

**TRS-80<sup>®</sup>**

Volume 4, Issue 4

**APRIL, 1982**

Price \$1.50

# Microcomputer News

Information Published for TRS-80 users.

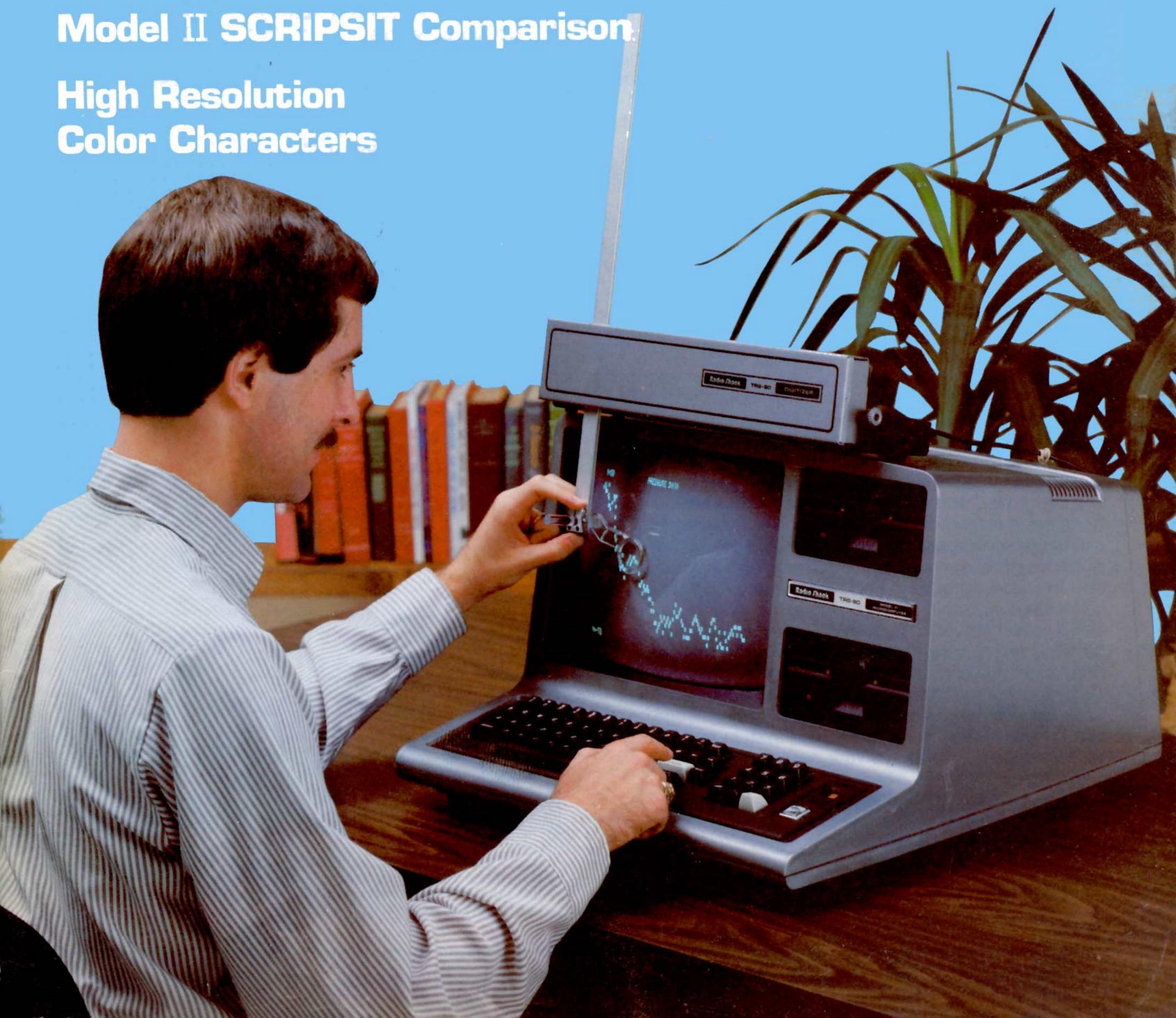
**Non-Printer Peripherals**

**Color Computer Assembly Language**

by William Barden

**Model II SCRIPSIT Comparison**

**High Resolution  
Color Characters**



# Fort Worth Scene



There is a change coming soon to Computer Customer Service which—not surprisingly—involves our telephone system.

What is the change? We are discontinuing the toll-free (WATS) lines and replacing them with regular toll lines.

Why the change? Simply because a WATS system is unable to provide you what we know you require most, i.e., prompt access to assistance. Any of you who have tried to reach the Customer Service Department can attest to the foregoing.

To those who may be asking: "Why not install more WATS lines?" The answer is WE HAVE. Fifty-three of them to be exact, with staff to match. The result was no significant improvement of accessibility. Instead, more lines simply seem to prompt more calls and many thousands of those calls are not related to our mandated purpose, which is helping our users with implementing our software or using our hardware.

How will the new telephone system improve accessibility? It will have a direct and indirect effect. First, it will eliminate some of the superfluous calls, leaving more representatives available to assist those who really need help, and it will eliminate those lengthy "holds." Adequate staffing of the lines is assured (although it will take us a month or two to get the proper balance between support groups after the new system is installed). If we need more representatives to staff the lines, we will get them. That is a promise from top management.

Secondly, and more indirectly, we hope this change will encourage many of our customers to rely upon the qualified Customer Service Representative at their local Radio Shack Computer Center. Very soon, we will have a Computer Center in every major (and many not-so-major) markets in the country. Our Vice President of Marketing is firmly committed to staffing the Customer Service positions in most of those Computer Centers with the most capable, informed individuals available. If these local people are unable to answer your questions or resolve your particular problem, they have access to support groups in Fort Worth who are dedicated to supporting only them (i.e., the Representatives at the Centers), in effect, a secondary line of support. Your question will be answered or your problem resolved with a minimum of delay in the majority of instances.

It should be pointed out that by using your RSCC Customer Service Representative, you get to know him and he in turn becomes familiar with you, your system and applications. In essence, you have your own consultant who can relate to your particular needs.

When will the change take effect? June 1, 1982.

It is important to us that you understand our motives for making this change. COST SAVINGS IS NOT THE REASON. As a matter of fact, Radio Shack's nationwide Customer Service Team will be expanded from its present 60-plus to over 200 nationwide and the sole purpose is to provide you

with better service. Decentralizing and enlarging our Customer Service Support function is a significant first step towards accomplishing that task.

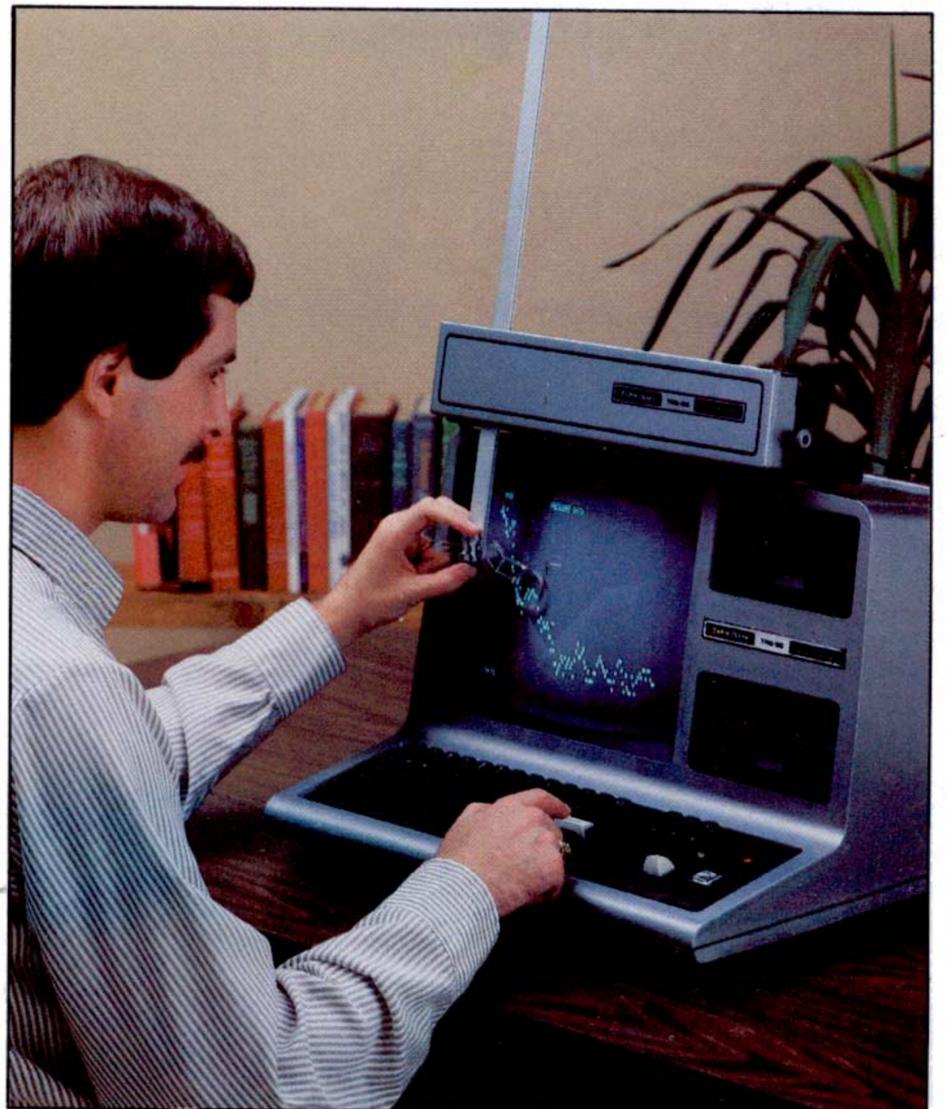
Please check the May Microcomputer News for more details on the forthcoming change.

—Director of Computer Customer Services

## Computer Clubs

WML TRS-80 Computer Group  
c/o Steve Bergers  
2040 Janes Avenue NE  
Grand Rapids, MI 49505  
1-616-3636-3561

Appraiser's TRS-80  
c/o Dan D. Swearingin, MAI, SREA  
1215 Fruit, NW  
Albuquerque, N.M. 87102  
1-505-842-1700



**Radio Shack's TRS-80 Digitizer is versatile enough to digitize from many surfaces, including your video display (as shown on this month's front cover). The adjustable crosshair piece gives you even more flexibility.**

Reproduction or use, without express written permission from Tandy Corporation of any portion of the Microcomputer News is prohibited. Permission is specifically granted to individuals to use or reproduce material for their personal, non-commercial use. Reprint permission for all material (other than William Barden's article), with notice of source, is also specifically granted to non-profit clubs, organizations, educational institutions, and newsletters.

TRS-80 Microcomputer News is published monthly by Radio Shack, a division of Tandy Corporation. A single six month subscription is available free to purchasers of new TRS-80 Microcomputer systems with addresses in the United States, Puerto Rico, Canada and APO or FPO addresses. Subscriptions to other addresses are not available.

The subscription rate for renewals and other interested persons with U.S., APO or FPO addresses is twelve dollars (\$12.00) per year, check or money order. Single copies of the Microcomputer News may be purchased from Radio Shack Computer Centers or Computer Departments for \$1.50 (suggested retail) each. The subscription rate for renewals and other interested persons with Canadian addresses is Fifteen dollars (\$15.00) 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 with 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 store 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 to the material. This assures that our readers may be permitted to photocopy 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 70007,535.

#### Notes to Program Users:

Programs published in the Microcomputer News are provided as is, for your information. While we make reasonable efforts to ensure that the programs we publish here work as specified, Radio Shack can not assume any liability for the accuracy either of the programs themselves, or of the results provided by the programs.

Further, while Microcomputer News is a product of Radio Shack, the programs and much of the information published here are not Radio Shack products, and as such can not be supported by our Computer Customer Service group. If you have questions about a program in the Microcomputer News, your first option is to write directly to the author of the program. When possible, we are now including authors' addresses to facilitate communications. If the address is not published, or if you are not happy with the response you get, please write us here at Microcomputer News. We will try (given the limited size of our staff) to find an answer to your question and, in many cases, will publish the answer in an up-coming issue of Microcomputer News.

Comments on our program listing style:  
In order to make the program listings we publish easier to read, we have adopted a style of inserting spaces to enhance readability, and we separate each program statement onto a separate line. While these techniques increase program readability, they also require more memory, and may execute more slowly than the original program did.

When you are entering a program for your own use, you may wish to eliminate many of the extra blanks (see your owners manual for required blanks), and you should certainly move multiple statements up to a single line where possible.

#### Trademark Credits

CompuServe™	CompuServe, Inc.
DIF™	Software Arts, Inc.
Dow Jones NEWS/RETRIEVAL Service®	Dow Jones & Co., Inc.
Personnel Manager™	Image Producers
Project Manager™	Image Producers
ReformatTer™	Microtech Exports
Time Manager™	Image Producers
VisiCalc®	VisiCorp™
Program Pak™	Tandy Corporation
SCRIPSIT™	Tandy Corporation
TRSDOS™	Tandy Corporation
TRS-80®	Tandy Corporation

#### Contents:

Assembly Language Programming . . . . .	8
by William Barden, Jr.	
<b>Color Computer</b>	
Product Line Manager's Page . . . . .	41
<i>Questions and Answers</i>	
Programs	
Hi-Res Character Generator by Ron Van Dyke . . . . .	46
Linefeed With Carriage Return by Ed Hamilton . . . . .	45
Merge Cassette Programs by Jorge Mir . . . . .	42
Perpetual Calendar by Kenneth Heberle . . . . .	43
Printer Calendar by Harry Stern . . . . .	43
Revisions, Enhancements, Etc . . . . .	42
<b>Computer Clubs</b> . . . . .	2
<b>Computer Customer Service</b> . . . . .	13
<i>SCRIPSIT Ideas</i>	
Notes From Computer Customer Services . . . . .	14
<b>Data Bases</b>	
CompuServe . . . . .	15
<i>Network Expansion</i>	
Phone Numbers . . . . .	48
Dow Jones . . . . .	17
<i>Sports Data Base</i>	
<b>Educational Products</b>	
Graphical Analysis of Experimental Data . . . . .	29
Interpreting Graphs in Physics . . . . .	31
Southern New Jersey Computer Awareness Project . . . . .	32
<b>Feature Story</b>	
Non-Printer Peripherals . . . . .	5
<b>Fort Worth Scene</b> . . . . .	2
<b>General Interest</b>	
Compound Interest by Mike Riggs . . . . .	20
Wind Chill Factor by Bill Clough . . . . .	7
<b>Model I/III</b>	
Bugs, Errors, and Fixes	
Accounts Payable (26-1554) . . . . .	21
Accounts Receivable (26-1555) . . . . .	21
Manufacturing Inventory Control (26-1559) . . . . .	21
Medical Office System (26-1568) . . . . .	22
TRSDOS Model III (26-312) . . . . .	21
Product Line Manager's Page . . . . .	19
<i>Personnel Manager</i>	
Programs for Models I and III	
Arrays by Linda Miller . . . . .	23
Array Input Routine by Louis Self . . . . .	24
Client Activity Information Array by Kenneth E. Robinson . . . . .	25
Disk Head Cleaner by Lowell Johnson . . . . .	16
Revisions and Enhancements . . . . .	22
<b>Model II</b>	
Bugs, Errors, and Fixes	
Accounts Payable (26-4505) . . . . .	36
RSCOBOL (26-4703 and 26-4704) . . . . .	36
Product Line Manager's Page . . . . .	33
<i>SCRIPSIT 1.0 vs. SCRIPSIT 2.0</i>	
Programs	
Statistical Analysis by W. W. Schumacher . . . . .	35
Revisions, Enhancements, Etc . . . . .	36
<b>Notes on Previous Issues</b>	
December, 1981 Model I/III Calendar Program . . . . .	28
January, 1982 VisiCalc Model . . . . .	28
<b>Peripherals</b>	
Bugs, Errors, and Fixes	
Line Printer VII (26-1167) . . . . .	7
Revisions, Enhancements, Etc . . . . .	7
<b>Pocket Computer</b>	
Programs	
Compound Meters by Bruce K. Taub . . . . .	40
Cub Scout Song by Jules L. Schafer . . . . .	40
Multiple Linear Regression by Garland Bryant . . . . .	38
Pythagoras' Theorem by J. Strolin . . . . .	40
Step-Saving Ideas by Leonard Levine . . . . .	37
Revisions, Enhancements, Etc . . . . .	39
<b>View From the 7th Floor</b> by Jon Shirley . . . . .	4

---

# View From The Seventh Floor

Jon Shirley  
Vice President, Radio Shack Computer Merchandising

I had hoped I could devote this column to a subject other than one caused by my reading, but no such luck. I have been very busy what with our new models and got behind on my reading but a bunch of customer letters forced me to pick up the January '82 issue of 80 Microcomputing and read Mr. Wayne Green's column, 80 REMARKS. In this incredible column he states that he has heard a rumor from our dealers that the Color Computer will be phased out!

He did not call us because "I have not called the factory about this because if it is true they will probably deny it. If it is false they will also deny it. So why waste money on the phone call?" Well, if Mr. Green had not been so tight with his nickel he would have heard a denial so strong he might not have published such rubbish and saved us all a lot of aggravation.

My response to Mr. Green follows intact. It was also sent to all those many Color Computer owners who wrote:

January 21, 1982

Wayne Green, Editor  
80 Microcomputing  
Elm Street at Rt. 101  
Peterborough, New Hampshire 03458

Dear Mr. Green:

As the publisher of several magazines it seems you have the power to print virtually any personal opinion of yours as well as any unsubstantiated rumor no matter how far wrong you are. In the past we have tended to ignore your monthly misinformation about Radio Shack as it did no one harm except, perhaps, yourself.

I must, however, take strong exception to your 80 Remarks column of January, 1982, in which you state that we are likely to drop the TRS-80 Color Computer.

That statement is absolutely untrue and is causing unnecessary anxiety on the part of our Color Computer owners. We just finished a terrific Christmas season with the Color Computer in which we sold out our entire warehouse supply. Contrary to your comments about no advertising, the Color Computer had more advertising September through December than any other TRS-80, which included a great deal of national TV advertising.

Our March computer catalog will list 26 Program Paks and 8 cassette programs for the Color Computer, and we have over 40 Program Paks, cassette and disk packages in the works. In addition to the current peripherals there are three additional hardware products for the Color Computer in design.

So, Wayne, sorry, but you are dead wrong! The

Color will not be dropped in 1982, or in 1983 for that matter. It is a highly successful product and stands a good chance of being the number one unit selling computer in America this year.

If you believe in responsible journalism you will publish this letter as soon as possible, and, in the meantime, I will send it to those customers who are calling us who, unfortunately, believe that if something is in print, it is true.

Sincerely,  
Jon Shirley  
Vice President, Merchandising  
Computer Products

P.S. You are also wrong on the Pocket Computer. We sold more of those the first 12 months of its life than we sold Model I's the first year of its life. And we just introduced its new brother.

In the article Mr. Green also decided to comment on why we would drop the Color Computer. "neglected, underadvertised, underdeveloped . . . and then phased into oblivion." Well part of that phase into oblivion will occur in two months when the Color Computer factory will move into a brand new building about three times larger than the current one. And underadvertised? Did you watch the start of the Super Bowl? That Color Computer spot cost in the six figures and comes after the most intensive TV fall series we have ever run on a single product. I guess Wayne does not watch football games or the Today show or the late news or????

I think the Color Computer is a great product. And we have a lot of plans to make it a greater product. There is already more software available from us after one year of selling the Color Computer than we had for the Model I after two years and, as I said in my letter, there is a lot more coming.

If I sound a little emotional, I guess I am. I just do not think that journalists should be allowed to print a total untruth when it harms many other people. It is not my people I am talking about, it is those Color Computer owners who subscribed in good faith to 80 Microcomputing and who called us to ask why we would do such a stupid thing as dropping the Color Computer. So, Wayne, if you read my column, please call us the next time, even collect, before you draw your pen from its holster.

And you Color Computer owners—send Wayne a few articles. That might make him a believer since he does not seem to read his competition where I have seen a lot of C.C. coverage.

Until next month.

# Non-Printer Peripherals

A quick look at the non-printer peripherals listed in our current Computer Catalog (RSC-7) reveals a number of items, most of which seem to fall into three categories: RS-232C devices, disk drives, and cassette-port oriented devices. We will look at two of these categories, RS-232C devices and cassette-port devices.

## RS-232C DEVICES

With the exception of the Pocket Computer (PC-1, Cat. No. 26-3501) and the Level I Model III, each of Radio Shack's TRS-80 Computers either comes with at least one RS-232C port, or can have one added as an optional extra.

## MODEMS

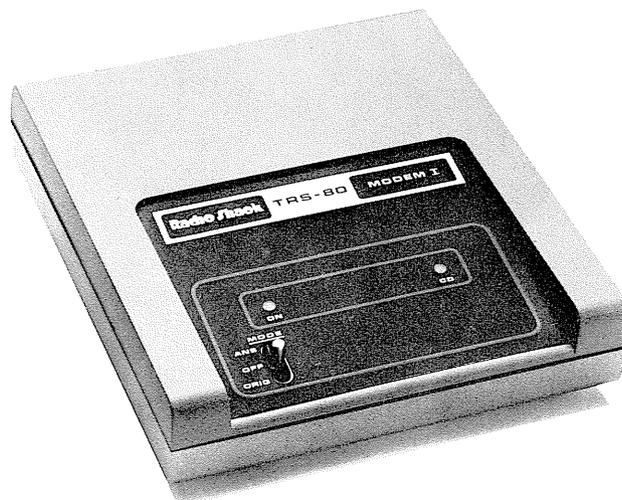
The most obvious RS-232C devices which Radio Shack offers are our three modems. A modem has two functions. The first is to modulate (translate) the information provided by the RS-232 port on your computer into tones which can be transmitted over ordinary telephone lines. The second function is to demodulate (again, translate) the audio tones which come over the telephone lines into RS-232C signals which can be understood by your computer. The word modem was created by combining the words for these two functions (MODulate/DEMODulate.)



**TRS-80 Telephone Interface II**

Assuming that you have the proper software, a modem can open up a whole new world to you and your computer. Two of the more obvious things you can do with a modem and software are to connect your computer, via telephone lines, with the CompuServe Information Service and the Dow Jones News/Retrieval Service. As the articles (run each month in Microcomputer News) on these two services point out, a whole world of information exists which you can access through a modem.

Another common use for modems is to let two individuals (again, with proper software), possibly with different



**TRS-80 Direct Connect Modem I**

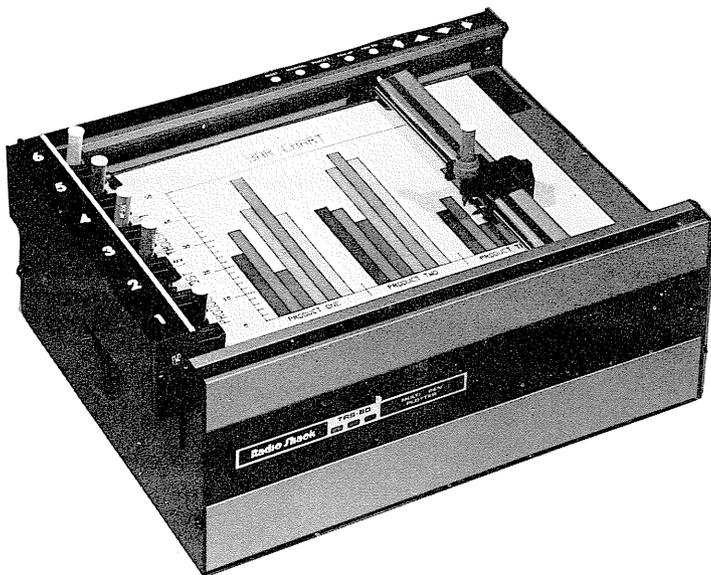
types of computers, communicate and transfer both information and programs. We have used a pair of Telephone Interface IIs here at the Microcomputer News to allow one of our Model IIs to transfer BASIC programs to our Color Computer. Eventually we will probably create a single cable to connect the two machines together, but until that happens, we use modems. And, each month two or three of the articles we run are sent via telephone lines and modems.



**TRS-80 Direct Connect Modem II**

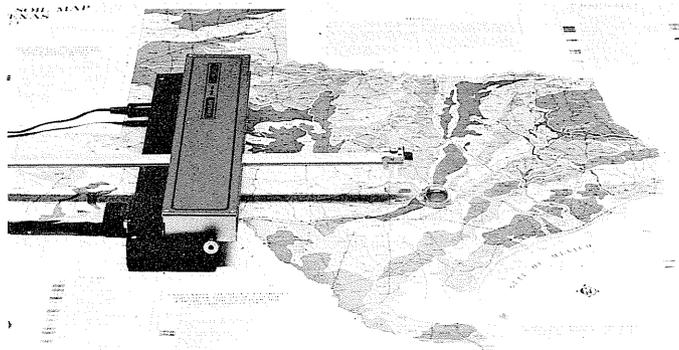
## OTHER RS-232 DEVICES

Radio Shack currently offers two other non-printer RS-232C devices. These are the Multi-Pen Plotter and our Digitizer.



### TRS-80 Multi-Pen Plotter

Radio Shack's Tandy-graph software package ( a simple graph creation routine) is available, at no additional cost, for Models I, II, III and the Color Computer with the purchase of the Multi-Pen Plotter. Tandy-graph can be used to create pie charts, line graphs and bar graphs.



### TRS-80 Digitizer

The Digitizer is a very versatile product which should let you input two-dimensional data to your programs from almost any type surface you want, including your video screen!

### NEW PRODUCTS

Radio Shack is now offering eight new RS-232C cables or adaptors. The first five of these items are RS-232C cables:

Stock Number	Description	Suggested Retail Price
26-1490	10 foot RS-232C Cable	\$29.95
26-1491	25 foot RS-232C Cable	\$39.95
26-1492	50 foot RS-232C Cable	\$54.95
26-1493	100 foot RS-232C Cable	\$89.95
26-1494	5ft. 4 pin to Network III	\$19.95

These cables have molded strain-relief plugs and round cables.

Since the Model III requires a flat RS-232C cable, we have also introduced an eight inch flat cable extender which will let you use the new longer cables with a Model III. The stock number of this item is 26-1497 (\$17.95).

Speaking of extenders, 26-1495 (\$29.95) is a Female to Female RS-232C adapter which will let you connect two RS-232C cables together.

The last new item is a null modem (26-1496 \$29.95). A null modem lets two computers, equipped with RS-232C capabilities, communicate without the use of modems.

Here at the Microcomputer News, we are constantly connecting our Model IIs to Models I and III. The reason for connecting the computers is to transfer software and articles from one to another, using either a combination of Host and Terminal programs, or SCRIPSIT and a Terminal program. The null modem lets us do this very well.

The following lines are available in the new Radio Shack RS-232-C cables and adapters.

Pin Numbers	Signals
1	Protective Ground
2	Transmit Data
3	Receive Data
4	Request to Send
5	Clear to Send
7	Signal Ground
8	Carrier Detect
15	I/O Transmit S.E.T.
17	Receiver Clock
20	Data Terminal Ready
24	Transmit Clock

### CASSETTE PORT DEVICES

When we look at cassette port peripheral devices, we find four items: Cassette Recorders, Plug 'n Power™, the Direct Connect Modem I and the Digitizer.

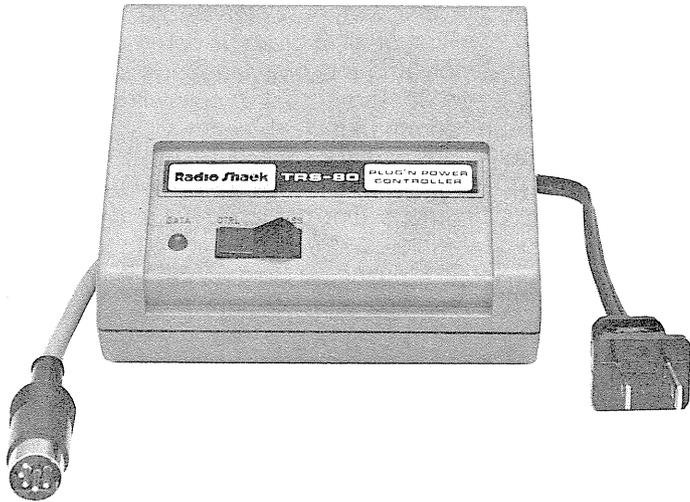
Both the Digitizer and the Direct Connect Modem I were built with special circuitry which allows owners of TRS-80 Model Is, who do not have the RS-232C serial interface, to enjoy the world of data communications. The special Cassette Software package (26-1139) and cable (26-3009) along with the Direct Connect Modem I give the Model I owner a useful package for communicating with data base systems such as CompuServe and Dow Jones.



CTR-80A Cassette Recorder

The Digitizer manual comes complete with a software routine which allows it to be used through the cassette port of a Model I.

The CTR-80A comes complete with the cable needed to connect to the cassette port of Model Is, IIIs, and Color Computers. One feature of the CTR-80A is the ability to Fast Forward or Rewind the cassette without the operator having to disconnect the motor control plug. If you already have a cassette recorder, cable 26-1207 will allow you to connect the recorder to a Model I, III or Color Computer.



#### TRS-80 Plug 'n Power Controller

To me, the Plug 'n Power controller is probably the most generally useful of the non-printer peripherals for TRS-80s. It can be used with Model Is, IIIs, and Color Computers to control electrical outlets, appliances, and lights throughout your home or business. I have one at home, and we use it to give the house that "occupied" look at all times. Our Color Computer is not always busy, especially if we are not home, and it easily manages the task of giving the house a lived in look without the fixed cycles of more conventional timers and other devices.

