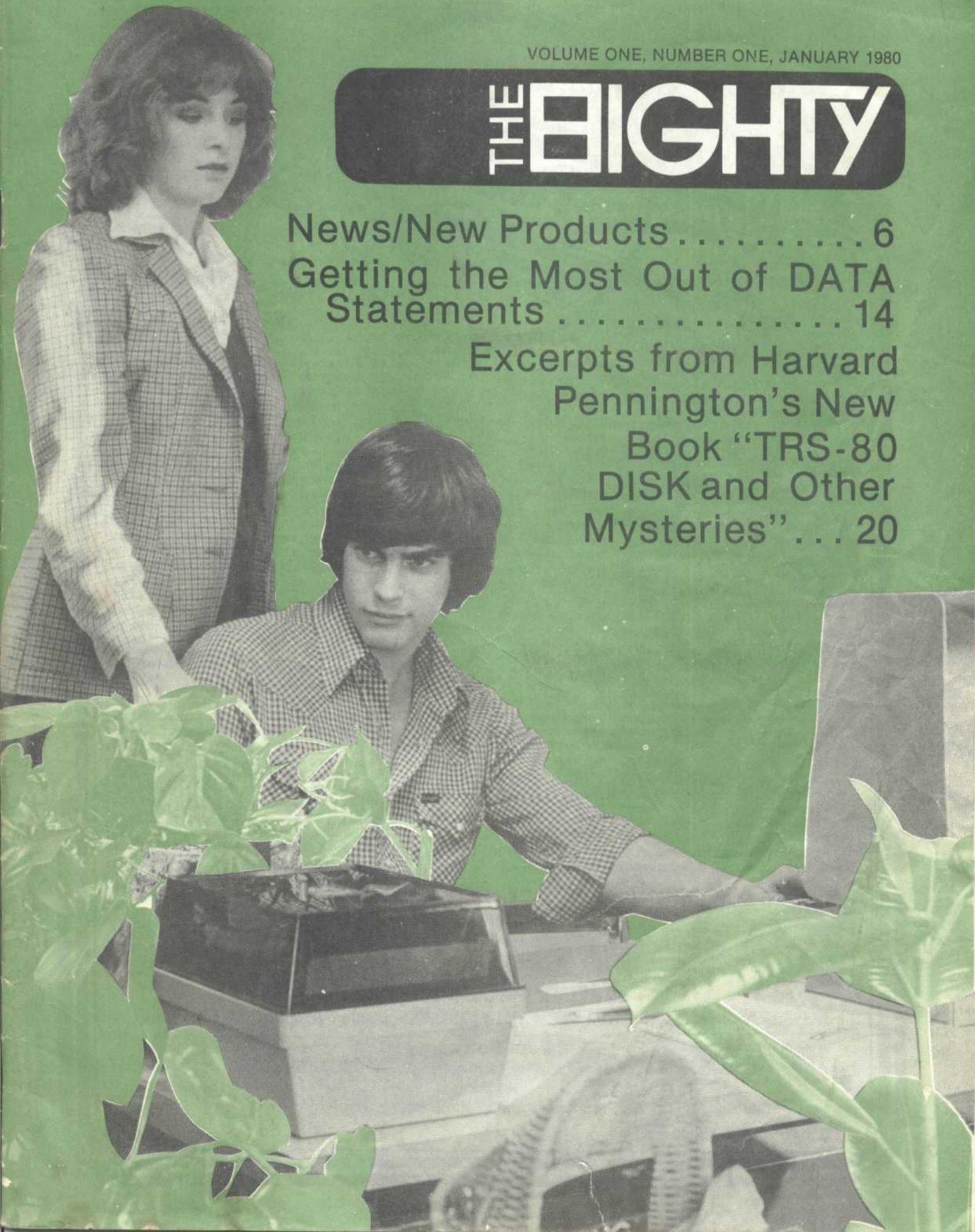


VOLUME ONE, NUMBER ONE, JANUARY 1980

# THE EIGHTY

|  |    |
|--|----|
| News/New Products .....  | 6  |
| Getting the Most Out of DATA<br>Statements .....   | 14 |
| Excerpts from Harvard<br>Pennington's New<br>Book "TRS-80<br>DISK and Other<br>Mysteries" .... | 20 |



## Welcome to the Pages of The Eighty

What, another new TRS-80 publication? Aren't there dozens already? We know of at least two dozen, but there is currently no reasonably priced advertising medium with a significant percentage of TRS-80 owners on the subscription list. Even our own SoftSide, one of the leading TRS-80 publications, reaches a mere 5% of TRS-80 owners; and SoftSide does not accept outside advertising.

