RESEARCH IN WORD PROCESSING NEWSLETTER

Dr. Bradford A. Morgan and Dr. James M. Schwartz, Editors
South Dakota School of Mines and Technology Rapid City, SD 57701 Phone: (605) 394-2481

IN THIS ISSUE . . .

Volume 4 Number 9 December 1986

1986 SOFTWARE REVIEW INDEX
see page 15

Research in Word Processing Newsletter. Volume 4, Number 9. Copyright © 1986 by the South Dakota School of Mines and Technology. All rights reserved. ISSN: 0748-5484. Also indexed by **ERIC, INSPEC, COMPUTER LITERATURE INDEX,** and **SOFTWARE REVIEWS ON FILE.** The *Research in Word Processing Newsletter* is published 9 times a year (September through May) by the South Dakota School of Mines and Technology, Rapid City, South Dakota 57701-3995; (605)394-2481. Postage paid at Rapid City, South Dakota. **SUBSCRIPTION:** $15 per year (U.S.); $21 per year (Canada); $27 per year (foreign). Address all subscription inquiries and manuscripts to the Editors, *Research in Word Processing Newsletter,* SDSM&T, 501 E. St. Joseph, Rapid City, SD 57701-3995. Please allow 4 to 6 weeks for subscription processing. **POSTMASTER:** Send address changes to RWPN, South Dakota School of Mines and Technology, 501 E. St. Joseph, Rapid City, SD 57701-3995.

A Customized *Apple Writer* Startup Program

Michael W. Meeker

Writers who make frequent use of a word processor find that most of the time they set the program's defaults for a particular purpose. They may regularly use a certain printer, require a specialized glossary, work with unique stationery, or format their screen display in an unusual way. They may also regularly catalog a data disk before beginning, go to a menu, or run a program.

Most simple, menu-driven word-processing programs require writers to move through a series of menus each time they start up. On the other hand, most professional, full-function word processors, which allow for user-modification, can daunt the uninitiated beginner with their needed mnemonics. The *Apple Writer II* word processing program combines the best of both worlds. It loads simply and easily without elaborate menus or log-in requirements. One can begin typing immediately. But if specialized startup procedures are needed, *Apple Writer* will allow writers to customize the program in almost any manner they choose. The secret to this flexibility is WPL, a programming language resident in *Apple Writer*.

WPL stands for "word-processing language." It is primarily used to set up and run the word processor's form letter and mailing-list routines. Most users happily ignore it and its manual, [*Apple II* Word Processing Language] which is supplied along with the standard program manual. It may be that they feel it is too difficult to learn. One doesn't need to learn WPL to use *Apple Writer*, but as WPL expert Don Lancaster writes, ". . . owning *Apple Writer IIe* and not using WPL is as absurd as keeping a Porsche in the driveway just to listen to its FM radio.(50)"

Other *Apple Writer* users who have discovered WPL are equally enthusiastic: "*Apple Writer* is a perfectly adequate word processor," says writer Myron Berger. "The ProDOS version, which adds telecommunications and a few other new features, is even better. WPL, however, makes *Apple Writer* a terrific program, one that can constantly change and grow with your needs.(44)" WPL is not difficult to learn, the manual is unusually excellent, and with a little diligence almost anyone can begin building a customized word processor.

Although WPL at first looks like a combination of Pascal and BASIC, it is quickly learned because it is primarily a language of *Apple Writer* commands. The usual CONTROL commands in *Apple Writer* such as CONTROL-L ([L]oad), CONTROL-S ([S]ave), CONTROL-F ([F]ind), CONTROL-B (go to [B]eginning), etc. are represented by the same letters in a WPL program. Here is an example of a program which will load a document named "myfile," find and replace all occurrences of the word "Apple" with "Orange," and save the new file with a new name:

```
NY
L myfile
B
F/Apple /Orange /a
S myfile2
```

The "NY" that begins the program is the *Apple Writer* command for CONTROL-N ([N]ew) followed by a "Y" for "yes." This clears the memory before loading the document. The "a" after the last delimiter in the "find" command indicates that all instances of the word are to be replaced. To use this simple program, first save it as a regular text file, perhaps with a "wpl." prefix to identify it (ie., wpl.sample). The program can then be run by typing in "do" and the name of the file after the print command prompt (ie., [P]rint: do wpl.sample). WPL programming is easy to learn because it only involves setting up a series of logical *Apple Writer* commands. As Berger says, ". . . if you have learned *Apple Writer*—learned all the control commands, that is—then you already know perhaps 80% of WPL.(39)"

Only 17 additional commands are required, and most of them seem logical or are easy to remember: DO (run), GO (goto), IN (input), QT (quit), CS (compare strings), LS (load string), and so on. The command ND (no display) will turn off the monitor display and allow *Apple Writer* commands to run five times faster. All these unique WPL commands, incidentally, are prefixed with the letter "P": ie., PND, PQT, PGO, etc. While it is not a full programming language, WPL also allows the use of subroutine calls, four string variables, and three numerical variables. These commands, and the ability to create conditional expressions (if statement is true, execute next command; if false, ignore and go on to next command), make WPL a very powerful tool within the word processing environment.

