step is to use the BUILD command (BUILD is available on both the Model II and III) to create a DO file which contains the PATCH commands. The first statement in the DO file should be a PAUSE command which contains a comment about this patch. The PATCH command(s) should follow the PAUSE command and the last item in the DO file is a DO command for this file itself. Name the patching files sequentially so that they can be traced in the same sequence that they were applied.

For example:

```
BUILD PATCHØ1 [ENTER]
PAUSE This patch is to make accounts transfer to next
month. [ENTER]
PATCH FRED:1 A=621F F=33Ø455 C=ØØØØØØ [ENTER]
PATCH HERMAN:1 A=9BA2 F=ØØ44 C=BCDF [ENTER]
DO PATCHØ1 [ENTER]
[ENTER]
```

This second step is important for two reasons. First, you can print out the DO file and compare it VERY CAREFULLY against the original document from which you were keying. There is now no reason for keystroke errors or for forgetting what was entered at the console. Sec-

ondly, it allows you to build and maintain a single "Master Patching Diskette" which contains all of the DO files from every patch which you have ever applied to your current system. This diskette gives you an audit trail in case of trouble.

If you are a multiple disk drive owner, you should make all of your PATCH commands patch a disk in drive 1. In this way you can create your master patching diskette and put it into drive 0 while you insert diskettes that require patching into drive 1. The reason for the PAUSE command contained in the DO file, is to enable you to insert the diskette in drive 1 before you press (ENTER) to patch it. The reason for the last DO command in the DO file is to allow you to continue automatically to the next diskette to be patched in the same way.

Take the printout of the DO file, which you have carefully checked against the original patch information several times, and clip it to the original document from which you were keying. Handwrite the date you applied the patch, which bug you were correcting, and which diskettes were patched. This is an important audit trail also. The more you write down, the less trouble you will have in retracing your steps in the case that problems occur later. File this paperwork in chronological sequence somewhere so that you can find it as needed. The printout also will enable you to remove a patch by reversing the "find and change" operands of the PATCH command.

3. The third step is to patch your backup copies (created in the first step). This should be done by using your verified and stored DO file (created in step two). Don't patch ANY OTHER DISKETTES until you have completed step four.
4. Step four is to TEST your patches by running the previously failing program from beginning to end. Use the backup copies to complete this run. If everything turns out okay, your completed run is now ready for you to continue with your normal procedures, using the backups as your production diskettes. If anything fails, you merely go back to the original (unpatched) diskettes and develop a new plan.
5. Step five is to apply this same patch to the rest of your failing diskettes. Note: never patch a diskette which is not backed up. Test each diskette which you are patching in the same manner as step four. After this step is completed, you have your problem fixed, you have an audit trail of what happened, and you have not been in a panic. You have not lost any data or programs and generally you have a good feeling that all is well.

With this five-step procedure, anyone can maintain his own system to his own satisfaction. You can immediately see that most of the headaches are gone and that you can always bring a diskette up to the current standards for your system by using the master patching diskette created in step two.

I wish all of you the best of luck in using these five simple steps as a way to being totally responsible for maintaining your own systems. Remember that no one else has your exact set of problems to deal with and that there is also no such thing as a perfect PATCH.

Model II Bugs, Errors, and Fixes

PAYROLL (26-4503)

A number of changes are needed to the 1.1 version of Model II Payroll to accommodate the Disability Withholding requirements for California, New Jersey, and Hawaii. If you are running this package in one of these states, please contact a Computer Center, or Computer Customer Services for the needed information.

SERIES I EDITOR/ASSEMBLER (26-4713)

The Series I Editor/Assembler assembles the OTDR instruction improperly. OTDR should assemble as "EDBB" not "ED8B."

You can correct the problem by applying the following patch to your EDTASM program:

```
PATCH EDTASM A=4E3Ø F=8B C=BB [ENTER]
```

You may also wish to correct the comment line in the SAMPLE/SRC listing from: ;ED8B to ;EDBB.

ReformaTTer™ (26-4714)

The ReformaTTer program contains an intermittent error when transferring TRSDOS records with a fixed length of 1 or variable length records to an IBM format with logical records of length 128. The result is that, depending on the data, spurious characters are added to the end of the record in the IBM file.

To correct the problem, make the following patch:

```
PATCH REFM A=46D3 F=13C13E8Ø9Ø28ØB C=C13E8Ø9Ø28ØC13
```

Christmas Tree for the Model II

by Kenneth Willoughby
P.O. Box 317
Fairacres, NM 88033

I am enclosing a series of CHRISTMAS TREE designs from the programs listed below. Please print it for your Christmas edition of the Newsletter. The program is for the Radio Shack TRS-80 Model II. It does not need much memory to run or store. After typing RUN (ENTER) the program is self-prompting and will trace some CHRISTMAS TREES on the screen using the Model II graphics capability.

```
1Ø DATA 143, 131, 138, 127, 126
2Ø READ X
25 ON ERROR GOTO 16Ø
3Ø CLS
4Ø PRINT@8Ø, "MERRY";
5Ø PRINT@1871, "SEASONS";
6Ø PRINT@9Ø, "CHRISTMAS!";
7Ø PRINT@1881, "GREETINGS!!";
8Ø L=4Ø-L
     : K=4Ø+K
     : M=Ø
9Ø FOR C=M TO 2Ø
1ØØ PRINT@(C, L), CHR$(X)
     : PRINT@(C, K), CHR$(X)
11Ø NEXT C
12Ø M=M+1
     : L=L-1
     : K=K+1
13Ø IF M<2Ø THEN 9Ø
14Ø SYSTEM"PAUSE"
15Ø L=Ø
     : K=Ø
     : M=Ø
     : GOTO 2Ø
16Ø RESTORE
     : END
```

Search Routine

by Robert C. Sanguinet

This Model II assembly language search subroutine and BASIC program were written by Mr. Robert C. Sanguinet of South Deerfield, Mass. The machine language sort routine is loaded into memory under DEBUG and accessed by Mr. Sanguinet's BASIC program. Using these two programs you can rapidly search for and locate a string in less

than one second. The BASIC program demonstrates the general method for using the routine by generating a set of random five character strings. These strings are randomly generated in line 450 and stored in the array A$(X). The randomly generated strings are printed out and line 575 asks for a string to be input.

First the machine language search routine.

ADR.	CODE															
F110	11	00	00	D5	ED	53	80	F1	5E	23	56	23	13	ED	53	83
F120	F1	5E	23	56	23	E5	EB	7E	32	86	F1	23	5E	23	56	E1
F130	4E	23	46	ED	43	89	F1	C1	3A	86	F1	4F	D5	2A	89	F1
F140	23	5E	23	56	23	22	89	F1	D5	ED	5B	80	F1	2A	83	F1
F150	13	ED	53	80	F1	AF	ED	52	E1	D1	28	0A	3E	16	CF	20
F160	DB	2A	80	F1	18	03	21	00	00	E5	2A	05	28	E3	C9	21
F170	10	F1	E5	2A	03	28	E9									

At TRSDOS Ready type DEBUG ON. When the prompt Debug is on appears enter DEBUG by typing DEBUG. Press (M) in response to the ?. When M= ? appears type F110. This is the address in memory where the above machine language code will be loaded. Press the (F1) key to move the cursor into the memory display area and enter the above code. After the code is entered, press (F2) and (0) to exit DEBUG. At TRSDOS Ready type DEBUG OFF (ENTER) and then DUMP SEARCHER START= F110 END=F176 (ENTER). Notice in line 275 of the BASIC program that the machine language search routine is loaded into memory under the name SEARCHER.

BASIC program:

```
1ØØ REMINDER--'ALSAD'-- ASSEMBLY LANGUAGE SEARCH
     ALGORITHM DEMONSTRATOR
125 REMODEL--LINES 275 & 375 TO INCREASE OR
     DECREASE SEARCH PARAMETERS
15Ø REM
175 REMARK             INITIALIZE MEMORY
2ØØ REM
225 RANDOM
     : SYSTEM"FORMS T"
25Ø CLEAR 1Ø1ØØ,61711
     : DIM A$(2ØØØ),X(4)  'Space saver
275 SYSTEM"LOAD SEARCHER"                  'The
     program (ALS)
3ØØ DEFINT X,Z                             'Define
     integers
325 DEF USRØ=&HF16F                        'Open
     Sesame ALS
35Ø REM
375 REMARK           RANDOM WORD GENERATOR
4ØØ REM
425 DEF  FNX$(X)=CHR$(RND(26)+64) +
     CHR$(RND(26)+64) + CHR$(RND(26)+64) +
     CHR$(RND(26)+64) + CHR$(RND(26)+64)
45Ø FOR X=1 TO 8Ø
     : LET A$(X)=FNX$(1)
     : LPRINT A$(X)+STR$(X),
     : NEXT X
475 LPRINT
5ØØ REM
525 REMARK           SEARCH DEMONSTRATOR
55Ø REM
575 INPUT B$
6ØØ T1$=TIME$
625 Z=Ø
     : X(Ø)=8Ø
     : X(1)=VARPTR(B$)
     : X(2)=VARPTR(A$(1))
65Ø Z=USRØ(VARPTR(X(Ø)))
     : T2$=TIME$
675 LPRINT Z,A$(Z),B$,T1$,T2$
     : GOTO 575       'Found it??
```

If the input string matches one of the random strings, the following is printed.

The array position number of the string
The array string
The matching input string
The time that the search began
The time that the search ended

Beginning and ending search times are usually the same which means that the match is made in less than one second. If there is no match, all of the above except the array string is printed.

Changing a few lines in the BASIC program allows 10 strings to be entered through input statements. Delete lines 375-450 and insert:

```
375 REMARK            ENTER 1Ø LAST NAMES
4ØØ REM
425 FOR X = 1 TO 1Ø
     : PRINT "ENTER LAST NAME #";X;
     : INPUT A$(X)
     : LPRINT X" " A$(X)
     : NEXT X
```

Now the program requests input of 10 names which are stored in array A$. In this sample run the ten names are: JONES, BOYD, ZUCKERMAN, ADAMS, CROW, DAVIS, WILLIAMS, MOORE, SMITH, NELSON. After each entry, the name and it's array position number are printed on the printer to yield the following.

1 JONES
2 BOYD
3 ZUCKERMAN
4 ADAMS
5 CROW
6 DAVIS
7 WILLIAMS
8 MOORE
9 SMITH
10 NELSON

After the 10 names are entered the ? prompt returns asking for input of the name to search for (B$ in line 575). If the name ZUCKERMAN is entered the following is LPRINTed.

```
3 ZUCKERMAN    ZUCKERMAN    ØØ.Ø2.42     ØØ.Ø2.42
```

The number 3 is the array subscript for the name ZUCKERMAN. The first ZUCKERMAN is the matching array string. The second ZUCKERMAN is B$ or the input string to be matched in the search. The fourth and fifth items printed are the beginning and ending times for the search. If B$ does not match an array name no array name is printed and the printout would appear like this.

```
Ø              JAMES          ØØ.Ø2.52     ØØ.Ø2.52
```

Input will be continually requested until program execution is broken.

If array string A$(X) which is an exact match for the input string B$, is contained in the leftmost characters of a second array string A$(N) and A$(N) precedes A$(X) in the array, the match returned for B$ will be A$(N) even though the exact match is A$(X). For example, B$="Ho" and A$(57)="Ho." Stored in the array are the following names which begin with the letters "Ho."

A$(10)="Hollingsworth"
A$(25)="Hobgood"
A$(57)="Ho"

Since A$(10) or "Hollingsworth" precedes A$(57) or "Ho" in the array, the match returned for "Ho" will be "Hollingsworth."

The way that the string array is built could be modified to suit a variety of applications. Strings could be input as in the above example, stored in data statements, or stored on disk and loaded in to be used again and again.

Rumors, Rumors . . .

December is the time for giving, and that is what I'm going to attempt to do, (give you a little dose of the truth to clear up some misunderstandings, that is . . .). It's been awhile since I've climbed up on my soapbox, so here goes . . .

I've been reading some of the trade publications and monthly newsletters that have sprung up to "support" the TRS-80 Color Computer and by and large, I think they are great. I'll be the first to admit that Radio Shack doesn't offer everything for everybody. Nobody could do that. These publications cater to specific types of users and are certainly accepted (or they wouldn't be in business very long!).

There have been many articles written about the Color Computer and the language it uses. Most of these articles have been fair and objective, coming to a favorable conclusion concerning the Color Computer. For this I am thankful. However, (here comes the speech, guys) while a large portion of the information is accurate and knowledgeable, some of the so-called "insiders information" is so far off that it's amusing to me to read it. The month by month tales of what we at Radio Shack are doing with our products, why and when we're doing it, etc. seems to be very informative (but most of it comes as a surprise to me!).

Let me relate some "for examples":

The statement of (supposed) fact that our disk drives for the Color Computer are manufactured by Shugart one month and the next month, again (supposed) fact, by TEAC. Neither of these are correct. (Sorry, we don't normally disclose information on who does what for Radio Shack.)

