TEXT TECHNOLOGY

AD DOCENDUM ◦ INVESTIGANDUM ◦ SERVIENDUM ▲ WRIGHT STATE UNIVERSITY 1967 ▲

Volume 1 Number 5
September 1991

In this issue . . .

EDITOR'S CHOICE

Jim Schwartz

Errata

Last issue, a figure was inadvertently left out of Eric Johnson's "Column One." Here is what should have been Fig 4. in Eric's article, "Fast Processing for Text."

PROGRAMS	486-25	386-33	386-25	386-20
Icon 1	-.-	1.7	3.1	4.0
Icon 2	-.-	1.8	3.3	4.3
Icon 3	-.-	1.7	3.1	4.1
Icon 4	-.-	1.9	3.4	4.4
SPITBOL 1	-.-	1.8	3.1	4.2
SPITBOL 2	-.-	1.7	2.9	3.9
SPITBOL 3	-.-	1.8	3.0	4.2
SPITBOL 4	-.-	1.6	3.0	3.8
SPITBOL 5	-.-	1.8	3.3	4.2

Table 4. Relative speed of execution as a factor of 486-25 timings.

TEXT TECHNOLOGY

AD DOCENDUM INVESTIGANDUM SERVIENDUM 1967 WRIGHT STATE UNIVERSITY

Editors
Jim Schwartz
Arthur A. Molitierno

Contributing Editors
Randal Baier
Cornell University
Paul Fritz
University of Toledo
Eric Johnson
Dakota State University
Stephen Miller
Oxford University
Brad Morgan
South Dakota School of Mines & Technology
Guy Pace
Washington State University
Bryan Pfaffenberger
University of Virginia
Ian Richmond
University of Western Ontario

This publication is funded in part by a research grant from the *Wright State University President's Club.*

TEXT Technology. Volume 1, Number 5. Copyright ©1991 by Wright State University. All rights reserved. ISSN: 1053-900X. Also indexed by ERIC, COMPUTER AND CONTROL ABSTRACTS (INSPEC), COMPUTER LITERATURE INDEX, LITERARY CRITICISM REGISTER, and SOFTWARE REVIEWS ON FILE. *TEXT Technology* is published bimonthly by Wright State University—Lake Campus, 7600 State Route 703, Celina, Ohio, USA 45822-2921; (419) 586-2365. Postage paid at Celina, Ohio. SUBSCRIPTION RATES US funds: $20 (US); $27 (Canada); $35 (Foreign). Address all subscription inquiries and manuscripts to the Editors, *TEXT Technology*, Wright State University—Lake Campus, 7600 State Route 703, Celina, Ohio, USA 45822-2921. Note that the Editors are not responsible for any unsolicited manuscripts. Allow 4 to 6 weeks for subscription processing. Selected back issues are also available; contact the Editors for more details. POSTMASTER: Send address changes to *TEXT Technology*, Wright State University—Lake Campus, 7600 State Route 703, Celina, Ohio, USA 45822-2921.

WRIGHT STATE LAKE CAMPUS

TEXT Technology is composed on a Northgate Elegance 486-25 computer and printed using an IBM Personal Page Printer II Postscript printer. Images and text are scanned with a Hewlett-Packard ScanJet Plus scanner. Software used includes Microsoft *Windows*, Aldus *PageMaker*, *Caere Omnipage 386*, *WordPerfect*, *HP Scan Gallery* and *Paintbrush*, Micrografx *Designer*, *CorelDRAW!*, Lotus *Ami Professional*, Microsoft *Windows Write*, and Symantec's *Q&A*.

Eric Johnson

A Limitation of Computers

In recent years, there have been spectacular advances in non-numeric computing, especially for text processing: accurate and fast optical character recognition, sophisticated desktop publishing, powerful programs for analysis of texts. Future progress will probably make the present seem stupid and amateurish by comparison. Yet, there seem to be signs of inherent limitations of computing for text applications.

Programs can be written that identify and categorize the words and sentences of texts, and such information can be extremely useful to the researcher in determining how literary works achieve their artistic effects. However, it is only with great difficulty that a computer can identify literary images, and such identification is far from unerring. It is uncertain whether a computer program can be written to deal accurately with metaphoric language—let alone to deal with questions of literary value. Even the simple meaning of words is often beyond the scope of a computer. There may be a principle of inherent limitation of computers that would be useful to identify.