Rather then attempt to explain the workings of WPL, which the *Apple Writer* WPL manual does wonderfully well, I will describe a startup program and suggest ways it can be modified. (This version is written for the ProDOS version of *Apple Writer II;* it will work equally well on the DOS 3.3 version if the ProDOS pathnames in the program are changed to DOS filenames followed by the required ",d1" or ",d2.")

The best way to learn WPL programming is to study and modify existent programs that perform familiar tasks. There is no startup program on the stock *Apple Writer*, but there is sufficient space to insert one. When *Apple Writer* is booted, it loads ProDOS, then the appropriate system file for the computer used (depending on its memory capacity), and then it looks for a file named "STARTUP." If it doesn't find one, it turns on the editing screen; if it does, it executes the startup program before it presents the editing screen—all automatically.

A startup program, then, can be totally transparent, quietly loading print program value files, glossaries, and screen parameters. Or it can pause and display options or menus. The following program presents both approaches. Note that the syntax of a WPL program statement can include labels (START, SELECT, OPTIONS), commands (PND, L, PGO), and arguments—additional information or modifiers (everything else). Labels begin in the leftmost position of the line, may contain any character or number (upper or lowercase) except a space, may be any length, and must be followed by at least one space. They can be on a line alone or be followed by the command. Commands must be preceded by at least one space, whether or not they follow a label. Arguments are everything else, including all the informational statements and the information which appears on the screen, and are always ended with a return.

Here then is a "startup" program. The P command prints a statement that will not be displayed. Information following PPR (Print) commands will be displayed on the screen. A PPR command followed by nothing creates a blank line. The [L] represents CONTROL-L, a form-feed command which clears the screen. (Such control characters must be preceded and followed by a CONTROL-V.) The first thing the user will see is the menu called "PRINTER OPTIONS."

```
START
 P        this program is called STARTUP
 PND
 PPR[L]
OPTIONS
 PPR
 PPP         PRINTER OPTIONS
 PPR            (1) Brother (Dynax, etc.)
 PPR            (2) Okidata (Library)
 PPR            (3) Epson FX 80, FX 100
 PPR            (4) Diablo (English Dept)
 PPR            (5) Imagewriter (I, II)
 PPR
PIN ◄press RETURN for default
     printer► ______ = $A
 PCS /$A/ /
 PAS /aw2master/prtglsry/GLOSS = $C
 PCS /$a/5/
 PAS /AW2master/prtglsry/IGLOSS = $C
 PCS /$A/4/
 PAS /AW2master/prtglsry/DGLOSS = $C
 PCS /$A/3/
 PAS /AW2master/prtglsry/EGLOSS = $C
 PCS /$A/2/
 PAS /AW2master/prtglsry/OGLOSS  =  $C
 PCS /$A/1/
 PAS /AW2master/prtglsry/DGLOSS = $C
```

The PIN command stops the program and asks for input, which is stored in string variable $A. The program then begins again, the PCS command comparing the string to various possibilities, finally assigning the string to variable $C. If the results of a compare string command is not "true," that is, if the input $A is **not** "5" for example, then the program skips the next command line. If true, then the string is assigned to variable $C and the program goes on to the next line. The first statement (PCS /$A//) looks for a response of "nothing," that is, only a RETURN, and then sets the default printer. This speeds the familiar user through the menu. If input other than a return or the numbers 1-5 is entered, the program continues but does not load any glossary. (It would be possible to "trap" such unaccepted responses and send the user back to the options menu. My own preference is to let the computer handle the problem by skipping the load glossary command which comes later in the program.)

The next section of the program operates without an input statement to stop the program. Instead, it uses a delay loop to momentarily display messages on the screen. Here it reminds the first time user how to access the *Apple Writer* help menu and how to display a special printer glossary menu. (The ability to create self-prompting glossary displays is another powerful feature of *Apple Writer*—a capability which would require a separate article to explain.) This next routine begins again by clearing the previous menu from the screen.

```
REMINDER
  PPR[L]
  PPR               OPEN APPLE KEYS
  PPR
  PPR     OPEN APPLE - ? for Apple Writer II
                    command summary
  PPR
  PPR  OPEN APPLE - z for printer controls menu
  PPR
SET
  PSX 25
LOOP
  PSX -1
  PAS (X) =$A
  PSZ 1
  PAS (Z) =$B
  PCS /$A/$B/
  PGO PREFIX
  PGO LOOP
```

The command PSX 25 in the delay loop sets the value of the numerical variable at 25. Note that the time the message appears on the screen can be modified by changing the original value of X. (When X = 25 the message is displayed for about 4 seconds.) When the loop decrements X to 1, when it is the same as Z, then the program will jump to the Label named PREFIX. Until then it jumps back to LOOP, causing a delay and allowing the message to remain on the screen.

The last two sections of the startup program set the default drive, using the normal commands within *Apple Writer*: [O] gets the DOS menu; [H] selects the option "set prefix," and the argument "D2" sets the default drive (the ProDOS pathname prefix) to drive 2. The seemingly superfluous P command is necessary to issue a carriage return which allows the program to exit the DOS commands menu. If there is no disk in drive 2, the computer will supply its own I/O ERROR message. Pressing RETURN will continue the program and automatically set the default drive back to drive 1.