Or the claim of how we are expanding to 32K RAM. At first, some assumed it would be by the method the after-market suppliers suggested; that is, by piggy-backing 2 sets of 16K RAM chips. Of course, if we were strictly an after-market supplier, this is probably the way we would have done it. But we are more than an after-market supplier! So, what we did do was to replace the 16K chips with 32K chips. It makes for a very neat installation. (More on 32K later.)

Then there is the "expert" definition, based on "reliable" sources of the "1.1" version ROM for the Color Computer, why it was done and what effects it had on the machine and software.

Read on and hopefully the reason for the 1.1 ROM will become clear along with the explanation of the 32K upgrade.

As you can see, some of the statements made by various sources have people confused about the Color Computer and the various things happening to it and with it. The major problem is that conjecture is intermixed with the accurate information and who's to know what is truth and what is fiction?

We can't feasibly publicize everything that we do here at the Shack in regards to changing our products, any more than record companies can tell you when they change the color of the label on their records or the weight of the cardboard used in the jackets that the record goes into. It just isn't necessary (and tends to cloud the issue), as long as the record still works with all the turntables "in the field," and you can still read the label on the record. What we do tell you about, (as well we should) are those changes which affect the operation of your machine and/or software. Then we will tell you if you need to bring in your machine for modification (heaven forbid); or we will tell you about any patches available for existing software.

What that whole last paragraph means is: We try to supply all of the relevant information we can about your machine, what we designed it to do, how to do it, (or how not to do it) as well as offering a Computer Customer Service group to help disseminate that information.

What we won't tell you is who our sources of supply are, how much we buy or sell, and why we did something the way we did, or how we did it. (Helps to maintain that competitive edge, you know.)

There are engineer-types who own Color Computers, who have opened the machine up, disassembled it and can tell you all the resistors, capacitors, assorted PIA chips, microprocessor chips, etc. that are used to manufacture our machine. What they probably cannot tell you is what is inside of those chips, what standards or specifications those chips have to meet to be used in our machines. They also can't tell you if the parts are standard "off-the-shelf" products or custom manufactured. Based on this, I'm going to attempt to "fill you in" on the 32K machine and upgrade.

First of all, as was reported in the October Newsletter article on the Color Computer Disk Drives, 32K RAM is NOT required to use the disk drives. Pure and simply, the 32K option offers you more memory to play with.

The next little tidbit (remember it's Christmas time) I'll offer is about the ROM in the 32K machine. There is nothing wrong with the 1.0 version in the Color Computer and therefore, there is no reason not to keep it if that is what you have. The disk drives work fine with either version (1.0 or 1.1) of the ROM in your Extended BASIC Color Computer.

What you might notice in some of the later machines (without Extended Color BASIC), is a sign-on stating the presence of a 1.1 version of the ROM. There is ONE big reason for offering this version instead of the 1.0 version. As originally designed (1.0), the Color Computer would only expand to 16K RAM. Therefore, the ROM addressed a maximum of only 16K RAM. We had to change the ROM (1.1) so it would recognize the full 32K of memory. Any machine sold with the 1.1 ROM or any machine upgraded to 32K RAM should include an information sheet telling you the differences between the ROMs to watch out for.

The 1.1 ROM slightly affects the following of our software packages (Program Paks™) currently in production as of 9/81:

Chess (26-3050):	if you are using the joystick
Checkers (26-3055):	option and you wish to pick up or put down the playing piece, you must use the space bar instead of the "fire-when-ready" button.
Bustout (26-3056):	to get to the next ball, use the spacebar instead of the "fire-when-ready" button.

If you have a Line Printer VII or a Line Printer VIII and you want to do dot-addressable graphics, you no longer need to load the eight bit driver program.

With the 1.1 ROM, the screen will display an S (for search) or F (found) when it encounters an "ungapped" file when reading from the cassette recorder. You can also have a full 255 character data file without worrying about losing any information (I never have worried about that).

So much for the ROM.

When you have Radio Shack install the 32K RAM kit (26-3017) in your Color Computer, the technician will check to make sure that you have the appropriate ROM in your machine and will supply you with a fully operational 32K machine. If you are upgrading to 32K and your Color Computer has a 1.0 ROM, the technician will swap your old 1.0 ROM for a new 1.1 ROM!

This means, that if you can't survive without a 1.1 ROM in your Color Computer, the only way to get one is to have a 32K kit installed in your machine. (No, the ROM is NOT included in the 32K kit. No, the ROM is not available as an upgrade by itself. If Radio Shack does the installation, all necessary retrofitting is done for the charge of the kit and installation alone.)

The end result is this: if you bought the first Color Computer off the production line, it will cost you no more to upgrade your machine to 32K (if Radio Shack does the installation) than if you were to buy one today and have it upgraded to 32K tomorrow.

One final note: Since Radio Shack does not support the 32K piggy-back expansion for the Color Computer, I cannot guarantee that this particular non-Radio Shack modification will still be compatible with the 1.1 version of the ROM. (For that matter, I can't guarantee that it works with the 1.0 version either!)

I hope this clears up some of the confusion that is being created by the "experts" out there. We at Radio Shack are doing our best to provide you with high quality, current technology merchandise. Whether your Christmas includes a 4K BASIC Color Computer or a 32K upgrade to your Extended BASIC machine, you can rest assured that many hours of design, research, and testing have gone into making that computer the best product available in its class.

Merry Christmas, and may you never run out of RAM!

A Christmas Wreath and Song

by Gerald Plueard
501 North 143 Street
Seattle, WA 98133

```
5 ' THIS PROGRAM WAS MADE ON THE (TRS-80) COLOR
    16K COMPUTER AND PRINTED ON THE (TRS-80)
    LINE PRINTER VII.  CSAVED ON TAPE USING THE
    (TRS-80) COMPUTER CASSETTE RECORDER.
10 'THIS PROGRAM MAY BE PUBLISHED, I HAVE NO COPY
    RIGHTS TO IT NOR IS ANY OF IT COPY RIGHTED
    TO MY KNOWLEDGE.
20 'FINISHED ON AUGUST 30,1981 AT 501 NORTH 143
    ST. SEATTLE WA.98133 BY JERRY PLUEARD
30 'LINES 110 THROUGH 320 IS A WAY OF DISPLAYING
    A MOVING CURSOR SO YOU KNOW HOW MUCH TIME
    YOU HAVE TO READ THE SONG WORDS.
40 'LINES 350 THROUGH 400 MOVES THE SONG WORDS
    ACROSS THE CENTER OF YOUR SCREEN
50 'LINES 590 THROUGH 900 DEVELOPS THE CHRISTMAS
    WREATH IN A WINDOW.
55 CLS
60 PRINT@64, STRING$(32,"*")
70 PRINT@352, STRING$(32,"*")
80 PRINT@199, "JOY TO THE WORLD"
90 FOR X=1 TO 1000
    : NEXT X
    : CLS
110 PRINT@256, "JOY TO THE WORLD"
120 GOSUB 280
130 PRINT@256, "THE LORD IS COME"
140 GOSUB 280
150 PRINT@256, "LET EARTH EARTH RECEIVE HER KING"
160 GOSUB 280
170 PRINT@256, "LET EVERY HEART"
180 GOSUB 280
190 PRINT@256, "PREPARE HIM ROOM"
200 GOSUB 280
210 PRINT@256, "AND HEAVEN AND NATURE SING"
220 GOSUB 280
230 PRINT@256, "AND HEAVEN AND NATURE SING"
240 GOSUB 280
250 PRINT@224, "AND HEAVEN AND HEAVEN AND NATURE
    SING"
260 GOSUB 280
270 GOTO 340
280 FOR C1=0 TO 31
290 PRINT@288+C1, CHR$(128);
300 FOR D1=1 TO 15
    : NEXT D1
310 PRINT@288, " "
320 NEXT C1
330 RETURN
340 CLS
350 Q$="           JOY TO THE WORLD THE LORD IS

    COME LET EARTH RECEIVE HER KING LET EVERY
    HEART PREPARE HIM ROOM AND HEAVEN AND NATURE
    SING AND HEAVEN AND NATURE SING AND HEAVEN
    AND HEAVEN AND NATURE SING "
360 FOR Z=1 TO LEN(Q$)
370 PRINT@260, MID$(Q$,Z,22)
380 PLAY"L64; O5; 1"
390 FOR D2=1 TO 50
    : NEXT D2
400 NEXT Z
410 GOTO 570
420 A$="T4; O3; L2; C; L4; O2; B; L8; A; L2.; G;
    L4; F; L2; E; D;"
430 G$=A$
    : RETURN
440 B$="L2.; C; P32; L4; G; L2; A; L4; P32; A;
    L2.; B; P32; L4; B; O3; L1.; C"
450 G$=B$
    :RETURN
460 C$="L4; C; C; O2; L4; B; A; G; L4.; G; L8; F;
    L4; E; O3; C"
470 G$=C$
    : RETURN
480 D$="O3; L4; C; O2; B; A; G; P32; L4; G; L8;
    F; L4; E; P32; E; P32; E; P32; E; P32; E;
    P23; L8; E; F"
490 G$=D$
    : RETURN
500 E$="L2.; G; L8; F; E; L4; D; P32; D; P32; D;
    P32; L8; D; E; L2.; F; L8; E; D"
510 G$=E$
    : RETURN
520 F$="O2; L4; C; O3; L2; C; O2; L4; A; L4.; G;
    L8; F; L4; E; F; L2; E; D; L1; C"
530 G$=F$
    : RETURN
540 X$="XA$; XB$; XC$; XD$; XE$; XF$;"
550 G$=X$
    : RETURN
560 PLAY X$
570 PMODE 3, 1
580 PCLS
590 SCREEN 1, 0
600 LINE(0, 0)-(255, 191), PSET, B
610 LINE(128, 0)-(128, 18), PSET
620 FOR I=0 TO 6.5 STEP .02
630 LA=60*SIN(I)
640 LB=60*COS(I)
650 I2=I2+.3
660 SA=20*SIN(I2)
670 SB=20*COS(I2)
680 LINE-(128-LA-SA, 96-LB-SB), PSET
690 NEXT I
700 PAINT(10, 40), 4, 0
710 CIRCLE(127, 95), 44, 3
```

```
72Ø GOSUB 78Ø
73Ø PAINT(128, 6Ø), 2, Ø
74Ø LINE(12Ø, 95)-(136, 132), PSET, BF
75Ø PAINT(128, 8Ø), 1, 2
76Ø GOSUB 84Ø
77Ø GOTO 82Ø
78Ø LINE(12Ø, 89)-(128, 7Ø), PSET
79Ø LINE-(136, 89), PSET
8ØØ LINE-(12Ø, 89), PSET
81Ø RETURN
82Ø GOSUB 84Ø
83Ø GOTO 72Ø
84Ø K=K+1
85Ø IF K>2Ø THEN 86Ø ELSE 9ØØ
86Ø J=J+1
87Ø ON J GOSUB 42Ø, 44Ø, 46Ø, 48Ø, 5ØØ, 52Ø, 54Ø
88Ø IF J=>7 THEN J=Ø
89Ø PLAY G$
9ØØ RETURN
91Ø END
```

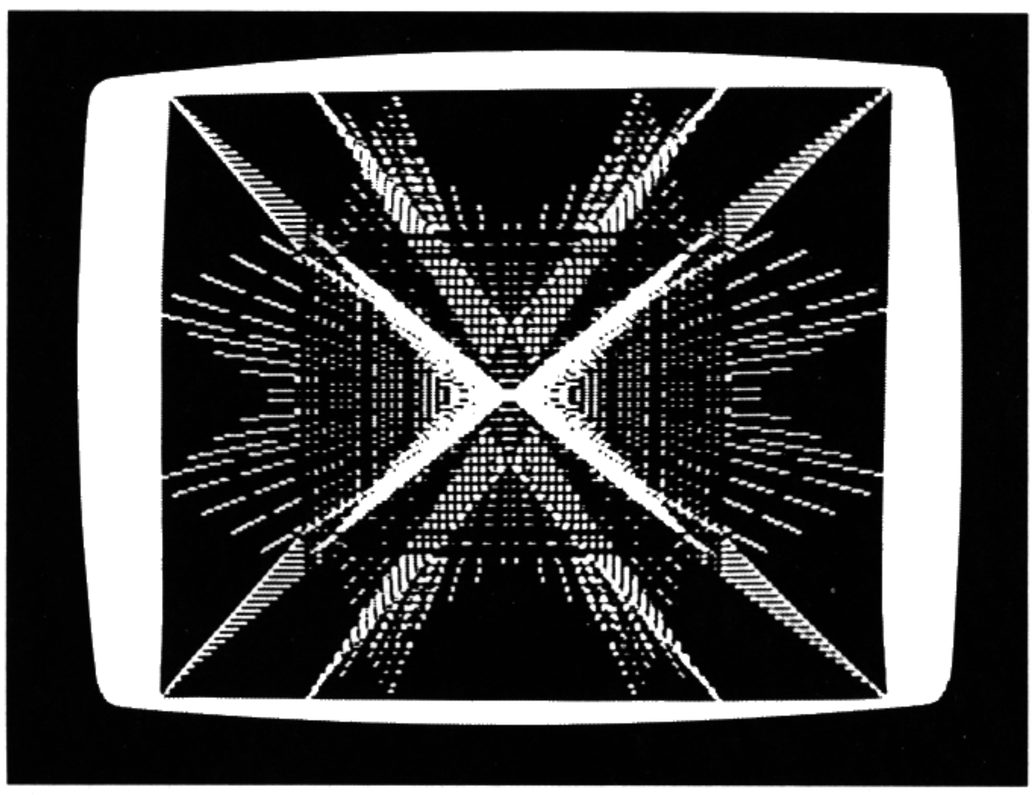
Mindtrip and Designs

For the Color Computer

by C. Ross Chamberlain
490 West 187th Street, #4K
New York, NY 10033

I have just put this program together to use my Line Printer VII as a sort of typewriter, and there may still be bugs, so please forgive any errors in the appearance of this letter.