The TRS-80 Plug 'n Power controller is the most versatile of the Plug 'n Power controllers since it is not limited to a single house code. By being able to access all sixteen of the house codes as well as the sixteen module codes within each house code, you can set each room of the house to a different house code (assuming no conflicts with the neighbors) and then control individual lights or appliances within that room. This gives you maximum flexibility in working out the different control programs you will want to use both while you are at home, and while the computer is home alone.

## Peripheral Bugs, Errors and Fixes

### LINE PRINTER VII (26-1167)

Forms for the Line Printer VII: The forms capacity specification for the LP VII given in both the LP VII Owner's Manual and the November, 1981 Microcomputer News is

misleading. The LP VII will handle paper (single or multi-part) with no more than a total thickness of 6.8 mil., which usually means single part forms only. The LP VII will not feed generally available multi-part forms smoothly.

### Revisions, Enhancements, Etc.

The following are the stock numbers for the Computer Peripheral miscellaneous items. These items can be ordered through your local Radio Shack store.

700-2110 Model I Tandygraph disk for 26-1191	
Multi-pen Plotter	N/C
700-2111 Model II Tandygraph disk for 26-1191	
Multi-pen Plotter	N/C
700-2112 Model III Tandygraph disk for 26-1191	
Multi-pen Plotter	N/C
700-2113 Color Computer Tandygraph disk for 26-1191	
Multi-pen Plotter	N/C
700-3010 Line Printer V (26-1165) Head	
Cleaning Kit.	\$1.50

## Wind-Chill Factor

W. C. Clough, Jr.  
Education Director  
Radio Shack Computer Center  
Dallas, Texas

Just knowing the temperature during winter isn't enough, as the U. S. military discovered thirty years ago. Something is needed to reflect the effect of wind, which makes the human body lose heat in apparent proportion to the wind speed—in other words, the harder the wind blows, the colder it feels.

Hence the famous wind-chill factor. I know that April is a little late in the year for this information, but this is being written during one of the coldest Januaries in Texas so it seems appropriate at the moment.

Here is a program to supply the wind chill factor based on the current Fahrenheit temperature and the wind velocity in miles-per-hour. This version, written for the Model III, is easily translated to any machine using BASIC.

```
10 INPUT "TEMP (F): "; T
   : INPUT "WIND VELOCITY: "; V
   : IF V<4 THEN V=4
20 C=((10.45 +(6.686112 *SQR(V)) -(.447041 *V))
   /22.034 *(T -91.4)) +91.4
30 PRINT USING "WINDCHILL ###.###"; C
```

In the formula, T represents Fahrenheit temperature, V the wind velocity in m.p.h., and C the wind chill factor.

Chill factors produced by wind velocities below four m.p.h. are negligible, but the formula produces erroneous answers with any velocity less than four m.p.h., therefore a conditional statement is used in line 10 to set V to four if necessary.

Similar results are obtained with speeds above 45 m.p.h. For instance, zero degrees at 45 m.p.h. gives a -54 factor; at 60 m.p.h. the factor drops only one more degree.

# Moving Dots

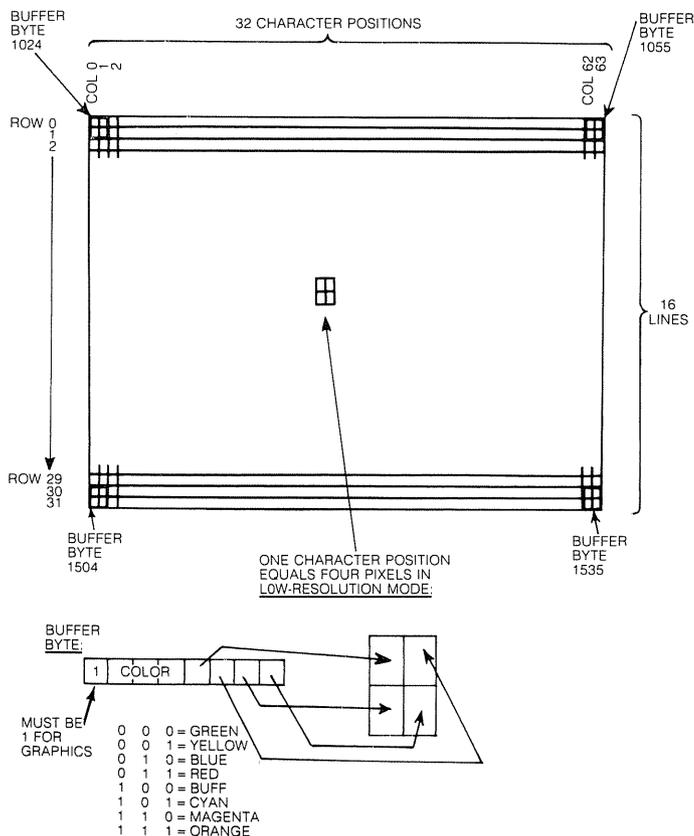
by William Barden, Jr.  
©William Barden, Jr.

Last month we showed you a SET/RESET subroutine that enabled us to set or reset any of the 2048 "pixels" that are present on the screen in the "low-resolution" graphics mode. This month we'll show you how SET/RESET may be used to program a "moving dot" for 16K Color Computer systems. The moving dot is exactly that—it moves at high speed in any of 16 directions and can be used to create graphics effects such as artillery projectiles or laser blasts. While we're looking at the moving dot code we'll explain in detail how we designed and implemented the program. This is a typical assembly language program—not too hard and not too easy.

## THE SET/RESET SUBROUTINE

For those of you that are reading your first issue of this newsletter, let's review SET/RESET, covered last month.

SET/RESET performs the same function as SET/RESET in Color BASIC—it sets or resets one of 2048 pixels. The 2048 pixels are arranged in 64 columns by 32 rows, as shown in Figure 1. Each pixel is addressed by an x (column) and y (row) coordinate.



Why implement another way of doing SET/RESET when we have a perfectly good way in Color BASIC? I thought you'd never ask . . . First of all, it gives us some practice in typical 6809E assembly and machine language code. Secondly, it allows us to do some neat things without being bounded by BASIC commands.

The assembly language code for SET/RESET is shown in Listing 1. The code is identical to the code given last month with three minor differences. One call to \$B4F4 and two calls to \$B3ED have been deleted. These calls were to the "integer/floating" conversion routines in Color BASIC ROM. They are the standard way of "passing arguments" to assembly language code and are described in the Extended Color BASIC manual. The x and y values were passed to SET and RESET and a color code was passed back to BASIC by calls to these ROM subroutines. In this month's example we're going to use an alternative way to pass arguments. The third change is the addition of the MOVEDT routine, which we will talk about later.

## Listing 1

```

3F00          00100      ORG      $3F00
00110 *****
00120 * MOVING DOT SUBROUTINE. MOVES ONE PIXEL IN ONE OF *
00130 * 16 DIRECTIONS UNTIL BOUNDARY COLOR ENCOUNTERED *
00140 * INPUT: $3FF0=START X *
00150 * $3FF1=START Y *
00160 * $3FF2=DIRECTION CODE. 0=0 DEGREES, *
00170 * CLOCKWISE IN 22.5 DEGREE INCR *
00180 * $3FF3, 4=DELAY COUNT *
00190 * $3FF5=RESERVED FOR COMPLETION CODE *
00200 * OUTPUT: $3FF0=COMPLETION X *
00210 * $3FF1=COMPLETION Y *
00220 * $3FF5=COMPLETION COLOR CODE *
00230 *****
3F00 F6 3FF2 00240 MOVEDT LDB $3FF2 GET DIRECTION CODE
3F03 58 00250 LSLB TIMES 2
3F04 4F 00260 CLRA NOW IN D
3F05 C3 3F3F 00270 ADDD #INCTAB POINT TO INCREMENT
3F08 1F 01 00280 TFR D,X NOW IN X
3F0A 20 1D 00290 BRA MOV060 BYPASS SET FIRST TIME
3F0C FC 3FF0 00300 LDD $3FF0 GET CURRENT X,Y
3F0F 34 10 00310 PSHS X
3F11 8D 4C 00320 BSR SET SET POINT
3F13 35 10 00330 PULS X
3F15 5D 00340 TSTB TEST COMPLETION
3F16 26 23 00350 BNE MOV090 GO IF BOUNDARY
3F18 10BE 3FF3 00360 LDY $3FF3 GET DELAY COUNT
3F1C 31 3F 00370 MOV050 LEAY -1,Y DECREMENT COUNT
3F1E 26 FC 00380 BNE MOV050 LOOP
3F20 FC 3FF0 00390 LDD $3FF0 GET CURRENT X,Y
3F23 34 10 00400 PSHS X
3F25 8D 51 00410 BSR RESET RESET POINT
3F27 35 10 00420 PULS X
3F29 B6 3FF0 00430 MOV060 LDA $3FF0 GET CURRENT X
3F2C AB 84 00440 ADDA ,X FIND NEXT X
3F2E B7 3FF0 00450 STA $3FF0 STORE
3F31 B6 3FF1 00460 LDA $3FF1 GET Y
3F34 AB 01 00470 ADDA 1,X FIND NEXT Y
3F36 B7 3FF1 00480 STA $3FF1 STORE
3F39 20 D1 00490 BRA MOV040 GOTO NEXT SET, RESET
3F3B F7 3FF5 00500 MOV090 STB $3FF5 STORE BOUNDARY COLOR
3F3E 39 00510 RTS RETURN
3F3F 00FF 00520 INCTAB FDB 255 0 DEGREES
3F41 01FE 00530 FDB 256+254 22.5
3F43 01FF 00540 FDB 256+255 45
3F45 02FF 00550 FDB 512+255 67.5
3F47 0100 00560 FDB 256+0 90
3F49 0201 00570 FDB 512+1 112.5
3F4B 0101 00580 FDB 256+1 135
3F4D 0102 00590 FDB 256+2 157.5
3F4F 0001 00600 FDB 0+1 180

```

```

3F51 FF02 00610 FDB 65280+2 205.5
3F53 FF01 00620 FDB 65280+1 225
3F55 FE01 00630 FDB 65024+1 247.5
3F57 FF00 00640 FDB 65280+0 270
3F59 FEFF 00650 FDB 65024+255 292.5
3F5B FFFF 00660 FDB 65280+255 315
3F5D FFFE 00670 FDB 65280+254 337.5
00680 *****
00690 * SET SUBROUTINE. SETS X,Y POINT IN LOW-RES MODE *
00700 * CALLED FROM BASIC WITH X,Y IN ARGUMENT 8,8 *
00710 *****
3F5F 8D 1F 00720 SET BSR ADDMSK FIND ADDRESS, MASK
3F61 1F 89 00730 TFR A,B SAVE MASK IN B
3F63 A4 84 00740 ANDA ,X TEST ALREADY SET
3F65 27 0A 00750 BEQ SET010 GO IF NOT SET
3F67 E6 84 00760 LDB ,X SET, FIND COLOR
3F69 C4 70 00770 ANDB #570 MASK OUT COLOR
3F6B 54 00780 LSRB ALIGN FOR 1-7 CODE
3F6C 54 00790 LSRB
3F6D 54 00800 LSRB
3F6E 54 00810 LSRB
3F6F 20 05 00820 BRA SET090 RETURN
3F71 EA 84 00830 SET010 ORB ,X SET PIXEL
3F73 E7 84 00840 STB ,X RESTORE
3F75 5F 00850 CLRB 0 FOR SET ACTION
3F76 4F 00860 SET090 CLRA CLEAR A FOR CONVERT
3F77 39 00870 RTS RETURN
00880 *****
00890 * RESET SUBROUTINE. RESETS ONE PIXEL IN LOW RES MODE*
00900 * CALLED FROM BASIC WITH X,Y IN ARGUMENT 8,8 *
00910 *****
3F78 8D 06 00920 RESET BSR ADDMSK FIND ADDRESS, MASK
3F7A 43 00930 COMA FLIP MASK FOR RESET
3F7B A4 84 00940 ANDA ,X RESET PIXEL
3F7D A7 84 00950 STA ,X RESTORE BYTE
3F7F 39 00960 RTS RETURN
00970 *****
00980 * FIND LOW-RES ADDRESS, MASK *
00990 * INPUT: A,B=X,Y *
01000 * OUTPUT: X=BUFFER ADDRESS *
01010 * A=MASK *
01020 *****
3F80 34 06 01030 ADDMSK PSHS D SAVE X,Y
3F82 54 01040 LSRB FIND START OF LINE
3F83 86 20 01050 LDA #32 32 IN A, Y IN B
3F85 3D 01060 MUL START OF LINE
3F86 C3 0400 01070 ADDD #5400 ADD START OF BUFFER
3F89 34 06 01080 PSHS D NOW ON STACK
3F8B E6 62 01090 LDA 2,S GET X
3F8D 4F 01100 CLRA NOW IN D
3F8E 54 01110 LSRB X/2
3F8F E3 E1 01120 ADDD ,S++ BUFFER+LINE ST+X/2
3F91 1F 01 01130 TFR D,X NOW IN X
3F93 A6 E4 01140 LDA ,S GET X
3F95 43 01150 COMA COMPLIMENT BIT
3F96 84 01 01160 ANDA #1 FIND ODD/EVEN
3F98 4C 01170 INCA EVEN TO 10, ODD TO 01
3F99 A7 E4 01180 STA ,S SAVE
3F9B E6 51 01190 LDB 1,S GET Y
3F9D C4 01 01200 ANDB #1 GET ODD/EVEN
3F9F 26 02 01210 BNE ADD090 GO IF ODD (MASK=10 OR 01)
3FA1 48 01220 LSLA ALIGN (1000 OR 0100)
3FA2 48 01230 LSLA
3FA3 32 62 01240 ADD090 LEAS 2,S RESET STACK
3FA5 39 01250 RTS RETURN
00000 01260 END
00000 TOTAL ERRORS

```

SET and RESET in this case will be "called" from the MOVEDT subroutine with the x and y values in the A and B registers, respectively. (If you're not familiar with the A and B registers, see the introductory column in this series two months previous, or review the introductory material in a good book on 6809 programming.)

SET sets the specified pixel by turning on one of the four bits in the byte associated with the "character position" containing the pixel. In this "low-resolution" graphics mode, each of the 512 character positions on the screen is set to graphics mode by turning on the most significant bit, and putting a "color code" of 0 through 7 into the next three bit positions, as shown in Figure 1.

Initially, the entire screen should be cleared by POKES of 128 + color \* 16 from locations 1024 through 1535. This sets all character positions to graphics and stores a color code into each character position. The four pixels associated with each character position are turned off by being set to zero. Subsequent SETs will turn on a pixel by storing a one in one of the four bit positions and RESETs will turn off the pixel by storing a zero. The four pixels of each character position must all be the same color.

## THE MOVING DOT WRITES AND HAVING WRIT . . .

Now that we have a SET/RESET subroutine, how can we implement a "moving dot" subroutine in assembly language? In theory, it's easy—just turn on a pixel, wait an interval, turn off the pixel and then turn on the next pixel, as shown in Figure 2. However, there are some problems that come to mind. Solving these problems constitutes a major part of any assembly language programming task. Coding is relatively easy once the problem is defined.

Step 1. Turn on Pixel



Step 2. Turn off Pixel, Wait



Step 3. Turn on Next Pixel

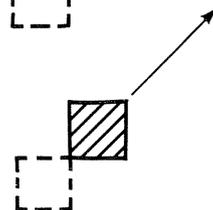
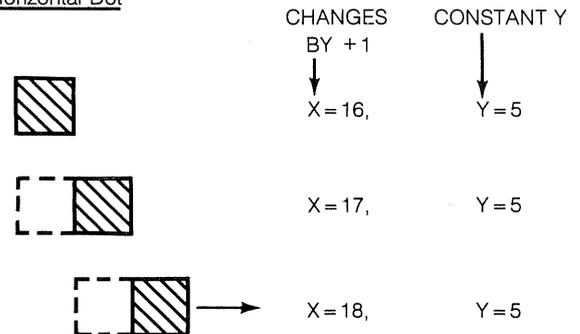


Figure 2. Moving Dot Action

The first obvious problem is that of figuring out where the next pixel is. Horizontal and vertical lines are easy—the next pixel is directly above, to the sides, or below. All we have to do in this case is to "bump" the x or y value by one or minus one to give the next position, and do the SET/RESET, as shown in Figure 3.

### Moving Horizontal Dot



### Moving Vertical Dot

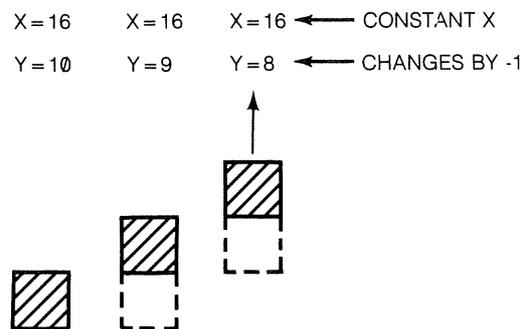


Figure 3. Moving Dot for Horizontal and Vertical Lines

Diagonal lines are also not too difficult. In this case we can bump both x and y to get the new position, as shown in Figure 4.

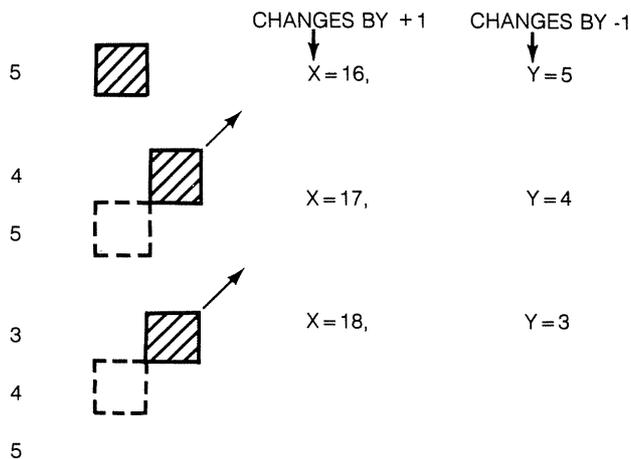


Figure 4. Moving Dot for Diagonal Lines

Horizontal, vertical, and diagonal lines, however, don't give us a great deal of flexibility. Ideally we'd like to be able to move a dot over any angle.

As a practical matter, however, we can't easily implement trigonometric functions such as SINE and COSINE in assembly language code. (To do so we'd need a major chunk of "floating-point" subroutine code, allowing us to work in numbers other than integers. We'll keep this subject for the April, 1989 issue.) As a compromise we'll allow only angles that are multiples of 22.5 degrees—22.5 degrees, 45 degrees, 67.5 degrees, and so forth. That will give us 16 angles, from 0 degrees (towards top) clockwise to 337.5 degrees, as shown in Figure 5.

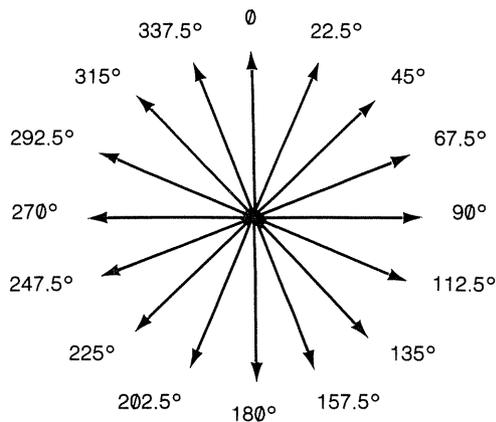


Figure 5. MOVEDT Angles

The angles that we haven't covered—22.5 degrees, 67.5 degrees, 112.5 degrees, and so forth, can be generated by setting and resetting a pixel, and by bumping the x and y values by 1 or 2, as shown in Table 1. The table gives the increments for moving a dot in any of 16 directions.

This table is held in the MOVEDT program as a 32-byte table and is the heart of the subroutine. The table is located at

Index	Degrees	X Increment	Y Increment
0	0	0	-1
1	22.5	1	-2
2	45	1	-1
3	67.5	2	-1
4	90	1	0
5	112.5	2	1
6	135	1	1
7	157.5	1	2
8	180	0	1
9	202.5	-1	2
10	225	-1	1
11	247.5	-2	1
12	270	-1	0
13	292.5	-2	-1
14	315	-1	-1
15	337.5	-1	-2

Table 1. Increment Table (INCTAB)

"INCTAB" (see Listing 1). There are 16 "entries" arranged to correspond to the angles of 0, 22.5, 45, 67.5 degrees, and so forth, up to 337.5 degrees. Each entry consists of an increment for x in the first byte, followed by an increment for y in the second byte.

Each increment byte is a "signed" value. The increment represents an 8-bit number of +1 (01), +2 (02), -1 (FF), or -2 (FE). We could have arranged the table as 32 bytes, each with a "FCB" or "Form Constant Byte" "pseudo-op." To save paper, however, we made each one a "Form Double Byte" or "FDB" pseudo-op. (Don't let the term "pseudo-op" throw you. It's anything other than an instruction operation code mnemonic. Pseudo-ops are used to tell the assembler to do things like setting the origin of the program, to generate tables such as INCTAB, or to END the assembly language statements.)

The FDB pseudo-ops generate two bytes of data for each operand. We've used a somewhat odd form for the operand to get the proper 8-bit values for x and y. 65024 (our value for x = -2) is really 11111110XXXXXXXX in binary, where XXXXXXXX is the y value. Likewise, 65280 represents a -1 in the x value, 256 a +1 in the x value, and 512 a +2 in the x value. Values of 255 and 254 are really -1 and -2 for y. If you're confused about this, look at the hexadecimal values in the third column. A value of 01FE, for example, represents an x increment of +1 and a y increment of -2.

Two other problems that we must resolve before coding are these: Where should the dot start and how far should it travel?

Answering the first problem is easy—it'll start where we tell it to start by an x and y coordinate. The x,y coordinate will be passed to the assembly language code as an "argument".

The second question could be handled by several options. We could specify a length of travel and count pixels. However, let's make the assumption that the MOVEDT subroutine will be used for "zapping" things in games. In this case it would be nice to have the dot stop when it encountered a target or boundary. Fortunately for us, the SET subroutine returns a color code if the pixel is already set, and we can use this feature to detect a target or boundary.

At this point we've blocked out how the MOVEDT should work: We'll call it from BASIC with a starting x,y and direction. MOVEDT will then move the dot along the specified angle until a target or boundary is encountered. Is there anything we've forgotten?

One thing could be added. We might put in a parameter to vary the delay between resetting the old dot and writing a new dot. The speed of a ping-pong ball is quite different from a tank shell, and this parameter would allow us to simulate both.

The final parameters that are passed between BASIC and MOVEDT are shown in Figure 6. They are located near the top of 16K memory in the \$3F00 area. (The "\$" stands for "hexadecimal number following".)

\$3FF0	START X	0-63
\$3FF1	START Y	0-31
\$3FF2	DIRECTION CODE	0-15
\$3FF3	DELAY	0-65,535
\$3FF4	COUNT	
\$3FF5	COMPLETION COLOR CODE	0 (NORMAL), 1-7 (BOUNDARY)

Figure 6. MOVEDT Parameters

The x parameter may be 0 through 63 and can be held in one byte at \$3FF0. The y parameter may be 0 through 31 and is held in one byte at \$3FF1. The direction code is a value from 0 through 15 and is held in \$3FF2. The delay count is a 16-bit value of 0 through 65,535 and is held in two bytes at \$3FF3 and \$3FF4. The greater the value, the longer the delay. Location \$3FF5 is reserved for the color code when the target or boundary is encountered. This color code is passed back as a value of 0 through 7. The code is one less than the Color BASIC color code values.

## OPERATION OF MOVEDT

Now that we've solved some of the basic design problems, we can list the steps in MOVEDT. In larger programs we might do this as a "flowchart".

The basic operation of MOVEDT is this:

1. Use the direction code to find the increment from the INCTAB.
2. Call SET to set the next point. Test the code returned. If it is 0, the pixel was set. If it is non-zero, the pixel was in a target or boundary. In this case, return to BASIC with the color code.
3. Use the delay count to "waste time".
4. Reset the point.
5. Add the x,y increment from INCTAB to adjust the old point location to the new point location.
6. Go back to step 2.

## DOWN TO THE NITTY GRITTY CODE

Let's see how these steps are done in MOVEDT.

First of all, the direction code is loaded into the B register from location \$3FF2. Now, the B register is shifted left one bit position by a "Logical Shift Left." This multiplies the direction code by 2, as each INCTAB entry is 2 bytes long. The A

register is now cleared by CLRA. At this point we've got twice the direction code in A and B. A and B taken together represent the D register.

The ADDD instruction adds the contents of D to the immediate value of INCTAB. The immediate value is held in the instruction itself (\$3F3F) and represents the location of "INCTAB" (see INCTAB location). The result (in D) is the actual location of the INCTAB entry for the increment values. A TFR transfers the location from D to the X index register. This location "pointer" will remain in X throughout the remainder of MOVEDT.

The code from MOV040 to one instruction before MOV090 is the "main loop" of MOVEDT.

The current x,y value is loaded into the D register at MOV040. The current value of x,y is always located at locations \$3FF0 and \$3FF1; these locations are used as "working storage." Now a call to SET is done. Before the call, however, a PSHS X is done. This saves the contents of X (INCTAB location) in the hardware stack. This action is necessary as SET uses the X register. After the BSR to SET, the X register is restored by a PULS X.

The PSHS and PULS used the "stack area" in RAM. The 6809E uses two stacks, a "hardware" stack, designated by S, and a "user" stack, designated by U. The S stack is used to automatically store return addresses after a BSR (or LBSR or JSR) or for temporary storage of any registers by a PSHS. The U stack may be used by the programmer for any purpose. In this case we've only used the S stack. The S register points to the stack area. As the BASIC interpreter also uses the S stack, the S register points to a valid stack area on entry to MOVEDT. As a matter of fact we must use (or at least restore) the S register so that the final "Return from Subroutine" instruction properly returns to the BASIC interpreter.

After the call to SET, the B register contains a code of 0 or 1 through 7. If the code is 0, the SET operated without a hitch. If the code is 1 through 7, the pixel to be set was already set, indicating a target or boundary. The TSTB tests B and if the contents of B are not equal to zero (BNE), a return to BASIC is made at MOV090 after first storing the target or boundary color code in \$3FF5.

Next (MOV050), the delay count is picked up from \$3FF3 and put into the Y register. The delay count is a 16-bit value of 0 through 65,535. The LEAY -1,Y instruction takes the contents of Y, adds -1 to it, and puts the result back into Y. If Y is not equal to zero, a loop is made back to MOV050. When Y finally becomes zero, the LDD \$3FF0 instruction is executed. Note that a value of 0 is the **longest** delay. This is true because the BNE is done **after** the decrement. At this point, the value in Y is 65,535 if it was originally 0.

The next four instructions repeat the load of x,y, the push of x, the call (to RESET), and the pull of x. The RESET resets the pixel just set before the delay.

The code at MOV060 is the code to increment x,y to find the next pixel. This is done as two 8-bit adds. If we did a 16-bit add here (via ADDD) we might get an erroneous result as a spurious carry occurred to the upper 8 bits in A. First, the INCTAB value for x is added to A. The result is stored back in \$3FF0. Next, the INCTAB value for y is added to A. The result is stored back to \$3FF1. Both of these adds used the X register in an "indexed addressing" mode where the second operand for the add was pointed to by the X index register.

The BRA (Branch Always) branches back to MOV040 for the next set, reset operation.

## USING MOVEDT

To use MOVEDT, first CLEAR 200, &H3EFF. This protects the area in high memory that will be used for MOVEDT. Next, use the program shown in Listing 2 to relocate the machine language form of MOVEDT into the \$3F00 area. Once this relocation is done, you won't need to repeat the operation.

### Listing 2

```
100 DATA 246, 63, 242, 88, 79, 195, 63, 63
130 DATA 31, 1, 32, 29, 252, 63, 240, 52
160 DATA 16, 141, 76, 53, 16, 93, 38, 35
190 DATA 16, 190, 63, 243, 49, 63, 38, 252
220 DATA 252, 63, 240, 52, 16, 141, 81, 53
250 DATA 16, 182, 63, 240, 171, 132, 183, 63
280 DATA 240, 182, 63, 241, 171, 1, 183, 63
310 DATA 241, 32, 209, 247, 63, 245, 57, 0
340 DATA 255, 1, 254, 1, 255, 2, 255, 1
370 DATA 0, 2, 1, 1, 1, 1, 2, 0
400 DATA 1, 255, 2, 255, 1, 254, 1, 255
430 DATA 0, 254, 255, 255, 255, 255, 254, 141
460 DATA 31, 31, 137, 164, 132, 39, 10, 230
490 DATA 132, 196, 112, 84, 84, 84, 84, 32
520 DATA 5, 234, 132, 231, 132, 95, 79, 57
550 DATA 141, 6, 67, 164, 132, 167, 132, 57
580 DATA 52, 6, 84, 134, 32, 61, 195, 4
610 DATA 0, 52, 6, 230, 98, 79, 84, 227
640 DATA 225, 31, 1, 166, 228, 67, 132, 1
670 DATA 76, 167, 228, 230, 97, 196, 1, 38
700 DATA 2, 72, 72, 50, 98, 57
730 FOR I=16128 TO 16293
760 READ A
    : POKE I, A
790 NEXT I
```

You're now set to "interface" from your BASIC program to MOVEDT. MOVEDT is designed to operate with a "black" screen for any unused areas. Boundaries and targets may be any color except green (a 0 color code conflicts with a normal return from MOVETB).

Before using MOVETB, clear the entire screen by doing POKES of locations 1024 through 1535. Use the following code for the POKES:  $128 + c * 16$ . The 128 sets each character position byte to "graphics" and the  $c * 16$  stores a color code of 1 through 7 (yellow, blue, red, buff, cyan, magenta, orange).

Draw a boundary of a color that will not be used for targets. The boundary will terminate any line of moving dots if they do not hit a target.

Listing 3 shows a demonstration program for MOVEDT. Line numbers 850 through 910 clear the screen and set each character position byte to code 7, or orange ( $128 + 7 * 16 = 240$ ).