Finally, the specialized printer glossary ($C) is loaded and a message momentarily displayed on the screen while the operation takes place. A final touch is to catalog the data drive and prompt the user to load a file. If the user presses RETURN, the PQT command turns the startup program off and displays the editing screen.

```
PREFIX
  PND
  PPR[L]
  PPR    DEFAULT DRIVE WILL NOW BE SET TO
                    DRIVE #2
  PPR
  PPR      (i/o error if empty—press return)
  OH,D2
  P
  PGO LOAD
LOAD
  PND
  PPR[L]
  PPR            "Loading Glossary: $C"
  QE$C
  PPR
  PPR[L]
  PPR
  PPR                 "Catalog"
  PPR
  OA
  P
  PPR
  PPR
  PPR = = = = = = = = = = = = = = = = = = = =
  PIN Enter file name or press ◄RETURN►: = $A
  L$A
  PQT
```

This program can easily be copied, modified to fit individual needs, and saved as a file named STARTUP on the *Apple Writer* program disk. One can easily change printer selections, add or delete messages, or omit whole sections. Even a very simple startup program, one which merely catalogs drive 2 or sets the default drive, would save much time. But be forewarned. Once you start working with WPL you will find more and more ways to customize your word processor, making it respond to your unique writing process.

WORKS CITED

Berger, Myron. "Automated Writing." *A+* Nov. 1985: 39-44.

Lancaster, Don. "Cracking the Code." *inCider* Feb. 1985: 50-3; 98-101.

Michael W. Meeker is an Associate Professor of English at Winona State University, Winona, MN 55987.

Bibliography Update

Bradford A. Morgan

Baker, Russell. "Quill-pen Days Vs. Oh-So-Easy Word Processing." *Celebrating the Computer Age: Man, Computers and Society* in *Computerworld*. 20:44 (November 3, 1986), p. 90.

Beil, Don. "*More*: A Review of a New Program That Gracefully Combines Idea/Outline Processing with Selected Desktop Publishing Features, Giving You More Power To Publish." *Personal Publishing*. 2:9 (September 1986), pp. 51-54.

Boot, Martin. "BOBRA: An Expert System for Automated Language Description As a Basis for Statistical Studies." *Literary & Linguistic Computing*. 1:2 (1986), pp. 55-62.

Boyle, F. Ladson. "Memory-Resident Spelling Checkers." *The Lawyer's PC*. 4:5 (November 1, 1986), pp. 8-11.

Breeze, Bill and Tony Iannotti. "XyWrite III: This Word Processor Is Really Much More Than a Word Processor. You Can Use It To Build a Complete Editorial/Production System. But You Do Have To Build It." *Personal Publishing*. 2:9 (September 1986), pp. 34-36, 40-42.

Brown, C.C.; J.L. Falk, and R.D. Sperline. *Preparing Documents with UNIX*. Old Tappan, NJ: Prentice-Hall, 1986.

Brown, Mariestelle M. "Inspiring Iowa Writers: DECtalk Allows Students To See, Hear, and Improve Their Writing in a Non-threatening Environment." *EDU: The Education Magazine of Digital Equipment Corporation*. 1:42 (Fall 1986), pp. 14-16.

Camp, John. "Computing & Composition: A Macintosh Alliance. Text-analysis Programs Go beyond Word Processing for College Students." *Electronic Learning*. 6:3 (November/December 1986), pp.49-50.

Crider, Janet. "*Word*'s Responsive Merge: Here's a Simple Method for Inserting Variable Text in Form Letters—without Creating a Separate Data File. *Microsoft Word*'s Powerful Merge Function Lets You Build On-Screen Prompts and Branching Systems That Can Import Different Text or Files for Each Possible Response." *PC World*. 4:11 (November 1986), pp.324-328.

D'Acquisto, Domonic R. "DDL: Standard #3? There Aren't Currently Any Printers That Understand Imagen's Document Description Language, but with Hewlett-Packard's Endorsement, It Could Be a Solution in the Future." Personal Publishing. *2:10 (October 1986), pp. 62-63.*

"*Directory of Word Processing Software.*" *T.H.E. Journal*. 14:4 (November 1986), pp. 99-101.

Donovan, Ronald John. "Word Processing: Online Outlines." *Writer's Digest*. (November 1986), pp. 64-65.

Eiser, Leslie. "I Luv To Rite: Spelling Checkers in the Writing Classroom." *Computer Classroom Learning*. 7:3 (November/December 1986), pp. 50-57.

Esplin, Kathryn. "VAX Word Processing Chases PC-Based Packages: Do Users Want WYSIWYG?" *Digital News*. 1:3 (November 3, 1986), pp. 39-45. (comparative review)

Frase, Lawrence T. and Mary Diel. "UNIX Writer's Workbench: Software for Streamlined Communication." *T.H.E. Journal*. 14:3 (October 1986), pp. 74-78.

Freeman, Terry. "A Trainer's Guide to Desktop Publishing: It May Be a Brave New World Out There, But It's Not Without Pitfalls for the Unwary Trainer." *Data Training*. 5:12 (November 1986), pp.45-47, 60-61.

Gancher, David. "Desktop Publishing Overview." *Computerland Magazine*. 2:1 (November/December 1986), pp. 40-43.

Gehani, Narain. *Document Formatting on the Unix System*. Summit, NJ: Silicon Press, 1986.

Grimm, Susan J. *How To Write Computer Documentation for Users*. Second Edition. New York: Van Nostrand Reinhold, 1986.