Radio Shack is in no way eager to advise its customers that there are many independent suppliers of excellent peripherals and software for the TRS-80 microcomputer. Yet, in fact, many of the suppliers offer products far superior to those of Radio Shack, and usually at better prices. For example, other drives outperform Radio Shack drives, yet cost significantly less. Much of Radio Shack's software is months behind the market, and often of dubious value.

We feel that the solution is to offer a bargain to both TRS-80 owners and advertisers. The Eighty is free to any TRS-80 owner who requests it. This first issue is going to over 17,000 TRS-80 owners and users: people who subscribe to SoftSide or its sister publication, PROG/80; customers of The Software Exchange, the largest independent source of software for Radio Shack computers; and customers of HardSide, one of the largest retailers of new and used Radio Shack computer equipment.

We have modest expectations. We have been disturbed and annoyed by the outrageous claims of some in the microcomputer industry. We have even seen a publisher go so far as to claim 50,000 readers for a TRS-80 magazine that didn't even exist at the time of the claim, with an attempt to charge advertisers accordingly! We simply invite advertisers to try us at minimal cost, and compare the results.

We do not intend to inflate our circulation artificially by dropping bundles of The Eighty in the mail to computer clubs and college campuses where only one copy in twenty will ever be picked up. One of our staff was an officer in a midwest computer club where dozens of newsletters and free magazines sat untouched on a side table during the meeting, only to be thrown away at the end of the meeting. We have seen the same process with the same publications at college computer centers.

If a club does desire copies of The Eighty, we shall be happy to provide a modest number of copies for one-time distribution, but ask that those who wish to continue receiving The Eighty ask to be placed on the mailing list. There is no charge for this, but we ask that only people who own, operate, or plan to purchase a TRS-80 microcomputer subscribe.

We are happy to receive articles, reviews, and manufacturer's press releases for publication. Only products directly related to the TRS-80 will be covered. We do pay for articles and reviews not originating from a manufacturer or supplier.

## THE EIGHTY

### Publisher

Roger Robitaille

### Editor

Bob Liddil

### Production

Sharon Demmerle

### Advertising Manager

Bill York

### Address

The Eighty  
SoftSide Publications  
P.O. Box 68  
Milford, NH 03055

### Advertising Rates

|         | 1x    | 2x  | 5x  | 12x |
|---------|-------|-----|-----|-----|
| Full    | \$400 | 375 | 350 | 325 |
| Half    | \$240 | 225 | 210 | 195 |
| Quarter | \$144 | 135 | 126 | 117 |

### Submissions

Articles, Reviews, and Press Releases are welcome and will be published as space permits. No compensation is offered for commercial submissions. Individual authors are compensated at \$20 a page.

\*Note: Radio Shack and TRS-80 are registered trademarks of the Radio Shack, Division of Tandy Corporation.

As I look back to the time when I first started programming in BASIC (after buying a Radio Shack computer and working my way through the Level I User's Manual), I really don't think I can do a lot more now than I could then. In programming, experience usually teaches you better ways of doing the same things.

For example, if you want to put a value in a variable, there are several ways to do it. If you want to put the word ALBATROSS in the string variable A\$, here are some ways to accomplish it:

- 10 A\$ = "ALBATROSS"
- 10 INPUT A\$
- 10 FOR A = 1 TO 9  
20 I\$ = INKEY\$: IF I\$ = " " THEN 20  
30 A\$ = A\$ + I\$: NEXT A enter from keyboard
- 10 INPUT #1, A\$ enter from cassette or disk sequential file, if open
- 10 OPEN "R", 1, "BIRDS"  
20 FIELD #1, 9 AS A\$ enter from random disk file  
30 GET 1, 1
- 10 DATA "ALBATROSS"  
20 READ A\$

All of these methods accomplish the same thing. As a general rule, if you have only a few items to be stored as variables, the first method is often best. If the variable is chosen by the person using the program, the second or third methods are preferred. If there is a lot of data to be processed by the program, the cassette or disk files are necessary. In between, when you have a moderate amount of data that's the same for each run of the program the statements READ and DATA are probably the best choice, especially if you have to switch variables back and forth. When you need to use the same data more than once, a RESTORE statement makes DATA and READ even more powerful.