### Listing 3

```
100 DATA 246, 63, 242, 88, 79, 195, 63, 63
130 DATA 31, 1, 32, 29, 252, 63, 240, 52
160 DATA 16, 141, 76, 53, 16, 93, 38, 35
190 DATA 16, 190, 63, 243, 49, 63, 38, 252
220 DATA 252, 63, 240, 52, 16, 141, 81, 53
250 DATA 16, 182, 63, 240, 171, 132, 183, 63
280 DATA 240, 182, 63, 241, 171, 1, 183, 63
310 DATA 241, 32, 209, 247, 63, 245, 57, 0
340 DATA 255, 1, 254, 1, 255, 2, 255, 1
370 DATA 0, 2, 1, 1, 1, 1, 2, 0
400 DATA 1, 255, 2, 255, 1, 254, 1, 255
430 DATA 0, 254, 255, 255, 255, 255, 254, 141
460 DATA 31, 31, 137, 164, 132, 39, 10, 230
490 DATA 132, 196, 112, 84, 84, 84, 84, 32
520 DATA 5, 234, 132, 231, 132, 95, 79, 57
550 DATA 141, 6, 67, 164, 132, 167, 132, 57
580 DATA 52, 6, 84, 134, 32, 61, 195, 4
610 DATA 0, 52, 6, 230, 98, 79, 84, 227
```

```
640 DATA 225, 31, 1, 166, 228, 67, 132, 1
670 DATA 76, 167, 228, 230, 97, 196, 1, 38
700 DATA 2, 72, 72, 50, 98, 57
730 FOR I=16128 TO 16293
760 READ A
    : POKE I, A
790 NEXT I
820 DEFUSR0=&H3FF0 'POKE 275, 63 POKE 276, 0 FOR
    COLOR BASIC
850 FOR I=1024 TO 1024+511
880 POKE I, 240
910 NEXT I
940 FOR I=0 TO 31
970 POKE 1024+I, 255
1000 POKE 1504+I, 255
1030 NEXT I
1060 FOR I=1 TO 14
1090 POKE 1024+I*32, 255
1120 POKE 1024+31+I*32, 255
1150 NEXT I
1160 T=10000
1180 D=RND(16)
    : IFD=16 THEN D=0
1210 POKE 16368, 32
    : POKE 16369, 16
1240 POKE 16371, INT(T/256)
    : POKE 16372, 0
1270 POKE 16370, D
1300 A=USR0(0) 'USR(0) FOR COLOR BASIC
1330 GOTO 1180
```

Lines 940 through 1030 draw the top and bottom boundary lines. Lines 1060 through 1150 draw the side boundary lines. The values used in the POKES are 240 for the graphics bit and orange, plus 15 to turn all pixels on.

The time delay value used is  $INT(10000/256)$ . This is POKEd into location &H3FF3, while a zero is POKEd into location &H3FF4.

The starting x,y is 32,16, approximately the middle of the screen. These values are POKEd into locations \$3FF0 and \$3FF1.

The direction is generated by  $D = RND(16)$ . A value of 16 is converted to 0, so that the direction code is 0 through 15. D is POKEd into location &H3FF2.

The USR0 call calls MOVEDT and returns after the dot has encountered the boundary. A GOTO 1180 generates random lines of moving dots that emanate from the center of the screen outward to the boundary.

Change T to observe the effects of different time delays.

## NO WONDER GAME PROGRAMMERS MAKE SO MUCH MONEY!

This is a simple example of what can be done to implement some special graphics features by assembly language. What we've created certainly isn't a complete games package, but it might show you how assembly language subroutines might be strung together to create the high-speed effects found in many games. Some excellent high-speed games can be designed with a dozen or so "modules" as we've considered here. In future columns we'll look at additional techniques for graphics, and maybe you, too, can MAKE BIG MONEY! in computer games . . .

Next month we'll look at some additional assembly language topics. In the meantime, I'd recommend getting the Editor/Assembler for the Color Computer if you don't already have it. It is possible to do hand assembly of machine language programs for the 6809, but it's very tedious. It's best to save your energy for solving those design problems and for debugging the code!

# SCRIPSIT™ Ideas

This is the third article in a row written by the Model I/III Software Support Group. We hope you are not getting tired of hearing from us. If you are, perhaps it will be some comfort to you to know that we are not scheduled to write another one until September. If I had been allowed to use SCRIPSIT while a freshman in college, the composition would not have been any easier, but at least the typing of the papers would have been eased considerably. Speaking of SCRIPSIT, that is the program that will be featured this month. (How's that for transition?)

SCRIPSIT is one of our most popular programs, as perhaps you've found if you tried to order it early last year. We definitely had a hard time keeping up with the demand for it. After receiving my own copy and using it for several months, I have become an enthusiastic user. As you must expect, I am using SCRIPSIT to write and edit this article. Sometimes we hear about people using SCRIPSIT to do things that it wasn't originally intended to do, and sometimes it does them well, and other times, it doesn't do them at all. It is this second group that we wish we could help, but there are a few limitations.

We would like to share with you some of the things that we have found to be easier with SCRIPSIT. But no matter how hard you try to read the manual and listen to the training tapes, if you don't jump in with both feet, you'll probably never use the program to its fullest potential. I suggest trying out various combinations to see what can be done. As an example, recently I have been working on the writing of a play (perhaps skit would better describe it) with a group of friends. When they found out that I had access to a computer and Word Processing program, guess whose job it became to type and revise it. Fortunately, we only get together once a week, so I didn't have to reprint it on a daily basis. I don't think it takes much imagination to see that it's easier to make only corrections and reprint than it is to re-type an entire play (about 20 pages) every week.

While working on the play, I wanted to have the character's name on the left, and all text indented. Naturally, though, the player's lines were longer than the document lines, and I got the automatic "wrap-around," which went back to the left margin. After finishing the typing of the text, I went back through to edit it for a pleasant visual effect. The procedure I used was to place the cursor on the first character at the left margin, type `<CTRL><S><CTRL><X>` to insert a line, `<CTRL><TAB>` to tab the cursor to the position I wanted, followed by `<CLEAR>` to bring the text back to the cursor. I might point out that if the indentation is very large, it may take more than one time to move it all the way over. The first time will move the spaces on the right side over to the margin, and subsequent repetitions will move one word at a time down to the next line. With practice, you will get so fast that you will think you've been doing it for years. (Note: This procedure should work for outlines, also. One other precaution - make sure that the video width matches the printer margins, other-

wise the spaces will come in the strangest places! So now you want to know how to make the video width change? Use `<Break> W = n` where n is the desired number of characters per line.)

During the Thanksgiving Holidays, I took a computer with me to my parents' home. After all our guests had left, my mother asked me if the computer would store and print recipes. I decided to try it, and it works for that, too.

We began by typing in the recipe just as we wanted it to appear, but we did not include any format information, and we ended each recipe with a page boundary marker (make sure to use `<CLEAR>` after the page marker, before saving). After saving several recipes, we exited and went to BASIC. Since I had a Line Printer VIII with me, we sent an `LPRINT CHR$(27);CHR$(56)` to the printer to cause it to print eight lines to the inch instead of the usual 6, and then sent `LPRINT CHR$(27);CHR$(20)` to select the condensed (16.7 char/in) print mode. (If you're using a Line Printer V, the commands are slightly different. Use `LPRINT CHR$(27);CHR$(14)` to set condensed print. For other printers, check your manual to see if condensed print and line feeds are possible, and how to set them.) We then went back into SCRIPSIT and created a "Format Document" which included the following format line:

```
>TM = 3 BM = 22 PL = 24 LM = 5 RM = 78
```

The only two items which are worth mentioning here are the Page Length of 24 and the Right Margin of 78. These correspond to the size of a 3x5 index card (with the printer set as previously defined). We then "Chain Loaded" several recipes together and printed out the recipes, then cut the paper to 3x5 and placed them into some clear plastic recipe sleeves.

Another concern, at times, is making the name of the document (in this instance, a recipe) conform to TRSDOS filename requirements. (If you're using tape, this won't be a problem at all. Just label the cassette.) We had fun trying to think of a descriptive, one-word name with not more than eight letters. Some examples: Green Bean Casserole became GRBEAN and Pumpkin Ice-Box Pie became PUMIBPIE. (Please, don't call us to ask for these recipes, they are standard recipes, nothing special.)

If you have come up with an interesting use for SCRIPSIT, one that you feel would be of interest to other readers, we would like to hear from you. If we hear of enough interesting uses to warrant it, we will use them in a future article.



## Computer Customer Services Address and Phone Numbers

8AM to 5PM 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

## Notes From Computer Customer Services

Computer Customer Services recently supplied us with the following Color Computer Routines which they thought you might be interested in. Please test these programs carefully in your computer and with your programs. These routines are not "official" Radio Shack programs. Disclaimer: Radio Shack does not support the use of these programs. Use of any of these programs is left strictly to the user's discretion and Radio Shack is not responsible for any results they may produce.

### COLOR COMPUTER TOP-OF-FORM

```

100 '*****
*   COLOR COMPUTER   *
*   TOP OF FORMS    *
*   DRIVER          *
*   ( 12 = TOF )    *
*****
110 '*****
* 1) PROTECT MEMORY *
* 2) ENTER AND RUN PROGRAM AS WRITTEN *
* 3) TYPE <N><E><W> <ENTER> *
* 4) THE START ADDRESS PLUS 32 IS THE LINE*
120 ' COUNTER. TO CLEAR THE LINE COUNTER *
* POKE THAT ADDRESS WITH A 255. EX: *
* IF YOU LOADED THE DRIVER AT 10000 *
130 '* THEN TO CLEAR THE LINE COUNTER YOU *
* WOULD EXECUTE THE FOLLOWING: *
* POKE 10032, 255 *
* 5) THE START ADDRESS PLUS 56 CONTROLS *
140 '* THE NUMBER OF LINES TO BE PRINTED PER*
* PAGE. *
* 6) THE START ADDRESS PLUS 73 CONTROLS *
* THE PAGE LENGTH. *
* 7) DEFAULT VALUES: *
150 '* LINES TO BE PRINTED PER PAGE = 60 *
* LINES PER PAGE = 66 *
*****
160 CLS
170 INPUT "START ADDRESS"; START

```

```

180 FOR ADDRESS = START TO START + 116
190 READ CODE
200 POKE ADDRESS, CODE
210 NEXT ADDRESS
220 DEFUSR0 = START
230 A = USR0(0)
240 END
250 DATA 052, 018, 182, 001, 103, 190, 001, 104,
167, 141, 000, 102, 175, 141, 000, 099, 134,
126, 048, 141, 000, 011, 183, 001, 103, 191,
001, 104
260 DATA 053, 146, 018, 018, 255, 052, 022, 198,
254, 209, 111, 038, 060, 129, 012, 039, 014,
013, 156, 038, 052, 108, 140, 236, 230, 140,
233, 193
270 DATA 060, 037, 042, 190, 001, 104, 175, 140,
221, 048, 141, 000, 045, 191, 001, 104, 198,
066, 224, 140, 211, 231, 140, 208, 134, 013,
198, 254
280 DATA 215, 111, 173, 159, 160, 002, 106, 140,
195, 038, 241, 048, 140, 188, 191, 001, 104,
053, 022, 129, 012, 038, 007, 134, 008, 173,
159, 160
290 DATA 002, 079, 018, 018, 018

```

```

100 '*****
1) ENTER AND RUN THE PROGRAM AS WRITTEN
2) TYPE <N><E><W><ENTER>
3) ADDRESS 1013 CONTROLS SPEED
(0=NORMAL, 255=SLOW)
*****
200 DATA 052, 016, 142, 000, 001, 048, 031, 038,
252, 053, 016
210 FOR X=1010 TO 1020
220 READ A
230 POKE X, A
240 NEXT X
250 POKE 1021, PEEK(359)
260 POKE 1022, PEEK(360)
270 POKE 1023, PEEK(361)
280 POKE 359, 126
290 POKE 360, 003
300 POKE 361, 242

```

CAUTION: I/O functions, including Keyboard input and Tape saves and loads may not function with this routine. Test this routine carefully before using it!

### COLOR COMPUTER DIRECTORY TO PRINTER

```

100 '* THIS PROGRAM PUTS THE DRIVE 0 DIRECTORY
OUT TO THE PRINTER
110 CLS
120 INPUT "DISK NAME"; N$
130 PRINT #-2, N$
140 PRINT #-2
150 CLEAR 2000
160 T=17
: S=3
170 DSKI$ 0, T, S, A$, B$
180 FOR X=1 TO 100 STEP 32
190 FOR Y=0 TO 12
200 C$=MID$(A$, X+Y, 1)
210 IF C$=CHR$(255) AND Y=0 THEN FOR L=1 TO 4
: PRINT #-2
: NEXT L
: END
220 IF Y=0 AND C$=CHR$(0) THEN 330
230 IF Y=8 THEN PRINT #-2, " ";
240 IF Y<11 THEN PRINT #-2, C$;
250 IF Y<>11 THEN 300
260 IF C$=CHR$(0) THEN PRINT #-2, " BASIC ";
: GOTO 300
270 IF C$=CHR$(1) THEN PRINT #-2, " DATA ";
: GOTO 300
280 IF C$=CHR$(2) THEN PRINT #-2, " OBJECT";
: GOTO 300
290 PRINT #-2, " SOURCE";
300 IF Y<>12 THEN 320

```

(Continued on Page 18)

# Network Expansion

*How TRS-80 Owners Will Benefit*

*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.*

CompuServe Incorporated has given birth to a third major business division which will allow the CompuServe Information Service (CIS) to be accessed in an ever increasing number of U.S. cities.

CompuServe now offers the communications services of its value-added network.

The CompuServe telecommunications network will be available for the first time to commercial customers not using the company's computers in a time sharing arrangement.

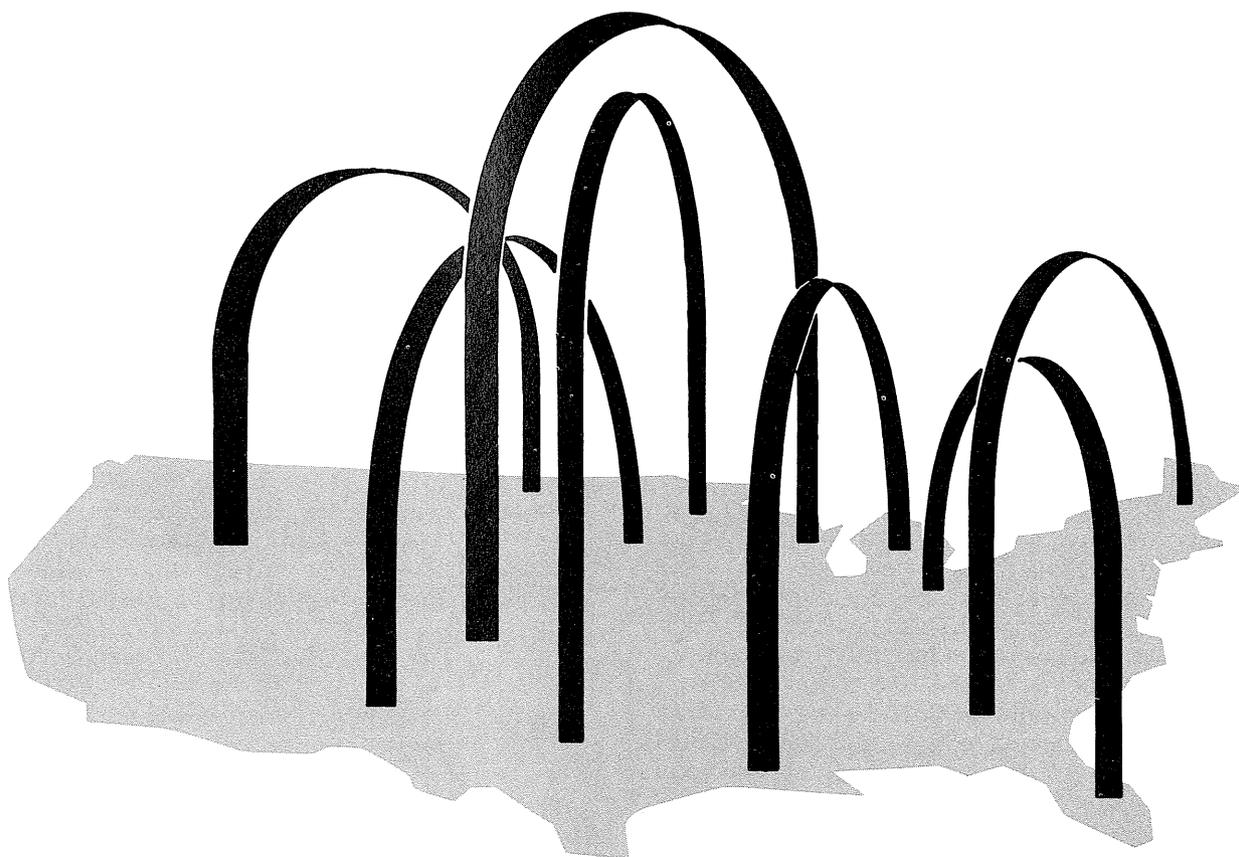
As the network reaches new cities, CompuServe Information Service customers can dial directly into the service, saving surcharges associated with alternate access through a supplementary network.

The CompuServe network has already expanded to more than 60 U.S. cities. Plans call for the network to reach 120 cities by April 1982 and 300 cities by the end of 1983.

That expansion will greatly reduce the number of CompuServe Information Service customers who now must dial in through a TYMNET network telephone number. TYMNET is the common carrier telecommunications network used as a "back up" or supplementary network to the CompuServe network. CIS customers who dial in through the CompuServe network save the \$2 an hour communication surcharge associated with dialing in through a TYMNET telephone number.

CIS customers consistently tell the customer service department that access through a CompuServe number is more reliable than through supplementary network numbers.

The reliability stems from the years of development and technological innovation which has gone into the CompuServe telecommunications network. CompuServe began building the network in 1972 to link its many commercial customers to its computers in Columbus, Ohio. Today, more than 600 commercial customers use the network to



access CompuServe's computers and the many value-added software programs available. CompuServe's commercial customers include one-fifth of the Fortune 500 companies, large financial institutions, mining companies and government agencies.

In addition to years of networking experience, the CompuServe network's reliability also comes as a part of the physical construction of the network. It has progressed from direct lines to multiplexers to packet technology to the current highly effective and efficient virtual circuit network.

One of the unique aspects of the CompuServe network is enjoyed by CIS customers who utilize our home banking service.

United American Bank customers access CIS and are switched to the bank's computers in Knoxville, Tenn. by the CompuServe network. The switching activity is transparent to the banking customer and allows the customer to conduct banking activities within the security of the bank's own system. The network switches the customer back to the CompuServe Information Service when he is finished.

CompuServe's network consists of many small to medium scale computer systems, strategically located across the country, which communicate data over leased lines and satellite circuits. These MicroNode computers, designed, built and maintained by CompuServe's own data communications staff, communicate with each other and with other host computer systems.

The CompuServe network's dynamic block size assures maximum throughput on transmission lines of varying quality, and, unlike other networks that depend on centralized control processes, the network's distributed intelligence guards the network against any large geographic outages.

The CompuServe network's interconnection "ring" configuration provides reliable computer access from remote locations in a truly effective and efficient manner. The benefit gained from this flexible configuration, combined with intelligent alternate routing, is the most dependable service in the industry.

Each MicroNode in the CompuServe network has its own Uninterruptible Power Supply system to ensure continued operation. The MicroNodes are also equipped with diagnostic control microprocessors and separate maintenance circuits. All MicroNodes feature multiple paths into the network, error detection and retransmission software and down-line software updating. The CompuServe network also maintains alternate routes.

All this adds up to a documented reliability rating of between 99.6 and 99.8 percent for "uptime" on the CompuServe system. CompuServe's computer technology and advanced networking knowledge together have achieved that remarkable statistic.

CIS customers should read the telephone number update section of the monthly Update newsletter to find out when CompuServe access has reached your city. Current telephone access information is also available through CompuServe User Information under "CIS Telephone Access Numbers," or type Go CIS-174.

*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.*

# Model I Head Cleaner Program, Version 2

Lowell Johnson  
JOHNSON CAR WASH COMPANY  
106 Belleville Court  
Thief River Falls, MN 56701

Although the idea of your head cleaning diskettes is very good, I questioned using the "DIR" command to operate the disk drive. To operate the recommended 30 seconds it is necessary to "unlock" the computer with the reset button and repeat the procedure described in the instructions. Also, the head will travel to track 17 and remain there, using only a small part of the cleaning diskette's surface.

The enclosed program will step the head of the selected drive from track 0 to 34 and back, and will operate the drive for approximately 30 seconds. It will operate only on the Model I, as I could find no source for the disk controller addresses on the Model III.

```
1000 'SUBMITTED BY: LOWELL JOHNSON
      THIEF RIVER FALLS MN
1010 CLS
1020 PRINT @ 7, " DRIVER PROGRAM FOR HEAD
      CLEANING DISKETTES"
1030 PRINT @ 137, "REMOVE PROGRAM DISKETTES FROM
      ALL DRIVES"
1040 DR$=" "
1050 DN=0
1060 PRINT @ 393,;
1070 INPUT "DRIVE NUMBER (0-3) OR <ENTER> TO
      QUIT"; DR$
1080 IF DR$=" " THEN CLS
      : END
1090 IF VAL(DR$)>3 THEN 1060
1100 DR=2 [ VAL(DR$)
1110 PRINT @ 595, "HEAD IN POSITION ";
1120 F=0
1130 L=34
1140 S=1
1150 GOSUB 1220
1160 IF DN=1 THEN 1010
1170 F=34
1180 L=0
1190 S=-1
1200 GOSUB 1220
1210 GOTO 1010
1220 FOR X=F TO L STEP S
1230 PRINT @ 612, X
1240 POKE &H37E0, DR
1250 POKE &H37EF, X
1260 POKE &H37EC, 18
1270 IF PEEK(&H37EC)<2 THEN 1320
1280 FOR Y=1 TO 110
1290 NEXT Y
1300 NEXT X
1310 RETURN
1320 PRINT @ 785, "DEVICE NOT AVAILABLE !!";
1330 DN=1
1340 FOR Y=1 TO 2900
1350 NEXT
1360 RETURN
```

# Dow Jones Sports Data Base

And for you sports fans,  
Let's go to the video display!  
Maybe you'd like to know . . .

- Who's leading the National League?
- What was the final score of the Ohio State-Michigan Game?
- How many points did "Magic Johnson" score last night?
- Who played the Celtics on Sunday?
- How many strokes under par is Nicklaus?
- Who won the U.S. Open? . . . the Indy 500? . . . the Kentucky Derby? . . . Wimbledon?

Find out fast . . . with just the touch of a few buttons!

If you are looking for immediate access to what's happening in the world of sports, let Dow Jones save you time and keep you informed with the new Sports Report data base.

## Learn About Your Favorite Sports, Teams and Players!

- Major League Baseball
- NFL Football
- NBA Basketball
- NHL Hockey
- PGA & LPGA Golf
- Soccer
- Tennis
- NCAA Football
- NCAA Basketball
- And More!

You simply select the sports category of your choice and retrieve the latest scores and statistics provided by United Press International over your Radio Shack computer. Whether it be the schedule of your favorite pro football team, the National League baseball standings or scores of Saturday's top college games—the information is at your fingertips. These statistics are updated every morning, Monday through Friday.

## THE COST

The usage rate for the Sports Report data base is the same as our regular news data base. (See your Dow Jones price list for details.)

## HOW TO GET THE SPORTS REPORT

To enter the data base on the DOW JONES NEWS/RETRIEVAL® Service, simply type two slashes, SPORTS and press the <ENTER> key.

</></><S><P><O><R><T><S>  
 <ENTER>

This is what you will see:

NEWS/RETRIEVAL SPORTS REPORT  
 (C) DOW JONES & CO., INC.  
 TUESDAY, OCT. 13, 1981

THIS DATA BASE, ASSEMBLED FROM  
 THE WIRES OF UNITED PRESS  
 INTERNATIONAL, OFFERS STORIES  
 AND STATISTICS ON PROFESSIONAL  
 AND TOP AMATEUR SPORTS

PRESS

FOR

- 1 RESULTS, STANDINGS, STATS
- 2 SPORTS NEWS

## **TOP MENU**

Once in the data base, select the sports information you want from the menus (indices) as they appear on your screen or print-out.

RESULTS, STANDINGS,  
 STATS

PRESS

FOR

- 1 NFL FOOTBALL
- 2 MAJOR LEAGUE  
BASEBALL
- 3 COLLEGE FOOTBALL
- 4 NBA BASKETBALL
- 5 NHL HOCKEY
- 6 SOCCER
- 7 GOLF
- 8 TENNIS

## **SEASONAL SUB-MENU**

For example, to view college football statistics, type:  
 3 <ENTER>

## **NOTES**

The following commands will assist you in moving about the data base:

1. To go forward to the next page within a section, simply press the <ENTER> key.

2. To go back to the previous page within a section, press <R> (for reverse) followed by the <ENTER> key.

3. Menu pages are numbered from 0-99. Text pages are numbered beginning with 100. You will find the page number at the top of each page, showing the page you are on and the last page of the section.

For example, "P103 OF 108" means you are on page 103 of the text and that page 108 is the end of the section.

To proceed to any page of the data base from any other page (forward or back), press <P> (for Page) followed by the page number and the <ENTER> key.

Example: To view page 105, type:  
<P><1><0><5><ENTER>

4. To return to a previous sub-menu within the data base, press <M> (for Menu) and the <ENTER> key until the desired menu page is reached.

5. For a shortcut back to the Top Menu, press <T> (for Top) followed by the <ENTER> key.

The Dow Jones Sports Report . . . for Monday morning quarterbacks who want Monday morning's facts.

DOW JONES NEWS/RETRIEVAL® Service  
P.O. Box 300  
Princeton, New Jersey 08540  
1-800-257-5114 (in NJ, 609-452-1511)

## Customer Service Notes (From page 14)

```
310 IF C$=CHR$(0) THEN PRINT #-2, " BINARY" ELSE
      PRINT #-2, " ASCII"
320 NEXT Y
330 NEXT X
340 IF FL=0 THEN A$=B$
      : FL=1
      : GOTO 180
350 FL=0
      : S=S+1
      : IF S=12 THEN END
360 GOTO 170
```

## CARRIAGE RETURN/LINE FEED FOR THE COLOR COMPUTER

This program sends a line feed after a carriage return (CR-LF).

```
100 CLS
110 DATA 052, 020, 214, 111, 193, 254
120 DATA 038, 011, 129, 013, 038, 007, 190
130 DATA 160, 002, 173, 003, 134, 010, 053
140 DATA 020, 057
150 FOR D=1000 TO 1021
160 READ E
170 POKE D, E
180 NEXT D
190 POKE 1021, PEEK(359)
200 POKE 1022, PEEK(360)
210 POKE 1023, PEEK(361)
220 POKE 359, 126
230 POKE 360, 3
240 POKE 361, 232
250 END
```

- 1) Type in the program as written
- 2) Type: <R><U><N><ENTER>
- 3) When the 'OK' prompt returns, type:  
<N><E><W><ENTER>

From this point the computer should react as normal in either Color BASIC or Extended Color BASIC (depending on which computer you have.) No memory needs to be set aside for this program and no memory is taken from the user RAM space.

Note: This program is only effective as long as the computer is left on after following the above directions.

## COLOR COMPUTER LINE PRINTER LINE WIDTH DRIVER

```
100 'LINE PRINTER LINE WIDTH DRIVER'
110 DATA 182, 001, 103, 167, 141, 000, 046, 190,
      001, 104, 175, 141, 000, 040, 134, 126, 183,
      001, 103, 048, 141, 000, 004, 191, 001, 104,
      057, 052
120 DATA 002, 150, 111, 129, 254, 038, 016, 150,
      156, 139, 001, 145, 155, 037, 008, 015, 156,
      134, 013, 173, 159, 160, 002, 053, 002, 018,
      018, 018
130 CLS
140 INPUT "START ADDRESS"; START
150 FOR ADDRESS=START TO START +55
160 READ CODE
170 POKE ADDRESS, CODE
180 NEXT ADDRESS
190 EXEC START
```

- 1) Enter the program as written.
- 2) Before RUNNING the program execute any CLEAR statements or other procedures necessary to protect memory for the program.
- 3) RUN the program.
- 4) Respond to the 'START ADDRESS?' prompt with the address at which you wish the computer to load the machine language routine, and press <ENTER>. The routine is 56 bytes long.
- 5) POKE decimal address 155 with the number of characters per line that you desire PLUS 1. (e.g., If you want 32 characters per line on the printer then POKE 155,33.)



# Personnel Manager

We talked in the December and January issues about Time Manager, the first program of the Manager's Series. Time Manager™ along with Personnel Manager™ and Project Manager™ will not do your job for you, but this collection of tools is as invaluable as an extra arm, a 26 hour day, or a "bark" that moves mountains. These amazing aids for any manager keep track of priorities, expenses, people, projects, tasks and much more. This month I have included an article from Image Producers, Inc. the authors of Personnel Manager. Personnel Manager (26-1581, \$99.95) will be available soon and is designed for the TRS-80 Model I or III with 48K and two drives.

Written by Phyllis M. Schneider

Personnel Manager is a specialized filing system for organizing all types of information. Due to the extreme versatility of this program, applications extending far beyond personnel management are also possible.

Personnel Manager records business and personal information matched to your individual requirements. Its four levels: General, Detailed, Project, and Resume, allow you to examine a general overview of your topic or a very specific report.