Harding, Tom. "New Role for a Veteran Performer: Desktop Publishing with the Apple II?" *Publish*. 1:2 (November/December 1986), pp. 70-71.

Harris, Larry R. "Speaking Up for Natural Language: In a World Full of Misconceptions, Here's What Natural Language Technology Is Not." *Information Center*. 11:10 (October 1986), pp. 21-23.

Hashway, Robert M. "From Text Processing to Communication: The Evolution of Computer Training." *T.H.E. Journal*. 14:3 (October 1986), pp. 88-89.

Hennig, Doug. "The Font Foundry: With Font Foundry's Character Editor You Can Create Your Own Hi-Res Character Sets. Its Character Generator Makes It Easy To Display Your Custom Characters on the Hi-Res Screen." *Nibble*. 7:11 (November 1986), pp. 15-16, 19-22, 27, 30, 35-36, 38-40, 44-47.

Hoenig, Alan. "Typesetting with TEX: This Program Runs on a Wide Variety of Computers, Handles Mathematic Typesetting, and with the Right Coding, Can Produce Beautiful Type on Dozens of Output Devices." *Personal Publishing*. 2:9 (September 1986), pp.30-33.

Holtz, Matthew. "The 3rd Word: Microsoft's Word Processor Just Got Better for Desktop Publishers." *Publish*. 1:2 (November/December 1986), pp. 54-55.

Huizhong, Yang. "A New Technique for Identifying Scientific/Technical Terms and Describing Science Texts: An Interim Report." *Literary & Linguistic Computing*. 1:2 (1986), pp.93-103.

Kirst, J. *Electronic Publishing*. London: Croom Helm, 1986.

Marshak, R. *Word Processing Software for the IBM PC*. New York: McGraw Hill, 1985.

Marshall, John C. "Microcomputer Software. 2. Scientific and Technical Word Processing on a Personal Computer: Has the Time Come?" *Journal of Chemical Information and Computer Sciences*. 26:3 (August 1986), pp. 87-92.

Mendelson, Edward. "Word Processing: A Continuing Guide for the Perplexed." *Yale Review*. 75:3 (Spring 1986), pp. 454-480.

Miller, Donald W., Jr. "Font Blaster: Now You Can Print Tool Kit Characters on Your Prowriter or ImageWriter, or You Can Create Your Own Custom Printer Characters." *Nibble*. 7:11 (November 1986), pp. 50-54, 58-60, 63-64, 67-68.

Nace, Ted. "Grappling with Text and Graphics: Merging Words and Images on the PC Takes Commitment." *Publish*. 1:2 (November/December 1986), pp. 56-61.

"A Perspective on Desktop Publishing." *The Edutech Report*. 2:8 (November 1986), pp. 1, 6.

Robinson, Kit. "*Racter*: Computerized Conversation." *Computerland Magazine*. 2:1 (November/December 1986), pp. 55, 58.

Rosenthal, Morton. "Checking Out Publishing Software." *Computerworld Focus*. 20:45A (November 12, 1986), pp. 31-32.

Roth, Steve. "PostScript Typefaces: PostScript Currently Offers More Versatility in Working with Type Than Any Other Inexpensive System. Here Are Some Tricks for Getting the Most Out of PostScript Printers." *Personal Publishing*. 2:10 (October 1986), pp. 30-31, 34.

Schleifer, Neal. "Making the Leap to Desktop Publishing: Whether You Put Out a School Newspaper, a Literary Magazine, a Yearbook or Just a PTA Newsletter, You (and Your Kids) Can Do It More Efficiently with Today's New Desktop Publishing Tools." *Computer Classroom Learning*. 7:3 (November/December 1986), pp.39-41.

Sell, William. "Headers and Footers: Keeping Your Head on Straight." *Access 86: The Magazine for Wang System Users*. 4:3 (November 1986), pp. 36-37.

Semar, Catherine. "Selecting a Software Package and Laser Printer To Meet Your Electronic Publishing Needs: A New Class of Document Production in the VAX and MicroVAX Environments Is Available through a Variety of Packages." *Hardcopy*. 6:11 (November 1986), pp. 158-161.

Steinke, Steve. "Desktop Publishing on the Macintosh." *Computerland Magazine*. 2:1 (November/December 1986), pp. 46-49.

"Students React Favorably to Freshman Writing Lab at Lee College in Texas." *T.H.E. Journal*. 14:3 (October 1986), pp. 61-62.

Ulick, Terry. "Personal Publisher: A Look at One of the First 'What You See Is What You Get' Programs for the IBM PC, and Why It Is Causing Such Controversy." *Personal Publishing*. 2:9 (September 1986), pp. 20-23, 25, 28.

Wallia, C. J. "Desktop Publishing: Preparing Master-Copy on LaserWriter and Linotronic 100." *Printing Journal*. 13:9 (September 1986), pp. 17, 36-37.

Wohl, Amy. "Classifying Desktop Publishing." *Computerworld Focus*. 20:45A (November 12, 1986), pp. 27, 30.

Writing for the Computer Industry

The Fourth Annual Conference on Writing for the Computer Industry will be held on Saturday, August 15, 1987, at MIT. The 1987 conference will include 30-minute oral presentations and practical hands-on workshops on how to write manuals and online user aids, human-computer interaction, information design, linguistic style, management of writing in the computer industry, and current research into artificial intelligence. Proposals were collected until January 9, 1987. Contact Dr. Edward Barrett, Conference Director, MIT Writing Program, Massachusetts Institute of Technology, Room 14E-310B, Cambridge, MA 02139.