Your call for special programs for the special December issue prompted me to send you the enclosed two programs for what I think are beautiful and ever-changing color patterns. One, the "MINDTRIP" program, is actually a set of specific cases of the other one, more simply called "DESIGNS."



In one sense, they are derivative of the Sample Program #5, "In-Out," in "Going Ahead With Extended Color Basic," but they represent a sort of culmination of a lot of experimenting I have been doing with PMODE 4,1's capacity for color. I do not know how basic this kind of experimentation is, of course. Your people may have discarded similar programs long ago — as, no doubt, have many of your customers — but as a writer and sometime artist in my 40's I come upon this with the often earnest wish that either I had been born much more recently, or the "Computer Revolution" had gotten underway years ago.

Anyway, I have spent hours listening to music and letting these designs do their thing on my TV screen, and I think others who have not already discovered similar ideas of their own may find them enjoyable.

Incidentally, I have tried to set up these programs to be as reproducible as possible, as simple to read and work with, but of course you may find better ways of doing that — I assume that your ability to get a printout with each command on a separate line is not available on the printer with the Color Computer, at least not without some software I do not have or have not learned about.

```
2 '  D E S I G N S
3 '          B Y
4 ' C. ROSS CHAMBERLAIN
5 '
6 '
1Ø CLS
2Ø PRINT
     : PRINT "              DESIGNS"
3Ø PRINT "    BY C. ROSS CHAMBERLAIN"
4Ø PRINT "      FOR USE BY FRIENDS"
5Ø PRINT
     : PRINT "           AUGUST 1981"
6Ø PRINT
     : PRINT "THERE ARE THREE VARIATIONS:  THEY
     ARE NUMBERED AS FOLLOWS: "
     : PRINT "1. LIGHT IN THE DARKNESS        2.
     DARKNESS IN THE LIGHT        3. ALTERNATING"
     : PRINT "THIS CAN BE CHANGED AT ANY TIME BY
     PRESSING THE APPROPRIATE KEY"
7Ø PRINT "...HOW DO YOU WISH TO START";
8Ø K$=INKEY$
     : IF VAL(K$)>3 OR VAL(K$)<1 THEN 8Ø
9Ø V=VAL(K$)
     : V1=V
     : IF V=3 THEN V=1
1ØØ PRINT
11Ø IF V1=3 THEN 13Ø
12Ø PRINT "VERY WELL--NOW, ASSUMING THAT,
     EVENTURALLY, YOU WILL BE WANTING TO USE THE
     ALTERNATING FEATURE, ";
13Ø PRINT "HOW OFTEN SHOULD IT CHANGE (ANY
     NUMBER) FROM ONE ALTERNATE TO THE OTHER
     (ENTER [R] IF RANDOM ALTERNATION DESIRED)";
14Ø INPUT V$
     : IF V$="R" THEN 16Ø
15Ø IF VAL(V$)<1 THEN PRINT "GOTTA ENTER
     SOMETHING, EVEN IF YOU NEVER PLAN TO USE
     THIS FEATURE. SO, WILL IT BE AN [R] OR A
     NUMBER OVER ZERO";
     : GOTO 14Ø
16Ø PRINT "   (THIS MAY BE CHANGED, LATER, BY
     PRESSING THE SPACEBAR)"
17Ø PRINT
     : INPUT "  FIRST PARAMETER (1 TO 128)"; P
18Ø IF P>128 THEN PRINT "...MUST BE 128 OR
     LESS..."
     : GOTO 17Ø
19Ø P=P+127
2ØØ INPUT "  SECOND PARAMETER (1 TO 96)"; R
21Ø IF R>96 THEN PRINT "...MUST BE 96 OR LESS..."
     : GOTO 2ØØ
22Ø R=R+95
23Ø INPUT "    STEP RANGE (2 TO 25)"; S
24Ø IF S>25 THEN PRINT "...MUST BE 25 OR LESS..."
     : GOTO 23Ø
25Ø IF S<2 THEN PRINT "...MUST BE 2 OR MORE..."
     : GOTO 23Ø
26Ø PMODE 4, 1
27Ø PCLS
28Ø SCREEN 1, 1
29Ø IF X<=Ø THEN X=Ø
     : A=SA
3ØØ IF Y<=Ø THEN Y=Ø
     : B=SA
```

```
31Ø IF X=>P THEN X=P
     : A=-SA
32Ø IF Y=>R THEN Y=R
     : B=-SA
33Ø IF SA<=1 THEN SA=1
     : SB=1
34Ø IF SA=>S THEN SA=S
     : SB=-1
35Ø D=255-X
     : E=191-Y
     : H=H+1
36Ø ON V GOSUB 4ØØ, 43Ø
37Ø X=X+A
     : Y=Y+B
     : SA=SA+SB
38Ø K$=INKEY$
     : IF K$<>"" THEN 46Ø ELSE 29Ø
39Ø GOTO 29Ø
4ØØ LINE(X, Y)-(D, E), PRESET, B
     : LINE-(X, Y), PSET
     : LINE(D, Y)-(X, E), PSET
41Ø IF V1=3 AND H=>V2 THEN GOSUB 54Ø
42Ø RETURN
43Ø LINE(X, Y)-(D, E), PSET, B
     : LINE-(X, Y), PRESET
     : LINE(D, Y)-(X, E), PRESET
44Ø IF V1=3 AND H=>V2 THEN GOSUB 54Ø
45Ø RETURN
46Ø IF K$=CHR$(32) THEN 49Ø
47Ø IF VAL(K$)<1 OR VAL(K$)>3 THEN 29Ø ELSE
     V1=VAL(K$)
     : V=V1
     : IF V=3 THEN V=1
48Ø GOTO 29Ø
49Ø PRINT
     : PRINT "YOU WISH TO ALTER THE FREQUENCY OF
     THE ALTERATIONS  VERY WELL--HOW OFTEN (ANY
     NUMBER, OR [R] DO YO WANT THEM TO OCCUR";
5ØØ INPUT V$
51Ø IF V$="R" THEN 28Ø
52Ø IF VAL(V$)<Ø THEN 5ØØ
53Ø V2=VAL(V$)
     : GOTO 28Ø
54Ø IF V=1 THEN V=2 ELSE V=1
55Ø IF V$<>"R" THEN 57Ø
56Ø V2=RND(5Ø)
57Ø H=Ø
     : RETURN
```

MINDTRIPS

```
1Ø CLS
2Ø PRINT
     : PRINT "          " CHR$(128);
     "MINDTRIPS"CHR$(128)
3Ø PRINT "     BY C. ROSS CHAMBERLAIN"
4Ø PRINT "       FOR USE BY FRIENDS"
5Ø PRINT
     : PRINT "           AUGUST 1981"
6Ø P=253
     : R=19Ø
     : V=1
     : X=Ø
     : Y=Ø
     : SA=Ø
7Ø PRINT
     : PRINT "              PROGRAM: "
71 PRINT " 1. ROBOT DREAMS OF COURTLY LOVE"
72 PRINT "    2.BUTTERFLIES OF PARADISE"
73 PRINT "      3. FANTASY IN TIFFANY'S"
8Ø PRINT
     : PRINT "CHANGE NUMBER AT ANY TIME JUST"
9Ø PRINT "BY PRESSING KEY OR PRESS SPACE"
1ØØ PRINT "BAR TO START OVER WITH NEW MENU
11Ø K$=INKEY$
     : IF K$="" THEN 11Ø
12Ø IF VAL(K$)<1 OR VAL(K$)>3 THEN 11Ø
13Ø S=VAL(K$)+1
14Ø PMODE 4, 1
15Ø PCLS
16Ø SCREEN 1, 1
17Ø IF X<=Ø THEN X=Ø
     : A=SA
18Ø IF Y<=Ø THEN Y=Ø
     : B=SA
19Ø IF X=>P THEN X=P
     : A=-SA
2ØØ IF Y=>R THEN Y=R
     : B=-SA
21Ø IF SA<=1 THEN SA=1
     : SB=1
22Ø IF SA=>S THEN SA=S
     : SB=-1
23Ø D=255-X
24Ø E=191-Y
25Ø ON V GOSUB 3ØØ, 31Ø
26Ø X=X+A
     : Y=Y+B
27Ø SA=SA+SB
28Ø K$=INKEY$
     : IF K$<>"" THEN 32Ø
29Ø GOTO 17Ø
3ØØ LINE(X, Y)-(D, E), PSET, B
     : LINE(X, Y)-(D, E), PRESET
     : LINE(X, E)-(D, Y), PRESET
     : V=2
     : RETURN
31Ø LINE(X, Y)-(D, E), PRESET, B
     : LINE(X, Y)-(D, E), PSET
     : LINE(D, Y)-(X, E), PSET
     : V=1
     : RETURN
32Ø IF K$=CHR$(32)THEN 1Ø
33Ø IF VAL(K$)<1 OR VAL(K$)>3 THEN 17Ø
34Ø S=VAL(K$)+1
     : X=Ø
     : Y=Ø
     : SA=Ø
     : GOTO 17Ø
```

Christmas Wrapping Paper

For the Color Computer

by Ron Beatty
Grand Rapids, MI

I have had a TRS-80 Color Computer since January of this year. My family and I have enjoyed our addition very much.

In the August 1981 TRS-80 Newsletter I read that you are having a special issue for December. I have a program that I hope you feel would be worth publishing in order to share it with your other readers. The program listing is enclosed.

The program is called "Christmas Paper." It started out as an experiment in using BASIC's poke graphics in the highest resolution (64 x 192) that still allows the use of all of the colors. When I saw the first few displays, they reminded me of Christmas paper. They also reminded me about the December issue of the TRS-80 Newsletter.

At that time, the program had nothing that I felt was really worth sharing so I added some other features. The program is now my first to use a machine language (ML) subroutine and ML graphics. Here is a list of the features.

1. The program used the highest resolution (64 x 192) that still allows the use of all colors.

2. It demonstrates how to setup and call ML routines and pass data from BASIC to these routines.
3. The ML routines are used for the graphics display and because the poke graphics are still in the program, it shows the difference in speed between BASIC and the ML.
4. The display is in random patterns and colors using the graphics characters represented by the decimal numbers 128 through 255.

The program, of course, requires the Color Computer and 16K.

When executed, a brief display of the title will appear followed by a request to enter either (M) for ML routine or (B) for the BASIC routine. If BASIC is entered, the next request will be to enter the step number. If ML is to be used, the next request will be to enter the speed for the ML routine and then the step number.

The speed is for the ML graphic display routine only and will allow you to see more of a painting effect at the slower speeds. The valid entries are 1 through 255. The fastest speed is 1, the slowest is 255. The step number is used to determine the number of screen locations to skip before the next display. The number entered is the maximum that will be skipped and is used to generate a random step number for each new screen display. If the number 25 is entered, the instruction used will be S = RND (SS) where SS is 25 and S is the new random step number. The lower step numbers (2-20) seem to create the best displays, but try them all.

Other options in the program, an explanation of the BASIC code and an explanation of the ML routines, are listed below.

If the R key is pressed during the BASIC display routine the program will return to the beginning. If the R key is pressed during the ML routine, it should be held down until the current display is completed. It will then return to the beginning.

The keys 0 through 8 have a special use. Each key represents its corresponding color code in Color BASIC.

0 = black, 1 = green, 2 = yellow, etc.

When pressed, the screen will be cleared to the requested color. This will change the background color of the "Christmas Paper," and adds a nice variety. This will work in the BASIC routine and the ML routine. Notice the difference in speed.

The E key will end the program with a final message.

If you find a paper you like, you can use the combination shift @ to hold the pattern.

All the key functions work like the R key. During the BASIC display routine the function takes effect immediately. During the ML routine, you must hold the keys down until the current display is completed.