## RESUME REPORT FOR ANDERSON

```

1/5/81  POSITION: HIRED--WRITER/MARKETING DEPT.
        -SALARY ADJUSTMENT- $14,500
        NOTE: BACKGROUND--AD AGENCY, BS--JOURNALISM
6/5/81  NOTE: WORKING ON M.B.A.--MARKETING
        -SALARY ADJUSTMENT- $17,000
        NOTE: GOOD PROGRESS; HANDLING LARGE PROJECTS
1/5/82  POSITION: ASSISTANT EDITORIAL MANAGER
        -SALARY ADJUSTMENT- $20,000
1/27/82 CLASS: M.B.A. MARKETING ADMIN- GRADE: A
        CLASS: M.B.A. NEW PROD. MGMT. GRADE: A
6/5/82  POSITION: ASSISTANT PRODUCT MGR.
        -SALARY ADJUSTMENT- $23,000

```

<ENTER> <CLEAR> <ARROWS> NAME / :

The data disk packaged with the program includes personnel management examples organized under sample formats. These examples, when used in conjunction with the manual, provide an overview of the program and its capabilities.

Once familiar with the program, you can change and resort labels and formats to efficiently organize whatever type of information you choose. By making simple modifications, the concept of the program as strictly a personnel management tool becomes individually redefined.

The program is user-definable, and its applications are almost endless. You can easily create bulk mailing lists (pre-sorted by zip code), a directory of frequently dialed telephone numbers, a file of your favorite recipes, a social calendar, a list

of important birthdays, anniversaries, holidays and obligations, a duty roster, a vacation schedule, a project assignment/completion table, or any other report appropriate to the program's capabilities and your specific requirements.

## DETAILED REPORT: MAILING LIST

```

>NANCY HART
  RESEARCH DIRECTOR
  LRT CORPORATION
  565 S. MAIN STREET
  LA MIRADA, CA 90638

```

```

SUE ELLEN SIMMONS
  SALES REP.
  S & L DIST.
  33 PEACH TREE STREET
  ATLANTA, GA 56842

```

```

-BILL B. JOHNSON
  SALES MANAGER

```

<ENTER> <CLEAR> <ARROWS> NAME / :

You can even specify exactly which portions of your data you do or do not wish to see by setting Personnel Manager's selection feature. By doing so, those records which do not meet certain criteria will be filtered out of the resulting display.

Before leaving on a business trip, for example, you could print a complete list of your clients in Chicago, including phone numbers and other essential information. You could, in a similar manner, obtain a list of all your clients outside of the Chicago area.

## GENERAL REPORT: TELEPHONE DIRECTORY

```

> ANDERSON, JUDY L.           H-(312) 446-8765
  SOFTWARE PUBLISHING CO.   B-(312) 329-4100 X24

  BELL, DIANE R.           H-(312) 764-5841
  SOFTWARE PUBLISHING CO.   B-(312) 329-4100 X7

  BERGMAN, SANDY K.       H-(312) 251-7782
  SELF-EMPLOYED            B-(312) 251-7781

  CARLISLE, JOSEPH O.     H-(312) 564-4840
  SOFTWARE PUBLISHING CO.   B-(312) 329-7761 X5

  DEVON, KATHY M.         H-(312) 945-6700
  SOFTWARE PUBLISHING CO.   B-(312) 329-4100 X10
  <ENTER> <CLEAR> <ARROWS> NAME / :

```

Personnel Manager maintains twenty-six category codes of information, originally set to codes relevant to personnel management. The category codes may, however, be altered to suit your particular program application.

CATEGORY CODES

A	ALL DATA	N	SALES STAFF
B	NAME AND ADDRESS	O	SALES REPS
C	TELEPHONE DIRECTORY	P	CONSULTANTS
D	COMPANY DIRECTORY	Q	ACTIVE ACCOUNTS
E	MAILING LIST	R	INACTIVE ACCOUNTS
F	MANAGEMENT	S	PROSPECTIVE ACCOUNTS
G	SUPERVISORS	T	VENDORS
H	CLERICAL	U	PROFESS. SERVICES
I	PRODUCTION	V	OFFICE SERVICES
J	MARKETING	W	SUPPLIES & EQUIPMENT
K	ADVERTISING	X	ASSOCIATIONS
L	DATA PROCESSING	Y	TRAVEL
M	CUSTOMER SERVICE	Z	SOCIAL

CATEGORY? (A-Z)

Once you have defined and entered information using your category codes, data can be extracted from the Personnel Manager program based solely on those codes. When combined with the selection criteria, very specific information is accessible quickly and easily. A mailing list of all the households in a predefined target market could be prepared with a minimum amount of time and/or effort.

Eight user-definable notepads can be used to store reference information such as phone numbers, miscellaneous notes, review comments, project results, meeting summaries or any other information you will occasionally refer to. The notepads contain command syntax and information for ready reference as you begin using Personnel Manager.

NOTEPAD A: REPORT COMMANDS

CONTROLS	DESCRIPTION
ENTER/CLEAR	MOVE FROM ONE REPORT LEVEL TO ANOTHER.
DOWN ARROW	MOVE POINTER TO NEXT RECORD.
UP ARROW	MOVE POINTER TO PREVIOUS RECORD.
RIGHT ARROW	MOVE POINTER TO NEXT ALPHABETICAL RECORD.
LEFT ARROW	MOVE POINTER TO PREVIOUS ALPHABETICAL RECORD.
NAME	MOVE POINTER TO NAME OR INITIAL ENTERED.
/	DISPLAY LIST OF COMMANDS.
CTRL-P	TYPE SCREEN CONTENTS ONTO PRINTER FOR ANY DISPLAY EXCEPT OPENING FRAME.

An optional line printer for your Model I or III computer will greatly enhance your use of the Personnel Manager program. Printouts can be obtained of any screen (other than the opening screen), record or report using the printing command. The program allows you to format the page to whatever form is necessary, including mailing labels.

A variety of other special features have been included with the Personnel Manager program. For instance, if you have the Time Manager™ program, you can make and update entries between the two programs without having to re-enter or type additional information. For example, you might transfer actual completion dates from Time Manager to Personnel Manager's Project report to track an individual's record for meeting his deadlines. By using identical category

codes, the two programs can be used in conjunction with each other, increasing the usefulness of both products.

The manual included with the program incorporates the examples provided on the original data disk, and can be used as a "walkthrough" guide to Personnel Manager. However, once familiar with the program you will find the applications almost limitless, since all the information on the data disk can be changed, deleted or saved, depending on your individual application requirements. Complete appendices and a reference card outline summarize procedures and program commands.

## Compound Interest and Savings

Mike Riggs  
Moscow, Idaho

This is a program that I developed during a week-long workshop for educators on programming. The purpose of this program is to demonstrate to students the effect of compound interest on savings accounts.

```

10 REM /PROGRAM WRITTEN BY MIKE RIGGS 'BANKER'
   6/19/81
20 CLS
   : PRINT "HELLO! WOULD YOU LIKE TO FIND OUT
   HOW COMPOUND INTEREST WORKS ON YOUR SAVINGS
   ACCOUNT? TYPE IN THE AMOUNT OF PRINCIPLE,
   THE RATE OF INTEREST, AND THE"
   : PRINT "NUMBER OF YEARS.";
30 PRINT
   : PRINT
   : X=0
   : PRINT
40 INPUT"AMOUNT OF PRINCIPLE =";A
50 PRINT
   : INPUT "RATE OF INTEREST (PLEASE EXPRESS
   THE PERCENTAGE AS A DECIMAL ) =";B
60 PRINT
   : INPUT "NUMBER OF YEARS =";Y
70 PRINT
   : FOR Z=1 TO Y
80 C=A*B
90 A=C+A
100 X=X+1
105 PRINT
   : PRINT"AFTER YEAR ";X;"----"
110 PRINT"SAVINGS BALANCE = $";A; "INTEREST PAID
   = $";C
120 NEXT Z
130 END

```



# Model I/III Bugs, Errors and Fixes

## MODEL III TRSDOS (26-312)

Operational note: If you add external drives to your Model III after TRSDOS has booted, or if you boot TRSDOS with the external drives turned off, and then turn the drives on, TRSDOS will not recognize the added drives. Always reboot TRSDOS after adding or turning on external disk drives so TRSDOS will "see" that the drives are available.

Programs like Manufacturing Inventory Control, Accounts Payable and Accounts Receivable can be expanded to use external drives. If you expand to external drives, and the drives were not connected and turned on when you powered up the Model III, it is important that you exit your program properly and reboot TRSDOS with the external drives on before you expand the program capacities.

## ACCOUNTS PAYABLE (26-1554)

In versions 3.0 and earlier of Model I Accounts Payable, Vendors can not be added to APS after the maximum number have been added and then some are deleted. The problem is corrected by following the steps listed below.

1. Backup the diskette(s) and make the changes on the Backup copy of the program
2. In BASIC load the program by typing:

```
LOAD "APS" <ENTER>
```

3. Make the following corrections:

CHANGES (Retype the line or refer to the Edit section of the owners manual)

VERSION 3.0:

```
Old Line: 161 P1=0: J=VN: IFPV=VN-1 THEN PL=704: W1$="VENDOR  
FILE IS FULL"+STRING$(24," "): GOSUB 121:  
GOTO 157 ELSE TN=0: FOR W9=1 TO 7: V$(W9)="": NEXT
```

```
New Line: 161 P1=0: J=VN: IFPV=VN-1-VD THEN PL=704:  
W1$="VENDOR FILE IS FULL"+STRING$(24," "):  
GOSUB 121: GOTO 157 ELSE TN=0: FOR W9=1 TO 7:  
V$(W9)="": NEXT
```

VERSIONS PRIOR TO 3.0:

```
Old Line: 510 P1=0: J=VN: IFPC=VN THEN PL=704: W1$="VENDOR FILE  
IS FULL": GOSUB 430: GOTO 500  
ELSE VV=0: FOR W9=1 TO 7: V$(W9)="": NEXT
```

```
New Line: 510 P1=0: J=VN: IFPV=VN-VD THEN PL=704: W1$="VENDOR  
FILE IS FULL": GOSUB 430: GOTO 500  
ELSE VV=0: FOR W9=1 TO 7: V$(W9)="": NEXT
```

4. Type:

```
SAVE "APS" <ENTER>
```

to save the changes in the program.

5. At TRSDOS Ready, make a backup copy of the corrected diskette.

## ACCOUNTS RECEIVABLE (26-1555)

For Model I Accounts Receivable versions prior to 3.0, the manual states: "Once you choose the storage capacity option, you cannot change the capacity until 'End of Period' Processing." At this point, you can increase the number of accounts if you desire. You cannot decrease the account capacity. The problem is that when you attempt to decrease the number, an error message "NO FILES ON DATA DISK" occurs.

The correction given in the March, 1981 Microcomputer News was not correct. Please change line 880 of the program SETUP to read:

```
880 PD=2: PC=500: PT=2500: IFQ$="M" THEN ON ERROR  
GOTO 895: KILL PT$: PT$=LEFT$(PT$, LEN(PT$)-1)+"2"  
: CLS: PRINT@458, "INSERT DATA DISK IN DRIVE 2  
AND PRESS <ENTER>" ELSE RETURN.
```

Also, please add these additional lines:

```
885 IF INKEY$ <> CHR$(13) THEN 885 ELSE  
OPEN "R", 3, PT$: CLOSE 3: ON ERROR GOTO 600: RETURN
```

```
895 IF ERR/2+1=54 THEN RESUME NEXT ELSE ON ERROR GOTO 0
```

After you have made these changes, be sure to save a copy of the corrected SETUP program.

If you are running out of extents in Model III Accounts Receivable version 3.0, the problem can be corrected by following the steps listed below.

1. Backup the diskette(s) and make the changes on the Backup copy of the program.
2. In BASIC load the program by typing the appropriate filename.
3. Make the following corrections:

Changes to the program ARS (Retype the line or refer to the Edit section of the owners manual):

```
Old Line: 2560 PI$="CUSINDEX."+"P$+"": 1":  
PD$="CUSDATA."+"P$+"": 1"
```

```
New Line: 2560 PI$="CUSINDEX."+"P$+"": 1":  
PD$="CUSDATA."+"P$+"": 0"
```

After making this change, save ARS and load the program SETUP. Make the following changes to SETUP:

```
Old Line: 470 PI$="CUSINDEX."+"P$+"": 1":  
PD$="CUSDATA."+"P$+"": 1":  
PT$="TRANSACTION."+"P$:"+"PS$="CUSSETUP."+"P$+"": 1":  
PG$="GLF FILE."+"P$+"": 1": IF PD=2 THEN  
PT$=PT$+"": 2": RETURN ELSE PT$=PT$+"": 1": RETURN
```

```
New Line: 470 PI$="CUSINDEX."+"P$+"": 1":  
PD$="CUSDATA."+"P$+"": 0":  
PT$="TRANSACTION."+"P$:"+"PS$="CUSSETUP."+"P$+"": 1":  
PG$="GLF FILE."+"P$+"": 1": IF PD=2 THEN  
PT$=PT$+"": 2": RETURN ELSE PT$=PT$+"": 1": RETURN
```

4. After making these changes, save the corrected copy of SETUP and return to TRSDOS READY.

5. Type:

```
COPY CUSDATA.(PASSWORD):1 TO CUSDATA.(PASSWORD):0
```

6. When this is done, Type:

```
KILL CUSDATA.(PASSWORD):1
```

7. This should eliminate extent problems with the customer file.

\*\*\* NOTE \*\*\* It will now be necessary to back up both disks as a set.

## MANUFACTURING INVENTORY CONTROL (26-1559)

In all Model I versions of Manufacturing Inventory Control prior to 3.0, it has been found that a problem can occur in Finish Good Maintenance which causes the machine to lock up when updating the files if more than 300 raw materials are in the finished good. The problem is corrected by following the steps listed below.

1. Backup the diskette(s) and make the changes on the Backup copy of the program.
2. In BASIC load the program by typing:

```
LOAD "FGMAINT" <ENTER>
```

3. Make the following corrections:  
**CHANGES** (Retype the line or refer to the Edit section of the owners manual)

**Old Line:** 4 DEFINTI-K,F,R,W-Z:DIMW,WS,WL,WD,PT,X,Y,  
 F1\$(8,5,4),F(7),IX\$(128),FR(1,610),J(3),  
 FG\$(25,3),FG(25),P\$(63),Q\$(63)

**New Line:** 4 DEFINTI-K,F,R,W-Z:DIMW,WS,WL,WD,PT,X,Y,  
 F1\$(8,5,4),F(7),IX\$(128),FR(1,610),J(4),  
 FG\$(25,3),FG(25),P\$(63),Q\$(63)

**Old Line:** 8510 J(1)=VARPTR(IX(1)):J(2)=VARPTR(FR(0,1)):  
 J(3)=RQ

**New Line:** 8510 J(1)=VARPTR(IX(1)):J(2)=VARPTR(FR(0,1)):  
 J(3)=RQ:J(4)=HR

**Old Line:** 10090 FIELD1\*146ASUS\$:GET1\*285:SR\$=US\$:RETURN

**New Line:** 10090 FIELD1\*149ASUS\$:GET1\*285:SR\$=US\$:RETURN

Save the corrected program, then make the following changes to "MFGINIT" (Initialization disk)

**Old Line:** 16000 SR\$=STRING\$(146,32):FORX=1TO146:READY:  
 MID\$(SR\$,X,1)=CHR\$(Y):NEXT

**New Line:** 16000 SR\$=STRING\$(149,32):FORX=1TO149:READY:  
 MID\$(SR\$,X,1)=CHR\$(Y):NEXT

**Old Line:** 16010 FIELD1,146ASQS\$:LSETQS\$=SR\$:PUT1,285:  
 SR\$="":RETURN

**New Line:** 16010 FIELD1,149ASQS\$:LSETQS\$=SR\$:PUT1,285:  
 SR\$="":RETURN

Replace lines 16030 through 16120 with these lines:

16030 DATA 205,127,10,94,35,86,213,221,225,35,94,  
 35,86,213,253

16040 DATA 225,35,94,35,86,35,78,35,70,3,19,197,  
 213,209,193

16050 DATA 11,121,176,200,197,221,35,221,35,213,  
 253,229,225,1,4

16060 DATA 0,27,122,179,40,67,213,221,126,0,190,  
 35,32,69,221

16070 DATA 126,1,190,32,63,209,43,94,35,86,213,  
 35,94,35,86

16080 DATA 213,229,209,43,62,4,245,197,213,229,  
 237,184,225,209,193

16090 DATA 241,61,32,243,209,253,114,3,253,115,2,  
 209,253,114,1

16100 DATA 253,115,0,253,229,225,17,4,0,25,229,  
 253,225,209,27

16110 DATA 122,179,213,32,159,209,193,201,17,3,  
 0,25,229,197,225

16120 DATA 19,25,229,193,225,209,27,122,179,40,  
 138,213,24,159

4. Save the corrected MFGINIT program.

5. At TRSDOS Ready, make a backup copy of the corrected diskette.

### MEDICAL OFFICE SYSTEM (26-1568)

In Model III version 1.2 of Medical Office System, the date does not change from the initial date. The problem is corrected by following the steps listed below.

1. Backup the diskette(s) and make the changes on the Backup copy of the program.

2. In BASIC-load the program by typing:

LOAD "DLYJOUR" <ENTER>

3. Make the following corrections:

**CHANGES** (Retype the line or refer to the Edit section of the owners manual)

**Old Line:** 1160 LPRINTUSING"TOTAL RECEIPTS: #####.##";  
 TT#(2)+TT#(3):LPRINTUSING"TOTAL CHARGES:  
 #####.##";TT#(1):FORE=1TO3:TT#(E)=0D0:NEXTE:  
 LPRINTCHR\$(12);:POKE16425,1:IFFQANDMH<=LHTHEN  
 D8\$=DH\$:CN\$="":GOSUB1055:GOTO1080ELSEIFNQ>0AN  
 DFQTHEN120ELSE560

**New Line:** 1160 LPRINTUSING"TOTAL RECEIPTS: #####.##";  
 TT#(2)+TT#(3):LPRINTUSING"TOTAL CHARGES:  
 #####.##";TT#(1):FORE=1TO3:TT#(E)=0D0:NEXTE:  
 LPRINTCHR\$(12);:POKE16425,1:IFFQANDMH<=LHTHEN  
 D8\$=DH\$:GOSUB340:CN\$="":GOSUB1055:GOTO1080  
 ELSEIFNQ>0ANDFQTHEN120ELSE560

4. Type:

SAVE"DLYJOUR" <ENTER>

to save the changes in the program.

5. At TRSDOS Ready, make a backup copy of the corrected diskette.

### Revisions, Enhancements, Etc.

The following is a list of current stock numbers for Computer Software revisions, enhancements and some miscellaneous items. These items can be ordered through your local Radio Shack store. If no price is shown, the item is available at no cost upon proof of purchase of the original program.

#### MODEL I

700-2007 LPC Driver N/C

With some Radio Shack printers and some early versions of Radio Shack software the "top of form" is increased by one line on each page. The LPC driver corrects this problem.

700-2009 Profile (26-1562 ver. 3.2) N/C

Corrects printing problems that existed in earlier versions of Profile.

700-2110 Tandygraph disk for 26-1191 N/C

Multi-pen Plotter

700-2220 VisiCalc (26-1566) DIF Documentation N/C

Programmer's Guide to the Data Interchange Format. Not included with any Model I VisiCalc packages.

700-2222 Payroll Disks & Manual (26-1556) \$25.00

Includes the 1982 tax tables and a program change to allow the W-2 program to be user formattable. (There are no W-2 changes for 1981.)

700-3004 TRSDOS 2.3 (26-2104) \$ 5.00

This manual incorporates all the changes that have been made to the previous TRSDOS 2.3 manual

700-5210 FORTRAN Source Files for 26-2201 N/C

Diskette containing the Sample Source Files for 26-2201

700-6001 Mailing List Exchange \$60.00

Upgrade Disk Mailing List (26-1551) to Business Mailing List (26-1558).

#### MODEL III

700-2006 Disk Scripsit (26-1563 Ver. 3.2) N/C  
 on TRSDOS 1.3

700-2008 Versafile (26-1604) on TRSDOS 1.3 N/C

700-2010 Profile (26-1562 ver 3.2) N/C  
 on TRSDOS 1.3

700-2012 Microfiles (26-1565) on TRSDOS 1.3 N/C

700-2112 Tandygraph disk for 26-1191 N/C  
 Multi-pen Plotter

700-2214 General Ledger Disks & Addenda N/C  
 (26-1552 Ver. 3.0) on TRSDOS 1.3

700-2215 Inventory Control System version 3.0 N/C  
 Disks & Manual (26-1553)

700-2216 Accounts Payable version 3.1 Disks & Addenda (26-1554)	N/C
700-2217 Accounts Receivable version 3.1 Disks & Addenda (26-1555)	N/C
700-2218 Manufacturing Inventory Control version 3.0 Disks & Addenda (26-1559)	N/C
700-2219 Standard & Poor's Disks & Manual (26-1507) on TRSDOS 1.3	N/C
700-2220 VisiCalc DIF Documentation Programmer's Guide to the Data Interchange Format. Not included with Model III VisiCalc packages prior to enhanced version of VisiCalc (26-1569)	N/C
700-2232 TRSDOS 1.3 Disk & Manual For owners of Model III Disk Systems who never re- ceived the final Disk System Owner's Manual and TRSDOS 1.3 diskette.	N/C
700-3210 VisiCalc Replacement Disk (26-1566)	\$25.00
For owners who have destroyed or have unusable origi- nal VisiCalc program diskettes.	
700-3211 Scripsit Replacement Disk (26-1563)	\$25.00
For owners who have destroyed or have unusable origi- nal Scripsit program diskettes.	
700-6200 VisiCalc Exchange Disk	\$99.05
Enhanced version of Model III VisiCalc. Owner must provide the exchange card included in the VisiCalc man- ual and pay the difference between the earlier program and the enhanced Mod III VisiCalc (\$99.05) to obtain the enhanced version.	

## Arrays

Linda Miller

Arrays are everywhere. As you walk into a cafeteria you are met with arrays of salads, entrees, desserts, vegetables, breads, and other miscellaneous edibles. An array is an assortment of things or elements which are usually related in some way. The elements in the vegetable array are carrots, peas, spinach, cauliflower, etc. A telephone directory is an array of names, addresses, and phone numbers as well as other miscellaneous information. Array elements are whatever you want them to be, and the order of the elements depends on how they are input into the array.

In BASIC, a single array element is represented by a subscripted variable. Variable N\$(10) tells us that:

1. N\$ is the name of a string array. N(10) would refer to a numeric array.
2. Number 10 in parentheses (the subscript) indicates that this is element number 11 (zero (0) is element number one) of array N\$.

Enter the following program.

```
10 CLEAR 500
20 FOR X = 1 TO 10
30 PRINT "ENTER LAST NAME # "; X; " ";
   : INPUT N$(X)
40 NEXT
```

The first time line 30 is executed the computer notes that there will be a string array named N\$ that will have at most 11 elements. Eleven, because that is the default size of an undimensioned array. Default means that unless specified otherwise by a dimension statement (DIM) the array will hold not more than eleven elements.

A dimension statement overrides the default. DIM X\$(15) creates a string array named X\$ that will have up to 16 elements. An array can have one dimension— e.g. DIM K(50)—or more than one dimension—e.g. DIM L(15,30), DIM M\$(5,14,3), etc. A single dimensioned array was created in the example program where 10 last names were entered. As an additional word of caution when dimensioning an array, if an array has previously been dimensioned by default and the same array is dimensioned again with a DIM statement, the result will be a DD or Redimensioned array error message. The following brief program illustrates this.

```
10 A$(5)="HELLO"
20 DIM A$(15)
RUN
DD ERROR IN 20
```

Elements in an array are numbered from 0 to N where N is the highest numbered element. Enter and run the following program.

```
10 FOR X = 0 TO 10
20 INPUT "ENTER A NUMBER"; NU(X)
30 NEXT X
```

If there were no mistakes when keying in the program then you should have been able to enter 11 numbers. Edit line 10 to read:

```
10 FOR X = 1 TO 11
```

Input of eleven numbers is still requested, but when the eleventh element is entered the video displays:

```
?BS ERROR IN 20
```

A BS error, or "subscript out of range," occurred because the eleven elements in the array are numbered NU(0), NU(1), NU(2), . . . NU(10). The first element in any array is subscripted with 0. Often, as in the previous program to input 10 last names, the 0 element is ignored. When a subscript exceeds

1) the number 10 of an array that has been subscripted by default,

or

2) the subscript numbers specified in the DIM statement, then a BS error occurs.

As long as the subscript of an array does not exceed 10 in any one dimension and more than three dimensions, the array does not have to be dimensioned. For example array X\$(10,10,10) does not have to be dimensioned but array Y\$(2,3,4,4) does because it has more than three dimensions.

The two example programs—one a list of ten last names and the other a list of ten numbers—used single dimensioned arrays. The length of either list could be increased (or decreased) by using a DIM statement and changing the limits of the FOR/NEXT loops. The statement

```
10 DIM CLIENT$(14, 3)
```

creates a two dimensional string array named CLIENT\$. It has 15 rows (0 - 14) and 4 columns (0 - 3). The positions of the elements can be visualized like this.

0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
.	.	.	.
.	.	.	.
14,0	14,1	14,2	14,3

Enter and run the following program inputing your own "dummy" data.

```

10 CLEAR 1000
20 DIM CLIENT$(14, 3)
30 FOR N = 1 TO 14
40 PRINT " ENTER NAME OF CLIENT # "; N;
   : INPUT CLIENT$(N, 1)
50 PRINT "ENTER THE OCCUPATION OF CLIENT # "; N;
   : INPUT CLIENT$(N, 2)
60 PRINT "ENTER THE AGE OF CLIENT # "; N;
   : INPUT CLIENT$(N, 3)
70 CLS
   : NEXT N
80 ' THE ARRAY CLIENT$ WILL BE PRINTED OUT ON THE
   DISPLAY
90 FOR N = 1 TO 14
100 PRINT CLIENT$(N, 1) TAB 30 CLIENT$(N, 2)
   TAB(45) CLIENT$(N, 3)
110 NEXT N

```

Because positions 0,0; 0,1; . . . 0,3 and 1,0; 2,0; . . . 14,0 were ignored by the program, the data is stored in the following array positions:

```

1,1  1,2  1,3
2,1  2,2  2,3
3,1  3,2  3,3
.
.
.
.
.
.
.
.
.
.
.
.
14,1 14,2 14,3

```

Adding line 130 causes the program to print out a list of the client names only.

```

130 FOR N = 1 TO 14
   : PRINT CLIENT$(N,1)
   : NEXT

```

All of the TRS-80 computers provide for the use of arrays. The Pocket Computer allows the creation of an array named A (or A\$) which is single dimensioned. Similarly the Model I/III Level I computers provide for a single array named A subscripted by one number. The Model I Level II, Model III with Model III BASIC, Model II, Color Computer (4K and extended Color BASIC) enable you to have multiple arrays with more than one dimension.

TRS-80 users have created a wide variety of applications using arrays. If you have discovered a way of using an array that you think might be interesting to others, please write to us.

NOTE: When clearing string space always put the CLEAR statement before the DIM statement or the array will be redimensioned to the default setting.

## Array Input Routine

Louis Self  
Phoenix, AZ

It is often necessary to input arrays of information into your BASIC programs. For example, if we needed to input the following values from weather observations:

Time	Wind			Rainfall	Humid.	Sky
	Temp	Dir.	Vel.			Cover
8	78	090	10	.00	.36	.1
9	80	110	5	.00	.40	.1
10	84	090	7	.00	.50	.2
11	88	085	3	.00	.45	.3
12	90	094	6	.00	.60	.2
1	95	088	4	.00	.70	.3
2	98	090	2	.00	.85	.45
3	100	090	1	.00	.95	.6
4	100	090	0	.00	1.00	.8
5	96	090	10	.40	.95	.9
6	93	098	5	.40	.95	.7
7	90	095	2	.40	.8	.5
8	85	100	1	.40	.8	.4
9	80	105	3	.40	.75	.1

A typical INPUT routine might look like this:

```

90 DIM A(13, 6)
100 FOR X=0 TO 13
110 FOR Y=0 TO 6
120 PRINT X; Y;
   : INPUT A(X, Y)
130 NEXT Y, X

```

This routine will allow us to INPUT the information, but error checking may be difficult. It is also very easy to "get lost" while entering information with this type of routine, you enter one item twice, or skip a time completely.

What is needed is an array input routine which would allow us to see our information in its array format as we input it. This will make error detection much easier, since it will be obvious if we input a wind direction value into the rainfall column.

The following program is designed to simplify input of information into arrays. The routine uses a combination of PRINT TAB and INPUT to both get the information into the array and display the information in array format. Here is the program:

```

20 CLS
30 DIM A(14, 7)
40 PRINT"ENTER   A(X,0) A(X,1) A(X,2) A(X,3)
   A(X,4) A(X,5) A(X,6)"
50 FOR X=0 TO 13
60 PRINT "X ="X
70 FOR Y=0 TO 6
80 PRINT TAB(Y * 8 + 8) CHR$(27);
90 INPUT A(X, Y)
100 NEXT Y
110 NEXT X

```

Line 30 DIMensions the array A to the size needed (14 by 7 in this example).

Line 40 PRINTs an initial line on the video to help prompt the user for the needed information.

Line 50 sets up the FOR-NEXT loop for the 14 (0 to 13) X elements.



Line 60 prints a row identifier (X=) so the user will know which row of information is being requested.

Line 70 sets up the FOR-NEXT loop to input each of the 7 (0 to 6) items of information associated with each row.

Line 80 uses the value of Y (column number) to position the next INPUT statement. The columns will be INPUT beginning at TAB(8) when Y=0 ( $0*8 + 8=8$ ), and continue to the last column of TAB(56) when Y=6 ( $6*8 + 8$ ). The CHR\$(27) moves the cursor up one line on the Model I and III. (CHR\$(27) has an entirely different function on the Model II.)

Line 90 actually INPUTs the value into array element A(X,Y).

Lines 100 and 110 end the two FOR-NEXT loops.

When you run this program, the screen will clear, the heading row will appear, and one line down, X=0 will be displayed.