ACH at the 1987 MLA Convention in San Francisco

The Association for Computers and the Humanities is making plans to develop a program on "The Nature of Text" at the Modern Language Association's 1987 convention in San Francisco. Inquiries and proposals should be addressed to either ■ Nancy Ide, Dept. of Computer Science, Vassar College, Poughkeepsie, NY 12601, or to ■ Donald Ross, Department of English, University of Minnesota, Minneapolis, MN 55455.

20th Annual Small College Computing Symposium

Macalester College will host the 20th Annual Small College Computing Symposium on April 10-11, 1987. Several papers and panel sessions at SCCS, including featured speakers, will report on recent advances in the use of computers to teach writing, linguistics, and literature. Contact Dr. Richard K. Molnar or Dr. G. Michael Scheider, Department of Mathematics and Computer Science, Macalester College, 1600 Grand Avenue, St. Paul, MN 55105.

European Electronic Publishing Exhibition

Frankfurt, West Germany, will be the site of the European Corporate Electronic Publishing Exhibit and Conference to be held June 10-12, 1987. Contact David Henry Goodstein, InterConsult, Inc., 48 Brattle Street, Cambridge, MA 02138, or call (800) 424-2377 or (617) 547-0332.

1987 NCTE Convention in Los Angeles

Both individual and group concurrent session program proposals are being considered until January 23, 1987, by the National Council of Teachers of English for the 77th Annual Convention to be held in Los Angeles between November 20-25, 1987. "Making Connections" is the announced theme. A few one- and two-day workshops are also open. Contact the NCTE 1987 Program, 1111 Kenyon Road, Urbana, IL 61801.

1987 National Educational Computing Conference in Philadelphia

June 24-26, 1987, is the time set for the 8th National Educational Computing Conference to be held in Philadelphia. Sessions, tutorials, and workshops will cover a variety of computer applications at all educational levels. As usual, an extensive exhibit of software and hardware products is also planned. Contact Laurie Shteir, Department of Computer and Information Science, Room 303 Computer Activity Building, Temple University 038-24, Philadelphia, PA 19122, or call (215) 787-1681.

Desktop Publishing To Be Featured at Small Press Expo '87

Small Press Exp '87 will be held in New York City on March 2-4, 1987. *Small Press* magazine editor Michael Coffey will serve as conference director. He notes that "right now the hottest topic in the publishing industry is desktop publishing...Since we naturally cover this field in our magazine, it too will be featured in our program." Both seminars and exhibit booths are planned. Contact Michael Coffey, Meckler Publishing Corporation, 11 Ferry Lane West, Westport, CT 06880.

SNOBOL4 Programming as Word Processing

Eric Johnson

SNOBOL4 is a computer language that is designed to do exactly the kinds of things that those interested in word processing want to do: text analysis, searching, replacing, editing, concordance and index generation, and all kinds of word manipulations. It is intended for non-numeric computing.

The name SNOBOL was taken, rather capriciously, from StriNg Oriented and symBOlic Language. This very powerful and original programming language was developed over a period of years by computer scientists at Bell Laboratories. Strictly speaking, the language that is now available is SNOBOL4. The playfulness of the naming indicates that the inventors of SNOBOL were interested in doing things with words as well as with computer code.

HOW SNOBOL WORKS

It is perhaps easiest to understand the power and flexibility of SNOBOL4 by briefly describing several programs.

Suppose you are using a word processor to write an article or an assignment that requires, say, 3,000 words. You can get a rough idea of the total words in your manuscript by counting the number of words in an average line and then counting the number of lines and multiplying. Or, if a more precise number were wanted, you could try to count all the words. Either way is time consuming and not likely to yield an exact answer.

You could also quickly write a short program in SNOBOL4 that would give the precise number of words in a document. Following is such a program:

```
          LTRS  =  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
+                  "abcdefghijklmnopqrstuvwxyz'0123456789"
          WORD.PATTERN  =  BREAK(LTRS)  SPAN(LTRS)
GET.LINE  LINE  =  INPUT                        :F(OUT)
GET.WORD  LINE   WORD.PATTERN   =               :F(GET.LINE)
          WORDS  =  WORDS + 1                   :(GET.WORD)
OUT       OUTPUT  =  "There are " WORDS " words in the ms."
END
```

These eight lines are the complete program.

It is always an advantage to write a program yourself, if that is possible, since if you do write the program, you can be sure it does exactly what you want; you are controlling the computer. For example, the first three lines of this program define what you want to consider to be a word: you are instructing the computer to span any string of letters and numbers plus the apostrophe; it breaks off and discards any other characters. If you do not want numbers to count as words, take them out of the second line. If you want hyphenated words to count as one word, include the hyphen somewhere within quotation marks in the first two lines.

After the initialization of the first three lines, the program reads in a line of text (line 4). Line 5 is the heart of this program; it uses the operation of pattern matching for which SNOBOL4 is famous. The first three lines construct a pattern of what is to be considered a word (it is called WORD.PATTERN in this program). Line 5 picks out such patterns from each line. As each word is found, 1 is added to a counter (line 6). When all the words in a line have been counted, a new line is read, and when there is no more text to read, a statement telling how many words are in the file is printed (line 7).