As a college student, I worked at a gas station. The owner of the station posted a list of names of people from whom we were not to accept checks because they were likely to bounce. We were told to be sure that the signature on a check was not one of the names on that list. One morning the owner and I were recording the checks from the previous night. He read the name of the signature on each check, and I wrote it on his bank deposit slip. About halfway through the stack of checks, he started to read a name, stopped, and uttered an expletive. The check was signed "U. R. Stuck." I considered pointing out to him that this was not a name on his list, but I thought I had better not.

Accepting a check signed "U. R. Stuck" is exactly the sort of thing a computer would do. It would solemnly determine that this was not a name on the do-not-accept-checks-from list. If the name "U. R. Stuck" were added to the list, the computer would probably nevertheless accept a check from "U. R. Stuck-Again," and, of course, the computer would immediately accept a check signed "I. Gotcha."

Suppose I ask the students in a class to tell me their names. If one student replies, "I will not tell you," I might react in various ways, but I would never mistake "I will not tell you" for a student's name. When I want to log onto a mainframe, the computer asks me to enter my user name; if I enter "I will not tell you," I will get some kind of message that says that is not a valid user name.

I will call it the Johnson Principle that is demonstrated in a computer's inability to correctly identify "U. R. Stuck" and "I will not tell you." The name of this principle is based on the fact that I was as naive as a computer in accepting the check. A kind of imaginative jump or an intuitive leap is required to recognize the difference in kind between a name like "T. S. Eliot" and "U. R. Stuck" or between "user 45" and "I will not tell you." The Johnson Principle holds that a computer cannot make such a leap, and therefore it is not possible for a computer to recognize a particular kind of unanticipated information that a person could or should instantly identify. As the Johnson Principle demonstrates, there can be a profound difference between the processing of computers and the operation of human minds.

I created a grammar and style checker to help college students identify and correct blunders and weaknesses in their writing. Students and teachers have found my computer program generally useful, but it has limitations. Students were fond of writing phrases like "It was not all that cold" and "She is not all that hungry." The checker would flag "not all that" and ask the student to consider rewriting the sentence using more specific words. One day a student gleefully showed me that the checker had flagged "Not all that glitters is gold."

The checker attempts to identify gerunds and participles that end in "ing," and it encourages their use. The idea is that a word such as "running" signals a difference in sophistication between a sentence like "While I was running a 10K race, a dog bit me" and "I ran a 10K race. Then a dog bit me." The checker uses a combination of algorithms (there must be at least one vowel before the "ing") and tables to avoid encouraging the writer to use words like "sing," "bring," "thing," or "something."

A published review of the checker sarcastically paraphrased one of its messages: "Words like PUDDING should be used more frequently." Obviously the checker misidentified the word "pudding" as a gerund or participle because the algorithms and lookups did not anticipate the form. Had I recognized what I now call the Johnson Principle when I created the checker, I might have known that there would be problems, and I might have avoided some of the pitfalls.

Computer languages such as Prolog that are used for so-called artificial-intelligence (AI) programming are different in form and syntax from languages like *SNOBOL4* and Icon that are used for analysis of texts, but the Johnson Principle is equally applicable. For example, we could write a simple AI program that would identify classes of animals based on key characteristics.

The program might establish the relation that anything that has feathers is a bird. Thus when a user tells the program that a robin has feathers, it would reply that a robin is a bird. If a user tells the program that a pillow has feathers, the program would reply that a pillow is a bird!

Now it is certainly possible for a programmer to modify the program so that if "pillow" is entered, it is not given the predicate "is a bird," but so far from invalidating the Johnson Principle, such a modification proves the Principle: a human being must intervene in certain cases because a computer cannot anticipate them.

The Johnson Principle is relevant to some kinds of mathematical programming. For example, a human being can recognize the usefulness of both 3.1416 and 22 divided by 7 as approximations of the ratio of the circumference of a circle to its diameter, while rejecting 3.1421 as meaningless although it falls between 3.1416 and 22 divided by 7. A computer will have all kinds of problems comparing these numbers with the concept of pi.