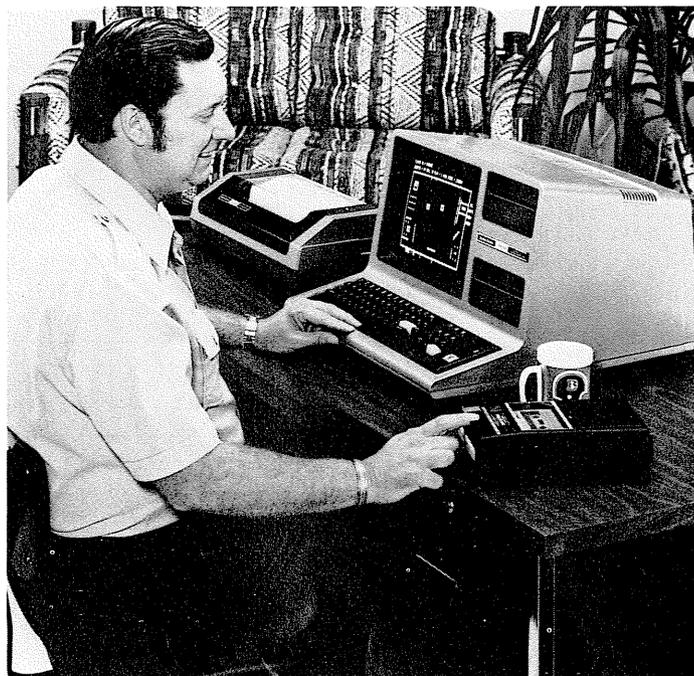
Next, a question mark will appear just to the right of X=0. Input the first Time value. Just to the right a second question mark will appear on the same line. Enter the first Temperature. As you continue to enter values, the computer continues across the line printing question marks for INPUT.

After you have entered the Sky Cover for X=0, the computer will move to the beginning of the next line, PRINT X=1, and begin requesting the next row of values.

As you continue to enter values, the information will be lined up in the same rows and columns as the original table. This makes it very easy to see when and where errors have occurred, and makes it easier to return to data entry if you happen to be interrupted.

Mr. Self included a second program for data output so you would have a complete set of Input/Output routines:

```
120 '----THE FOLLOWING PRINTS OUT THE ARRAY----
130 INPUT"PRESS ENTER TO PRINT OUT THE ARRAY";X
140 FOR X=0 TO 13
140 FOR Y=0 TO 6
160 PRINT A(X, Y);
170 NEXT Y
180 PRINT
190 NEXT X
```



## Client Activity Information Array

**Kenneth E. Robinson**  
7409 N. Garfield  
Gladstone, MO 64118

Client Activity Information Array is a program designed to keep track of clients, customers, patients, etc. When I was asked if a program existed that would keep track of about 140 clients at a rest home and keep track of which activities they attended such as bingo games, I could not find such a program. At least I could not find a program that would do it for a 16K computer. So I set out to reach into the recesses of my mind for a way to keep track of that much information. An array using only yes or no seemed the only way.

The Client Activity Information Array program will allow you to list each client, describe each item, and tell the computer if that item applies to the client. The computer will then read which items apply to the client, how many clients are listed for an item, and what the names of the clients are. The computer will also let you change the information in the array as to name of the client, description of the item, and if the item applies to the client. Information can be stored on a cassette tape and put back into the computer later.

```
10 CLS
   : PRINT "CLIENT ACTIVITY INFORMATION ARRAY"
20 PRINT, "written by Kenneth E. Robinson, July
   16, 1981"
30 FOR T=1 TO 800
   : NEXT T
40 CLEAR 5075
   : ON ERROR GOTO 1200
50 G=1401
55 CLS
60 INPUT "HOW MANY CLIENTS DO YOU EXPECT TO
   HAVE"; B
70 INPUT "HOW MANY ITEMS DO YOU WISH TO MONITOR";
   C
80 IF B*C<G GOTO 90 ELSE PRINT "TOO MUCH DATA FOR
   A 16K COMPUTER"
   : GOTO 60
90 DIM G$(B, C)
100 INPUT "DO YOU HAVE DATA ALREADY STORED ON
   CASSETTE TAPE (tape data must match client
   and item amounts just selected)"; F$
110 IF LEFT$(F$, 1)="Y" GOTO 1000 ELSE 120
120 CLS
   : FOR A=1 TO C
130 PRINT "WHAT IS ITEM #"; A
140 INPUT G$(P, A)
150 NEXT A
160 FOR P=1 TO B
170 A=0
180 PRINT"FOR CLIENT #"; P
190 PRINT"WHAT IS THE CLIENT'S NAME"
200 INPUT G$(P, A)
210 FOR A=1 TO C
220 PRINT"THIS IS FOR ITEM #"; A; "("; G$(0, A);
   ")" FOR CLIENT #"; P; "("; G$(P, 0); ")"";
230 PRINT" INPUT <Y> OR <YES> IF THIS ITEM
   APPLIES TO THIS CLIENT, OTHERWISE INPUT <N>
   OR <NO> THEN <ENTER>."
240 INPUT G$(P, A)
250 NEXT A
260 IF P=B GOTO 300
270 PRINT "DO YOU HAVE MORE INFORMATION TO BE
   STORED"
280 INPUT I$
290 IF LEFT$(I$, 1)="Y" THEN GOTO 300 ELSE P=B+1
```

```

300 NEXT P
305 CLS
310 PRINT"PRESS <1> TO READ CLIENT INFORMATION"
320 PRINT" <2> TO READ TOTAL CLIENTS LISTED
FOR AN ITEM"
330 PRINT" <3> TO READ NAMES OF CLIENTS
LISTED FOR AN ITEM"
340 PRINT" <4> TO CHANGE CLIENT INFORMATION"
350 PRINT" <5> TO CHANGE ITEM DESCRIPTION"
355 PRINT" <6> TO CHANGE ONE ITEM FOR ONE
CLIENT"
360 PRINT" <7> TO RESTORE INFORMATION ON
CASSETTE TAPE"
370 PRINT" <8> TO RETRIEVE INFORMATION FROM
CASSETTE TAPE"
380 INPUT K
: IF K<1 GOTO 380
390 IF K>8 PRINT"ONLY SELECTIONS LISTED ARE
PERMITTED"
: GOTO 310
400 ON K GOTO 410, 500, 600, 700, 800, 1100, 900,
1000
410 CLS
: INPUT "WHAT IS CLIENT'S NUMBER - All
clients must be assigned a number starting
with 1"; P
: J=0
420 IF P<1 OR P>B THEN PRINT "THAT CLIENT IS NOT
LISTED IN COMPUTER"
: GOTO 310
425 PRINT "CLIENT NO."; P; "("; G$(P, 0); ")"
430 FOR A=1 TO C
440 PRINT "ITEM NO."; A; "("; G$(0, A); ")" ;
450 IF LEFT$(G$(P, A), 1)="Y" THEN PRINT "YES"
ELSE PRINT "NO"
460 J=J+1
: IF J>14 INPUT "PRESS <ENTER> TO CONTINUE
READING ITEMS"; Z
: J=0
470 NEXT A
480 INPUT "PRESS <ENTER> TO CONTINUE"; Z
490 GOTO 305
500 CLS
: INPUT "WHICH ITEM ARE YOU INTERESTED IN";
A
510 IF A<1 OR A>C THEN PRINT "THAT ITEM IS NOT
LISTED"
: GOTO 310
520 E=0
530 FOR P=1 TO B
540 IF LEFT$(G$(P, A), 1)="Y" THEN N=1 ELSE N=0
550 E=E+N
560 NEXT P
570 PRINT "TOTAL NUMBER OF CLIENTS LISTED FOR
ITEM #"; A; "("; G$(0, A); ")" IS"; E"."
580 INPUT "PRESS <ENTER> TO CONTINUE"; Z
590 GOTO 305
600 CLS
: INPUT "WHAT ITEM ARE YOU INTERESTED IN"; A
: J=0
610 IF A>C THEN PRINT "THAT ITEM DOES NOT EXIST
IN COMPUTER FILES"
: GOTO 310
620 FOR P=1 TO B
630 IF LEFT$(G$(P, A), 1)="Y" THEN PRINT G$(P, 0)
ELSE 650
640 J=J+1
: IF J<15 GOTO 650 ELSE INPUT "PRESS <ENTER>
TO CONTINUE READING NAMES"; Z
: J=0
650 NEXT P
660 INPUT "PRESS <ENTER> TO CONTINUE"; Z
670 GOTO 305
700 CLS
: INPUT "FOR WHICH CLIENT WOULD YOU LIKE TO
CHANGE THE INFORMATION"; P
710 IF P<1 OR P>B PRINT "CLIENT NUMBER NOT
LISTED. TO CHANGE NUMBER OF CLIENTS YOU
MUST RE-INPUT ALL INFORMATION"
: GOTO 310
720 INPUT "CLIENT'S NAME"; G$(P, 0)
730 FOR A=1 TO C
740 PRINT "ITEM NO."; A; "("; G$(0, A); ")" FOR
CLIENT NO."; P
750 INPUT "INPUT <YES> IF THIS ITEM APPLIES TO
THIS CLIENT, OTHERWISE INPUT <NO>"; G$(P, A)
760 NEXT A
770 GOTO 305
800 CLS
: INPUT "WHICH ITEM DESCRIPTION DO YOU WISH
TO CHANGE"; A
810 PRINT "ITEM"; A; "IS--"; G$(0, A); "--THIS
ITEM DESCRIPTION IS BEING CHANGED FOR ALL
CLIENTS."
820 INPUT "HOW DO YOU WANT TO DESCRIBE THIS
ITEM"; G$(0, A)
830 INPUT "DO YOU WISH TO CHANGE CLIENT
INFORMATION FOR THIS ITEM"; L$
840 IF LEFT$(L$, 1)="N" GOTO 305
850 FOR P=1 TO B
860 PRINT "THIS IS ITEM #"; A; "("; G$(0, A); ")"
FOR CLIENT #"; P; G$(P, 0)
870 INPUT "DOES THIS ITEM APPLY TO THIS CLIENT";
G$(P, A)
880 NEXT P
890 GOTO 305
900 CLS
: INPUT " PRESS <ENTER> WHEN CASSETTE IS
READY TO BEGIN RECORDING INFORMATION ON
TAPE. BE SURE TAPE IS REWOUND AND RECORDER
IS IN RECORD MODE. USE A 'CLEAN' LEADERLESS
90 MIN. TAPE."; Z
910 PRINT "THE DATA IS NOW BEING INPUT FROM THE
COMPUTER INTO THE CASSETTE RECORDER."
920 FOR P=0 TO B
: PRINT P;
930 FOR A=0 TO C
940 PRINT # -1, G$(P, A)
950 H=H+1
: IF H>1400 INPUT "PRESS <ENTER> WHEN NEW
TAPE IS READY TO RECORD INFORMATION"; Z
: H=0
960 NEXT A, P
970 PRINT "THE DATA HAS BEEN STORED ON TAPE. YOU
MAY TURN OFF THE RECORDER NOW."
975 PRINT "N O T I C E : You may want to make
extra copies of data tape!"
980 INPUT "PRESS <ENTER> TO CONTINUE"; Z
990 GOTO 305
1000 CLS
: INPUT " PRESS <ENTER> WHEN CASSETTE IS
READY TO BEGIN PLAYING STORED DATA BACK
INTO COMPUTER. BE SURE TAPE IS REWOUND AND
PLAYER IS IN PLAY MODE."; Z
1010 PRINT " THE DATA IS NOW BEING INPUT FROM THE
CASSETTE PLAYER INTO THE COMPUTER."
1020 FOR P=0 TO B
: PRINT P;
1030 FOR A=0 TO C
1040 INPUT # -1, G$(P, A)
1050 H=H+1
: IF H>1400 INPUT "<ENTER> WHEN NEW TAPE IS
READY TO INPUT MORE INFORMATION INTO
COMPUTER"; Z
: H=0
1060 NEXT A, P
1070 PRINT "THE DATA HAS BEEN READ INTO THE
COMPUTER. YOU MAY TURN OFF THE PLAYER NOW."
1080 INPUT "PRESS <ENTER> TO CONTINUE"; Z
1090 GOTO 305
1100 CLS
1110 INPUT "WHAT IS THE CLIENT NUMBER"; P
1120 IF P<1 OR P>B PRINT "THAT CLIENT NUMBER IS
NOT LISTED"
: GOTO 1110
1130 INPUT "WHAT IS THE ITEM NUMBER THAT NEEDS
CHANGED"; A
1140 IF A<1 OR A>C PRINT "THAT ITEM IS NOT IN
COMPUTER"
: GOTO 1110

```

```

1150 INPUT "INPUT A <YES> OR <NO> FOR ITEM";
    G$(P, A)
1160 PRINT "FOR CLIENT NO."; P; "("; G$(P, 0); ")"
    INFORMATION IN ITEM NO."; A; G$(0, A); " HAS
    BEEN CHANGED TO "; G$(P, A)
1170 INPUT "PRESS <ENTER> TO CONTINUE"; Z
1180 GOTO 305
1200 IF ERR/2+1 = 14 PRINT "OOPS...OUT OF SPACE."
    : RESUME 310
1210 IF ERR/2+1 <> 14 PRINT "!!! PROGRAM IS IN
    TROUBLE !!!"
    : RESUME 60

```

To fast test this program after entering, change the following lines:

```

140 G$(P, A)="TEST ITEM"
200 G$(P, A)="KENNETH E. ROBINSON"
240 G$(P, A)="YES"
280 I$="YES"

```

Then look for screen outputs similar to these:

---

CLIENT ACTIVITY INFORMATION ARRAY  
written by Kenneth E. Robinson, July 16, 1981

---

```

HOW MANY CLIENTS DO YOU EXPECT TO HAVE? 150
HOW MANY ITEMS DO YOU WISH TO MONITOR? 25
TOO MUCH DATA FOR A 16K COMPUTER
HOW MANY CLIENTS DO YOU EXPECT TO HAVE? 140
HOW MANY ITEMS DO YOU WISH TO MONITOR? 10
DO YOU HAVE DATA ALREADY STORED ON CASSETTE TAPE (tape data
must match client and item amounts just selected)? N

```

Computer screen should start changing while information is being input by program into array. Then, if the program is good, the following screen will display.

---

```

PRESS <1> TO READ CLIENT INFORMATION
<2> TO READ TOTAL CLIENTS LISTED FOR AN ITEM
<3> TO READ NAMES OF CLIENTS LISTED FOR AN ITEM
<4> TO CHANGE CLIENT INFORMATION
<5> TO CHANGE ITEM DESCRIPTION
<6> TO CHANGE ITEM FOR ONE CLIENT
<7> TO STORE INFORMATION ON CASSETTE TAPE
<8> TO RETRIEVE INFORMATION FROM CASSETTE TAPE

```

? 1

---

WHAT IS CLIENT'S NUMBER - All clients must be assigned a number starting with 1? 150

```

THAT CLIENT IS NOT LISTED IN COMPUTER
PRESS <1> TO READ CLIENT INFORMATION
<2> TO READ TOTAL CLIENTS LISTED FOR AN ITEM
<3> TO READ NAMES OF CLIENTS LISTED FOR AN ITEM
<4> TO CHANGE CLIENT INFORMATION
<5> TO CHANGE ITEM DESCRIPTION
<6> TO CHANGE ONE ITEM FOR ONE CLIENT
<7> TO STORE INFORMATION ON CASSETTE TAPE
<8> TO RETRIEVE INFORMATION FROM CASSETTE TAPE

```

? 1

---

WHAT IS CLIENTS' NUMBER - All clients must be assigned a number starting with 1? 1

---

```

ITEM NO. 1 (TEST ITEM) YES
ITEM NO. 2 (TEST ITEM) YES
ITEM NO. 3 (TEST ITEM) YES
ITEM NO. 4 (TEST ITEM) YES
ITEM NO. 5 (TEST ITEM) YES
ITEM NO. 6 (TEST ITEM) YES
ITEM NO. 7 (TEST ITEM) YES

```

2

---

WHICH ITEM ARE YOU INTERESTED IN? 1  
TOTAL NUMBER OF CLIENTS LISTED FOR ITEM #1 (TEST ITEM) IS 140.  
PRESS <ENTER> TO CONTINUE?

3

---

WHAT ITEM ARE YOU INTERESTED IN? 1

---

```

KENNETH E. ROBINSON
PRESS <ENTER> TO CONTINUE READING NAMES?

```

4

---

FOR WHICH CLIENT WOULD YOU LIKE TO CHANGE THE INFORMATION? 1

```

CLIENT'S NAME? JANET L. WILLINGHAM
ITEM NO. 1 (TEST ITEM) FOR CLIENT NO. 1
INPUT <YES> IF THIS ITEM APPLIES TO THIS CLIENT, OTHERWISE
INPUT <NO>? NO
ITEM NO. 2 (TEST ITEM) FOR CLIENT NO. 1
INPUT <YES> IF THIS ITEM APPLIES TO THIS CLIENT, OTHERWISE
INPUT <NO>? YES

```

---

WHICH ITEM DESCRIPTION DO YOU WISH TO CHANGE? 2  
ITEM 2 IS —TEST ITEM—THIS ITEM IS BEING CHANGED FOR ALL  
CLIENTS  
HOW DO YOU WANT TO DESCRIBE THIS ITEM? CHANGED TEST  
DO YOU WISH TO CHANGE CLIENT INFORMATION FOR THIS ITEM? NO

5

---

WHICH ITEM DESCRIPTION DO YOU WISH TO CHANGE? 3  
ITEM 3 IS—TEST ITEM—THIS ITEM IS BEING CHANGED FOR ALL  
CLIENTS.  
HOW DO YOU WANT TO DESCRIBE THIS ITEM? CLOSED SALE  
DO YOU WISH TO CHANGE CLIENT INFORMATION FOR THIS ITEM? YES  
THIS IS ITEM #3 (CLOSED SALE) FOR CLIENT #1 JANET L. R OBINSON  
DOES THIS ITEM APPLY TO THIS CLIENT? YES  
THIS IS ITEM #3 (CLOSED SALE) FOR CLIENT #2 KENNETH E.  
ROBINSON  
DOES THIS ITEM APPLY TO THIS CLIENT? NO

6

---

WHAT IS THE CLIENT NUMBER? 4  
WHAT IS THE ITEM NUMBER THAT NEEDS CHANGED? 4  
INPUT A <YES> OR <NO> FOR THIS ITEM? YES  
FOR CLIENT NO. 4 KENNETH E. ROBINSON INFORMATION IS ITEM  
NO. 4  
TEST ITEM HAS BEEN CHANGED TO YES  
PRESS <ENTER> TO CONTINUE?

7

---

PRESS <ENTER> WHEN CASSETTE IS READY TO BEGIN RECORDING  
INFORMATION ON TAPE. BE SURE TAPE IS REWOUND AND RECORDER  
IS IN RECORD MODE. USE A 'CLEAN' LEADERLESS 90 MIN. TAPE?  
THE DATA IS NOW BEING INPUT FROM THE COMPUTER INTO THE  
CASSETTE RECORDER.

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

8

---

PRESS <ENTER> WHEN CASSETTE IS READY TO BEGIN PLAYING  
STORED DATA BACK INTO COMPUTER. BE SURE TAPE IS REWOUND  
AND PLAYER IS IN THE PLAY MODE?

THE DATA IS NOW BEING INPUT FROM THE CASSETTE PLAYER INTO THE COMPUTER.

Congratulations. If you have made it this far, you are ready to reset lines 140, 200, 240, and line 280 to values listed in program. After the program is returned to its original values you should run the program using small values for client and item amounts (the small values are needed to save time). Remember all programs should be tested to see if it will work for your conditions. You are the only one who knows if this program does what you need it to do.

The client activity information array is a program in BASIC written to allow the small business user to take a microcomputer and use it to keep track of his clients. This array will give you a yes or a no for items that you may want to keep track of for each client that you may have. The program is written in BASIC for easy input. All outputs are written in easy to understand instructions and meaningful information. After data is put into your program changes are easy and the computer will keep track of your clients, customers, patients, etc.

## Notes on Previous Issues

### December, 1981

#### MODEL I/III CALENDAR PROGRAM CORRECTIONS

Several people have written with suggested changes or corrections to the Model I/III Monthly Calendar program we printed in the December, 1981 issue.

The most obvious problem was that the calendar had two Augusts and no September. Change line 290 to read:

```
290 DA=DA+243
    : E=30
    : LPRINT TAB(20) "SEPT"; YR
    : RETURN
```

The second problem deals with leap years. Here is a loose excerpt from the letter we received from Mr. Norman Carlile:

Mr. Carlile stated that he hated to be "nit-picking" since the first time in the future that the produced calendar will be wrong will be for January and February 2100. The same will be true for the years 2200 and 2300. However, if some historian uses this program in connection with his studies for January and February of the years 1800 & 1900 he will be a day off. Also, all calendars from January 1700 through December 1752 will likely be wrong.

He went on with . . . My research shows that when Pope Gregory XIII introduced his calendar in 1582 he corrected an error in the Julian calendar by changing Julian 5 October to Gregorian 15 October. I have been unable so far to determine what adjustment was made when Great Britain and the American Colonies finally adopted the Gregorian calendar during 1752, but any calendar produced by the program prior to 1753 is likely incorrect.

Mr. Carlile also states that to reduce the error in the calendar Pope Gregory XIII restricted centesimal years as leap years to only those evenly divisible by 400. Therefore century years such as 2000, 2400, 2800 are leap years while

others such as 1700, 1800, 1900, 2100, etc. are not leap years.

Mr. Carlile then produced results from the published program which show that 31 Dec 1799 and 1 Jan 1800 are both computed to be on the same day of the week, while December, 1999 and January, 2000 are correct since 2000 is a leap year.

Mr. Carlile and Mr. George Clendenin provided us with revised or new calendar programs which overcome these problems.

To correct the program we published in December, Mr. Jim Johnston suggested adding these lines:

```
57 IF (M=1 OR M=2) AND (INT(YR/100)=YR/100) GOSUB
    450
450 IF INT(YR/400)=YR/400 RETURN
460 IF M=2 THEN E=E-1
470 B=B+1
480 IF B=7 THEN B=0
490 RETURN
```

Mr. Johnston also commented "Line 480 can be omitted as the program overflows at year 2328, before B ever is 7. This is of little consequence as the calendar will probably be revised long before then. I might note that the Gregorian calendar did not start in the American colonies until Sept. 14, 1752."

Mr. John Henson suggests that the leap year problem can be solved by simply changing line 41 to read:

```
41 IF (YR/4=INT(YR/4)) THEN FD=1
    : IF (YR/100=INT(YR/100)) AND
    (YR/400<>INT(YR/400)) THEN FD=0
```

Our thanks to L. H. Erb and Jeffrey W. Pettiross who also commented on the program.

### January, 1982

In the VisiCalc article (Page 17), the formula for B11 (calculation for Principal and Interest) contains an error. The correct formula for Model I/III computers is:

$$+ B7 * (B8/100) / (1 - ((1 + (B8/100))^{-(B4 * B5 - 1)}))$$

Notice the addition of a set of parenthesis around  $B4 * B5 - 1$  just after the exponentiation symbol (^ use an up arrow on Model I/III).

This same formula will work with Model II VisiCalc, but only to a point. In the example given, you will only be able to use 28 years. After 28, you will get a lot of ERROR flags. The problem is in the Model II VisiCalc exponentiation function.

Mr. Ralph K. Schmitz of International Marketing Consultants, Inc. of Fort Worth provided us with the following alternate formula for B11 on a Model II:

$$+ B7 * (B8/100) + (B7 * (B8/100)) / ((1 + (B8/100))^{-(B4 * B5 - 1)})$$

Mr. Schmitz commented that there will be a few cents difference in the monthly payment using the two different formulas, but the second formula works on the Model II and is not affected by the exponentiation problem.

# Educational Courseware for Graph Analysis in High School Physics

Graphical Analysis of Experimental Data and Interpreting Graphs in Physics are two problem-solving courseware programs designed for use in the secondary physics classroom. Like Advanced Graphics and Vector Addition, featured in last month's Education article, these programs were designed and programmed by Richard Born, a Radio Shack consultant with fourteen years of experience in teaching high school math, physics, and computer science.

## GRAPHICAL ANALYSIS OF EXPERIMENTAL DATA

Graphical Analysis of Experimental Data enables high school students to use the computer when graphing and studying data that they have gathered through experiment or research. The program is especially suited for use in the lab, where your students can use the TRS-80 to analyze experiment results as soon as data have been gathered.

The Graphical Analysis of Experimental Data program was first used in author Richard Born's physics class, where high school students were busy performing experiments in which they controlled one variable and measured another variable. Born has found that the program supports in-class analysis of experiment results by allowing students to see important patterns in their data while they skip some of the tedious work involved in constructing graphs manually. Because students can perform linear analysis quickly with the program, they often discover significant tendencies which suggest promising variations on the experiments. Students can then explore these variations immediately, in class.

The program begins by offering a special timing option. If the student wishes, he or she can use the computer to time an experiment in minutes and seconds. Once an experiment has been performed and the data collected, the student can begin setting up to analyze the data. The computer will graph numeric data pairs consisting of an independent variable and a dependent variable. Students name their own variables, then type in the data pairs. Once all data pairs have been entered, the computer will display a full-screen graph of the data and will then make available a list of options for working with the graphed data.

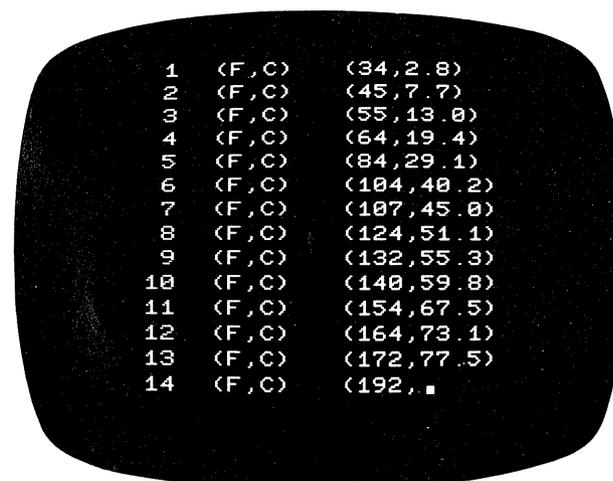
Let's take a brief look at a few things the program can do. For example, let's see how the TRS-80 and the Graphical Analysis of Experimental Data program could be used to graph the relationship between the Fahrenheit and Celsius temperature scales. Suppose that you gradually heated a pan of ice on a stove, while taking simultaneous temperature readings with both a Fahrenheit and a Celsius thermometer. You recorded the following temperatures:

F	C
34	2.8
45	7.7
55	13.0

64	19.4
84	29.1
104	40.2
107	45.0
124	51.1
132	55.3
140	59.8
154	67.5
164	73.1
172	77.5
192	84.0

Your two variables for this example will be "F" for "Fahrenheit" and "C" for "Celsius." "F" will be your independent variable while "C" will be your dependent variable. When asked to name your independent variable, you'd type <F> and press <ENTER>. When asked to name your dependent variable, you'd type <C> and press <ENTER>.

An instruction screen with directions for entering your data pairs would then appear. Once you had read the instruction screen and pressed <ENTER>, you'd see the screen on which data pairs are to be typed. When you had typed in part of your data, the screen would look like screen one.



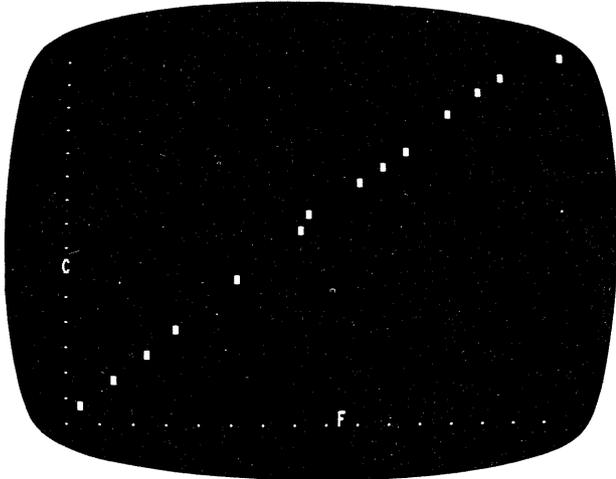
Screen 1

Once all data pairs were entered, a graph of all data points would appear as shown on screen two.

After studying the graph, you could press the space bar to see the list of options shown on screen three.

In analyzing the data in this particular example, you might want to begin by pressing <E> to eliminate the data points that did not fit with the smooth curve of the graph. Pressing <E> would make the graph reappear. Then, starting with the data point on the far left, each data point would in turn blink. To delete a data point you'd type <D> when the point

was blinking. To keep a data point you'd type <K> when the point was blinking.



Screen 2

To see a table of the variable values that remained, you could press <T>. A display like the one shown on screen four would appear.

```

GRAPHICAL ANALYSIS OF EXPERIMENTAL DATA
OPTION LIST

A  ADD DATA POINTS
C  COMPUTE INTERCEPTS, SLOPE, EQUATION
E  ELIMINATE BAD DATA POINTS
F  FUNCTION SELECTION
G  GRAPH DATA
L  LIMITS FOR GRAPH DISPLAYED
P  PREDICT DEPENDENT VARIABLE VALUE
R  READ IN NEW DATA (OLD DATA DESTROYED)
T  TABLE OF DATA DISPLAYED

PRESS LETTER OF SELECTION
    
```

Screen 3

If you next wanted to have the computer calculate and display the slope and intercepts of your graphed data, and the equation of the straight line of best fit to your data points, you'd press <C>. You'd see a display like the one on screen five.

Once you had used the <C> option, you'd be able to use the <P> option to test the accuracy of the equation as a predictor of dependent variable values. To use this option, you'd press <P> and type in one or more independent variable values. As you typed each value, the computer would predict and display a dependent variable value based on your independent variable value.