The output from this word-count program (using this article as input text) follows:

```
There are 2418 words in the ms.
```

It would be quite easy to have the word-count program also compute average sentence length. It would only be necessary to add a pattern for sentences (anything ending in ! or ? or . followed by two spaces), perform pattern matching to count them, then when all sentences and words have been counted, divide the number of words by the number of sentences, and output the answer.

Some users of word-processing software say they do not want to learn to program. All right, don't think of it as programming. Think of it as giving commands to software. And indeed, a SNOBOL4 compiler is software, and it has so many built-in features that writing programs in it is like issuing word-processing commands.

There are many useful programs that can be created in SNOBOL4 in fewer than twenty lines.

How about a program to produce a word-frequency list so that you can tell if you are using the same words over and over. It requires only a little modification of the word-count program:

```
           LTRS  =  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
+                     "abcdefghijklmnopqrstuvwxyz'0123456789"
           WORD.PATTERN  =  BREAK(LTRS)  SPAN(LTRS)  .  WORD
           H  =  TABLE( )
GET.LINE   LINE  =  INPUT                      :F(OUT)
GET.WORD   LINE  WORD.PATTERN  =               :F(GET.LINE)
           H◄WORD►  =  H◄WORD►  +  1           :(GET.WORD)
OUT        J  =  SORT(H,  1)
           S  =  0
PRINT      S  =  S  +  1
           OUTPUT  =  J◄S,1►  '      '  J◄S,2► :S(PRINT)
END
```

This twelve-line program picks out words (lines 1-3, 5-6) in about the same way as the word-counter program, but the WORD.PATTERN is a little different. When a word pattern is found, it is stored in the variable WORD and thus is available to the program. A table, H, is defined in line 3. When words are identified by pattern matching (line 6), they are added to the table by being used as subscripts and 1 is added to the table as a counter for each word. When all words have been added to the table and counted, the table is converted to an array which is sorted in alphabetical order (line 8). The words and the number of occurrences are then printed (lines 9 to 11). The SORT function which converts the table to an array and sorts it alphabetically (in a single line of code, line 8) is available only in one version of SNOBOL4 for microcomputer. A small portion of the program output (using an earlier version of this article as the text) looks like this:

```
with            12
within           2
without          1
word            22
words           18
worked           1
working          1
works            1
workstations     1
would            2
write            6
writer           1
writing          1
written          5
wrote            5
years            1
```

```
yield          1
you           19
your           2
yourself       1
zip            1
```

Some SNOBOL4 programs take a while to run. For example, this word-frequency program took several minutes to read in the text of this article from disk and do the computations. SNOBOL4 is designed to be efficient for the programmer—not necessarily efficient for the computer (which, if a choice must be made, is the way most of us like it).

Not all computer languages are as easy to use as SNOBOL. COBOL, a business programming language, which, despite the rhyme, is absolutely unrelated to SNOBOL, might require a hundred lines to accomplish the same thing as my twelve-line program. Understandably, those interested in word processing refuse to learn languages like COBOL. However, microcomputer users can learn to use SNOBOL4 and have powerful programs at their finger tips with little more trouble than learning the arbitrary commands of most word-processing programs. Moreover, the logic of SNOBOL programming is different from that of other languages, and it may appeal to those who dislike programming in general.

In an effort to prove that Marlowe wrote the works attributed to Shakespeare, T.C. Mendenhall argued that word length is an indicator of an author's style. With just a very few changes, the word-frequency program can list word lengths instead.

```
          LTRS  =  ''ABCDEFGHIJKLMNOPQRSTUVWXYZ''
+                  ''abcdefghijklmnopqrstuvwxyz'0123456789''
          WORD.PATTERN  =  BREAK(LTRS)  SPAN(LTRS)  .  WORD
          A  =  ARRAY(' 19')
GET.LINE  LINE  =  INPUT                           :F(OUT)
GET.WORD  LINE  WORD.PATTERN  =                    :F(GET.LINE)
          SZ  =  LT(SIZE(WORD),19) SIZE(WORD)      :S(PUT.IN)
          SZ  =  19
PUT.IN    A◄SZ►  =  A◄SZ►  +  1                    :(GET.WORD)
OUT       S  =  0
PRINT     S  =  LT(S,18)  S  +  1                  :F(LAST.L)
          OUTPUT  =  DUPL(' ',2  -  SIZE(S) )  S'      '
+           DUPL(' ',4  -  SIZE(A◄S►) )  A◄S►         :(PRINT)
LAST.L    OUTPUT  =  'Over 18'  DUPL(' ',4  -  SIZE(A◄19►) )  A◄19►
END
```

Except for changing the table to an array (line 4), the first six lines of this program are the same as the word-frequency program. The heart of this letter-counting program is in line 7. The function SIZE(WORD) returns the number of letters in each word. The way I have coded it, if the number returned is less than 19, 1 is added to the count in an element of the array using the number of letters as a subscript; if there are more than 18 letters, one is added to the count in the 19th element of the array. When there are no more words, the array is printed out. The DUPL function is used in the last lines to insert spaces in order to print numbers in aligned columns such as the following.

```
 1      40
 2     176
 3     166
 4     159
 5     116
 6      52
```

7	87
8	60
9	28
10	18
11	18
12	4
13	7
14	5
15	1
16	
17	
18	
Over 18	2

Suppose you have lines of names and addresses with first name first; you want to alphabetize the list by last name, but it does not always begin in the same column; all that you know is that the last name follows the first name and a space. A SNOBOL program about the same length as those above can do this; I wrote it in 17 lines of code. That is not many considering it is doing something that is almost impossible in most languages (in standard programming, names to be sorted have to start in the same column), and if it could be written in another language, it would take hundreds of lines of code. Such a SNOBOL program can be modified easily to sort the names and addresses by zip code.