Applications of the Johnson Principle occur mainly in the arena of text processing because reading and understanding natural language often requires some kind of imaginative leap or intuitive jump—precisely the sort of actions a computer cannot perform. The impact of a literary work commonly requires complex recognition of multiple meanings and shades of meaning of words in their context, and thus the effects of literary works are very difficult to process with a computer. Because humor is often based on multiple meanings and unexpected imaginative leaps, the Johnson Principle would indicate the impossibility of writing a computer program to correctly identify all humorous passages in a novel, say, by P. G. Wodehouse.

The Johnson Principle is a recognition of a limitation of computers. Knowing about the Principle may allow programmers to write better programs, and it should teach all of us to form more realistic expectations about what computers will do in the future.

* * * * * * * * * *

Eric Johnson is Professor of English and Dean of the College of Liberal Arts at Dakota State University, Madison, SD 57042. He is the Director of the International Conference on Symbolic and Logical Computing, and he has published more than fifty articles and reviews about computers, writing, and literary study. His BITNET address is ERIC@SDNET.

PERFECT TECHNIQUES

Guy Pace

Grammar & Style Checking,
or Strunk and White Never Had It So Good!

I'm a software junkie. At work I get to use the latest release of products. I really do have to keep ahead of my students and the people I help. At home I keep certain programs as current as my wallet allows. Those programs include *WordPerfect*, *Right Writer*, *Turbo Pascal*, tax software and bookkeeping software. Other programs languish in semi-usefulness, several versions out of date and ready to become "shelfware."

When I got the notice that *Right Writer* 4.0 was available, I immediately ordered the upgrade. That is the most essential *WordPerfect* add-on in my software library. Without it, my prose would degenerate into—well, let's not get obscene.

I first bought *Right Writer* in version 2-point-something. It was good, as far as grammar checkers went. I didn't like having to exit *WordStar* (what I was using then) to run *Right Writer*. I learned to live with inconveniences in exchange for the services of the grammar checker.

I replaced WordStar with *WordPerfect*, and life improved. I could edit the original file while I had the marked up file in the second window. *WordPerfect* Library saved me the trouble of having to get completely out of *WordPerfect* (CTRL-F1 shells you out to the library). I wanted something better, of course. Doesn't everyone?

Then came version 3-point-something. It had a hot-key for *WordPerfect*. The hot-key was really no more than a macro or script that saved the document, popped you out of *WordPerfect* and then ran *Right Writer* on your document. It worked well, but the program was still slow. A large document—say one as large as this little column—would sometimes take as long as 10 minutes to process on the XT. When something takes that long, you avoid using it if you are in a hurry. That's usually when you most need a grammar checker's help.

Now comes version 4.0. The price is right, and the folks at Que (who now distribute the program) pack a copy of Strunk and White's *The Elements of Style* in the box.

I like *Elements of Style*. It's a simple, clear concise book. It isn't perfect. What is? When I had journalism interns under my wing, I insisted they always carry a copy with them. I also insisted they READ it!

Well, when you aren't writing fast and furiously and pounding the basics of decent grammar into young heads every day, you get sloppy. Your prose gets sluggish. Your verbs get passive, and you sometimes split infinitives. You need a hard-nosed editor looking over your shoulder. *Right Writer*.

True, you can turn off any or all the rules. You can even make *Right Writer* look the other way at "to boldly go where no man has gone before." But, if you turn on some of the most obvious rules and make it flag the really bad stuff, you finish with passable writing. You might even learn something as you move through the marked-up copy.

One change in the new version is the way *Right Writer* handles files. When you run the program on a document, *Right Writer* creates a file with the extension OUT—the copy. After *Right Writer* rips your copy apart, it puts you into *WordPerfect* with the marked-up copy in Document 1. You edit the marked-up copy, taking or leaving the comments as you please. Then you press ALT-S (strip) to have *Right Writer* strip out the markup and give you back a clean file. When the disks finish grinding away, *Right Writer* created a new copy of your document with the file extension CLN. This is now your corrected document.

Don't worry. Your original file (in my case, the one with the DOC extension) still exists on the disk, undamaged, in case you want to go back and start over.