Before I give any examples, there is a flaw in Radio Shack Level II BASIC that should be mentioned. Sometimes READ statements will repeatedly read the first item of DATA. This is because the RESTORE flag is improperly set. If you have this problem, you will have to enter the statement:

(line number) POKE 16554, 255

before the first READ statement. Some computers even require this before each READ statement. The problem is often limited to certain programs on the same computer, and doesn't seem to occur in Level I or disk BASIC.

The statement DATA tells the computer that information on the same line is to be used in READ statements. In most computers, the data can be either numbers or letters. More than one item can be on the same line of items are separated by commas. Most computers require quotes around string data.

The statement READ tells the computer to fill a variable with the next unused item of data. Most computers will give an error message if you try to string data into a numerical variable, a number into a string variable, or if you have run out of data.

The statement RESTORE tells the computer to forget which data has already been read and start over with the first item in a DATA statement. This is how the RESTORE statement works. The computer keeps count of how many DATA items have been read, so that the next READ statement will read the proper data. RESTORE sets the "data pointer" back to the first item of DATA.

This program will demonstrate the three statements:

```
10 DATA "DATA", "MORE DATA", "STILL MORE DATA"
20 READ A$, B$
30 PRINT A$, B$
40 RESTORE
50 READ A$
60 READ A$, B$
70 PRINT A$, B$
```

Enter the program into your computer and run it. The sample run should look like this:

```
DATA          MORE DATA
MORE DATA    STILL MORE DATA
```

Line 10 is our DATA statement. Actually, it could be anywhere in the program and the program would still run the same. You may want to be convinced, so delete line 10 and enter line 35:

```
35 DATA "DATA", "MORE DATA", "STILL MORE DATA"
```

You may wish to try the same line in other places, particularly at the end of the program.

Since the DATA statements will work anywhere in the program, does that tell us it doesn't matter where we put them? The answer is, "Not really". The practical locations are: near the beginning of a program, at the end of a program, and (in exceptional cases) near the part of the program that makes use of the data. Wherever you put them, all DATA statements should be together. This is because you need to be able to tell which item of data is next to be read, and you can't count easily if you have to look all over the place.

For speed, data statements should be at the beginning of the program, as the interpreter starts to look at the first line. Most of the time, I like DATA statements at the end of a program. This is because my DATA statements are often edited, and I like to have them at the end when I type LIST. I've fallen into the habit of putting DATA statements at line 10000. I find putting them in the same place at different programs helps me get organized and work faster.



Because you often end up counting items of data, I like to put the same number of items in each statement (usually ten). It's easy to count by tens, and in addition, I can make the line numbers significant. For example, in a disassembler, I put the Z-80 Operation Codes in groups of ten, and the last three digits of the line number were the decimal equivalent of the starting Op Code.

When you're using RESTORE, careful thought should be given to the decision of whether or not to mix numerical and string data. In our sample program, line 50 was a "dummy READ". This is a READ statement that you don't use in the program, but rather one that's used to skip over an item of data to the next data item. If you put all your string data first and numerical data afterward, it's easy to find an item of data at random. If you had fifty items of string data and fifty items of numerical data, and you wanted the twenty-first item of numerical data, you could use this subroutine: (the DATA statements are not shown).

```
10 N = 21:GOSUB 100
```

```
100 FOR S = 1 TO 50:READ A$:NEXT S
110 FOR S = 1 TO N:READ V:NEXT S
120 PRINT V
```

Line 100 does a dummy READ on the fifty string variables and line 110 does a dummy READ on the first twenty numerical variables saving the twenty-first in variable V.