If you were analyzing data from a different type of experiment, you might want to graph the functions of independent and dependent variables. Pressing <F> at the options list would allow you to choose a function for your variables (see screen six).

```

GRAPHICAL ANALYSIS OF EXPERIMENTAL DATA

THE NUMBER OF DATA POINTS IS 12
F-INTERCEPT = 29.9735 °
C-INTERCEPT = -16.3121
SLOPE = .544218

THE EQUATION OF THE STRAIGHT LINE
OF BEST FIT TO YOUR DATA IS:

C = .544218 * F + -16.3121
    
```

Screen 5

The user's manual for Graphical Analysis of Experimental Data contains a Selected Investigations section which provides numerous applications problems for high school physics. Typical problems present sample data from physics experiments based on frictional forces, Hooke's Law, Kepler's Law of Periods, Boyle's Law, and radioactive decay, among other concepts. Methods for analyzing the data on the computer are then suggested. A problem on U.S. Population Growth is included in the Selected Investigations section to suggest that many non-physics applications of the program are also possible. The population exercise provides population data from every Census taken from 1790 to 1970. The student is then asked to use the computer to plot population versus time, to plot the natural log of population versus

F	C	F	C
34	2.8	34	2.8
45	7.7	45	7.7
55	13	55	13
64	19.4	64	19.4
84	29.1	84	29.1
104	40.2	104	40.2
124	51.1	124	51.1
132	55.3	132	55.3
140	59.8	140	59.8
154	67.5	154	67.5
164	73.1	164	73.1
172	77.5	172	77.5

THE TABLE IS COMPLETE.

Screen 4

```

GRAPHICAL ANALYSIS
OF EXPERIMENTAL DATA
FUNCTION SELECTION FOR T

PRESS   FOR
 1      T
 2      POW T
 3      SQRT
 4      SIN T
 5      COS T
 6      TAN T
 7      LOG T
 8      EXP T

SEE APPENDIX A
FOR FUNCTION DESCRIPTIONS
    
```

Screen 6

time, to eliminate particular data points, and then to use the prediction option of the program to project United States population for the years 2000 and 2025.

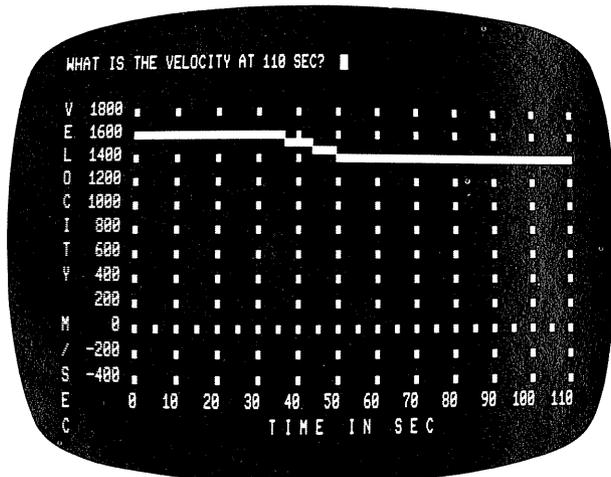
An Answer Key provided in the user's manual gives answers to all of the Selected Investigations problems. A Programming Guide also included in the manual outlines the structure of the Graphical Analysis of Experimental Data program.

## INTERPRETING GRAPHS IN PHYSICS

Richard Born designed the Interpreting Graphs in Physics program to fill a gap that he felt was not being filled by the available textbooks for high school physics.

Graphing the way objects move in space is a typical activity for physics students, and Born observed that students' understanding of position and velocity versus time is greatly aided by a graph, but he felt that the textbooks available did not provide enough graphs. To supplement the textbooks, Born therefore designed Interpreting Graphs in Physics, a program that generates graphs for students and asks questions about the graphs.

Once the Interpreting Graphs in Physics program is loaded and running, the computer presents two options. You can choose to work with a position versus time graph or a velocity versus time graph. If you type <P>, the screen will display a position versus time graph. If you type <V>, a velocity versus time graph like the one shown on screen seven will be displayed:



Screen 7

The computer will ask a series of five questions about the graph. For position graphs, these questions will be on position, speed, velocity, and displacement. You might, for example, be asked to identify the displacement of an object between two points in time. If you choose a velocity graph, you'll solve problems related to speed, velocity, displacement, and acceleration. For example, you might be asked to determine the average acceleration of an object between two points in time. Once you have answered five questions about a single graph, you'll be presented with a second option screen. Once again you'll have the option of requesting a position versus time graph or a velocity versus time graph. A third option—that of requesting a progress report—is also given. To see a report, you'd press <R>. Screen eight is an example of a progress report for a student who worked with one position graph and one velocity graph.

At this point, you could press <ENTER> to make a more detailed second half of the report appear (see screen nine).

Mary's PROGRESS REPORT			
PROBLEMS	P VS. T	V VS. T	COMBINED
NO. TRIED	5	5	10
% CORRECT 1ST TRY	60	80	70
% CORRECT 2ND TRY	40	0	20
% INCORRECT AFTER 2 TRIES	0	20	10

PRESS <ENTER> TO CONTINUE REPORT

Screen 8

When you pressed <ENTER> again, you'd be returned to the beginning of the program, where you could again select a position graph or a velocity graph.

CONCEPT SUMMARY					
*** POSITION VS. TIME GRAPHS ***	GENERAL	PROBLEMS ATTEMPTED	%-CORRECT 1ST TRY	%-CORRECT 2ND TRY	%-INCORRECT AFTER 2 TRIES
CONCEPT	POSITION	2	100	0	0
	SPEED	1	0	100	0
	VELOCITY	0	0	0	0
	DISPLACEMENT	2	50	50	0
*** VELOCITY VS. TIME GRAPHS ***					
	SPEED	1	100	0	0
	VELOCITY	0	0	0	0
	DISPLACEMENT	1	0	0	100
	ACCELERATION	3	100	0	0

PRESS <ENTER> TO BEGIN AGAIN  
PRESS <SHIFT> S TO STOP

Screen 9

Graphical Analysis of Experimental Data (26-1722) and Interpreting Graphs in Physics (26-1721) can be used with a TRS-80 Model I or Model III 16K tape system or 32K or 48K disk system. The suggested retail price of Graphical Analysis of Experimental Data is \$39.95, while Interpreting Graphs in Physics is listed at \$29.95. Prices may vary at individual stores and dealers. For more information, contact your local Radio Shack Store or Computer Center.



# The Southern New Jersey Computer Awareness Project

Glassboro State College  
Glassboro, New Jersey 08028

The Southern New Jersey Computer Awareness Project (SNJCAP) provides summer and academic year follow-up for middle and secondary school teachers whose districts indicate a need for faculty training in the development and implementation of a computer-related curriculum. The goals of the project are to increase the computer literacy and skills of the teachers, and to increase the teachers' knowledge about the implementation, integration, and use of education computing.

The project grew out of a need discovered in a 1980 survey of fifty high schools in Southern New Jersey; less than half of the schools were using computers in their instructional programs. While economic resources were a part of the problem, a major reason was that of untrained and inexperienced faculty. With SNJCAP's inception at Glassboro State College, area teachers, principals, and superintendents expressed a need for and support of a project that would provide training in the development and use of computer based programs and materials. Dr. Francis Masat and Dr. Tom Osler of the mathematics and computer science department of Glassboro State College worked together to develop the program after finding that South Jersey school districts had continually fallen behind North Jersey schools in technological areas.

The project, in its proposal stages, had the support of the New Jersey Educational Improvement Center-South, the New Jersey State Department of Education, and the Superintendents of Camden, Gloucester, and Cumberland counties. Subsequently, it was funded in part by the National Science Foundation.

An integral part of the project is the summer immersion experience that uses teaching and subject-related problems in a classroom and microcomputer laboratory situation. For four weeks the project participants attend morning group sessions on computer programming, and afternoon lab sessions on computer use and educational applications. The teachers use the Mathematics and Computer Science Department's Microcomputer Laboratory which has several TRS-80 microcomputer systems. With the use of TRS-80 systems, the teachers learn the logical structure of a computer, the algorithmic formulation of problems, and BASIC.

Software, including that available through Radio Shack, is used and evaluated. Each day's assignment and homework in the highly-accelerated program represents one week's worth of course work during a normal semester. The teachers get a lot of hands-on experience with the department's TRS-80 microcomputers. The teachers also investigate recent research and development in microcomputers and computer science education, and write and present projects that deal with developing and implementing computer awareness and use. These projects normally involve curricular and administrative applications, grant writing, and strategies and procedures for implementing computer-

based educational or administrative uses. Subsequently, the teachers prepare materials, projects, and curricula to use in their respective districts.

The thirty teachers presently in the program demonstrate the variety of uses that are beginning to occur in South Jersey schools. Although half of the teachers represent the fields of math and science, the other half come from such fields as business, gifted and talented, and compensatory education. But they all recognize the need for computer education for students of all ages and abilities.

The summer experience presents clear evidence that networks of microcomputers can offer more flexibility and output than large time-shared systems. Users do not compete for resources. If a CPU had failed, it would not have affected the other users. In fact, those teachers who had an opportunity to compare the department's time-sharing system with the microcomputer systems preferred the micros.

While the campus-based portion of the project happens during the summer, teachers, administrators and students return to the micro labs during the school year. The recent addition of a TRS-80 Network II has allowed many new and exciting applications to be developed. High school students and teachers have designed three-voice music programs, accounting and general ledger programs, CAI and Computer Managers Instruction (CMI) applications, and a variety of games complete with moving graphics.

To insure the project's goal of continued interest in computer education, site visits and roundtable discussions are used during the school year to keep in touch with teachers and find out how successful they have been in expanding or initiating interest in computer use in their districts. With the use of the TRS-80 microcomputers, the SNJCAP faculty and participants have helped to move many school districts into the computer age. Moreover, the impact is continuing and growing as the concepts, applications, and implications of microcomputers become part of the intellectual climate of the schools.



# Model II SCRIPSIT 1.0 vs Model II SCRIPSIT 2.0

*Or, What Does the Extra \$100.00 Do for Me?*

The following is a brief overview of the major differences between SCRIPSIT version 1.0 and 2.0.

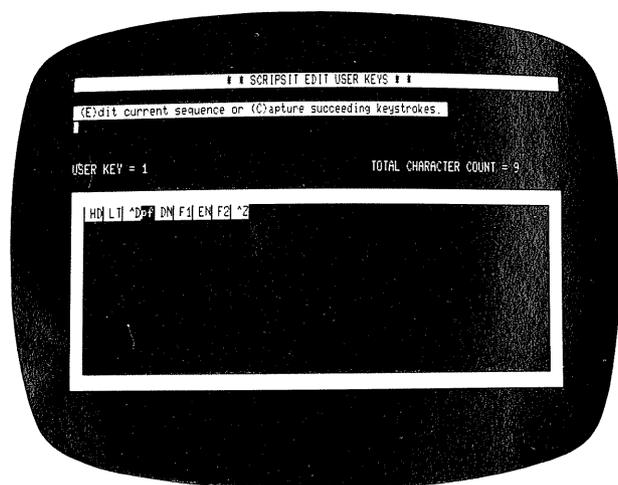
## USER DEFINED KEYS

SCRIPSIT 1.0 has five different user keys, each with a maximum length of 32 characters. These keys have to be created through disk utilities and there is no provision for editing. If you make a mistake, you have to start over defining that key. One user key can be linked to other user keys.

In SCRIPSIT 2.0, you have access to 20 different user keys. These keys can be defined to carry out a sequence of instructions, to display text and to do any number of things.

One example is a user key which can be defined to move all of the text files from one SCRIPSIT diskette to another. This type operation requires commands to be issued both within a document and at the SCRIPSIT main menu level.

Each of SCRIPSIT 2.0's user keys can contain up to 255 keystrokes, and other user keys can be called, or the key can call itself to create a recursive routine.



SCRIPSIT 2.0's user keys can be defined at either the main menu Utility level or at the document level, giving you the ability to define keys quickly and easily at any time during a SCRIPSIT session. There are two basic methods of creating user keys. In the first method you tell SCRIPSIT to capture all following keystrokes (until a Control Z or 255 characters are received). This is probably the most straight-forward method since it allows you to see the action of the keys being captured while you are defining the key. The second method allows you to edit a previously defined key by inserting or deleting keystrokes.

SCRIPSIT 2.0 user keys can be edited from either the main utility or from your document. This lets you customize keys for particular documents.

## PRINT FEATURES

SCRIPSIT 1.0 Special print codes are entered as a character (ex. '\_') or as a control code sequence. Code sequences use the backslash and/or tilde. This causes a problem when the user wants to print one of these characters: `_`, `|`, `\`, and `~`.

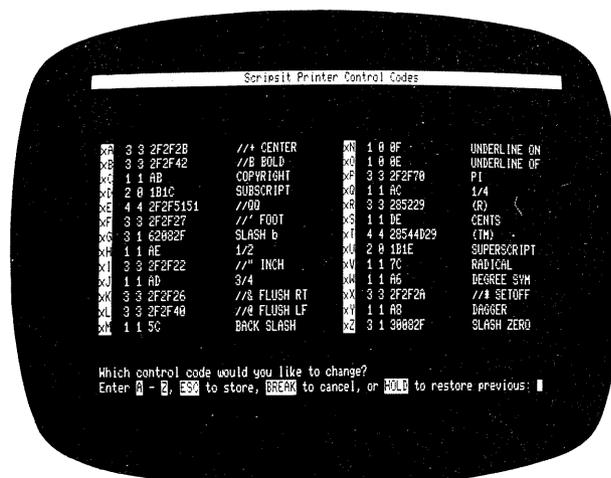
SCRIPSIT 2.0 print codes are entered as single characters and are displayed on the video using the Model II's graphic characters.

Printing special characters such as © and ™ in SCRIPSIT 1.0 requires patches to the program and is difficult for the operator to use.

In SCRIPSIT 2.0, the user can define up to 26 special printer control codes from the main utility menu. Each of the 26 special codes can have up to six ASCII codes to control printer functions. This gives you the ability to customize SCRIPSIT 2.0 to your particular printer, and use, if they are available, printer features like forward and reverse line feeds, multiple LPI (lines per inch) settings, or any other feature which your printer supports. You can also tell SCRIPSIT 2.0 how many (if any) of the characters in the printer control code are to be counted when justifying text that contains the defined character.

Line spacing in SCRIPSIT 1.0 can be set to 1-8 lines. In SCRIPSIT 2.0, you still use the digits 1-8, but they are now interpreted as:

- 1 - Single space, no blank lines
- 2 - Double space, 1 blank line
- 3 - Triple space, 2 blank lines
- 4 - 3 blank lines between each line printed.
- 5 - Half line spacing
- 6 - One and a half line spacing
- 7 - Two and a half line spacing
- 8 - Three and a half line spacing



Note that your printer must be capable of performing a half forward line feed for codes 5-8 to function.

## BACKGROUND PRINTING

In SCRIPSIT 1.0, to print in the background (print one document while you are working on a different document), the desired document must be opened and then captured to a disk file through the print routine and then printed from the disk file. There is no printer control (such as stopping the printer and then continuing or printing a document on single sheets of paper).

In SCRIPSIT 2.0, you can type or edit one document while printing a second document. When printing begins, the Print Monitor gives you three choices:

1. If you want to perform another SCRIPSIT function while printing is in progress, you press **<ESC>** to return to the directory. Once the directory appears on the screen, you can open or create another document to work on while printing is in progress. There are a few functions which can not be done while printing is in progress (like diskette BACKUPS or document Merge) but this will not prevent you from doing most of the things you will want to do while in a document.
2. You can stop the printing process.
3. You can continue printing after it has stopped.

If you have chosen to use background printing, SCRIPSIT will alert you if the printer needs attention (such as at the end of a page, ribbon out, etc.) by flashing "PRINT MONITOR" at the bottom of the screen. You can then return to the Print function in the normal manner. SCRIPSIT 2.0 also gives you a special printer code, which you can put in the text of your document. This code tells the printer to stop and alert you. This can be used to change print wheels on the Daisy Wheel II for special effects, or for any other purpose you might have.

## DIRECTORY FEATURES

In SCRIPSIT 1.0, ending a session is done through Disk Utilities using the Swap command. Exit to the operating system is not allowed. In SCRIPSIT 2.0, you are provided with an Exit option at the directory. After exiting you are given the choice of returning to SCRIPSIT, or exiting to TRSDOS.

SCRIPSIT 1.0 cursor movement on the directory screen is accomplished by use of the **<↑>**, **<↓>**, or **<ENTER>** keys. In

SCRIPSIT 2.0 these functions are still available, but they have been augmented. Each displayed directory entry is numbered and you may move directly to that document by pressing the appropriate number.

Another new feature of the SCRIPSIT 2.0 directory is that new documents are placed at the beginning of the directory where they are easily found rather than at the bottom of the directory as with SCRIPSIT 1.0

## BACKUP DISKETTE

In SCRIPSIT 1.0, the master diskette password was not required to backup a diskette. To better help you protect your information, the master diskette password is required in SCRIPSIT 2.0.

## MERGE

In SCRIPSIT 1.0 print prompts cannot be changed prior to merge except through Change Disk Defaults. SCRIPSIT 2.0 gives you a print document screen which lets Print parameters be changed prior to merging text.

## GET PAGE

By using the get page command in SCRIPSIT 1.0, the user can get the next page, the previous page, or a specified page by number. (CTRL G: N, P, or number). SCRIPSIT 2.0 adds two additional options, B for the Beginning page of the document, and E for the Ending page of the document.

## PRINT THE SCREEN

This feature not available on version 1.0. In SCRIPSIT 2.0, the video screen can be printed at anytime by pressing **<CTRL .>** (Control period). This allows the user to get a hard copy of Print Control Codes, User Defined Keys, Help text, menus etc.

## ESCAPE MENU

From an open document, pressing **<ESC>** in SCRIPSIT 1.0 displays a single menu on the bottom line. In SCRIPSIT 2.0, pressing **<ESC>** from an open document displays a menu that changes by pressing any key that is not an Escape Command.

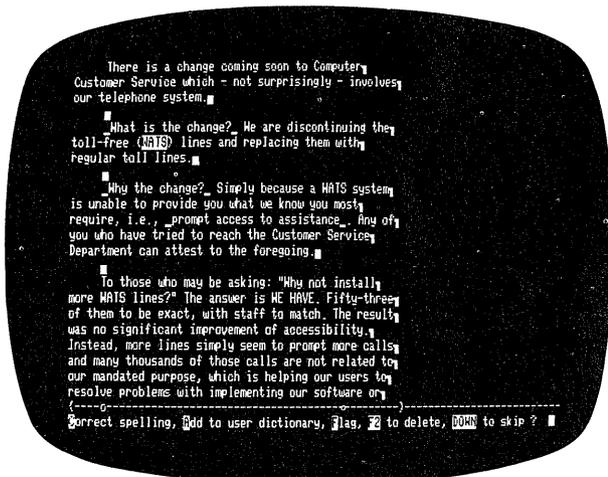
## RECALLING/SAVING FORMAT LINES

Format lines available for recall in SCRIPSIT 1.0 are 1 through 11 with no access to the default format line. When recalling a format line in SCRIPSIT 2.0, the choices are 1 through 11 or 'D' for default. If saving a format line, selection of 'D' will save the format as the new Create Document default.

## ENHANCEMENTS

SCRIPSIT's speed has been greatly increased. This is most noticeable when going from disk to disk in the directory, and page to page in a document. Another place where the speed increase is noticeable is in document printing. If you wish to print only pages 41-49 of a 50 page document, SCRIPSIT 1.0 had to display each page until it got to page 41 and began printing. SCRIPSIT 2.0 scans through the skipped pages very rapidly (without displaying them) and begins printing the requested pages almost immediately.

SCRIPSIT 2.0 has a new "cascading" margin feature for creating outlined documents. Pressing **<ESC>** **<→>** or



<ESC> <←> will reset the margins to the next tab position right or left respectively. A new paragraph will be started at the new position.

The system clock can be turned on or off from the directory. Also the current time can be printed on a document by enclosing a 'T' in braces - e.g. 00.25.09.

SCRIPSIT documents can be converted into ASCII text files and vice versa.

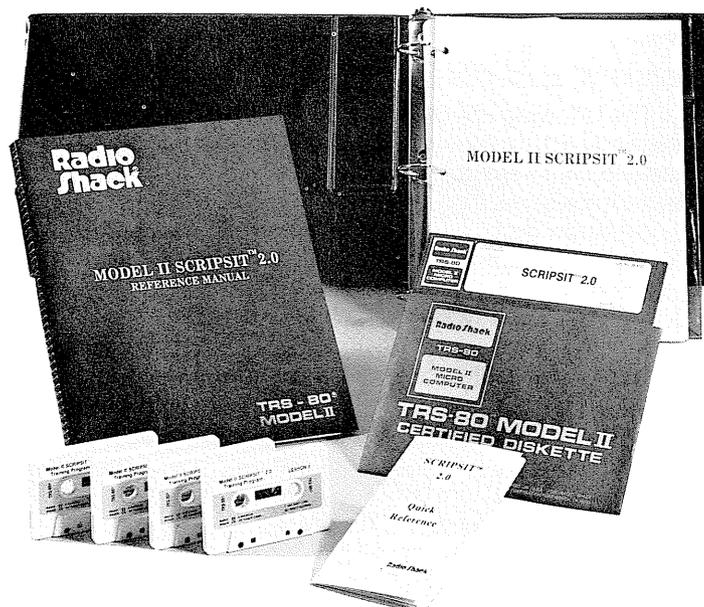
## UPDATING FROM OLD TO NEW

In order to use a version 1.0 document in version 2.0, the 1.0 document must be copied onto the 2.0 diskette in drive 0 from the 1.0 diskette in drive 1. After the copy has been made, the new copy of the document must be reformatted before using. Document copies between new and old versions cannot be made in a one drive system. A user key sequence can be set up which makes the copying of documents easy and 'painless'.

## SCRIPSIT DICTIONARY

SCRIPSIT 2.0 can be used with Radio Shack's new SCRIPSIT Spelling and Hyphenation Dictionary to save you time and increase the accuracy of your documents. The Dictionary program quickly checks your documents for spelling errors by comparing the words first to a Master List of over 100,000 words and then to a specialized User List (which you create for words not in the Master List). The program can also hyphenate your document, correctly dividing words between syllables.

The specialized User List enables you to personalize the program so that it will recognize words that are a part of your specialized vocabulary. This User List holds up to 2047 entries.



# Statistical Analysis

**W.W. Schumacher**  
**KEMPER GROUP**  
3 Huntington Quadrangle  
Melville, NY 11747

The Loss Control Engineering Department at KEMPER GROUP is currently studying staffing needs as they relate to our field inspection activities. As might be expected, a large amount of "number crunching" is involved with 19 branch staffs functioning nationwide

I have been looking at the methods and information available from our Home Office and branches in an attempt to confirm or deny the validity of both. Your Statistical Analysis package (Cat. #26-4540) for Model II has been most useful, however the correlation matrix program (CORRL/BAS - Version 2.5) contains several bugs, or perhaps oddities, that are most annoying, especially when printouts are sent to our people in Chicago or Syracuse. Since the problem is spelled out in the manual, but not noted as a bug, it appears to be intentional.

Please refer to the attached copy which states in part, "The variable numbers which appear on your output will be sequential (1 through the number of variables in the analysis) and are not related to those appearing on the file characteristics display."

This, in itself is odd, however, what is not stated is even more important.

The input routine asks how many variables are to be correlated (from 1 to 9) and then asks the user to input the variable numbers. This appears to be straight forward, but it is not!

In addition to using sequential numbers, as stated in the manual, the program uses sequential variable names. The result is that the output displays variable names and numbers that do not correspond to the data. Everything in the output is mislabeled, except in one specific case. The exception is the case where X variables are selected as variables 1 through X.

There is a fix and I cannot help but wonder why it was not included in the original program. Because of the manual comment, it seems the problem was intentional, but the logic escapes me.

In line 2115, the selected variables are input as a subscripted variable,  $WV\%(J)$ . Unfortunately, when the variables and variable names are printed in the PRINT and LPRINT routines, the variables are identified as "J" and the variable names are identified as  $VN\$(J)$ . The program will not and cannot print what was requested by the input statement.

The solution is simple, however, because of a lack of REMs, it took several hours to determine how the author set up the program. (Actually, the excessive time was spent trying to figure out why the method was intentional rather than accidental.)

The PRINT and LPRINT routines should be altered to print the Jth item of  $WV\%$  rather than "J" and  $VN\$(J)$ . Four program lines need to be changed as follows:

```
3360 - Change J to WV%(J)
4030 - Change J to WV%(J)
      Change VN$(J) to VN$(WV%(J))
5010 - Change J to WV%(J)
5110 - Change J to WV%(J)
      Change VN$(J) to VN$(WV%(J))
```

I have enclosed a complete listing of the program with the changes incorporated and underlined in red. The variable "J" was not altered in the FOR:NEXT loops, rather, only in the PRINT (or LPRINT) statements.

No changes were required in the plotting routine starting at 9000, however users should be aware of a potential problem in 9000 and 9010. When asked which variables are to be plotted on the X and Y axis, the user must use the numbers of the variables as they appear on the screen rather than the sequential numbers previously generated in the program.

This is the way it would be logically expected, but after making the above changes, it may not be obvious that they would be reflected here without additional changes.

Had the original version printed out the proper variable names, I could have lived with the sequential numbering. The fact that output will be sent far and wide eliminates the possibility of co-existing with improper variable names. It would create too much confusion and require too many explanations.

I recommend that other users of this package be informed of this situation and the solution in the "Microcomputer News" as soon as possible. An errata sheet or reprinted page should be put in the manual at an early date, and future releases of the program should be upgraded.

Probably your software development people or another user can develop a solution more elegant than mine and if so, it should be put in the field as soon as possible.

With the seemingly prompt bugs, errors, and fixes published routinely, I am somewhat surprised that I have seen no reference to this particular problem before. I do not think it would go uncommented for long by your astute audience.

From page 43 of the manual:

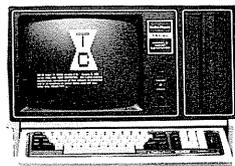
For each variable, the computer expects you to enter a variable number displayed on the screen. These numbers are used only for telling the computer exactly which variables it needs to read from disk. The variable numbers which will appear on your output will be sequential (1 through the number of variables in the analysis) and are not related to those appearing on the file characteristics display.

#### SUGGESTED PROGRAM CHANGES:

```

75 ' Version 2.5 - MODIFIED PRINT/LPRINT
  ROUTINES ON 5 NOVEMBER, 1981
3360 PRINT "VAR. # ";
  : FOR J=1 TO NV
  : PRINT USING "#####"; WV%(J);
  : NEXT J
  : PRINT
4030 PRINT USING "#####"; WV%(J);
  : PRINT TAB(21); VN$(WV%(J)); TAB(45);
5010 LPRINT STRING$(79, "-")
  : LPRINT "VAR. # ";
  : FOR J=1 TO NV
  : LPRINT USING "#####"; WV%(J);
  : NEXT J
  : LPRINT " "
5110 FOR J=1 TO NV
  : LPRINT USING "#####"; WV%(J);
  : LPRINT TAB(22); VN$(WV%(J)); TAB(45);

```



# MODEL II Bugs, Errors and Fixes

## ACCOUNTS PAYABLE (26-4505)

The Model II Accounts Payable program (26-4505 Version 1.0) will allow you to add more than one vendor with the same name, but will only process one of the vendor accounts. The following changes should be made to the program to prevent the addition of two or more vendors with the same name.

1. Backup the diskette(s) and make this change on the backup copy of the program.
2. In BASIC load the program by typing:  
LOAD"APS/BAS" <ENTER>
3. Add the following new line to the program:  
1703 IF VV=NV THEN PRINT@(3,50), R\$ " ALREADY  
THERE " N\$;: ER=1: GOSUB 1690: RETURN
4. Save the changes by typing:  
SAVE"APS/BAS" <ENTER>
5. At TRSDOS Ready, make a backup copy of the corrected diskette.

## RSCOBOL (26-4703) and RUNCOBOL (26-4704)

When using variable length records in ISAM files, RSCOBOL always displays an ERROR 97 when a WRITE record is attempted. Following the ERROR 97, COBOL returns immediately to TRSDOS Ready. To correct this condition, use the following patches:

```

PATCH RUNCOBOL A=3EDD F=229B4A C=CDE26F
PATCH RUNCOBOL A=6FE2 F=0000000000000000
C=229B4A22934AC9

```

Also, when using variable length records, the "OCCURS DEPENDING ON" clause must not be specified at level 01. This is stated in the manual, but it is not obvious when reading casually.

## Revisions, Enhancements, Etc.

The following is a list of current stock numbers for Computer Software revisions, enhancements and some miscellaneous items. These items can be ordered through your local Radio Shack store. If no price is shown, the item is available at no cost upon proof of purchase of the original program.

700-2014 Accounts Receivable (26-4504)	N/C
Version 1.1 on TRSDOS 1.2a	
700-2017 FORTRAN on TRSDOS 2.0a (26-4701)	N/C
700-2018 EDTASM on TRSDOS 2.0a (26-4702)	N/C
700-2024 Accounts Receivable (26-4504)	N/C
on TRSDOS 2.0a	
700-2111 Tandygraph disk for 26-1191 Multi-pen Plotter	N/C
700-3013 Profile II on TRSDOS 2.0a (26-4512)	N/C
700-3014 General Ledger (26-4501)	\$15.00
on TRSDOS 2.0a	
700-3015 Payroll (26-4503) on TRSDOS 2.0a	\$25.00
700-3016 Accounts Payable (26-4505)	\$15.00
on TRSDOS 2.0a	
700-5001 Accounts Receivable Source Code for 26-4504 Version 1.1 on TRSDOS 1.2a	\$299.00
700-6002 TRSDOS 2.0a Upgrade (26-4910)	\$24.95
TRSDOS 2.0a Disk & Replacement pages for manual.	
700-6005 Scripsit 2.0 Upgrade (26-4531)	\$100.00

# Step-Saving Ideas

*Time Calculation Algorithm and Saving Steps with the Pocket Computer*

Leonard M. Levine  
10 Fifth Avenue  
New York, NY 10011

As my first contribution to Microcomputer News, may I suggest the following:

In RUN or DEFINE mode, key:

T = 0, T = DEG + 0.00 + T, DMS T

Now hit <ENTER> to set T (time) to zero. Hit <→> or <←> to retrieve the formula and hit <SPC> four times to remove the initialization of T from this (recursive) time-calculation algorithm. Now hit <→> four times and you are ready to enter minutes, to the left of the decimal point and seconds to the right! Try this for summing cuts on a phonograph record for tape recording appointments or phone calls, etc. The first "+" may be changed to a minus or, for simplicity, eliminated. In RESERVE mode it uses 18 steps and will not take up even one of the 1424 PROGRAM steps!