BASIC and ML code (listings are included):

LINES	FUNCTIONS
10-55	Title and reserve memory for ML routine.
60-98	Enter language, speed and step number.
110-220	First ML routine is poked into memory locations 11000 -10012 and is called. This routine will clear video ram, 6144 bytes starting at location 10240. See routine number I on page 5.
230-330	Set the bit configuration for the three registers that control the graphics mode, the video display, the display control and the page select registers. This provides the resolution of 64 x 192 with all colors available.
340-480	This is the second ML routine. It will perform the graphics display if the ML was selected. The routine is poked into locations 10000-10032. See routine number II on page 5.
500-510	Generates the random graphics code and passes it to the ML routine in address 10150
525-540	Gets the step number form address 10155 and generates the random step number for the next screen display. The number is passed to the ML routine in address 10152.
550-560	Call the ML routine for graphics.
570-595	Check for the use of keys and loop to the next screen display for ML routine.
600-660	BASIC display routine using poke graphics (line 630), 645-649, check for special key use.
700-760	If keys 0 through 8 are pressed, this routine supplies the graphics character code to the display routines.
800-850	Display Final message and end the program.

The following addresses are used to pass data from BASIC to ML subroutines.

Decimal Address	Data
10150	Graphics character code
10152	Random step number
10154	0 = Basic, 1 = ML not used
10155	Maximum step number
10156	ML routine speed

Machine Language Routines

I. Clear Video Ram

	6809 Assembler	Hex	Decimal	Function
	LDA # 128	8680	134, 128	Get Graphic character black
	LDX # 10240	8E2800	142, 40, 0	Get screen start addr.
Loop 1	STA, X+	A780	167, 128	Clear screen addr. and increment addr.
	CMPX #16384	8C5400	140, 64, 0	End of screen?
	BLO Loop 1	25F9	37, 249	No, do it again
	RTS	39	57	Yes, return to Basic

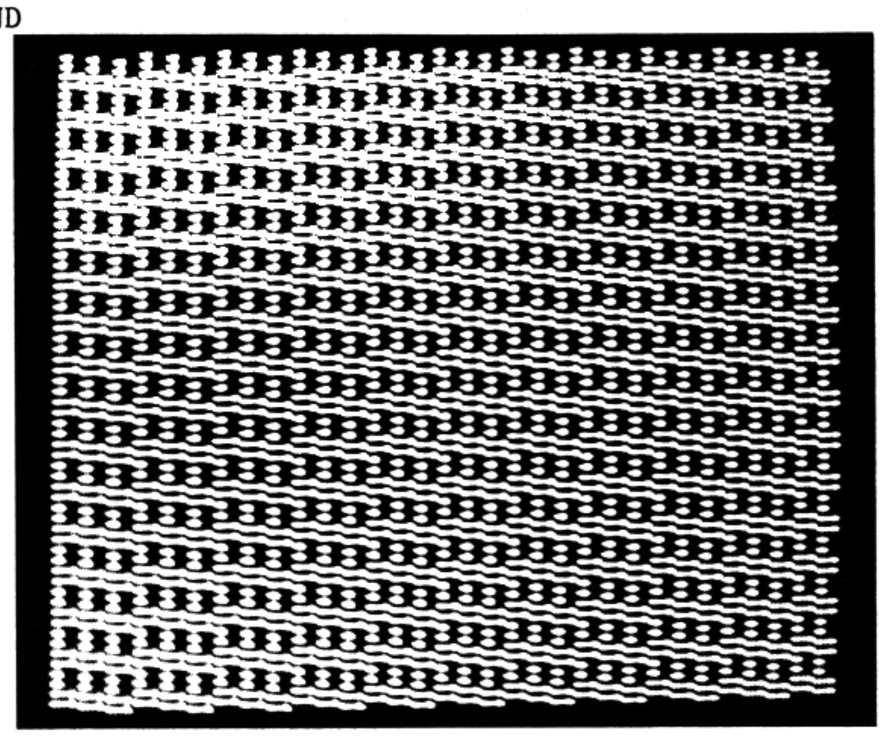
II. Display

	6809 Assembler	Hex	Decimal	Function
	LDX #10240	8E2800	142, 40, 0	Get screen start addr.
Loop 1	LDB $27A6	F627A6	246, 39, 166	Get graphics character
	STB, X+	E780	231, 128	Display and increment screen addr.
	LDB $27A8	F627A8	246, 39, 168	Get random step number
Loop 2	DECB	5A	90	Decrement step number
	CMPB # 0	C100	193, 0	Is step done
	BEQ CONT	2704	39, 4	Yes, continue program
	INX	3001	48, 1	No, increment screen addr.
	BRA Loop 2	20F7	32, 247	and do again
Cont	LDB $27AC	F627AC	246, 39, 172	Get speed
Loop 3	DECB	5A	90	Decrement speed
	CMPB #0	C100	193, 0	Is speed loop done
	BHI Loop 3	22FB	34, 251	No, do it again
	CMPX #16384	8C4000	140, 64, 0	Yes, is screen done
	BLO Loop 1	25E2	37, 226	No, do it again
	RTS	39	57	Yes, return to Basic

Here's the BASIC program.

```
10 CLS4
15 PRINT"MERRY CHRISTMAS 1981"
20 PRINT@192,"       CHRISTMAS PAPER"
30 PRINT@256,"       BY"
40 PRINT@320,"       RON BEATTY"
50 FOR L=1 TO 2000
```

```
     : NEXT
55 CLEAR 2ØØ,9999
6Ø CLS 3
63 PRINT"PLEASE ENTER <M> FOR MACHINE"
65 PRINT"LANGUAGE OR <B> FOR BASIC."
66 INPUT A$
     : IF A$="M" OR A$="B" THEN 67 ELSE 6Ø
67 IF A$="M"THEN POKE 1Ø154,1 ELSE POKE 1Ø154,Ø
68 IF A$="B" THEN 95
7Ø PRINT"PLEASE ENTER SPEED OF DISPLAY"
8Ø PRINT"1 THROUGH 255"
     : INPUT L
9Ø IF L<1 OR L>255 GOTO 7Ø
91 POKE 1Ø156,L
95 PRINT"PLEASE ENTER STEP#"
96 PRINT"2 THROUGH 255"
     : INPUT SS
97 IF SS<2 OR SS>255 THEN 95
98 POKE 1Ø155,SS
11Ø FOR A=ØTO 12
12Ø READ B
13Ø POKE 1ØØØØ+A,B
14Ø NEXT A
15Ø DATA 134,128
16Ø DATA 142,4Ø,Ø
17Ø DATA 167,128
18Ø DATA 14Ø,64,Ø
19Ø DATA 37,249
2ØØ DATA 57
21Ø DEFUSRØ=1ØØØØ
22Ø C=USR(Ø)
23Ø POKE 65478,Ø
24Ø POKE 6548Ø,Ø
25Ø POKE 65483,Ø
26Ø POKE 65484,Ø
27Ø POKE 65487,Ø
28Ø POKE 65488,Ø
29Ø POKE 6549Ø,Ø
3ØØ POKE 65472,Ø
31Ø POKE 65475,Ø
32Ø POKE 65477,Ø
33Ø POKE 65314,Ø
34Ø FOR A=Ø TO 33
35Ø READ B
36Ø POKE 1ØØØØ+A,B
37Ø NEXT A
38Ø DATA 142,4Ø,Ø
39Ø DATA 246,39,166
4ØØ DATA 231,128
41Ø DATA 246,39,168
415 DATA 9Ø
416 DATA 193,Ø
417 DATA 39,4
42Ø DATA 48,1
425 DATA 32,247
427 DATA 246,39,172
43Ø DATA 9Ø
44Ø DATA 193,Ø
45Ø DATA 34,251
46Ø DATA 14Ø,64,Ø
47Ø DATA 37,226
48Ø DATA 57
5ØØ TA=RND(127)
     : TG=128+TA
51Ø POKE 1Ø15Ø,TG
525 SS=PEEK(1Ø155)
53Ø S=RND(SS)
     : IF S<2 THEN 53Ø
54Ø POKE 1Ø152,S
545 IF A$="B" THEN 6ØØ
55Ø DEFUSRØ=1ØØØØ
56Ø C=USR(Ø)
57Ø B$=INKEY$
575 IF B$="R" THEN RESTORE
     : GOTO 6Ø
585 IF B$=>"Ø" AND B$<"9" THEN GOSUB 7ØØ
59Ø IF B$=>"Ø" AND B$<"9" THEN GOTO 545
592 IF B$="E" THEN 8ØØ
595 GOTO 5ØØ
6ØØ 'BASIC DISPLAY ROUTINE
61Ø TS=Ø
62Ø FOR LP=ØTO 6143 STEP S
63Ø POKE 1Ø24Ø+TS,TG
64Ø TS=TS+S
645 B$=INKEY$
646 IF B$="R" THEN RESTORE
     : GOTO 6Ø
647 IF B$=>"Ø" AND B$<"9" THEN GOSUB 7ØØ
648 IF B$=>"Ø" AND B$<"9" THEN GOTO 545
649 IF B$="E" THEN 8ØØ
65Ø NEXT LP
66Ø GOTO 5ØØ
7ØØ 'SCREEN TO ONE COLOR
71Ø C=VAL(B$)
72Ø IF C>Ø THEN C=(C-1)*16+143
73Ø IF C=Ø THEN C=128
74Ø POKE 1Ø15Ø,C
     : TG=C
75Ø POKE 1Ø152,1
     : S=1
76Ø RETURN
8ØØ 'END PROGRAM
81Ø CLS4
811 PRINT@Ø," "
813 PRINT@128," "
82Ø PRINT@192,"     HAVE A MERRY CHRISTMAS"
83Ø PRINT@256,"           AND   A"
835 PRINT@32Ø,"        HAPPY NEW YEAR!"
845 PRINT@448," "
85Ø END
```



The Fleet's In!

by CDR Frank A. Robinton

After playing around with the color computer house program in the April 1981 issue of the Microcomputer News, CDR Robinton began writing his own programs through trial and error experimentation. He is the owner of a 16K Extended Color BASIC computer which was a gift to him from his son. A retired Navy commander, he converted his knowledge and obvious love for seagoing vessels to visual representations of a variety of boats.

YACHT

The first boat program CDR Robinton sent was his own former home or YACHT.

```
5 PMODE 3,1
1Ø PCLS
15 SCREEN 1,Ø
2Ø LINE (2Ø,12Ø)-(12Ø,12Ø), PSET
21 LINE (12Ø,12Ø)-(24Ø,12Ø), PSET
22 LINE (1Ø8,1Ø8)-(24Ø,1Ø8), PSET
23 LINE (1Ø8,1Ø8)-(1Ø8,2Ø), PSET
24 LINE (1Ø8,1Ø8)-(2Ø,1Ø8), PSET
25 LINE (1Ø8,1Ø8)-(1Ø,1Ø8),PSET
26 LINE (24Ø,12Ø)-(24Ø,1Ø8),PSET
```

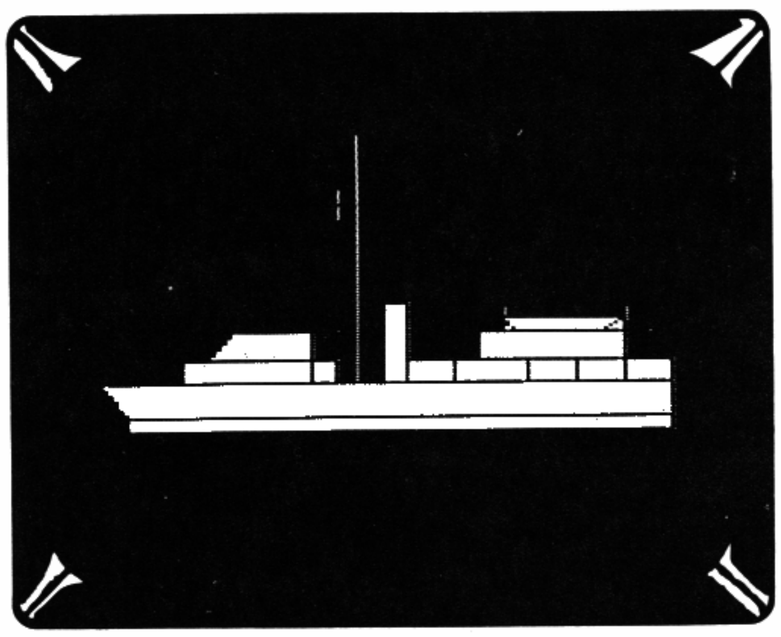
```
28 LINE (1Ø,1Ø8)-(2Ø,12Ø),PSET
3Ø LINE (1ØØ,4Ø)-(1ØØ,5Ø),PSET
31 LINE (4Ø,1ØØ)-(1ØØ,1ØØ),PSET
32 LINE (7Ø,9Ø)-(1ØØ,9Ø),PSET
33 LINE (9Ø,1ØØ)-(9Ø,1Ø8),PSET
34 LINE (1ØØ,1ØØ)-(1ØØ,1Ø8),PSET
35 LINE (9Ø,1ØØ)-(9Ø,9Ø),PSET
36 LINE (4Ø,1ØØ)-(4Ø,1Ø8),PSET
37 LINE (7Ø,9Ø)-(6Ø,9Ø),PSET
38 LINE (6Ø,9Ø)-(5Ø,1ØØ),PSET
39 LINE (15Ø,1ØØ)-(24Ø,1ØØ),PSET
4Ø LINE (15Ø,1ØØ)-(15Ø,1Ø8),PSET
41 LINE (24Ø,1ØØ)-(24Ø,1Ø8),PSET
42 LINE (2Ø,125)-(24Ø,125),PSET
43 LINE (2Ø,12Ø)-(2Ø,125),PSET
44 LINE (24Ø,12Ø)-(24Ø,125),PSET
45 LINE (12Ø,8Ø)-(13Ø,8Ø),PSET
46 LINE (12Ø,8Ø)-(12Ø,1Ø8),PSET
47 LINE (13Ø,8Ø)-(13Ø,1Ø8),PSET
48 LINE (1ØØ,45)-(1Ø8,45),PSET
49 LINE (2ØØ,1ØØ)-(2ØØ,1Ø8),PSET
5Ø LINE (18Ø,1ØØ)-(18Ø,1Ø8),PSET
51 LINE (22Ø,1ØØ)-(22Ø,1Ø8),PSET
52 PAINT (125,2Ø),3,Ø
6Ø GOTO 6Ø
```

After sending the above program CDR Robinton's "ragman" or sail boat enthusiast neighbor chided him for spending his time on a "stinkpot" yacht. In order to placate his neighbor, CDR Robinton designed a YAWL (pronounced like "you all" with a Texas accent).