Another way to accomplish the same task would be to pair string data and numerical data. Here is a short program to find the price of a sandwich, using paired data:

```
10 DATA "HAM", 1.5, "CHICKEN", 1.25, "CLUB", 2.25
20 INPUT "SANDWICH NUMBER";N
30 FOR A = 1 TO N:READ A$,P:NEXT A
40 PRINT A$;"SANDWICH-$",P
50 RESTORE
60 GOTO 20
```

Enter the program and try a few sample runs. INPUT the number 4 to discover what kind of message your computer gives for an OUT OF DATA error. For Radio Shack computers, you should get:

```
[Level I]          HOW?
[Level II]         OD ERROR IN 30
[Disk]            OUT OF DATA IN 30
```

If you consistently pair your string and numerical data, line 30 could handle more data than you're likely to put into a program. You could also group string and numerical data in any sequence using the same idea. For example, if you matched two string variables and three numerical values, you could change line 30 to:

```
30 FOR A = 1 TO N:READ A$, B$, P,Q,R:NEXT A
```

This could do a dummy READ on five variables at a time, so long as all your data was in this form:

```
10000 DATA "(string)", "(string)", (number), (number), (number)
10010 DATA "(string)", "(string)", (number), (number), (number)
```

When you group several variables like this, it's good practice to put each group in a separate DATA line.

As a final example of the power of a data statement, enter this program:

**[Radio Shack Level II and Disk BASIC only]**

```
10 FOR C = 1 TO 2:READ N
20 FOR A = 1 TO N
30 READ B:S$(C) = S$(C) + CHR$(B)
40 NEXT A
50 READ N$(C)
60 PRINT S$(C),N$(C):PRINT
70 NEXT C
80 DATA 12,130,188,188,188,188,191,189,188,188,188,188,140,
  "AIRCRAFT CARRIER"
90 DATA 7,176,176,188,176,176,144, "SUBMARINE"
```

This program mixes numerical and string data, uses data (N) to control the reading of more data, shows how to use data statements to form string variables of graphic characters, and demonstrates a method of using mixed numerical and string data in random format. To do a dummy READ, all you have to change is the subscripted variables. Try it by substituting S\$ for S\$(C), N\$ for N\$(C), and put the PRINT statement outside the FOR LOOP. Clear S\$ on each loop.

You can do a dummy READ because the first item in each data line tells the computer how many times to read the data.

The method of using string variables full of graphics characters is one of the most powerful techniques available for graphics. It makes fast animation possible, as is well demonstrated in Bee Wary, probably the best graphics program written in 1979 for the TRS-80 (Available from TSE, P.O. Box 68, Milford, N.H. 03055.)

Take the time to figure out for yourself what is happening in the sample program. When you understand it well, start trying to write your own programs using the same method. The ideas are simple, and to make them work for you, all you need is practice. Do it with DATA!

EIGHTY

**APL for the TRS-80\***

■■■ APL by Phelps Gates - APL - The powerful, mathematically elegant language of the IBM 5110, has come to the TRS-80. Deluxe version, for 32K Disk systems, includes self-teaching instructional material and the book APL - an interactive approach for only \$49.95 plus \$1.00 handling. Order from The Software Exchange, 8 South Street, Milford, NH 03055 or call TOLL FREE 1-800-258-1790.

## Book Excerpt

### TRS-80\* DISK AND OTHER MYSTERIES

\$19.95 plus \$1 shipping available from The Software Exchange, 6 South Street, Milford, NH 03055.

This book is the best source of information currently available on the TRS-80\* TM Disk Operating System. It gives instructions for recovering lost data, understanding and reconstructing the disk directory, and using disk related programs and utilities including Superzap, RSM-2D, Monitor 3, Debug, Dircheck, LMOFFSET, and Electric Pencil.

Having received Mr. Pennington's permission to excerpt his book, the problem was to select portions that were short enough, complete in themselves, and which gave a fair selection of the book. Two such excerpts follow. The first one is Mr. Pennington's analysis of different DOS's available. The second is a last ditch method to recover a bad diskette.

Both selections Copyright (C) 1979 by Harvard C. Pennington and used by permission.

#### 4.0 OPERATING SYSTEMS

This will be a brief review of the various operating systems that are available as of this writing. I will not dwell too long on the pros and cons of each and you must remember that the following is an OPINION, mine.