Perhaps you have a list of names and addresses with a variable number of blank lines between clusters of consecutive lines of names and addresses, and you want to print them on three-up sheets of labels. A 19-line program will do that. Once again, such a SNOBOL program allows the user to control all variables, such as the width of the lines printed on the labels.

There are many other useful programs that can be written in SNOBOL in a few lines: index programs, concordance programs, programs which point out basic grammar and usage blunders, readability checkers, and even artificial intelligence applications. Programs to search a text for a specified string of characters and replace them with others can be written with unusual economy. Following is a complete program to search a text for double quotation marks; if found, they are replaced with a single quotation mark; the entire line is then output surrounded with double quotation marks.

```
AGAIN  OUTPUT  =  '"'  REPLACE(INPUT,'"',"'")  '"'   :S(AGAIN)
END
```

Only one line of SNOBOL code, plus an END statement is required. This economy is achieved by embedding the INPUT function within the REPLACE function and what is returned from that function is given to the OUTPUT function along with enclosing quotation marks. Its ability to format and adjust lines of text should make SNOBOL4 very useful in desktop publishing applications.

SNOBOL COMPILERS

SNOBOL4 was originally designed for mainframe operation, and it is available for almost every size and make of mainframe, but with the advent of larger memory microcomputers, it can run on them. Three excellent SNOBOL compilers are available for microcomputers.

SNOBOL4+ was written by Mark Emmer. Although it was marketed by Prentice-Hall for a time, it should now be ordered directly from Emmer at Catspaw, Inc., P.O. Box 1123, Salida, Colorado 81201. The price of $95.00 includes *SNOBOL4+ The SNOBOL4 Language for the Personal Computer User*, a very helpful, two-hundred-and-fifty-page book of instruction. All of the programs in this article were tested with SNOBOL4+.

Viktors Berstis wrote Minnesota SNOBOL4 when he worked for IBM in Rochester, Minnesota (thus the name.) It may be purchased for $60.00 from Berstis International, P.O. Box 441, Millwood, New York 10546.

Both Minnesota SNOBOL4 and SNOBOL4+ will run on an IBM PC, XT, AT or any other microcomputers using 8086/88/186/286 processors; IBM compatibility is not required, only MS-DOS compatibility. A minimum of 128K bytes of memory is essential, but 192K or more is recommended. Both are full implementations of mainframe SNOBOL4, and SNOBOL4+ has some additional features (such as the SORT function).

Robert Dewar and A.P. McCann wrote a mainframe version of SNOBOL4 called SPITBOL (SPeedy ImplemenTation of snoBOL). David Shields and Dewar wrote an implementation of SPITBOL for microcomputer. PC SPITBOL requires 196K of memory. It produces an executable file (an EXE file) which may be run without the compiler. It can be ordered from Robert B.K. Dewar, SPITBOL Orders, 73 5th Ave., New York, NY 10003 for $195.00.

Mark Emmer is beta testing a new version of his SNOBOL4+ which will have additional functions and features; he is also working on an implementation of SPITBOL that will run on AT&T Unix and thus will be available for Apollo and Sun workstations.

FOR MORE INFORMATION

Several introductory books about SNOBOL have been written. Susan Hockey's *SNOBOL Programming for the Humanities* (Oxford University Press) is recommended. *A SNOBOL4 Primer* by Ralph and Madge Griswold is intended for those who have had no programming experience; it is published by Prentice-Hall.

The standard reference is *The SNOBOL4 Programming Language*, Second Edition, by R.E. Griswold, J.F. Poage, and I.P. Polonsky; it is also published by Prentice-Hall. It is often referred to as "The Green Book." Ralph Griswold publishes the *SNOBOL4 Information Bulletin*. He will gladly put you on the mailing list, free of cost, if you send your name and address to SNOBOL4 Project, Department of Computer Science, The University of Arizona, Tucson, AZ 85721.

ICEBOL, the International Conference on the Applications of SNOBOL and SPITBOL, has been held at Dakota State College. ICEBOL publishes a Proceedings of papers on machine translation, literary analysis, grammar identification, educational drill and practice, and many other topics. For information, write ICEBOL, 114 Beadle Hall, Dakota State College, Madison, SD 57042.

SUMMARY

Programs in SNOBOL4 do the kinds of things that those interested in word processing want to do. Programs are usually short and not time consuming to write. Programming in SNOBOL4 allows the writer to control the computer rather than merely selecting from options someone else offers in a word-processing package. Anyone seriously interested in word processing should learn and use SNOBOL4.

Eric Johnson is Head of the Division of Liberal Arts at Dakota State College, Madison, SD, and is the Director of ICEBOL, The International Conference on Applications of SNOBOL and SPITBOL. He holds a Ph.D. in English literature and has written articles on computerized text analysis and literary criticism.