YAWL

```
2 PCLEAR 4
5 PMODE 3,1
1Ø PCLS
15 SCREEN 1,Ø
2Ø DRAW "BM 2Ø,15Ø; R22Ø E1Ø L24Ø F1Ø"
25 DRAW "BM 8Ø,14Ø; U5 R25 D5 U5 R25 D5 U5 R25 D5
    U5 R25 D5"
3Ø DRAW "BM 14Ø,14Ø; L9Ø U9Ø G8Ø R45"
35 DRAW "BM 135,135; R45 U15Ø F14Ø"
4Ø DRAW "BM 135,135; R45 U13Ø G12Ø R12Ø"
45 DRAW "BM 14Ø,14Ø; L8Ø U3 L2"
5Ø DRAW "BM 14Ø,14Ø; L83 U5"
55 DRAW "BM 135,135; U3 L4Ø D3"
6Ø DRAW "BM 18Ø,13Ø; R65"
65 DRAW "BM 15,14Ø; U4 R1Ø D4 U4 R1Ø D4 U4 R1Ø
    D4"
7Ø DRAW "BM 24Ø,14Ø; U5 L2Ø D5 U5 L2Ø D5 E5 F5 E5
    F5 E5 F5 E5 F5 E5 F5"
8Ø PAINT (125,2Ø),8,Ø
85 PAINT (25Ø,4Ø),8,Ø
95 PAINT (2Ø,16Ø),3,Ø
1ØØ GOTO 1ØØ
```

The next letter from CDR Robinton included NOT one but four programs — YACHT TWO, YAWL TWO, DRAGGER, and SHIP. YACHT TWO and YAWL TWO are enhanced versions of the previously listed programs — YACHT and YAWL.



YACHT TWO enhancements include a sundeck and a dinghy on the top.

```
16 LINE (16Ø,9Ø) - (16Ø,1ØØ), PSET
17 LINE (23Ø,9Ø) - (15Ø,9Ø), PSET
18 LINE (22Ø,9Ø) - (22Ø,1ØØ), PSET
```

Lines 20-38 are unchanged.
Line 39 has a change in it.

Lines 40-51 are unchanged.
Lines 52-56 are new.

```
52 LINE (22Ø,85) - (17Ø,85), PSET
53 LINE (22Ø,85) - (21Ø,9Ø), PSET
54 LINE (17Ø,85) - (175,9Ø), PSET
55 LINE (17Ø,82) - (17Ø,9Ø), PSET
56 LINE (22Ø,82) - (22Ø,9Ø), PSET
```

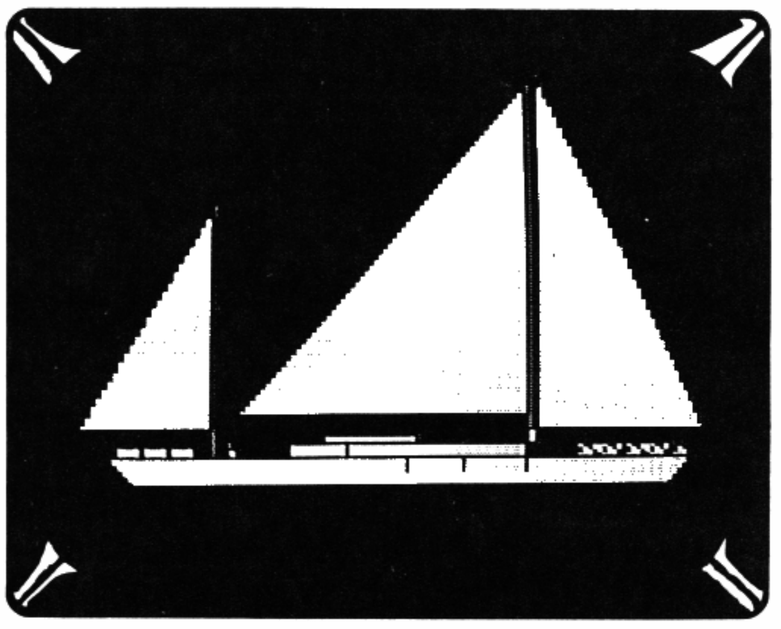
Line 58 is the same as 52 was previously.

```
58 PAINT (125,2Ø),3,Ø
```

Line 60 is unchanged.

YAWL TWO

One of the changes in YAWL TWO is the use of the color "sea" blue.



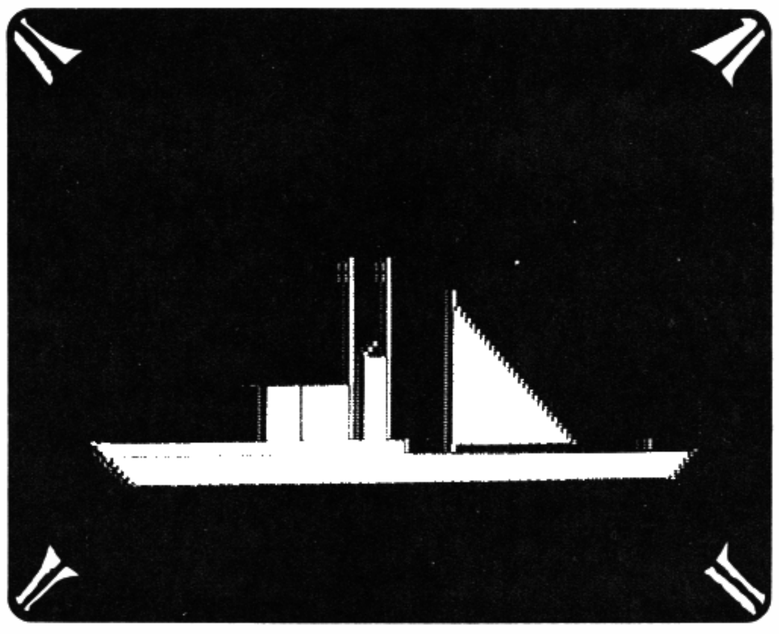
Lines 2-20 remain the same.
Lines 25-35 are new.

```
25 DRAW "BM 8Ø,14Ø; U5 R25 D5 UDR25 D5 U5 R25 D5
    U5 R25 D5"
3Ø DRAW "BM 135,135; R5Ø U15Ø F13Ø L5Ø"
32 DRAW "BM 135,135; L45 R9Ø D5"
35 DRAW "BM 14Ø,14Ø; L9Ø U9Ø G8Ø R48"
```

Lines 40-70 are the same.

```
8Ø PAINT (125,2Ø),3,Ø
85 PAINT (25Ø,4Ø),3,Ø
95 PAINT (2Ø,25Ø),3,Ø
12Ø GOTO 12Ø
```

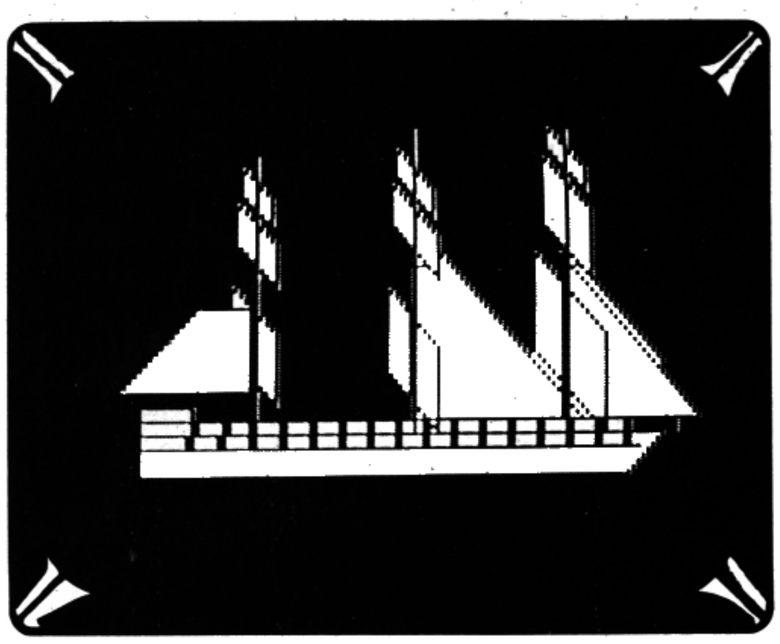
DRAGGER



DRAGGER AND SHIP

The next two programs — DRAGGER and SHIP — come in one version only (so far). A DRAGGER is a fishing boat which operates a dragnet.

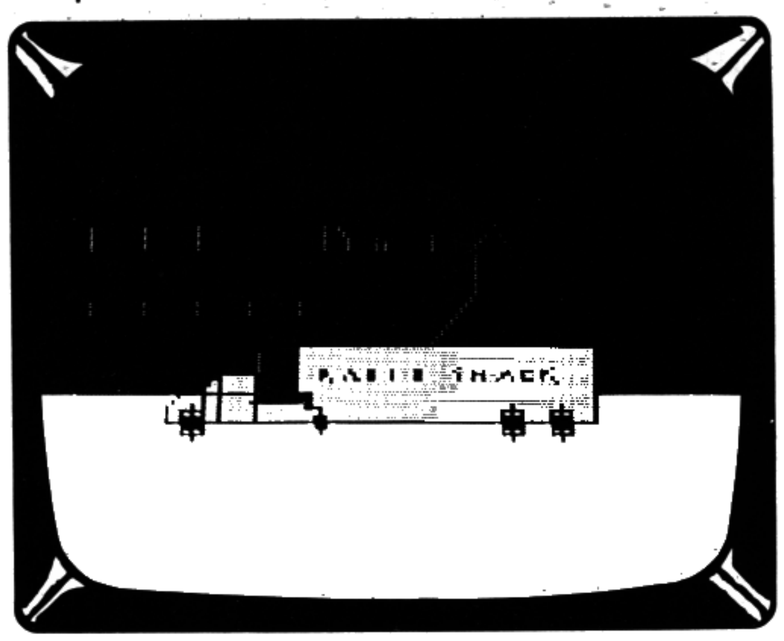
```
2 PCLEAR 4
5 PMODE 3,1
10 PCLS
15 SCREEN 1,0
20 DRAW "BM 20,150; R220 E10 L240 F10 H15"
25 DRAW "BM 8,135; R120 D5"
30 DRAW "BM 120,135; L10 U30 R10 D30 U30 L10 E5"
35 DRAW "BM 120,135; L15 U20 L40 R5 D20 R35 U65
     D2 L2 D2 R2 D2 L2 R2 D2 L2 U2"
40 DRAW "BM 120,135; L15 U20 L20 D10 U10 D20 U10
     L10 R10 U22 D2 L2 D4 R2"
45 DRAW "BM 230,140; U4 L2 D4"
50 DRAW "BM 149,140; U58 D5 F50 L50"
55 PAINT (125,20),3,0
100 GOTO 100
```



```
2 PCLEAR 4
5 PMODE 3,1
10 PCLS
15 SCREEN 1,0
20 DRAW "BM 20,150; R200 E15 L215 D15 U20 R45 U95
     D45 L10 R8 D55 R67 U110"
30 DRAW "BM 20,130; U5 R20 D5 R200 D5 U5 L45 U100
     D45 L5 R8 L5 D55 R45"
40 DRAW "BM 129,130; U55 R5 L15"
50 DRAW "BM 20,130;  R20 D5 R12 U5 R12 D5 R10 U5
     R2 D5 R10 U5 R2 D5 R10 U5 R2 D5 R10 U5 R2 D5
     R10 U5 R2 D5 R10 U5 R2 D5 R10 U5 R2 D5 R10
     U5 R2 D5 R10 U5 R2 D5 R10 U5 R2 D5 R10 U5 R2
     D5 R10 U5 R2 D5"
60 DRAW "BM 10,120; R55 L52 E30 R20 F10 H18 D8 R8
     F10 D25 H8"
70 DRAW "BM 20,140; R205 L5 U5 L2 D5 L10 U5 L2 D5
     L10 U5 L2 D5 L10 U5 L2 D5 L10 U5 L2 D5 L10
     U5 L2 D5 L10 U5 L2 D5 L10 U5 L2 D5 L10 U5 L2
     D5 L10 U5 L2 D5 L10 U5 L2 D5 L10 U5 L2 D5
     L10 U5 L2 D5 L10 U5 L2 D5 L10 U5 L2 D5 L10
     U5 L2 D5"
80 DRAW "BM 20,130; R20 D5 R2 U5 R13 D5"
90 DRAW "BM 20,130; R45 U55 F8 H16 U15 F15 D14"
100 DRAW "BM 20,130; R45 U75 F5 H11 U10 F11 D10"
120 DRAW "BM 20,130; R110 U35 F10 H22 D30 F22
     U30"
130 DRAW "BM 20,130; R110 U60 F10 H20 U15 F20
     D15"
140 DRAW "BM 20,130; R110 U80 F8 H16 U10 F16 D10"
150 DRAW "BM 20,130; R175 U10 F15 U35 H30 D35
     F15"
160 DRAW "BM 20,130; R175 U60 F10 H20 F20 U25 H20
     D25"
170 DRAW "BM 20,130; R175 U90 F8 H16 U8 F16 D8 H8
     U15"
180 DRAW "BM 20,130; R180 H60"
190 DRAW "BM 20,130; R230 H60"
200 DRAW "BM 20,130; R215 BU20 H30"
220 PAINT (125,20),3,0
250 GOTO 250
```