##### 4.1 "TRSDOS 2.1"

Except for the few unfortunate souls that started with 2.0 this is the operating system that most of us developed our first, genuine love-hate relationship with. For all practical purposes, due to the short life of 2.0, this was the 'FIRST' operating

system generally available for the TRS-80.

2.1 has many problems. Of course, Radio Shack never came out and admitted, in plain English, (at least to me -- did they tell you?) that the problems existed. TRSDOS 2.1 is adequate for most trivial programming requirements and a few serious applications IF you are prepared to tolerate an occasional lost file. If you contemplate any real serious applications I would not recommend that TRSDOS 2.1 be used, under any circumstance.

Data recovery on TRSDOS 2.1 generated disks is normal and routine for formatted data disks and system disks.

##### 4.2 "TRSDOS 2.2"

TRSDOS 2.2 is a huge improvement over 2.1. Most of the errors are corrected. However, it will still create errors. Most of the complaints I have about the system are that they still have not given the user any of the utility that is available with NEW DOS.

As far as data recovery goes, there is one major point. When you 'KILL' a file with 2.2, it ZEROS THE ENTIRE DIRECTORY ENTRY. There is not a single clue as to what was there or where it was! Since Radio Shack has no utility for looking at the disk, I presume it was to prevent all you "SUPERZAPPERS" out there from finding out too much! However, if you need to recover something, this makes it not impossible but a genuine bitch because you have to go 'mucking around on the disk' looking for the file.

For this reason alone, I would not use this system on a serious application where I MIGHT have to recover 'KILL'ed data.

Data recovery on TRSDOS 2.2 generated disks is normal and routine on formatted disks and system disks except for the above described 'KILL'ed files.

##### 4.3 "VTOS 3.0"

This is Randy Cook's version of 2.2 with quite a few bells and whistles. Cook is the

author of Radio Shack's 2.1 and, I have reason to suspect, most of 2.2. This system has some nice features but is, in my opinion, VERY AGGRAVATING to use because of its 'BACKUP' protection feature. In the version that I used for evaluation, some of the commands did not work entirely as advertised. I'm sure that this will be corrected in a later release. On the whole, the system is good and the concepts are excellent. I have not used it enough, at this time, to have detected any errors, if it has any.

If you find it necessary to recover data or files that have been 'SAVE'd to a VTOS 3.0 system disk, you will not be pleased with the recovery procedures.

This is due to the fact that as a function of the VTOS 3.0 protection features, you will NOT BE ABLE TO RECOVER THE DATA TO ANOTHER DISK AND THEN 'RUN' THAT DISK!

In spite of all the nice features in this system, it is for this reason that I would not recommend its use with applications of other than, a trivial nature. Data recovery on VTOS 3.0 system disks is VERY DIFFICULT. You must first format a disk and then use the "SUPERZAP" 'BACKUP' function to transfer the information to the 'working disk'. YOU WILL NOT BE ABLE TO 'BACKUP' TRACK 0, SECTOR 4. You must 'SKIP' this sector when "SUPERZAP" tries to 'read' it from the 'SOURCE' disk. Then, when you have finished recovering the file, you must 'COPY' it back to a 'system disk' MADE FROM THE MASTER VTOS 3.0 YOU RECEIVED FROM MRS. COOK'S SON, RANDY.

VTOS 3.0 WILL NOT FUNCTION UNLESS TRACK 0, SECTOR 4 IS UNFORMATTED! (At least that's the way it appears.) This is how Randy Cook is able to protect his software from pirating. It is a great idea but it makes it extremely aggravating to use. For a new user who is trying to use an applications package transferred to this system, who is not familiar

with computers, nor does he want to be -- he just wants to press a button and have the damn thing run his application -- this system will not find much favor at all.

#### 4.4 "NEW DOS 2.1"

It works! The current release has no known bugs and will do everything Radio Shack says cannot be done. It corrects every KNOWN error in TRSDOS 2.1. All in all, there are over 200 additions, corrections, and enhancements to TRSDOS. Many of the 'improvements' in TRSDOS 2.2 are poor 'implementations' of NEWDOS 2.1. (That's an opinion, and I cannot verify it, but from the looks of things, I'd give better than even odds that it's true.)