## SO LITTLE TIME, SO MUCH TO SAY . . .

1. Thanks for the PC (I call it that with great affection and no little respect) and thanks also for the Microcomputer News.
2. I bought the PC in January, 1981 but just learned of the Microcomputer News. The May, 1981 Pocket Computer Product Line Manager's News is very meaty; I am still digesting the meal!

## SOME FURTHER FOOD FOR THOUGHT —

3. Key in:

```
950PRINT"THIS IS A TEST OF THE":PA.QI. <ENTER>
```

Now cursor over to the Q and key a " over it. Next cursor to the end (I use EOL to mean "end-of-line") and add the rest of the line (as on page 21 of the May MN). Now RUN the line (RUN or DEF mode); note that there are 14 spaces left to key on line 950 (6 more!); RUN it again (you will probably want to change the PAUSE to a PRINT by now); note that it takes up only 65 bytes (including the 3-byte overhead) out of the 1424; RUN . . .

That is right! The characters [INPUT] appear on the instruction line and the execution PRINT line but take up only 1 (one) of the 79 positions on the instruction line and only 1 of the 1424 MEMORY steps! To paraphrase an old bromide, for the PC more than any other computer, "A step saved is a step earned."

Of course this works just as well for BEEP, RETURN, DEGREE, NEXT, LIST, END, LOG and all the others, many of which are useful in string constants or variables(!) as part of INPUT, PAUSE, PRINT statements or even as labels. Each use saves from three to six bytes of MEMORY. There is more to it; play with it and see. For example, key:

```
300CS=QI.
310AS=QN.P.L.
320P.CS;AS
```

Now replace the Q's with "s, note the MEMORY savings and RUN. Also try:

```
200G.QC.
210P."FAIL
220QC.Q B.2:P."OKAY!
```

Replace the three Q's as above and watch it RUN. (I usually omit quotes, at EOL only—"A step saved is . . ." There are cautions to be observed here also. Play with it.)

4. While we are on the subject of step-saving, see what happens when the line at the top of the second column (same issue, same page) is changed to:

```
100: X=(B>0) + (C>0: IF X=1 BEEP2
```

A step saved . . . Even within a multiple assignment, e.g.:

```
555: A=INT (B * (C + D, E=F * (COS (G + H * (J + 2π, K=...
```

the PC closes up all pending right parens for you! In calculator mode, too. A step . . . , well, you get the idea by now.

*Editor's Note: While PC-1 will close pending right parens as indicated, our programmers warn against the practice. To paraphrase one programmer, the PC may ambush the user if this is tried.*

5. Still speaking of "anti-inflationary" programming, the PC manual does mention (bottom of page 51) the kind of LET statement I used above but the GAMES I Package, Cat. No. 26-3515, uses colons instead of commas. Commas cost nothing extra and their use does save about three milliseconds per assignment; in some programs this adds up to real savings (I am as sick of that old saw "a second saved . . ." as you are by now).

For instance, the PC will solve the time honored mathematical problem posed by British mathematician G.H. Hardy: find the smallest integer expressible as the sum of two cubes in two different ways (answer: if  $N = X^3 + Y^3$  Then  $N = 1729$ ,  $X(1) = 1$ ,  $Y(1) = 12$ ;  $X(2) = 9$ ,  $Y(2) = 10$ ). However, it sits there thinking for an hour and forty minutes before beeping happily and displaying the right answer. This is about an hour and thirty-nine minutes more than the Color Computer requires and an hour and an forty minutes more than a large-scale IBM, so every millisecond counts!

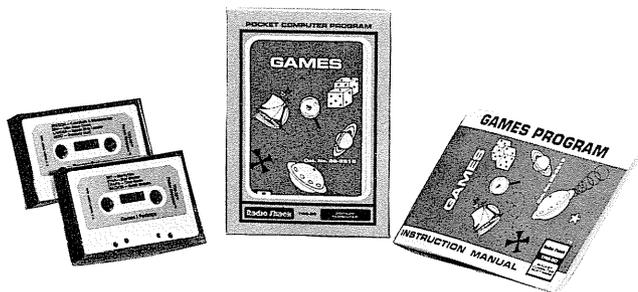
I would love to see a faster PC solution from a Microcomputer News reader, or any solution for that matter. It is not easy. I will send you my program and other programs of more practical value just as soon as I can get, or get at, one of those PC printers I have been hearing about.

6. Really enjoyed that Games I Package. Gave my dealer a correction to the Star Blaster program and write-up which he sent to One Tandy Center in March. No word yet. "The mills of the gods . . ." Here it is again, and one for Missionaries and Cannibals (not the one mentioned on the error-insert 875-9068 that came with the Package):

Star Blaster, page 17, "Change Pattern" chart, center-left section should be: and line 110 should be changed so the "U=2.7" reads "U=1.7"

Missionaries: line 165 is missing a quote after "BOAT STOLEN".

7. Back to the May column. Some readers may still be confused by your example showing that "... the line number takes up exactly as many steps as there are digits." When line 950 is edited so that the line number is 95 you do get another position on the line. You do not get another byte (step) of MEMory, i.e. one of the 1424. Every line, regardless of the number of digits in the line number, uses three steps (bytes) of MEMory before you have keyed in the first character following the line number. Two of those steps are for the line number itself and the other, though the manual does not say it in so many words, represents EOL when <ENTER> is keyed. A very minor point - the program in the second column will run just as well without line 110. I know it is an illustration; please pardon the proofreader in me.



## Multiple Linear Regression

Garland Bryant  
Kirkland, Washington

On March 1st, 1981, I purchased the Pocket Computer and Recorder. I will share with you and the readers one of my more useful programs. Using standard text book statistics (Multiple Linear Regression), it provides an and/or choice of a 2nd or 3rd order polynomial curve-fit for a two dimensional data array, i.e., one dependent and one independent variable. The notation is:

$$\hat{Y} = A + B_1X + B_2X^2$$

or

$$\hat{Y} = A + B_1X + B_2X^2 + B_3X^3$$

where  $\hat{Y}$  (Y hat) is the statisticians symbol for "best value".

In the program listing, line 10 is for data entry and lines 140 and 150 store the cumulative sums of squares. Notice that I've added line 139 to provide data print-out, but it is a nuisance when the program is being run without the printer. My only solution to this is to change GOTO 139 in line 10 to GOTO 140 when the printer is not being used. Any comments? Lines 20 through 100 calculate the residual sum of squares, the statistical parameters and the constants and coefficients for the 3rd order polynomial. Lines 160 and 170 calculate the same for the 2nd order fit. Lines 110 and 115 and lines 180 and 185 print out the program results. R

squared is the square of the correlation coefficient (unadjusted). A value of 1.000 . . . indicates a perfect fit; a value of zero, no connection between X and Y.

"S" is the residual standard deviation. It is an interval in a plus or minus sense about the mean line (our calculated curve) which contains 68% of the data points. Lines 120 and 190 permit numerical evaluation of the polynomial. Do not enter values of X outside the range of the data. This is an unreliable extrapolation of the data. Any combination of plus and minus values may be used as input data. One must use a minimum of four data pairs for the 2nd order polynomial and five for the 3rd order polynomial. There is no limit to the number of data points except register capacity. Lines 130 and 200 show how to recall curve data if desired. This is not necessary with the printer. Lines 132 and 205 allow selection of the other polynomial (if desired) without re-entering the data.

As an example I have chosen eight data pairs taken from an aircraft engine test stand. The smaller of the numbers is EPR, the power setting parameter and the larger, the gross thrust of the engine. R<sup>2</sup> (R squared) is .9999 which signifies very highly correlated data. (It should be—we are dealing with a machine.) The standard deviation is 14 lbs. Degrees of freedom are 5. This is the number of data points minus the number of constants in the fitted curve—in this case three: A, B<sub>1</sub> and B<sub>2</sub>. In the program, this number is used as a divisor which explains the minimum number of data points.

From previous experience with the rather complex test facility, I know that S should be 15. A value of 20 is acceptable but I'm not overly enthused. If I see a value of, say 30, I'm jumping up and down and screaming. The appropriate experts must comb the system.

The Pocket Computer is 100% dedicated to me and I can make these computations quickly without dependence on telephone lines or the big computer expert (it's two o'clock Saturday morning and he's away on a camping trip 200 miles away and the other one is on sick leave.)

I have also run a third order fit here. It can be seen from the deterioration of R<sup>2</sup> (S has gone completely to pot) that the 3rd order fit is just not the one to use on this particular data. Of course, in the general case, it's quite possible that neither the 2nd or 3rd order fit will produce a satisfactory fit for a given purpose.

### THE EXAMPLE (2nd ORDER)

NEW DATA	
X=?	1.229
Y=?	3892.
X=?	1.298
Y=?	4992.
X=?	1.327
Y=?	5422.
X=?	1.407
Y=?	6654.
X=?	1.551
Y=?	8752.
X=?	1.23
Y=?	3941.
X=?	1.299
Y=?	5020.
X=?	1.327
Y=?	5438.

```

<BREAK>
RUN 20 <ENTER>

2ND OR 3RD POLY?(2OR3) 2<ENTER>

```

```

R SQD= 9.999417804E-01
S= 14.2421782
DF= 5.
A= -19329.07277
B1= 21974.23452
B2= -2494.470727
X=1.229
Y HAT= 3909.510596
X=1.298
Y HAT= 4990.789378
X=1.327
Y HAT= 5438.150595
X=1.407
Y HAT= 6650.498719
X=1.551
Y HAT= 8752.263695
X=1.23
Y HAT= 3925.350927
X=1.299
Y HAT= 5006.285472
X=1.327
Y HAT= 5438.150595
RUN180 FOR CURVE DATA
RUN 32 FOR OTHER POLY

```

```
RUN 32 <ENTER>
```

```
2ND OR 3RD POLY?(2OR3) 3<ENTER>
```

```

R SQD= 9.913099265E-01
S= 194.5398198
DF= 4.
A= -19801.07329
B1= 23246.75325
B2= -3506.493506
B3= 242.4242424

```

### THE PROGRAM

```

1 "NEW DATA"
  : CLEAR
2 PAUSE "NEW DATA"
10 INPUT "X=?"; X, "Y=?"; Y
  : GOTO 139
20 M=C-BB/A
  : O=E-CC/A
  : Q=G-DD/A
  : P=F-CD/A
  : A(36)=E-BD/A
30 N=D-BC/A
  : W=J-BH/A
  : Z=K-CH/A
  : A(27)=L-DH/A
  : U=I-HH/A
32 INPUT "2ND OR 3RD POLY?(2OR3)"; A(37)
33 IF A(37)=2 GOTO 160
40 A(28)=MOQ+ 2NPA(36)- (A(36)A(36)O+ PPM+ QNN
50 A(29)=(WOQ+ ZPA(36) +A(27)NP- (A(27)A(36)O+
  PPW+ QNZ))/ A(28)
60 A(30)=(MZQ+ NA(27)A(36)+ WPA(36)-
  (A(36)A(36)Z+ A(27)PM+ NWQ))/ A(28)
70 A(31)=(MOA(27)+ NPW+ A(36)ZN- (A(36)OW+ PZM+
  A(27)NN))/ A(28)

```

```

80 A(32)=H/ A- A(29)B/ A- A(30)C/ A- A(31)D/ A
  : A(33)=A-4
90 A(35)=U- (A(29)W+ A(30)Z+ A(31)A(27))
95 R=1- (A(35)/ U)
100 S= (A(35)/ A(33))
110 PRINT "R SQD="; R
  : PRINT "S="; S
  : PRINT "DF="; A(33)
  : PRINT "A="; A(32)
115 PRINT "B1="; A(29)
  : PRINT "B2="; A(30)
  : PRINT "B3="; A(31)
120 INPUT "X=?"; V
  : PRINT "X="; V
  : A(34)=A(32)+ A(29)V+ A(30)VV+ A(31)VVV
  : PRINT "Y HAT="; A(34)
  : GOTO 120
130 PRINT "RUN 110 FOR CURVE DATA"
132 PRINT "RUN 32 FOR OTHER POLY"
  : STOP
135 GOTO 1
139 PRINT X, Y
140 A=A+1
  : B=B+ X
  : C=C+ XX
  : D=D+ XXX
  : E=E+ XXXX
  : F=F+ XXXXX
  : G=G+ XXXXXX
150 H=H+Y
  : I=I+ YY
  : J=J+ XY
  : K=K+ XXY
  : L=L+ XXXY
  : GOTO 10
160 A(28)=MO- NN
  : A(29)=(WO- NZ)/ A(28)
  : A(30)=(MZ- NW)/ A(28)
165 A(32)=H/ A- A(29)B/ A- A(30)C/ A
170 R=1- ((U- (A(29)W+ A(30)Z))/ U)
  : A(33)=A- 3
  : S= ((U- (A(29)W+ A(30)Z))/ A(33))
180 PRINT "R SQD="; R
  : PRINT "S="; S
  : PRINT "DF="; A(33)
  : PRINT "A="; A(32)
185 PRINT "B1="; A(29)
  : PRINT "B2="; A(30)
190 INPUT "X=?"; V
  : PRINT "X="; V
  : A(34)=A(32)+ A(29)V+ A(30)VV
  : PRINT "Y HAT="; A(34)
  : GOTO 190
200 PRINT "RUN180 FOR CURVE DATA"
205 PRINT "RUN 32 FOR OTHER POLY"
  : STOP
210 GOTO 1

```

## Pocket Computer Bugs, Errors and Fixes

*Revisions, Enhancements, Etc.*

The following is the stock number for a Pocket Computer Software revision. This item can be ordered through your local Radio Shack store and is available at no cost upon proof of purchase of the original program.

700-2400 Real Estate Tape 2 (26-3510 Ver 1.1) N/C  
 Corrects part B of REAL4 where the total amortization was not including the first month's payment.

# Compound Miters

Bruce Taub  
75 Captains Road  
North Woodmere, NY 11581

I happen to own a Texas Instruments TI-58, and a Radio Shack TRS-80 Pocket Computer. I enjoy using the PC 100% more than the TI-58. The PC is compact, has varied programs, constant memory (even when the unit is off), and is reasonably priced.

Recently I created a program for the PC to cut down the calculating time in any wood-working shop where you have to calculate saw angles for compound miters. All the angles are printed in degrees. If you are using a table saw, substitute the miter angle for the arm angle.

## PROGRAM DESCRIPTION

SHFTA—The PC takes the amount of sides and finds the number of degrees of the interior angle. The PC then calculates the tangent of the interior angle. Using another variable, the PC converts the tangent of the interior angle to the cotangent. The PC calculates the sine of the slope and multiplies this with the cotangent. The arctangent of the result is the ARM ANGLE.

SHFT B—The PC takes the amount of sides and finds the number of degrees in the interior angle. The PC then calculates the cosine of the interior angle. Once the slope angle is inputted the PC finds its cosine. After these two variables are multiplied together, the PC takes the arcsine of the answer and gives you the TILT ANGLE.

I hope that this program is as helpful to you as it is to me! Good-luck and happy computing.

## THE PROGRAM

```
5: PAUSE "SAW ANGLE SETTINGS FOR:"
  : PAUSE "    COMPOUND MITERS"
10:"Z" PAUSE "ARM ANGLE      - SHFT A"
20: PAUSE "BLADE TILT ANGLE- SHFT B"
25: PAUSE "END PROGRAM      - SHFT K"
  : END
30:"A" INPUT "# OF SIDES? "; N
  : A=90-180/N
40: B=TAN A
50: C=1/B
60: INPUT "SLOPE ANGLE? "; S
70: D=SIN S
80: J=C*D
90: Z=ATN J
  : H=Z
  : M=(H-INT (H))
  : IF M>.55 LET Z=Z+.05
100: PAUSE "THE ARM ANGLE IS"
  : PRINT USING "###.#"; Z
  : GOTO 10
110:"B" INPUT "# OF SIDES? "; N
  : A=90-180/N
120: F=COS A
130: INPUT "SLOPE ANGLE? "; X
  : Y=COS X
140: O=F*Y
150: Q=ASN O
  : H=Q
  : M=(H-INT (H))
  : IF M>.55 LET Q=Q+.05
160: PAUSE "THE TILT ANGLE IS "
  : PRINT USING "###.#"; Q
  : GOTO 10
210:"K" IF V=0 PAUSE ">END "
  : END
```

# Pythagoras' Theorem

Joseph Strolin  
21 Ellen Street  
Norwalk, CT 06851

Enclosed find a simple little program I wrote for the TRS-80 Pocket Computer. Use it any way you wish. It makes a nice small program with graphics!

```
10: PRINT "PYTHAGORAS, "
20: PRINT "THEOREM"
30: PRINT "BY JOE STROLIN"
40: PRINT "    *"
50: PRINT "    ***"
60: PRINT "    *  *"
70: PRINT "C=?  *  *"
80: PRINT "    *  * B"
90: PRINT "    *  *"
100: PRINT "    *  *"
110: PRINT "    *  *"
120: PRINT "*****"
130: PRINT "    A"
140: PRINT "SIDE A=?"
150: INPUT H
160: PRINT "SIDE B=?"
170: INPUT I
180: J=√(H*I)
190: PRINT " "
200: PRINT "SIDE A="; H
210: PRINT "SIDE B="; I
220: PRINT "SIDE C="; J
230: PRINT " "
240: PRINT " "
250: END
```

# Cub Scout Song

Jules L. Schafer  
3314 Cottman Avenue  
Philadelphia, Penna. 19149

This program is dedicated to all former and present Cub Scouts, their Parents, Den Leaders, and Pack Masters. It is sure to make the parents and Cub Leaders groan as they recall hearing this song, line by line, from 100 bottles down to one and then back up to 100.

It may be of interest to note that the program works, repeating from 100 down to one and back up to one hundred again with TWO FOR statements, but only ONE NEXT statement. It may also be of aid for those who are awed by computers: You can make the computer do what you want, for as long as you want, no matter how silly what you make it do is.

```
10 FOR A=100 TO 1 STEP -1
20 PAUSE A; "BOTTLES OF BEER"
30 PAUSE "ON THE WALL"
40 PAUSE A; "BOTTLES OF BEER"
50 PAUSE "ON THE WALL"
60 PAUSE A; "BOTTLES OF BEER"
70 PAUSE "TAKE ONE DOWN"
80 PAUSE "PASS IT AROUND"
90 NEXT A
100 FOR A=1 TO 100
110 GOTO 20
```

# Questions and Answers

This month I will continue answering some of the more frequently asked questions that we receive.

Question: "How can I figure out the addresses of a machine language program so that it can be saved to disk and tape?"

Answer: First, I assume you are trying to duplicate a program for BACKUP and archival purposes, and not for unauthorized distribution of copyrighted material.

Memory locations &H01DA to &H01E1 contain the name of your program loaded from tape. Memory locations &H01E5 to &H01E6 contain the EXECute address. Memory locations &H01E7 to &H01E8 contain the load or START address. All that is left for you to determine is the length of the program (or the ENDing address). To figure out what the proper values are, PEEK the locations (don't forget the "&H" in front of the numbers). The values returned are in decimal. You will need to convert the name to ASCII values using the CHR\$( ) function of standard BASIC. The START and EXECute address can be used in either decimal or converted to hexadecimal. If you are unsure of how large the program is, simply dump from the START address to the end of memory.

ADDRESS	CONTENTS
&H01DA-&H01E1	Eight Character Program Name (padded on right with spaces)
&H01E5, &H01E6	EXEC ADDRESS (MSB, LSB)
&H01E7, &H01E8	START ADDRESS (MSB, LSB)

This method will work with "normal" programs. If the program was written to defeat duplication (sometimes they do that you know), the programmer may have done something strange in ROM or in low RAM and you may not be able to duplicate the material without a lot of work figuring out what the programmer did.

Note: \*\*Unless you write your own driver routine, you will need Extended BASIC to CSAVEM a machine language program on tape.\*\*

One other thing to remember: If the program you are loading is on tape and you're going to save it to disk, you may need to offset the three addresses (Start, End, Execute) by about 2050 bytes so the program does not get garbled with the disk buffer.

Question: "How do you handle files on a micro system?"

Answer: Depending on whether you are using tape or a disk system as the storage medium, you have a choice of either Random (Direct) Access or Sequential Access Files. Tape storage offers only sequential files, while disk storage offers either or both. I have found that manipulating data in memory is a lot faster than doing it on a storage medium, but you usually run out of memory before you are finished. With the disk system, I have found that using a random file with a sequential index file (see next question) is the fastest, most economical method for my purposes.

Question: "How might a disk mailing list which prints out labels be set up?"

Answer: There are many different ways this could be accomplished. The following should get you started towards creating a disk mailing list. The easiest method I've found to manipulate data is by using Random Access Files (also called Direct Access Files). I also want to sort quickly, so I also have a sequential file which I keep in-memory. To start, I determine the various fields (e.g. name, address, city/state, zip, phone number.) Since writing small programs is easier than writing long, involved programs, write each function (add, delete, update, print) as a separate module. When all modules are completed, you can write a short Menu program to select the one you need.

The "add" module would simply display the various fields involved, allow input to each one of the fields, and then create one large string from the various data input and write this string to a file on the disk. The "print" function could be done much the same way, by loading in the long string and then breaking it down to the various fields involved. You can then select the information you want printed, format the manner in which you want it printed, etc.

The "update" module tends to be a little more tricky. Using the MID\$ function, you can replace data in the middle of a string without affecting the surrounding data. Each field must remain the same length. However, once you have set up the "add" module, the update module can use many of the program lines, input routines, field formats, etc.

The next major step is to decide which fields are to be used for sorting. Using a sequential file, you read the record number and the portion of the string which is going to be used for sorting (like name or zip). Using a two dimension array, you can sort the field you've selected and save it in another file, saving both the field and the corresponding record number. (This enables you to go directly to the record in the random file.) Each time you load a module, you will load this sequential file and store it in memory. Whenever you want to find a specific record, a simple FOR-NEXT loop will go through the array in memory, find the related record number, open the file and get the information. Sound confusing? Well, take it one module at a time. Be sure each module is debugged before proceeding with the next module. Use the disk for storing and retrieving the modules, as memory can be limited depending on the amount of information you want to save in each record.

Question: "How do I set up cassette files to migrate to disk if I ever want to?"

Answer: This is not a problem with the Color Computer Disk system. The syntax (or sentence structure) for writing data to tape is almost the same as writing sequential data to the disk. Instead of OPENing device #-1, you OPEN device #1. To convert your existing data files to disk, simply read them as you would normally, then write them out to disk changing the OPEN statement and the PRINT # statement.

After you have created your disk data file, change the OPEN statement and INPUT # statement to read from disk instead of tape. Remember we said that the Color Disk was "transparent" to the user, and that is what we meant! The only difference in creating/maintaining disk files as opposed to tape files is the device you specify when you open the buffer to the device. (-1 for tape, and 1 for disk.)

Hopefully these questions and answers have been of use to you. There are more in the works. (The questions come faster than the answers).

## Color Computer and Videotex Bugs, Errors and Fixes

*Revisions, Enhancements, Etc.*

The following is a list of current stock numbers for Computer Software revisions, enhancements and some miscellaneous items. These items can be ordered through your local Radio Shack store. If no price is shown, the item is available at no cost upon proof of purchase of the original program.

### COLOR COMPUTER

- 700-2013 8 Bit Printer Driver for CC N/C  
(LP VII, VIII) Driver which allows dot addressable graphic operation of LP VII and VIII. This program is needed by owners of the Color Computer with the standard 1.0 ROM only. (NOT required to operate 26-3021, Screen Print Program.)
- 700-2113 Tandygraph disk for 26-1191 N/C  
Multi-pen Plotter

### VIDEOTEX

- 700-2300 Dow Jones Service for any Videotex. N/C  
This software is for Videotex owners whose Videotex packages did not contain the Dow Jones Service.

## Merge Cassette Programs

Jorge Mir  
12851 W. Balboa Dr.  
Berlin, Wis. 53151

Here is a simple way to merge BASIC programs for the Color Computer.

1. Type PCLEAR 1 to clear as much memory as possible.
2. CLOAD the first program into memory.
3. RENUM the program to conserve space and to lower size of the largest line number. Be sure to write down the largest line number in the newly renumbered program. I use RENUM 100,,1 (don't forget the two commas.)

4. Make a note of where the first BASIC program begins by writing down the values from:

```
PRINT PEEK(25), PEEK(26) <ENTER>
```

The values for a typical program will be 12 and 1.

5. Change the BASIC beginning of program pointers to the end of the first program by typing:

```
A=PEEK(31): POKE 25, A <ENTER>
A=PEEK(32)-2: POKE 26, A <ENTER>
```

6. CLOAD the second program into memory.
7. RENUM the second program, making sure that the starting line number for the second program is larger than the ending number you wrote down for the first program in step 3. Use:

```
RENUM xxx,,1 <ENTER>
```

(where xxx is the new starting line number for the second program.)

8. Change the BASIC beginning of program pointer back to its original value (the values you wrote down in step 4.) If the values were 12 and 1, the command sequence would be:

```
POKE 25, 12: POKE 26, 1 <ENTER>
```

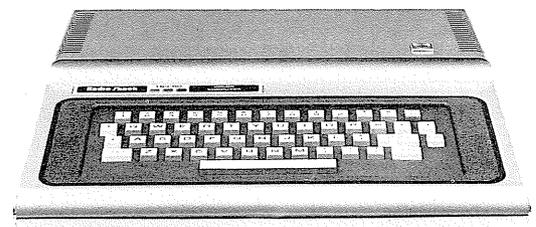
9. You are now ready to RUN, LIST, CSAVE, etc. the program.

Here are some hints or comments to help you with merging programs:

- The first step in each of the merged programs should be a CLEAR xxx statement to clear memory and reset any DIM statements you may have used in the earlier sections. (If you are merging a series of subroutines into one larger program, disregard this hint.)
- If you are using DATA statements in more than one section of the merged programs, you will have to check the order in which they are read to avoid conflicts.
- I use program steps 1-99 in the final program as a menu to get into the various merged programs.
- End each of the individual programs with a GOTO 1 statement so you can get back to the menu after running each section.
- If more than two programs are being merged, repeat steps 5-7 until the last program has been merged, then go on to step 8.

I have been using this merging technique for a while and it works very well.

Merging helps in reducing the number of tapes I use, and helps to eliminate hunting for various programs within a tape. One drawback to merging several programs and tying them together with a menu is that programs will, of course, be longer and it takes more time to load them into the computer. However, if you organize the merging process properly, you can merge into similar programs together, or programs which are used together can be merged.



# Printer Calendar

Harry Stern  
189 NE Sierra Drive  
Miami, FL 33179

The programs for the Color Computer in the December Microcomputer News were great. But I do not want any of my fellow owners of color computers and Line Printer VIIs to be jealous because they do not have a calendar program like the one for Model III.

## NOW THEY HAVE!

```
5 CLS
: PRINT " PROGRAM TO PRINT CALENDAR FOR ANY
MONTH FROM MARCH, 1700 TO FEBRUARY 2100"
: PRINT
10 INPUT "MONTH<EXAMPLE JAN=1>, YEAR"; M, Y
20 IF M<1 OR M>12 THEN 10
30 GOSUB 1000
40 IF M=1 THEN M$=" JANUARY"
: L=31
50 IF M=2 THEN M$=" FEBRUARY"
: L=28
: IF Y/4=INT(Y/4) THEN L=29
: IF (YR/100=INT(YR/100)) AND (YR/400<
INT(YR/400)) THEN L=28
60 IF M=3 THEN M$=" MARCH"
: L=31
70 IF M=4 THEN M$=" APRIL"
: L=30
80 IF M=5 THEN M$=" MAY"
: L=31
90 IF M=6 THEN M$=" JUNE"
: L=30
100 IF M=7 THEN M$=" JULY"
: L=31
110 IF M=8 THEN M$=" AUGUST"
: L=31
120 IF M=9 THEN M$="SEPTEMBER"
: L=30
130 IF M=10 THEN M$=" OCTOBER"
: L=31
140 IF M=11 THEN M$=" NOVEMBER"
: L=30
150 IF M=12 THEN M$=" DECEMBER"
: L=31
160 UL$=STRING$(40, "-")
170 PRINT #-2, CHR$(31); CHR$(16); "05"; M$; Y;
CHR$(30)
180 PRINT #-2, UL$
190 PRINT#-2, CHR$(16); "01"; "SU"; CHR$(16); "07";
"MO"; CHR$(16); "13"; "TU"; CHR$(16); "19";
"WE"; CHR$(16); "25"; "TH"; CHR$(16); "31";
"FR"; CHR$(16); "37"; "SA"
200 PRINT#-2, CHR$(13)
210 FOR D=1 TO L
220 BL=N+D-1
230 BL=BL-INT(BL/7)*7
240 BL=BL*6
250 IF D<10 THEN BL=BL+1
260 D$=RIGHT$(STR$(BL), 2)
270 IF BL<12 THEN D$="0"+RIGHT$(STR$(BL), 1)
280 PRINT#-2, CHR$(16); D$; D;
290 IF BL>35 THEN PRINT#-2, CHR$(13); CHR$(13)
300 NEXT D
305 IF BL<36 THEN PRINT#-2, CHR$(26)
310 END
1000 FM=M+1
: GY=Y
1010 IF M<3 THEN FM=M+13
: GY=Y-1
1020 N=INT(365.25*GY)+INT(30.6*FM)-621048
1030 IF ((Y=1700 AND M>2) OR Y>1700) AND ((Y=1800 AND
M<3) OR Y<1800) THEN N=N+2
1040 IF ((Y=1800 AND M>2) OR Y>1800) AND ((Y=1900 AND
M<3) OR Y<1900) THEN N=N+1
```

```
1050 N=N-INT(N/7)*7
1060 RETURN
```

# Perpetual Calendar

Kenneth J. Heberle  
116 Goodrich  
Erie, PA 16508

The enclosed is a perpetual calendar written for the Color Computer. I believe you had already printed a perpetual calendar in your newsletter before I started getting it, but this one has several features which make it unique.