SCENE

SCENE uses DRAW, CIRCLE, PAINT, and three SCALES to produce a "novel scene."



```
10 PMODE 3,1
15 PCLS
20 SCREEN 1,0
25 FOR RADIUS=1 TO 10 STEP 5
30 CIRCLE (50,130),RADIUS
   : NEXT RADIUS
35 FOR RADIUS=1 TO 6 STEP 2
40 CIRCLE (50,130),RADIUS
   : NEXT RADIUS
45 FOR RADIUS=1 TO 6 STEP 5
50 CIRCLE (100,130), RADIUS
     : NEXT RADIUS
55 FOR RADIUS=1 TO 6 STEP 2
60 CIRCLE (100,130),RADIUS
     : NEXT RADIUS
65 DRAW "BM 50, 130; S2 L20 U10 E10 R30 L15 E15"
68 DRAW "BM 100,130; S2 U10 L10 H5 L40"
70 DRAW "BM 100,130; S2 L90 U20 R40 D20 L30 U35
     R30 D30 U45"
75 FOR RADIUS=1 TO 10 STEP 5
80 CIRCLE (200,130),RADIUS
     : NEXT RADIUS
85 FOR RADIUS=1 TO 6 STEP 2
90 CIRCLE (200,130),RADIUS
     : NEXT RADIUS
95 FOR RADIUS=1 TO 10 STEP 5
100 CIRCLE(180,130),RADIUS
     : NEXT RADIUS
105 FOR RADIUS=1 TO 6 STEP 2
110 CIRCLE (180,130),RADIUS
     : NEXT RADIUS
115 DRAW "BM90,120; S2 U35 R250 D55 L230"
120 DRAW "BM90,120; S2 R15 D10"
125 DRAW "BM100,115; S1 U15 R10 D8 L10 R8 F8"
130 DRAW "BM110,115; S1 E15 F15 H1 L5"
135 DRAW "BM120,115; S1 U15 R15 D15 L15"
140 DRAW "BM130,115; U15"
145 DRAW "BM140,115; U15 R15 D15 L15"
150 DRAW "BM158,115; E6 H8 E6"
155 DRAW "BM165,115; U15 D7 R15 U7 D15"
160 DRAW "BM175,115; E15 F15 H1 L5"
165 DRAW "BM185,115; U15 R15 BD15 L15"
170 DRAW "BM195,115; BU9 E10 G10 F12"
175 DRAW "BM192,115; U15
178 DRAW "BM0,120; R255"
180 DRAW "BM255,120; L155"
185 PAINT (0,140),2,0
190 PAINT (140,0),3,4
195 DRAW "BM 0,70; S4 R165 D15 G18"
200 FOR RADIUS=1 TO 4 STEP 2
```

```
2Ø5 CIRCLE (9Ø,9Ø),RADIUS
    : NEXT RADIUS
21Ø FOR RADIUS=1 TO 4 STEP 2
215 CIRCLE(7Ø,9Ø),RADIUS
    : NEXT RADIUS
22Ø FOR RADIUS=1 TO 4 STEP 2
225 CIRCLE (5Ø,9Ø), RADIUS
    : NEXT RADIUS
23Ø FOR RADIUS=1 TO 4 STEP 2
235 CIRCLE(3Ø,9Ø),RADIUS
    : NEXT RADIUS
24Ø FOR RADIUS=1 TO 4 STEP 2
245 CIRCLE(1Ø,9Ø),RADIUS
    : NEXT RADIUS
25Ø DRAW "BM 5Ø,7Ø; U1Ø L2Ø D1Ø U1Ø L2Ø D1Ø U1Ø
    L2Ø"
255 DRAW "BM 6Ø,7Ø; U7Ø D68 E5Ø G1Ø H3Ø"
26Ø DRAW "BM 1ØØ,7Ø; U1Ø R5 F5 D5"
265 DRAW "BM 12Ø,7Ø; U5 R2 D5 U4 R5 U1 D4 U4 R19
    U1 D5"
27Ø DRAW "BM 163,7Ø;  E1Ø D2 F8 L2 H3 D2 F8 L2 H8
    D2 F8  L2 H8"
3ØØ GOTO 3ØØ
```

For a night scene make the following changes:

```
1Ø PMODE 4,1
2Ø SCREEN 1,1
```

SCENE2

Later changes to the SCENE program included the addition of the "small car" program from the June 81 issue of the Microcomputer News. This program is called SCENE2.

```
1 DRAW "S4"
5 CLS
1Ø PCLEAR 4
15 PMODE 3,1
17 PCLS
    : GOSUB 1ØØØ
2Ø SCREEN 1,Ø
25 FOR RADIUS=1 TO 1Ø STEP 5
3Ø CIRCLE(5Ø,15Ø),RADIUS
    : NEXT RADIUS
35 FOR RADIUS=1 TO 6 STEP 2
4Ø CIRCLE(5Ø,15Ø), RADIUS
    : NEXT RADIUS
45 FOR RADIUS=1 TO 1Ø STEP 5
5Ø CIRCLE(1ØØ,15Ø),RADIUS
    : NEXT RADIUS
55 FOR RADIUS=1 TO 6 STEP 2
6Ø CIRCLE(1ØØ,15Ø),RADIUS
    : NEXT RADIUS
65 DRAW "BM 5Ø,15Ø; L1Ø U5 E5 R15 L8 E8"
68 DRAW "BM 1ØØ,15Ø; U5 L5 H3 L2Ø"
7Ø DRAW "BM 1ØØ,15Ø; L45 U1Ø R2Ø D1Ø L15 U18 R15
    D15 U23"
75 FOR RADIUS=1 TO 1Ø STEP 5
8Ø CIRCLE(2ØØ,15Ø),RADIUS
    : NEXT RADIUS
85 FOR RADIUS=1 TO 6 STEP 2
9Ø CIRCLE(2ØØ,15Ø),RADIUS
    : NEXT RADIUS
95 FOR RADIUS=1 TO 1Ø STEP 5
96 CIRCLE(18Ø,15Ø),RADIUS
    : NEXT RADIUS
97 FOR RADIUS=1 TO 6 STEP 2
98 CIRCLE(18Ø,15Ø), RADIUS
    : NEXT RADIUS
1ØØ DRAW "BM5Ø,7Ø; U1Ø L2Ø D1Ø U1Ø L2Ø D1Ø U1Ø
    L2Ø"
11Ø DRAW "BM6Ø,7Ø; U7Ø D68 BE5 E5Ø G1Ø H3Ø"
12Ø DRAW "BM 1ØØ,7Ø; U1Ø R5 F5 D5"
13Ø DRAW "BM 12Ø,7Ø; U5 R2 D5 U4 R5 U1 D4 U4 R1Ø
    U1 D5"
14Ø DRAW "BM 163,7Ø; E1Ø D2 F8 L2 H8 D2 F8 L2 H8
    D2 F8   L2 H8"
145 DRAW "BM 9Ø.14Ø: U18 R125 D28 L115"
15Ø DRAW "BM 9Ø,14Ø; R8 D5"
155 DRAW "BM 1ØØ,14Ø; U4 R3 D2 L3 R2 F2"
16Ø DRAW "BM 11Ø,14Ø; E4 F4 H1 L4"
165 DRAW "BM 13Ø,14Ø; U4 R4 D4 L4"
17Ø DRAW "BM 145,14Ø; U4"
175 DRAW "BM 15Ø,14Ø; U4 R4 D4 L4"
18Ø DRAW "BM 165,14Ø; E1 H2 E1"
182 DRAW "BM 17Ø,14Ø; U4 D2 R4 U2 D4"
184 DRAW "BM 18Ø,14Ø; E4 F4 H1 L3"
186 DRAW "BM 195,14Ø;U4 R4 BD4 L4"
188 DRAW "BM 2Ø5,14Ø; U5 D3 E3 G3 F3"
195 DRAW "BM Ø,7Ø; R165 D15 G2Ø D16"
2ØØ DRAW "BM 255,12Ø; L155"
2Ø5 FOR RADIUS=1 TO 5 STEP 4
2Ø8 CIRCLE(9Ø,9Ø),RADIUS
    : NEXT RADIUS
21Ø DRAW "BM Ø,12Ø; R255"
22Ø FOR RADIUS=1 TO 5 STEP 4
225 CIRCLE(5Ø,9Ø),RADIUS
    : NEXT RADIUS
23Ø FOR RADIUS=1 TO 5 STEP 4
235 CIRCLE(3Ø,9Ø),RADIUS
    : NEXT RADIUS
24Ø FOR RADIUS=1 TO 5 STEP 4
245 CIRCLE(1Ø,9Ø),RADIUS
    : NEXT RADIUS
25Ø FOR RADIUS=1 TO 5 STEP 4
255 CIRCLE(7Ø,9Ø),RADIUS
    : NEXT RADIUS
26Ø PAINT (125,2Ø),3,4
265 PAINT (Ø,Ø),3,4
27Ø PAINT (2Ø,125),2,4
3ØØ FOR C=25Ø TO 15 STEP -5
31Ø PUT(C,162)-(C+36,178),W
32Ø PUT(C-15,162)-(C+21,178),V
33Ø NEXT C
34Ø PUT(15,162)-(51,178),W
    : GOTO 3ØØ
4ØØ DRAW "BM 192,155; U8"
1ØØØ DRAW "BM Ø,3Ø; R52 U3Ø"
1ØØ5 PAINT (Ø,Ø),2,4
1Ø1Ø DIM W(36,16)
1Ø2Ø GET (12,12)-(48,48),W
1Ø3Ø DRAW "BM 16,16; R8 E4 R8 F4 R8 G4 L8 NE4 H4
    L8 G4 L8 NE4 D4 R4 BR4 NU4 R8 NU8 R8 BR4 R4
    NU4 E4 U4 G4 L24"
1Ø4Ø CIRCLE(18,24),5
    : CIRCLE(38,24),5
1Ø5Ø PAINT (18,18),1,4
    : PAINT (3Ø,14),1,4
    : PAINT (42,18),1,4
    : PAINT (46,2Ø),1,4
    : PAINT (42,22),1,4
    : PAINT (32,22),1,4
    : PAINT (24,22),1,4
    : PAINT (14,22),1,4
    : PAINT (18,24),1,4
    : PAINT (38,24),1,4
1Ø6Ø DIM V(36,16)
1Ø7Ø GET(12,12)-(48,48),V
1Ø8Ø PCLS
    : RETURN
```

CDR Robinton has used his fertile imagination to write over thirty graphic and numeric programs which he has shared with Radio Shack stores and dealers in his area. When he writes a program he does not graph the drawings on paper but dreams up the scene and proceeds to write the program using a GOTO to check his progress. CDR Robinton challenges other TRS-80 Color Computer owners to produce a bit of competition.