RWPN Back Issues

Vol. 4 No. 8 [Nov. 1986], 18 pp.
Multi-Lingual Word-Processing with the Macintosh; Bibliography Update; Memory Resident Thesaurus Programs

Vol. 4 No. 7 [Oct. 1986], 18 pp.
FinalWord II: Word Processing for a College Writing Program; Word Processing as a Tool for Revision; Bibliography Update

Vol. 4 No. 6 [Sep. 1986], 18 pp.
Desktop Publishing That Anyone Can Do; Software Review: *AppleWriter II*; Computer Projected Thinking in the Classroom; Bibliography Update; Electronic Outlining Comes of Age

Vol. 4 No. 5 [May 1986], 23 pp.
Word Processing, Electronic Research, and Desktop Publishing: Annual Cumulative Bibliography

Vol. 4 No. 4 [Apr. 1986], 31 pp.
Personal Publishing on Microcomputers; Writing-Instruction Software with *HBJ Writer*; Software Review: *WordPerfect 4.1*; Bibliography Update; Scholar's Software Library: *Rightwriter 2.0*

Vol. 4 No. 3 [Mar. 1986] 15 pp.
More on Low-Cost Word Processing; Classroom Computers & Job Seeking Strategies; Bibliography Update; Scholar's Software Library: *Fancy Font 2*

Vol. 4 No. 2 [Feb. 1986], 21 pp.
A Typology of Word-Processing Programs; News & Notes; Bibliography Update; Addendum to *Microsoft Word* (Macintosh)

Vol. 4 No. 1 [Jan. 1986], 17 pp.
Diagrammatic Writing: "Larger Vision" Software; 1985 Software Review Index; Review: *Nota Bene*; Bibliography Update; *TELECOMMUTER*: Laptop to PC Link

Vol. 3 No. 9 [Dec. 1985], 21 pp.
The English Department Microlab: An Endangered Species; Software For Text Analysis & Writing Instruction; Bibliography Update; Software Review: *Microsoft Word* (Macintosh)

Vol. 3 No. 8 [Nov. 1985], 13 pp.
Effects of Word Processing on the Correctness of Student Writing; Review: *Quintillian Analysis*; Bibliography Update; Review: *NoteBook II*.

Vol. 3 No. 7 [Oct. 1985], 17 pp.
Introducing Word Processing to Students; Review: *Readability*; A Cautious View of Computers in Teaching Writing; Bibliography Update; Review: *SAMNA +*

Vol. 3 No. 6 [Sep. 1985], 15 pp.
Word Processing on a Budget; Bibliography Update; Scholar's Software Library: *ProofWriter*; Software Review: *MacWrite 4.5*

Vol. 3 No. 5 [May 1985], 19 pp.
Word Processing, Writing, Literature, and Linguistics: Annual Cumulative Bibliography

Vol. 3 No. 4 [Apr. 1985], 15 pp.
Computers, Word Processing, and the Teaching of Writing; Scholar's Software Library: *Framework*; Word-Processing Errors in Technical Writing; Bibliography Update; *ZYIndex*: State-Of-The-Art Text Management

Vol 3 No. 3 [Mar. 1985], 11 pp.
Micros, Minis, and Writing: A Critical Survey; Bibliography Update; Review: *Microsoft Word* (IBM)

Vol. 3 No. 2 [Feb. 1985], 11 pp.
Setting Up a Word-Processing Microlab; Bibliography Update; Scholar's Software Library: *ASCII*; Software Review: *WordStar 3.3*

Vol. 3 No. 1 [Jan. 1985], 11 pp.
A Scholar's Typology of Database Management Software; Bibliography Update; *Textra* Extra; *WordStar* Tips and Tricks

Vol. 2 No. 9 [Dec. 1984], 11 pp.
Teaching News Writing With a Computer; Database Management for Teachers and Researchers III; Bibliography Update; Software Review: *WordMARC*

BACK ISSUE PRICES: Volume 4—$3.00 each; Volumes 3 and 2—$2.00 each

Please specify volume and issue numbers on a separate sheet of paper. Make checks payable to *RWPN* and send to Editors, *Research in Word Processing Newsletter*, South Dakota School of Mines & Technology, 501 E. St. Joseph, Rapid City, SD 57701-3995. Please allow 4 to 6 weeks for order processing and delivery.

1986 Software Review Index

SOFTWARE	*ISSUE*
AppleWriter II	4-6 (Sep.) 4-9 (Dec.)
Fancy Font 2	4-3 (Mar.)
FinalWord II	4-7 (Oct.)
HBJ Writer	4-4 (Apr.)
Nota Bene	4-1 (Jan.)
PC-Outline	4-6 (Sep.)
Random House Reference Set	4-8 (Nov.)
Ready!	4-6 (Sep.)
Rightwriter 2.0	4-4 (Apr.)
SNOBOL 4	4-9 (Dec.)
TELECOMMUTER	4-1 (Jan.)
ThinkTank	4-6 (Sep.)
Turbo Lightning 1.0	4-8 (Nov.)
Webster's New World Writer/Thesaurus	4-8 (Nov.)
Word Finder 3.2	4-8 (Nov.)
WordPerfect 4.1	4-4 (Apr.)