NEWDOS 2.1 is oriented to the programmer as well as the user. Included in the NEW DOS+ package, are utilities such as "SUPERZAP", "DIR-CHECK", "LMOFFSET" and others. These utilities are especially designed to assist the user and are very necessary if you need to recover data.

Data recovery on NEWDOS 2.1 generated disks is normal and routine for formatted data disks and system disks.

#### 10.5 RECOVERING A PHYSICALLY DAMAGED DISK

This unfortunate circumstance can occur in many unpredictable ways. You could have accidentally scratched the disk while using it for a shoe horn or a pipe cleaner. It could have been carelessly handled by a store clerk who thought it was a Master Charge Card and ran it through his little machine. A friend who had just dropped by to visit after a taco eating contest picked it up with thumb and forefinger expertly placed on the head access slot.

In any case, this aggravation is handled in exactly the same way as described in 10.4.

**CAUTION - MAKE SURE THE FOREIGN MATERIAL THAT IS ON THE DISK IS SUFFICIENTLY ATTACHED SO AS NOT TO CON-**

**TAMINATE THE DISK 'READ/WRITE' HEAD.** If you determine that the disk surface has been contaminated with a foreign substance such as finger prints, coffee, hand lotion etc., here is one semi-drastic measure you may take that I have used successfully on one or two occasions:

- (1) CAREFULLY slit the back of the disk jacket and remove the disk. DO NOT TOUCH THE DISK SURFACE! HANDLE BY EDGES AND CENTER ONLY!
- (2) CAREFULLY wash the disk in warm soapy water using your WET AND SOAPY fingers to GENTLY STROKE (DO NOT RUB) the disk.

- (3) THOROUGHLY rinse the disk in warm water.
- (4) If soap and water did not do the job, add alcohol to the water and try again.
- (5) Repeat - 3.
- (6) Place the disk on a sheet of NEWSPAPER. WARNING - PAPER TOWELS LEAVE LINT! Lay another sheet on top. Press gently. Repeat until the disk is dry.
- (7) Under no circumstances rub the disk!
- (8) When the disk is dry, CAREFULLY reinsert it into A NEW JACKET - DO NOT TOUCH THE MEDIA! (Here is a good use for those diskettes that weren't any good and you couldn't bring yourself to throw away.)
- (9) BACKUP THE DISK IMMEDIATELY!

**EIGHTY**

**DATA BASE MANAGER IDM-IV** \$69  
You can use it to maintain a data base & produce reports without any programming. Define file parameters & report formats on-line. Features key random access, fast multi-key sort, field arith., label generator, audit log. MOD-II version with more than 50 enhancements \$199.

**ACCOUNTS RECEIVABLE ACCT-III** \$69  
One or more drives. Order entry calculates sales tax, shipping, amount for multiple items. Credit checking, aging, sales analysis, invoices, statements and reports. As opposed to most other A/R, ours can be used by doctors, store managers, etc. MOD-II version \$149.

**WORD PROCESSOR** 16K \$39, 32K \$49, MOD-II \$49.  
First word processor specifically designed for the TRS-80 that uses disk storage for text. Written in BASIC. No special hardware and text limit. Use for letters, manuals & reports. 32K version features upper/lower case without hardware change and multiple input text files.

**MAILING LIST advanced MAIL-V** \$59.  
Fast sort by any field. Multiple labels and reports. 4-digit selection code, new zip code ext., screen input, live keyboard, powerful report writer. MOD-II \$99.

**INVENTORY INV-V** \$99.  
9-digit alphanumeric key for fast key random access. Reports include order info, performance summary, etc. Calculate E.O.Q. Powerful report writer. MOD-II \$149.

All programs are on-line, interactive, random access, virtually bug free, documented and delivered on disks. MOD-I requires 32K, DOS. We challenge all software vendors to offer low cost manuals so you can compare and avoid those high-priced undocumented, 'on-memory' programs. Send \$5 for a MOD-I manual and \$10 for MOD-II.

MOD-II programs are extensively modified, guaranteed to run with 1 year newsletter & updates. 10% off for ordering more than 1 MOD-II programs.

**MICRO ARCHITECT**

96 Dothan St., Arlington, MA 02174