As a hearty and obviously active eighty year old he wonders if he is the oldest TRS-80 enthusiast. We wonder if he is, too, and would like to hear from you if you know differently.

BASIC Renumber Utility

by Van R. Malan

This small utility program will renumber BASIC programs on the Color Computer. It is a machine language program and is loaded into memory by the small BASIC program below. It is very simplistic and only changes line numbers and not such things as GOTOs, GOSUBs, and IF-THEN statements.

```
1Ø 'RENUMBER UTILITY LOADER
12 DATA 158, 25, 79, 95, 195, Ø
15 DATA 1Ø
25 DATA 237, 2, 174, 132, 14Ø
32 DATA Ø, Ø, 46, 244, 57
35 FOR J=32Ø TO 336
4Ø READ K
    : POKE J, K
45 NEXT J
52 POKE 157, 1
    : POKE 158, 64
```

This program stores the renumbering utility in a protected place in low memory untouched by any BASIC programs. This is locations 320-336 or 140-150 hex. Once this BASIC program has been run, type the command EXEC and any BASIC program in user memory will be renumbered. Try it with this program and it will be found neatly renumbered and stepped by 10. For those who do not like 10 as the step number, line 15 is the starting number and step. This may be changed to any positive number between 1 and 255 inclusive.

This program is very handy to clean up an existing program or to add space between lines if a section of code is to be added. One practical approach to using this utility is to save it on tape. When a program is to be renumbered, reload the program above and run it. Now reload the program to be renumbered and type EXEC. Remember to change the GOTOs, etc.! Once the renumber utility is in memory it will remain usable until the computer is turned off.

The machine language program listing is shown below along with brief comments.

```
        ORG     $14Ø      ;START PROGRAM AT 14ØH
        LDX     $19       ;X <- BEGIN ADDR OF PRG
        CLRA              ;CLEAR THE D ACCU
        CLRB              ;  D <- A & B
NEXT    ADDD    #$ØA      ;ADD 1Ø TO THE D ACCU
        STD     2,X       ;STORE LINE NUMBER
        LDX     Ø,X       ;GET NEXT PRG LINE
        CMPX    #Ø        ;COMPARE ADDR WITH Ø
        BGT     NEXT      ;IF >Ø GO
        RTS               ;ELSE RETURN TO BASIC
        END
```

Christmas Line Draw Routine

by Jason R. Gee
Thibodaux, LA

This program is for those of you who like art but do not have access to joysticks. To become an artist, simply use the coordinate keys: ↑, ↓, ← and → to draw a picture. If you would like to blockout a graphics space, press the space bar. Of course, since it is Christmas, I chose the colors red and green. Press the 'R' key if you want the space red. Otherwise, the space will be colored green.

If you would like to start over, press the 'S' key and if you are finished, press the 'F' key.

I just received my computer a few weeks ago, and I am really impressed at all the extra functions the 4K Color Computer can do, not to mention the EXTENDED BASIC for the 16K computer. Although I own a 4K Color Computer, I can do almost everything the EXTENDED BASIC can do. (Except for the special commands for graphics.)

By the way, I have a special Christmas surprise at the end of the program.

```
1 REM "THE CHRISTMAS ARTIST"
2 REM BY JASON GEE
3 REM 7/14/81
1Ø CLS(Ø)
2Ø H=31
3Ø V=15
4Ø C=6
5Ø SET(H, V, C)
6Ø A$=INKEY$
7Ø IF A$="" THEN 6Ø
8Ø IF A$="R" THEN C=4 ELSE C=6
9Ø IF A$=CHR$(8) THEN H=H-1
1ØØ IF A$=CHR$(9) THEN H=H+1
11Ø IF A$=CHR$(94) THEN V=V-1
12Ø IF A$=CHR$(1Ø) THEN V=V+1
13Ø IF H>63 THEN H=H-63
14Ø IF H<Ø THEN H=H+63
15Ø IF V>31 THEN V=V-31
16Ø IF V<Ø THEN V=V+31
17Ø IF A$=" " THEN RESET(H, V)
     : GOTO 6Ø
18Ø IF A$="S" THEN 1Ø
19Ø IF A$="F" THEN 21Ø
2ØØ GOTO 5Ø
21Ø CLS(Ø)
22Ø FOR X=23 TO 38
23Ø SET(X, 25, 6)
24Ø SET(X, 21, 6)
25Ø NEXT X
26Ø FOR X=21 TO 25
27Ø SET(23, X, 4)
28Ø SET(38, X, 4)
29Ø NEXT X
3ØØ FOR X=15 TO 47
31Ø SET(X, 2Ø, 6)
32Ø NEXT X
33Ø V=19
34Ø H=16
35Ø SET(H, V, 6)
36Ø V=V-1
37Ø H=H+1
38Ø IF H>31 THEN 4ØØ
39Ø GOTO 35Ø
4ØØ H=47
41Ø V=19
42Ø SET (H, V, 6)
43Ø V=V-1
44Ø H=H-1
45Ø IF H<31 THEN 47Ø
46Ø GOTO 42Ø
47Ø V=19
48Ø C=45
49Ø D=17
5ØØ X=RND(C)
51Ø F=RND(8)
52Ø IF X<D THEN 5ØØ
53Ø SET(X, V, F)
54Ø D=D+1
     : IF D>31 THEN 58Ø
55Ø C=C-1
56Ø V=V-1
     : IF V<5 THEN 58Ø
57Ø GOTO 5ØØ
58Ø PRINT@455, "MERRY CHRISTMAS!";
59Ø SOUND 89, 15
     : SOUND 133, 1Ø
     : SOUND 133, 1Ø
```

Helpful Programs

Are you stumped for a Christmas gift for that special someone on your list who 'has everything'? Well, I think we may just have the answer . . . the TRS-80 Pocket Computer . . . it's on sale now through December 27 for just $169.95, a whopping $60 dollars off our 1982 Catalog price! Or, if your favorite person already has a Pocket Computer, why not give him or her one of the matching peripherals which are also sale priced. The Cassette Interface is 39% off, at only $29.95 and the Printer and Cassette Interface is reduced $20 to $129.95, both prices also good through December 27, 1981.

You probably already know what a powerful tool this little computer is so it shouldn't be hard to think of the many ways that special student, engineer or business person on your list could use it every day. Anyway, I just thought that in case you have not seen our recent Pocket Computer ads, I would do my last good deed for this year and solve your gift-giving problems. And now for some of the more unusual uses I have put my Pocket Computer to lately.

Several weeks ago a good friend of mine was trying to create some cassette tapes with recorded music composed of selections or cuts from many different record albums. Most of the cuts had the times listed on either the record jacket or on the label of the record and a few she had to time with a stopwatch to get the exact cut playing time. The problem she was having was figuring out which cuts would make the best use of a C-90 cassette with 45 minutes of recording time on each side. As you can imagine, adding up a bunch of different times in minutes and seconds can be rather tedious, so I created the following short program for the Pocket Computer which does the job nicely and is quite versatile as we shall see shortly.

```
5: " " BEEP 3
     : PAUSE " *** TIME COMPUTER ***"
1Ø: USING "####.####"
     : X=Ø
     : T=Ø
2Ø: "K"INPUT "TIME (MM.SSHH)? "; T
     : T=T/1ØØ
3Ø: X=X+DEG T
     : T=1ØØ*(DMS X)
4Ø: "=" PRINT "TOT. (MM.SSHH)= "; T
     : GOTO 2Ø
```

The program is run in the DEF Mode by pressing SHFT SPC. This beeps 3 times, flashes the name of the program on the screen, sets the input time register (T) and the total time register (X) to zero. The program then requests the time be input in the form of MM.SSHH where MM is the time in minutes, SS is the time in seconds and HH is hundredths of seconds (which I'll explain a little later). Times may be input or accumulated in several ways. First, you can simply enter each record cut time, press ENTER after each entry and the PC will display the total time to that point in the same format as was input. Or, you could enter the maximum recording time of the cassette for instance, 45 minutes for a C-90 (per side) and then enter the record cut times as negative numbers and thereby subtract each cut time from the total each time the ENTER key is pressed. The program is quite versatile and my friend was really impressed at how easy it now became to get the most music on each cassette.

The total time is retained in the PC and even if it turns itself off while you are recording, simply turning it back on and pressing SHFT = will recall and display the previous total time. If you then want to continue inputting times, just press SHFT K or if you want to start over, then press SHFT SPC to clear out the registers.

Line 30 of the program is where all the computation and conversion is done. You've no doubt noticed by now that I have used both the DMS and the DEG functions of the PC. Why? First we convert the time to a decimal number and then add it to the total time register X. Then we convert the decimal value of the total time back to the Hours/Minutes/Seconds format using the DMS instruction and multiply the result by one hundred. The reason for multiplying by one hundred is so that we can display the time in the same format as was input because we divided by one hundred in line 20. This was done simply as a matter of convenience since in the basic format of the DEG instruction (see page 22 of the Owner's Manual) the two positions to the left of the decimal point would represent, in this case hours, and we are dealing only with minutes and seconds. So, it seemed more logical to input the time in minutes then a decimal point and then the seconds portion of the time. If you look at page 22 you will see that this function was designed for conversion of angles but since the units are the same as our 12 hour/60 minute/60 second time system, it works very nicely!

As I mentioned earlier, you can also enter hundredths of a second into the program and it will total and convert these to the next second when you go over 99 hundredths. I added this little feature a few days later when I ran into a friend of mine who is a coach at a local high school. He was trying to find an easy way of adding up chronograph times he had accumulated from various races and was finding it to be rather time consuming (pun intended). So, by putting this feature in, he can now add up these times rapidly and get his totals. I will probably modify this program shortly to allow him to compute the average of the times input, if desired, but this will give you an idea of the versatility of these two built-in functions.

Another way I use my Pocket Computer is to make price comparisons when shopping at the grocery store. Since many stores still do not post the price per unit of an item on the shelf or the price per unit which is posted is incorrect due to recent price changes, I find this very handy to tell what is the best buy when considering similar products. So, if you are price conscious these days and who isn't, you will probably find this program useful too.

```
9ØØ: "C" BEEP 3
     : PAUSE "PRICE/UNIT COMPARATOR"
91Ø: USING "####.##"
92Ø: "A"INPUT "BRAND-A= "; A$
     : INPUT "PRICE= "; B
     : INPUT "SIZE= "; C
93Ø: "B" INPUT "BRAND-B= "; D$
     : INPUT "PRICE= "; E
     : INPUT "SIZE= "; F
94Ø: "V" IF (B/C)>(E/F) THEN 96Ø
95Ø: PRINT "BEST BUY:  "; A$
     : GOTO 9ØØ
96Ø: PRINT "BEST BUY:  "; D$
     : GOTO 9ØØ
```

To use the program, just select the DEF Mode and press SHFT C. This flashes the program name, sets up the output format and goes to request the first brand to be compared. Type in the name of the product (remember, only the first seven characters will be used), press ENTER and the "PRICE=" is requested. ENTER the price in normal dec-

imal form and press ENTER again. The last entry is the SIZE which means the number of ounces, pounds etc. Pressing Enter after the SIZE for BRAND-A will request the same information for the second brand to be compared. The only thing to remember is that the units for both brands must be the same in order to get a valid comparison. After pressing ENTER for the second price/units entry, the PC compares the two entries and displays "BEST BUY:" and then the name that you input for whichever one computes to be the lowest in unit cost.

Looking at the above listing, program operation is very straightforward. Selecting either SHFT A or SHFT B will allow entering another brand A or B respectively for further comparison and then selecting SHFT V will again compute the best buy. You may think this to be a "waste of time" for the Pocket Computer because you can do it pretty easily on any old 'el cheapo' calculator. Right? Right! BUT, I also put my whole grocery list into the PC using PRINT instructions starting at line 100. First, in the RES Mode, I assign PRINT " to the SPC key. Then in the PRO mode I simply type in the line number, press SHFT SPC and then the item or items I want on that line for my list. Continue typing in line numbers and putting in your shopping list using the SHFT SPC to put in the PRINT " and avoid typing it in every time. When you are finished, just put an END as the last line number, before line 900 which is where you have your price comparator program. Now, if you have a PC Printer you can get a printed list of your items to be purchased by just RUNing the program in the DEF mode. You can also get it by LISTing the program but you get a little more extraneous information that way. When I go to the grocery store, I take the PC with me and as I go around picking up what is on my list, I simply delete each item in the PRO mode as I buy it. That way, when I am finished, anything remaining in the PC, I either couldn't find or decided to buy at a later time. When I want to make a price comparison, I just switch to DEF mode, select SHFT SPC and enter the information.

This program could be made a lot more elegant but it does the job handily for me, so I will leave it to those of you who can't resist, to create the "Coup de Elegance" on this program while you are overstuffed from whatever bird is you fancy this holiday season. And speaking of which, I hope all of you have a relaxing and Merry Christmas and a safe and rewarding New Year. Until then . . . more Pocket Power!