First of all, it is good until about the year 3000 with only minor changes as mentioned below. Second, and perhaps more important, most "movable" holidays are calculated, and are also good for over 3000 years. "Fixed" holidays, birthdays, etc. are easily added to the DATA statements at the end of the program.

The only maintenance required is the yearly updating of the current year in line #40 and the once-a-century changing of the current century in line #260. The "Yearshift" in line #320 is optional and can be used to start the current 12-month year at a month other than January. This can be handy in the last few months of a year. No other adjustments should ever be needed until around the year 3000, when our present calendar will need to be adjusted slightly.

The whole thing takes about 6K. I hope you can use this in your newsletter.

```
10 ' *** PERPETUAL CALENDAR ***
20 ' (YEARSHIFT @ 320)
30 CLEAR 400
: CLS(7)
: SOUND 50, 3
40 CY$="1981" ' CURRENT YEAR
45 DIM HOL$(31)
50 ED=99
: LM=31
55 MH$="0414040303230411033104180408032804160405032
50413040203204100330041704070327"
60 DY$="000031059090120151181212243273304334"
70 MN$="JANUARY FEBRUARY MARCH APRIL MAY
JUNE JULY AUGUST SEPTEMBER OCTOBER
NOVEMBER DECEMBER "
80 AD$="SUNDAY MONDAY TUESDAY WEDNESDAY
THURSDAY FRIDAY SATURDAY "
90 DZ$=" SUN MON TUE WED THU FRI SAT "
100 CLS ' BEGIN
105 SOUND 200, 2
110 PRINT
: PRINT
: PRINT
120 PRINT TAB(6) "perpetual calendar"
130 PRINT
: PRINT
140 PRINT " ENTER A DATE (YEARS 1753- )"
145 PRINT
150 PRINT " (FORMATS: MM/DD, MM/DD/YYYY, "
160 PRINT " MM/DD/YY, MM/YYYY, OR JUST MM)"
165 PRINT
: PRINT
170 SOUND 200, 2
: SOUND 200, 2
: INPUT "ENTER DATE OR END "; DT$
180 LL=LEN(DT$)
190 IF DT$="END" THEN 9999
200 IF LL=0 THEN 100
201 YR$=""
210 P1=INSTR(1, DT$, "/")
: P2=INSTR(P1+1, DT$, "/")
220 IF P1=0 THEN MO=VAL(DT$)
: DA=1
```

```

: YR$=CY$
: GOTO 300
230 MO=VAL(LEFT$(DT$, P1-1))
240 IF P2=0 THEN 270
250 DA=VAL(MID$(DT$, P1+1, (P2-1)-P1))
260 IF LL-P2>3 THEN YR$=RIGHT$(DT$, 4) ELSE
YR$=STR$(VAL("19"+RIGHT$(DT$, 2)))
265 GOTO 300
270 IF LL-P1>3 THEN YR$=RIGHT$(DT$, 4)
: DA=1 ELSE YR$=CY$
: DA=VAL(RIGHT$(DT$, LL-P1))
300 YR=VAL(YR$)
315 ' *** YEARSHIFT ***
320 IF LL<5 AND YR$=CY$ AND MO<9 THEN YR=YR+1
: YR$=STR$(YR)
325 ' *****
400 '
410 IF YR<1753 AND SW=0 THEN GOSUB 1500
420 IF MO>12 OR MO<1 THEN DA=1
: GOTO 1400
430 IF DA<1 THEN 1400
450 LP=0
460 IF (YR/100-INT(YR/100))=0 THEN I=INT(YR/400)*400
ELSE I=INT(YR/4)*4
470 IF I=YR THEN LP=1
480 LD=365+LP
500 IF MO=2 THEN 600
510 IF MO=4 OR MO=6 OR MO=9 OR MO=11 THEN 700
520 IF DA>31 THEN 1400
530 LM=31
: GOTO 750
599 '
600 LM=28+LP
610 IF DA>LM THEN 1400
620 GOTO 750
700 LM=30
710 IF DA>LM THEN 1400
740 ' PRINT ROUTINES
750 IF MO<3 THEN LP=0
760 N=MO*3
770 JUL=VAL(MID$(DY$, N-2, 3))
780 JUL=JUL+DA+LP
790 N=MO*9
800 PM$=MID$(MN$, N-8, 9)
810 DT=YR+ INT((YR-1)/4)- INT((YR-1701)/100)+
INT((YR-1601)/400)+ JUL
820 IF P1=0 OR (P2=0 AND LL>5) THEN 1000
840 '
850 ' DAY PRINT ROUTINE
855 SOUND 100, 2
: SOUND 100, 2
860 DW=(DT/7)
: WKDY=INT((DW-INT(DW))*7+.5)
870 O1=WKDY*10+1
880 WKDY$=MID$(AD$, O1, 10)
900 '
950 CLS
: PRINT
: PRINT
: PRINT
960 PRINT "THE DATE "; DT$; " = "; WKDY$
961 IF LL>5 THEN 963
963 PRINT
: JD$=YR$+"."+STR$(JUL)
965 PRINT " THE JULIAN DATE = "; JD$
969 SOUND 250, 4
970 PRINT
: PRINT
: PRINT "WOULD YOU LIKE TO SEE"
980 INPUT " THE WHOLE MONTH? (Y OR N) "; R$
985 IF R$="Y" THEN 1000
990 IF R$="N" THEN 100 ELSE DT$=R$
: GOTO 180
999 '
1000 ' PRINT WHOLE MONTH
1010 DD=0
: HS=0
1020 SOUND 100, 2
: SOUND 100, 2
1030 MS=(DT-(DA-1))/7
1040 D1=INT((MS-INT(MS))*7+.5)
1050 CLS

```

```

1060 PRINT TAB(8) PM$; TAB(18) YR$
1070 PRINT DZ$
1080 GOSUB 1300
1100 PL$=""
1105 FOR WK=1 TO 7
1110 IF DD=D1 THEN DP=1
1120 IF DP<10 THEN DP$="" +STR$(DP) ELSE DP$=STR$(DP)
1125 IF DP=0 THEN DP$=""
1126 IF DP>ED AND WK=1 THEN GOSUB 1250
1127 IF HOL$(DP)<>"" THEN DP$="" **
: HS=1
1130 PL$=PL$+" "+DP$
1140 DD=DD+1
1150 IF DP<>0 THEN DP=DP+1
1160 IF DP>LM THEN DP=0
: DP$=""
1170 NEXT WK
1180 PRINT
: PRINT PL$
1190 IF DP=0 THEN 1200
1195 GOTO 1100
1200 PRINT
1205 IF HS=0 THEN 170
1207 SOUND 200, 2
: SOUND 200, 2
1210 INPUT " ENTER DATE OR * "; R$
1220 IF R$="" THEN 2000
1230 DT$=R$
: GOTO 180
1250 IF HOL$(DP)<>"" THEN HOL$(DP)="EASTER
SUNDAY"+CHR$(13)+" & "+HOL$(DP) ELSE
HOL$(DP)="EASTER SUNDAY"
1260 ED=99
1270 RETURN
1300 '
1310 ' SET MONTH'S HOLIDAYS
1320 GOSUB 8000
1330 GOSUB 3000
1335 RESTORE
1340 READ HDT$, HOL$
1345 IF HDT$="END" THEN RETURN
1350 MT=VAL(LEFT$(HDT$, 2))
1355 IF LEN(HDT$)>5 AND YR<>VAL(RIGHT$(HDT$, 4)) THEN
1340
1360 IF MT<>MO THEN 1340
1365 HDT$=LEFT$(HDT$, 5)
1370 DX=VAL(RIGHT$(HDT$, 2))
1380 IF HOL$(DX)="" THEN HOL$(DX)=HOL$ ELSE
HOL$(DX)=HOL$(DX)+CHR$(13)+" & "+HOL$
1390 GOTO 1340
1399 '
1400 CLS
: PRINT@ 200, "invalid date"
1405 IF DA>LM THEN PRINT
: PRINT " FORMATS ARE: MM/YYYY OR MM/DD,"
: PRINT TAB(7) " - NOT MM/YY!"
1410 SCREEN 0, 1
: SOUND 1, 10
1415 S9=1
1420 FOR D= 1 TO 600
: NEXT D
1425 IF DA>LM AND S9<4 THEN S9=S9+1
: GOTO 1420
1430 GOTO 100
1499 '
1500 ' TOO OLD
1510 CLS
: PRINT@38, "*** caution ***"
1515 SOUND 1, 2
: SOUND 1, 2
1516 SOUND 1, 2
: SOUND 1, 2
1520 PRINT
: PRINT
: PRINT" OUR PRESENT CALENDAR WAS"
1530 PRINT" ADOPTED IN 1752 - ACCURACY"
1540 PRINT" FOR DATES EARLIER THEN 1753"
1550 PRINT" MAY REQUIRE CONVERSION"
1565 PRINT
: PRINT "(MOST DATES IN AMERICAN HISTORY HAVE
ALREADY BEEN CONVERTED)"

```

```

1570 PRINT
: PRINT
: INPUT "HIT <ENTER> TO CONTINUE "; R$
1575 SW=1
1580 RETURN
2000 ' PRINT HOLIDAYS
2010 CLS
: SOUND 250, 4
2025 IF HS=0 THEN 2200
2030 PRINT" IMPORTANT DATE(S) IN "; PM$
2035 PRINT STRING$(32, "*")
2040 FOR H=1 TO LM
2045 IF HOL$(H)="" THEN 2060
2050 PRINT H; TAB(5) HOL$(H)
2060 NEXT H
2065 PRINT@400, "";
2070 INPUT "ENTER DATE OR NULL "; R$
2080 IF R$="" THEN 1010
2090 DT$=R$
: GOTO 180
2200 PRINT " NO HOLIDAYS, BIRTHDAYS, OR "
2210 PRINT " ANYTHING AT ALL IN "; PM$
2250 GOTO 2065
2999 '
3000 ' MOVABLE HOLIDAYS
3010 ON MO GOSUB 3050, 3100, 3900, 3900, 3200, 3300,
3050, 3050, 3400, 3600, 3500, 3050
3050 RETURN
3099 '
3100 IF D1<2 THEN HX=16-D1 ELSE HX=23-D1
3120 HOL$(HX)="WASHINGTON'S HOLIDAY"
3130 RETURN
3200 HX=15-D1
3205 IF D1=0 THEN HX=8
3210 HOL$(HX)="MOTHER'S DAY"
3215 HX=30-D1
3220 IF D1=6 THEN HX=31
3230 HOL$(HX)="MEMORIAL DAY **"
3240 HX=21-D1
3250 HOL$(HX)="ARMED FORCES DAY"
3260 RETURN
3300 HX=22-D1
3305 IF D1=0 THEN HX=15
3310 HOL$(HX)="FATHER'S DAY"
3320 RETURN
3400 IF D1<2 THEN HX=2-D1 ELSE HX=9-D1
3410 HOL$(HX)="LABOR DAY **"
3420 RETURN
3500 IF D1<2 THEN HX=3-D1 ELSE HX=10-D1
3510 HOL$(HX)="ELECTION DAY"
3520 IF D1>4 THEN HX=33-D1 ELSE HX=26-D1
3530 HOL$(HX)="THANKSGIVING **"
3540 RETURN
3600 IF D1<2 THEN HX=9-D1 ELSE HX=16-D1
3610 HOL$(HX)="COLUMBUS DAY"
3620 RETURN
3900 ' EASTER SUNDAY
3910 FM=YR/19
3920 PFM=INT((FM-INT(FM))*19+.5)
3930 PX=PFM*4+1
3940 PFM$=MID$(MH$, PX, 4)
3950 EM=VAL(LEFT$(PFM$, 2))
: EH=VAL(RIGHT$(PFM$, 2))
3960 IF EM=MO THEN ED=EH ELSE ED=99
3970 D2=D1
: IF D2=0 THEN D2=7
3980 IF EM=3 AND MO=4 AND EH>31-D2 THEN ED=0
3990 RETURN
7999 '
8000 ' CLEAR HOLIDAYS
8010 FOR IX=1 TO 31
8020 HOL$(IX)=""
8030 NEXT IX
8040 RETURN
8999 '
9000 ' HOLIDAYS, ETC.
9010 DATA 04/15, INCOME TAXES DUE!!!
9020 DATA 01/01, NEW YEARS DAY **
9040 DATA 07/04, INDEPENDENCE DAY **
9050 DATA 08/13, OUR ANNIVERSARY
9080 DATA 12/25, CHRISTMAS DAY **
9090 DATA 06/07, MARCIA'S BIRTHDAY

```

```

9100 DATA 10/02, KEN'S BIRTHDAY
9110 DATA 02/12, LINCOLN'S BIRTHDAY
9120 DATA 02/14, VALENTINE'S DAY
9150 DATA 03/17, ST. PATRICK'S DAY
9160 DATA 01/24, JEFF'S BIRTHDAY
9170 DATA 07/23, DAVE'S BIRTHDAY
9190 DATA 04/24, TIM'S BIRTHDAY
9200 DATA 06/14, FLAG DAY
9220 DATA 10/31, HALLOWEEN
9230 DATA 11/11, VETERAN'S DAY
9300 DATA 03/20, 1ST DAY OF SPRING
9310 DATA 06/21, 1ST DAY OF SUMMER
9320 DATA 09/22, 1ST DAY OF AUTUMN
9340 DATA 12/21, 1ST DAY OF WINTER
9410 DATA 12/07, PEARL HARBOR DAY
9440 DATA 04/01/1980, 1ST DAY OF PASSOVER
9450 DATA 04/19/1981, 1ST DAY OF PASSOVER
9460 DATA 04/08/1982, 1ST DAY OF PASSOVER
9470 DATA 03/29/1983, 1ST DAY OF PASSOVER
9480 DATA 04/17/1984, 1ST DAY OF PASSOVER
9490 DATA 04/06/1985, 1ST DAY OF PASSOVER
9500 DATA 12/03/1980, 1ST DAY OF HANUKKAH
9510 DATA 12/21/1981, 1ST DAY OF HANUKKAH
9520 DATA 12/11/1982, 1ST DAY OF HANUKKAH
9530 DATA 12/01/1983, 1ST DAY OF HANUKKAH
9540 DATA 12/19/1984, 1ST DAY OF HANUKKAH
9550 DATA 12/08/1985, 1ST DAY OF HANUKKAH
9900 DATA END, ***
9999 STOP

```

## Linefeed with a Carriage Return

Ed Hamilton  
P.O. Box 943  
Santa Rosa, CA 95402

I am currently using a printer (TELETYPE MODEL 43) that does not perform an automatic linefeed when a carriage return is sent. Since the Color Computer expects this to be the case, my printouts all end up on one line - UGH! This 6809E code inserts a patch into the operating system to intercept the CR character and issue both the linefeed and carriage return.

The code works with either regular or Extended Basic. It resides in the highest 19 bytes of memory and is not affected by resets, although (obviously) the program must be entered after every power-on.

### THE BASIC PROGRAM

```

10 'O/S PATCH FOR LF ON CR
100 'PUT CODE AT MEMTOP- 19
110 MT=PEEK(116)*256+PEEK(117)
112 MT=MT-19+12
120 FOR I=0 TO 18
122 READ T
124 POKE MT+I, T
126 NEXT I
130 'SAVE O/S CODE
132 POKE MT+16, PEEK(359)
134 POKE MT+17, PEEK(360)
136 POKE MT+18, PEEK(361)
140 'PUT JMP IN O/S HANDLER
142 AH=INT(MT/256)
144 AL=MT-AH*256
146 POKE 359, 126
148 POKE 360, AH
: POKE 361, AL
150 'MACHINE-LANGUAGE CODE
151 DATA 129, 13, 38, 12, 150, 111
152 DATA 76, 42, 5, 134, 10, 189
153 DATA 162, 133, 134, 13, 0, 0, 0
200 'RESERVE MEMORY
210 CLEAR 200, MT
999 END

```

# Hi-Res Character Generator

Ron Van Dyke  
52 East McKinley  
Zeeland, MI 49464

This program will allow you to display alphanumeric characters on a graphic screen. Instructions are included in the program.

```
10 'CHAR-GEN BY ron van dyke
20 'DATE: 09/13/81
30 '
40 PMODE 1, 1
   : PCLS 1
   : SCREEN 1, 0
50 DRAW "S8; C3; BM70, 12"
   : AA$="CHAR-GEN"
   : GOSUB 130000
60 DRAW "S16; C2; BM60, 44"
   : AA$="C"
   : GOSUB 130000
70 DRAW "C3"
   : AA$="O"
   : GOSUB 130000
80 DRAW "C4"
   : AA$="L"
   : GOSUB 130000
90 DRAW "C2"
   : AA$="O"
   : GOSUB 130000
100 DRAW "C3"
   : AA$="R"
   : GOSUB 130000
110 DRAW "S8; C4; BM70, 64"
   : AA$="COMPUTER"
   : GOSUB 130000
120 DRAW "C2; BM55, 100"
   : AA$="++++++"
   : GOSUB 130000
130 DRAW "C4; BM65, 120"
   : AA$="CHARACTER"
   : GOSUB 130000
140 DRAW "BM65, 140"
   : AA$="GENERATOR"
   : GOSUB 130000
150 DRAW "BM60, 160"
   : AA$="SUBROUTINE"
   : GOSUB 130000
160 FOR T=0 TO 2000
   : NEXT T
170 CLS
   : PRINT TAB(11) "char-gen"
   : Q$=CHR$(34)
   : PRINT
180 PRINT "TO USE CHAR-GEN MERGE IT WITH"
190 PRINT "THE PROGRAM THAT YOU DESIRE AND"
200 PRINT "CALL IT AS A SUBROUTINE USING"
210 PRINT "THE FOLLOWING PROCEDURE:"
220 PRINT
230 PRINT "FIRST YOU MUST DECIDE WHICH"
240 PRINT "GRAPHIC MODE YOU WILL BE USING."
250 PRINT "IF THE GRAPHIC MODE IS ANYTHING"
260 PRINT "OTHER THAN PMODE4 YOU MUST"
270 PRINT "SCALE YOUR CHARACTERS TO"
280 PRINT "AT LEAST S8."
290 PRINT
   : INPUT "PRESS ENTER TO CONTINUE"; X
   : CLS
300 PRINT "NEXT USE THE DRAW COMMAND TO"
310 PRINT "MOVE THE STARTING POSITION TO"
320 PRINT "DESIRED BEGINNING POINT."
330 PRINT
   : PRINT CHR$(123) "note" CHR$(125) " THE
   : ROUTINE STARTS EACH"
```

```
340 PRINT "CHARACTER AT IT'S LOWER LEFT"
350 PRINT "CORNER. BE SURE TO GO LOW ENOUGH"
360 PRINT "NOW SET THE STRING VARIABLE"
370 PRINT "AA$ EQUAL TO THE STRING YOU"
380 PRINT "WISH TO BE PRINTED AND CALL"
390 PRINT "THE SUBROUTINE AT 130000"
400 PRINT
   : INPUT "PRESS ENTER TO CONTINUE"; X
   : CLS
410 PRINT "THE FOLLOWING IS A SAMPLE"
420 PRINT "PROGRAM SHOWING HOW TO USE"
430 PRINT "CHAR-GEN."
   : PRINT
440 PRINT "10 PMODE 1, 0:PCLS1:SCREEN 1, 0
450 PRINT "20 DRAW" Q$ "C3; S8; BM10, 95" Q$
460 PRINT "30 AA$=" Q$ "THIS IS A TEST" Q$
470 PRINT "40 GOSUB 130000
480 PRINT "50 DRAW" Q$ "C2; BM10, 180" Q$
490 PRINT "60 AA$=" Q$ "PRESS ENTER" Q$
500 PRINT "70 GOSUB 130000"
510 PRINT "80 IF INKEY$="" Q$ ; Q$ " GOTO 80
520 PRINT
   : INPUT "PRESS ENTER TO SEE SAMPLE"; X
530 PMODE 1, 1
   : PCLS
   : SCREEN 1, 0
540 DRAW"C3; S8; BM10, 95"
550 AA$="THIS IS A TEST"
   : GOSUB 130000
560 DRAW"C2; BM10, 180"
   : AA$="PRESS ENTER"
570 GOSUB 130000
580 IF INKEY$="" GOTO 580
590 CLS
   : PRINT
   : PRINT "DID YOU NOTICE THE C2 IN LINE"
600 PRINT "20 AND THE C3 IN LINE 50"
610 PRINT "THE 'C' SUBCOMMAND CHANGES THE"
620 PRINT "COLOR SET OF OUR CHARACTERS"
630 PRINT
   : PRINT "LET'S TRY ANOTHER SAMPLE USING"
640 PRINT "THE HIGHEST RESOLUTION MODE"
   : PRINT
   : PRINT
   : INPUT "PRESS ENTER TO CONTINUE"; X
650 CLS
   : PRINT
   : PRINT
   : PRINT "10 PMODE 4, 1:PCLS1:SCREEN 1, 0
660 PRINT "20 DRAW" Q$ "C0; S16; BM20, 50" Q$
670 PRINT "30 AA$=" Q$ "LARGE" Q$
680 PRINT "40 GOSUB 130000
690 PRINT "50 DRAW" Q$ "S4; BM20, 65" Q$
700 PRINT "60 AA$=" Q$ "SMALL" Q$
710 PRINT "70 GOSUB 130000
720 PRINT "80 DRAW" Q$ "BM20, 180" Q$
730 PRINT "90 AA$=" Q$ "PRESS ENTER" Q$
735 PRINT "100 GOSUB 130000
740 PRINT "110 IF INKEY$="" Q$; Q$ " GOTO 110
750 PRINT
   : INPUT "PRESS ENTER TO SEE SAMPLE"; X
760 PMODE 4, 1
   : PCLS 1
   : SCREEN 1, 0
   : DRAW"C0; S16; BM20, 50"
   : AA$="LARGE"
   : GOSUB 130000
770 DRAW"S4; BM20, 65"
   : AA$="SMALL"
   : GOSUB 130000
780 DRAW"BM20, 180"
   : AA$="PRESS ENTER"
790 GOSUB 130000
800 IF INKEY$="" THEN 800
950 CLS
   : PRINT "ONE FINAL NOTE:
960 PRINT "THIS ROUTINE USES INFORMATION"
970 PRINT "STORED IN DATA STATEMENTS AND"
980 PRINT "A RESTORE COMMAND, THIS CAN"
990 PRINT "CAUSE MAJOR PROBLEMS IF USED"
```

LARGE

SMALL

PRESS ENTER

```
1000 PRINT "WITH PROGRAMS CONTAINING DATA"
1010 PRINT "STATEMENTS. YOU MAY HAVE TO"
1020 PRINT "RENUMBER THE PROGRAMS SO THAT"
1030 PRINT "THE SUBROUTINE IS AHEAD OF"
1040 PRINT "YOUR PROGRAM AND THEN ADD"
1050 PRINT "DUMMY READ STATEMENTS TO READ"
1060 PRINT "ALL DATA THAT WOULD INTERFERE"
1070 PRINT "WITH YOUR PROGRAM. BEYOND THIS"
1080 PRINT "YOUR ON YOUR OWN."
1090 INPUT "PRESS ENTER"; X
1100 CLS
      : PRINT
      : PRINT
      : PRINT
      : PRINT "PRESSING ENTER AT THIS POINT"
1110 PRINT "WILL DELETE LINES 10 THRU 1140
LEAVING ONLY THE CHAR-GEN SUBROUTINE."
1120 PRINT "BE SURE TO SAVE THE SUBROUTINE AFTER
PRESSING ENTER."
1130 PRINT
      : PRINT
      : INPUT "PRESS ENTER"; X
1140 CLS
      : DEL 10-1140
13000 ' char-gen
13001 FOR XX=1 TO LEN(AA$)
13002 RESTORE
      : LL=0
13003 READ LL$, CC$
13004 IF LL$=MID$(AA$, XX, 1) THEN DRAW CC$
      : GOTO 13006
13005 LL=LL+1
      : IF LL<48 THEN 13003
13006 NEXT
      : RETURN
13007 DATA " ", "BM+7, 0"
13008 DATA "A", "U4; E2; F2; D2; NL4; D2; BM+3,
0"
13009 DATA "B", "U6; R3; F1; D1; G1; NL3; F1; D1;
G1; L3; BM+7, 0"
13010 DATA "C", "BM+1, -0; H1; U4; E1; R2; F1;
BM+0, +4; G1; L2; BM+6, 0"
13011 DATA "D", "U6; R3; F1; D4; G1; L3; BM+7, 0"
13012 DATA "E", "NR4; U3; NR2; U3; R4; BM+3, +6"
13013 DATA "F", "U3; NR2; U3; R4; BM+3, +6"
13014 DATA "G", "BM+1, -0; H1; U4; E1; R2; F1;
BM+0, +2; NL1; D2; G1; L2; BM+6, 0"
13015 DATA "H", "U3; NU3; R4; NU3; D3; BM+3, 0"
13016 DATA "I", "BM+1, 0; R1; NR1; U6; NL1; R1;
BM+4, +6"
13017 DATA "J", "BM+0, -1; F1; R1; E1; U5; NL1;
R1; BM+3, 6"
13018 DATA "K", "U3; NU3; R1; NE3; F3; BM+3, 0"
13019 DATA "L", "NU6; R4; U1; BM+3, +1"
13020 DATA "M", "U6; F2; ND1; E2; D6; BM+3, 0"
13021 DATA "N", "U6; F1; D1; F2; D1; F1; NU6;
BM+3, 0"
```

```
13022 DATA "O", "BM+1, 0; H1; U4; E1; R2; F1; D4;
G1; L2; BM+6, 0"
13023 DATA "P", "U6; R3; F1; D1; G1; L3; BM+7, 3"
13024 DATA "Q", "BM+1, 0; H1; U4; E1; R2; F1; D3;
G1; NH1; NF1; G1; L1; BM+6, 0"
13025 DATA "R", "U6; R3; F1; D1; G1; L2; NL1; F3;
BM+3, 0"
13026 DATA "S", "BM+0, -1; F1; R2; E1; U1; H1;
L2; H1; U1; E1; R2; F1; BM+3, +5"
13027 DATA "T", "BM+2, +0; U6; NL2; R2; BM+3, +6"
13028 DATA "U", "BM+0, -1; NU5; F1; R2; E1; U5;
BM+3, 6"
13029 DATA "V", "BM+0, -6; D2; F1; D1; F1; ND1;
E1; U1; E1; U2; BM+3, +6"
13030 DATA "W", "NU6; E2; NU1; F2; U6; BM+3, 6"
13031 DATA "X", "U1; E4; U1; BM-4, 0; D1; F4; D1;
BM+3, 0"
13032 DATA "Y", "BM+0, -6; D2; F2; ND2; E2; U2;
BM+3, 6"
13033 DATA "Z", "NR4; U1; E4; U1; L4; BM+7, 6"
13034 DATA "1", "BM+1, 0; R1; NR1; U6; G1; BM+6,
+5"
13035 DATA "2", "NR4; U1; E1; R1; E2; U1; H1; L2;
G1; BM+7, +5"
13036 DATA "3", "BM+0, -1; F1; R2; E1; H2; E2;
H1; L3; BM+7, 6"
13037 DATA "4", "BM+3, 0; U2; NR1; L3; U1; E3;
D3; BM+4, 3"
13038 DATA "5", "BM+0, -1; F1; R2; E1; U2; H1;
L3; U2; R4; BM+3, +6"
13039 DATA "6", "BM+4, -5; H1; L2; G1; D4; F1;
R2; E1; U1; H1; L3; BM+7, +3"
13040 DATA "7", "U1; E4; U1; L4; BM+7, +6"
13041 DATA "8", "BM+1, -0; H1; U1; E1; H1; U1;
E1; R2; F1; D1; G1; NL2; F1; D1; G1; L2;
BM+6, 0"
13042 DATA "9", "BM+0, -1; F1; R2; E1; U4; H1;
L2; G1; D1; F1; R2; BM+4, +3"
13043 DATA "/", "U1; E4; U1; BM+3, 6"
13044 DATA "?", "BM+0, -5; E1; R2; F1; D1; G2;
BM+0, +1; D1; BM+5, +0"
13045 DATA "!", "BM+2, +1; U1; BM+0, -2; U5;
BM+5, 7"
13046 DATA ".", "BM+2, 0; U1; BM+5, +1"
13047 DATA ":", "BM+2, -1; U1; BM+0, -2; U1;
BM+5, +5"
13048 DATA ";", "BM+1, 0; E1; U1; BM+0, -1; U1;
BM+5, +4"
13049 DATA ",", "BM+2, 0; NU1; G1; BM+6, -1"
13050 DATA "=", "BM+1, -5; E2; BM+4, +7"
13051 DATA "-", "BM+0, -3; R4; BM+3, +3"
13052 DATA "+", "BM+2, -1; U2; NU2; NL2; R2;
BM+3, +3"
13053 DATA "=", "BM+1, -2; R3; BM-3, -2; R3;
BM+4, +4"
```

CHAR-GEN  
COLOR  
COMPUTER

+++++++  
CHARACTER  
GENERATOR  
SUBROUTINE