DEC/HEX—HEX/DEC Conversion

by Albert A. Livingstone, II
Chicago, IL

It is 2:00 A.M. You are doing a machine language program. You have unintelligible scrapes of scribbled notes spread about you. You are verging on a nervous breakdown while attempting to do Hex to Decimal/Decimal to Hex conversions. You can not use the fancy new program you have because your trusty TRS-80 is tied up. You must resort to pencil and paper calculations.

In attempting to maintain my sanity and secure a decent night of sleep, I had to find a better solution. My answer came in the form of Radio Shack's incredible little Pocket Computer and the following program I wrote. The operation of the program is simple. You are able to enter and receive Hex characters in their exact form.

For example, to convert Hex FDCE to Decimal, press (SHFT) H. You will be greeted with the name of the program and a prompt to input the First Integer. Type F and press (ENTER). You will now be prompted to input the Second Integer. Continue this process until all four integers have been entered. The computer will then display: Decimal = 64974. To continue, press (ENTER). Note: You must enter all four prompts with a value. For example, FF should be input: Ø (ENTER) Ø (ENTER) F (ENTER) F (ENTER); and Hex 234 would be entered as Ø (ENTER) 2 (ENTER) 3 (ENTER) 4 (ENTER).

To convert Decimal to Hex is equally simple. Press (SHFT) D. You will be greeted with the name of the program and prompted to input the Decimal number. Type in the number, press (ENTER). The first Hex value will be displayed. Press (ENTER) and you will receive the second Hex value. Continue this process until you receive all four values. For example, entering the Decimal number 65535, you will receive F F F F after pressing (ENTER) for each value. Entering the Decimal number 564, you will receive the Hex value 0. 2. 3. 4.

There you have it. It is fast, simple and accurate. It should save you countless hours of pencil and paper calculations.

```
2ØØ: "H":CLEAR
2Ø1: PAUSE "HEX TO DEC CONVERSION"
2Ø2: A=1Ø:B=11:C=12:D=13:E=14:F=15
21Ø: INPUT "FIRST INTEGER? ";J
22Ø: INPUT "SECOND INTEGER? ";O
23Ø: INPUT "THIRD INTEGER? ";S
24Ø: INPUT "FOURTH INTEGER? ";H
249: X=4Ø96:Y=256:Z=16
25Ø: V=(X*J)+(Y*O)+(Z*S)+H
26Ø: PRINT "DECIMAL = ";V
27Ø: GOTO 2ØØ
3ØØ: "D":CLEAR
3Ø1: PAUSE "DEC TO HEX CONVERSION"
3Ø2: INPUT "DECIMAL NUMBER ";A
311: B=4Ø96
312: C=A/B
313: D=INT (C)
314: E=D*B
315: F=A-E
316: G=256
317: H=F/G
318: I=INT (H)
319: J=I*G
32Ø: K=F-J
321: L=16
322: M=K/L
323: N=INT (M)
324: O=N*L
325: P=K-O
326: Q=INT (P)
38Ø: IF D<1Ø PRINT D
384: IF D=1Ø PRINT"A"
385: IF D=11 PRINT"B"
386: IF D=12 PRINT"C"
387: IF D=13 PRINT"D"
388: IF D=14 PRINT"E"
389: IF D=15 PRINT"F"
39Ø: IF I<1Ø PRINT I
391: IF I=1Ø PRINT"A"
392: IF I=11 PRINT"B"
393: IF I=12 PRINT"C"
394: IF I=13 PRINT"D"
395: IF I=14 PRINT"E"
396: IF I=15 PRINT"F"
4ØØ: IF N<1Ø PRINT N
4Ø1: IF N=1Ø PRINT"A"
4Ø2: IF N=11 PRINT"B"
4Ø3: IF N=12 PRINT"C"
4Ø4: IF N=13 PRINT"D"
4Ø5: IF N=14 PRINT"E"
4Ø6: IF N=15 PRINT"F"
41Ø: IF Q<1Ø PRINT Q
411: IF Q=1Ø PRINT"A"
412: IF Q=11 PRINT"B"
413: IF Q=12 PRINT"C"
414: IF Q=13 PRINT"D"
415: IF Q=14 PRINT"E"
416: IF Q=15 PRINT"F"
42Ø: GOTO 3Ø1
```

Rounding With the Pocket Computer

by Newell Claudy
Elkhart, IN

This algorithm for rounding is the sort of technique that is often very useful for novice programmers but seldom found in reference texts, so you may wish to call it to your readers' attention. It can be handy whether you are rounding off items on your income tax return to the nearest dollar, computing sale taxes to the nearest penny, or working on a corporate budget and rounding to the nearest $500. Pocket Computer users may find it especially useful, as the "USING" function of that machine does not incorporate rounding of the least significant digit.

Given numerical variables R and N, the algorithm returns M rounded to the nearest whole multiple of R. It works equally well whether R and N are positive or negative, and whether R is an integer or a decimal fraction. (I do not know why anyone would want to use a negative R, but it works if anyone does.)

The algorithm would probably be used most often as a subroutine, but the following illustrates how it works.

```
1Ø INPUT "ROUND TO "; R
2Ø INPUT "N="; N
3Ø GOSUB 1ØØ
4Ø PRINT "M="; M
5Ø GOTO 1Ø
1ØØ M=SGN(N)*INT(ABS(N/R)+.5)*R
    : RETURN
```

Some examples:

R	N	M
.01	1.555	1.56
.1	1.555	1.6
1	1.555	2
.01	−1.555	−1.56
−.01	−1.555	1.56
5	123456	123455
10	123456	123460
500	123456	123500
2000	123456	124000

I have never seen this published anywhere, but it is so fundamental that I doubt it is original with me.

P.S. Pocket Computer users who are only interested in rounding up the results of a calculation to a predetermined number of decimal places should consider using:

```
1Ø INPUT A
2Ø INPUT B
3Ø C=A/B
4Ø USING "####.##"
5Ø PRINT C+.ØØ5
6Ø END
```

The value added to the variable to be printed should be "5" preceded by the same number of zeroes as the number of #'s to the right of the decimal, plus a decimal point.

For example, if A = 20 and B = 3, C = 6.666666667, C + .005 = 6.671666667, and 6.67 is displayed or printed.

Fort Worth Scene (From page 12)

Our first step in providing you with more information came in July when we added a regular monthly column from the busy people in Computer Customer Services. The Customer Service people were eager to have a way to communicate with many customers at the one time.

August saw the first monthly article from CompuServe. We added a bi-monthly series of articles from Personal Software (VisiCalc) in September, and October saw the addition of monthly articles from both Dow Jones and the small Computer Company (Profile).

We thank CompuServe, Dow Jones, Personal Software, and the small Computer Company for taking the time to prepare these articles and get them to us within our frequently changing schedule.

The purpose of these articles is to direct special material to users of particular products. We hope that these articles will be a way for you to learn more about each of these products, what the product can do for you, and how you can use it more effectively. We hope you will learn about new features and new techniques that will make your TRS-80 ever more valuable to you.

We have one more series of articles that we will be presenting you, but more about THAT in another issue.

Tacked to my bulletin board are article ideas in 18 different categories. The article topics range from definitions of commonly used "buzz-words" to articles on the effective use of COBOL's multi-key ISAM files and high resolution, three dimensional color graphics techniques.

So, how are all of those articles going to get written? Well, most of them will have to be written by you, the readers. We will print special articles from internal people, when those articles are available.

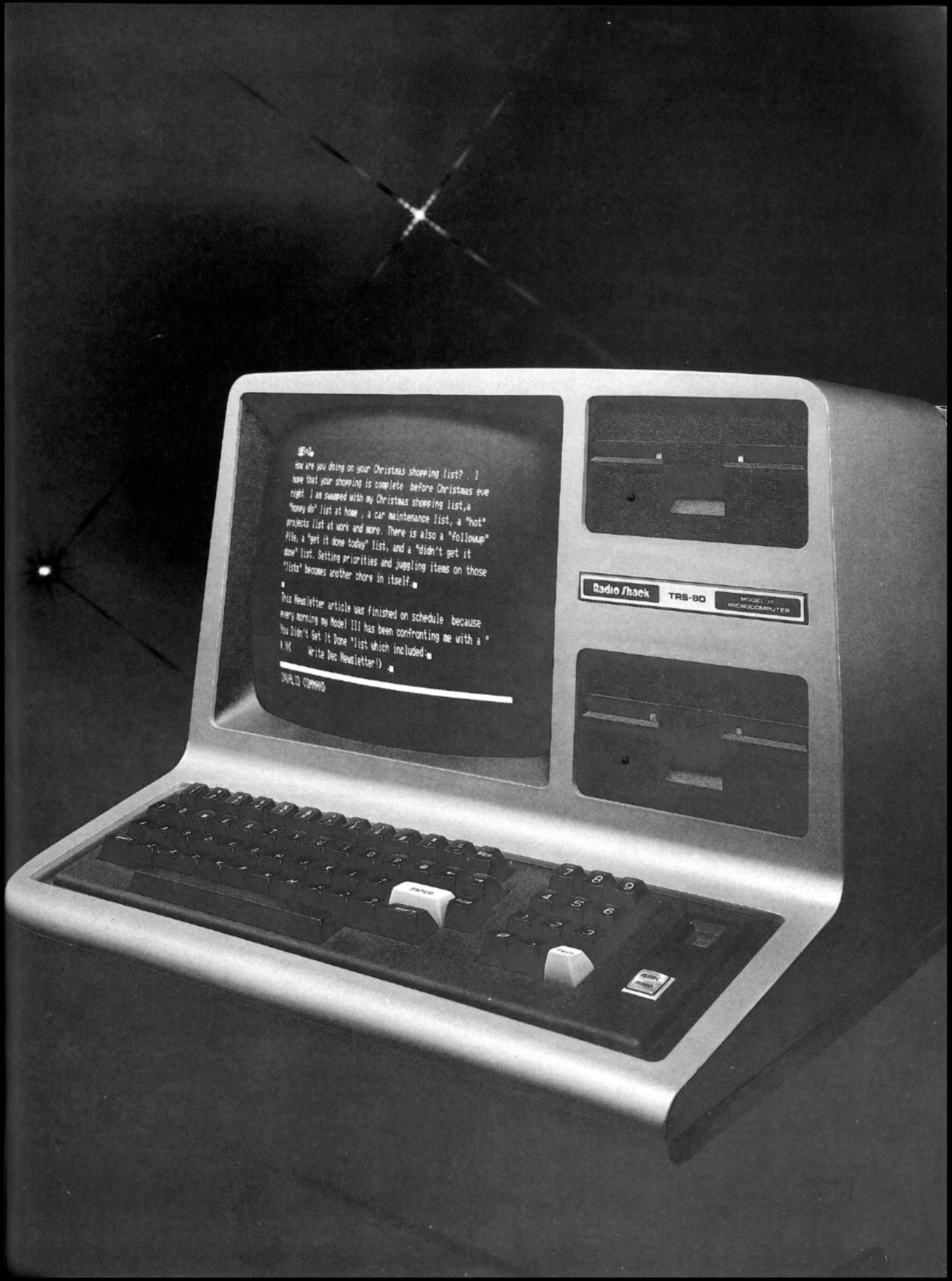
To help you know what may be coming in the next year, here is a list of general topics which we will use to help focus each issue during the coming year:

January	— Sequential Files (Tape and Disk)
February	— Direct/Random Access Files
March	— Review of Accounting Software
April	— Non-Printer peripherals
May	— Languages
June	— Graphics
July	— Word Processing
August	— Printers
September	— Education, VisiCalc, Spectaculator
October	— Operating Systems
November	— Data Bases/Data Base Management
December	— Christmas Issue

If you have material or ideas which will fit these topics, send it to us and we will try to integrate your material with ours. If you have material, but you don't think it will fit these topics, send it anyway!

Remember, if you don't see what you want here it may be because you haven't sent us anything!

Bruce W. Elliott — Editor
Linda Miller — Writer


Now are you doing on your Christmas shopping list? . I
hope that your shopping is complete before Christmas eve
night. I am swamped with my Christmas shopping list,a
"honey do" list at home , a car maintenance list, a "hot"
projects list at work and more. There is also a "followup"
file, a "get it done today" list, and a "didn't get it
done" list. Setting priorities and juggling items on those
"lists" becomes another chore in itself.
This Newsletter article was finished on schedule because
every morning my Model III has been confronting me with a "
You Didn't Get It Done "list which included:
Write Dec Newsletter!).
INVALID COMMAND
Radio Shack
TRS-80
MODEL III
MICROCOMPUTER
ENTER
CLEAR
SHIFT
4BK
RUNO

Radio Shack®

TRS-80 Microcomputer News
P.O. Box 2910 • Fort Worth, Texas 76113-2910

ADDRESS CHANGE

☐ Remove from list
☐ Change as shown

Please detach address label and mail to address shown above.

IF UNDELIVERABLE DO NOT RETURN