The value of the program is in what you learn as you work with it and in improving your ability to communicate.

There is one small problem. *Right Writer* leaves a series of spaces and carriage returns at the end of your file. I noticed this right off because my *WordPerfect* displays happy-faces for hard returns. Those remained behind when *Right Writer* stripped out the "readability," "strength," "descriptive," and other information at the end of the marked-up document.

Cleaning up the extra hard returns is not difficult if you catch them each time. After a few trips through *Right Writer*, though, the end of the document could have several pages of hard returns.

During the first draft of this installment of the column, I ran *Right Writer* through the text up to the last paragraph. *Right Writer* picked up some passive verbs (nearly everyone uses a few). It flagged "junkie" in the first sentence as a colloquial expression. Of course, it picked out the split infinitive in the "to boldly go" phrase. I had the program set for a general public audience and a technical report or article. Those settings work well for my kind of writing.

Remember, the writing style settings in the Change Settings menu are general. When you select an audience and a type of document, you establish a general style. You will want to double-check the rules and grammar settings and make sure they meet your requirements.

For example, I took a short piece of high school English writing and ran it through *Right Writer*. I set the audience to high school and the type of document to proposal (this was a scholarship letter). I found that, with this audience selected, *Right Writer* turned off the passive verb rule and a few others. I turned on all the rules and processed the text. It picked out the passive verbs, the incomplete sentences, and subject/verb disagreements like an old warhorse news editor.

Right Writer is not the last act in editing my work. I go over it several more times to massage a phrase here, check an odd sentence there, and run the spell checker one more time. Then my wife looks it over—she has the final say.

Of course, the purpose of a grammar checker like *Right Writer* is to help you with your writing. You can take or leave the suggestions and recommendations it makes. You can turn rules on and off and set the program for the style that most suits your writing. The value of the program is in what you learn as you work with it and in improving your ability to communicate.

Communicating, after all, is what we're supposed to be doing with grammar.

* * * * * * * * * * *

Guy L. Pace lives in Pullman, WA, with his wife and family. He is an Information Services Evaluator (in his words, "a fancy name for trouble-shooter.") at Washington State University's Cooperative Extension. A large part of Guy's job involves helping people learn to use computer hardware and software more efficiently. His degree is in public relations, and he was a journalist off and on for 15 years. If you have a question or idea that you want addressed in a column, please contact Guy via US Postal Service at NW 375 Dillon, Pullman, WA 99163 or via BITNET at PACE@WSUVM1

RANDOM ACCESS

Bryan Pfaffenberger

Knowledge, Competence, and Documentation

In reflecting on my experience writing trade books on computer software for publishers such as Que, Osborne-McGraw Hill, Sybex, and others, I've developed a working theory of just what we authors are trying to convey in these books. Guided much more by market pressure and some very savvy editors than by technical writing theory, computer book authors have increasingly come to differentiate the forms of knowledge and competence that we're trying to teach in our books, and increasingly we are using distinctive formats (such as "Tips" or Quick Starts") to identify and segregate sections of the text that convey such knowledge. Here's a working overview of the way I'm currently mapping out the forms of knowledge and competence that I'm trying to convey in my current books (such as *Using Microsoft Word for the Mac, Special Edition*, which will be published this fall by Que).

Conceptual Knowledge

A minimal conceptual foundation for computer usage includes the ability to define and recognize computer hardware and software entities (disks, drives, system units, files, menus, etc.) that must be successfully manipulated to accomplish the tutorials. Such entities should be defined and their functions shown by analogy and graphics.

Tutorial Knowledge

This knowledge is acquired through step-by-step tutorials. Although such tutorials build confidence, this type of knowledge is inherently limited because it is learned by rote. An adequate tutorial guides the reader directly toward the most frequently used (and useful) aspects of the program. Every instruction should involve just one operation and should be set off clearly from the rest of the text.

Heuristic Knowledge

Usually ignored or incompletely treated in computer self-help books, this vital knowledge guides experienced computer users toward a solution when something doesn't work. Heuristic knowledge is often tacit and is acquired by experience; it can, however, be expressed in the form of IF-THEN rules: "If the program runs funny, then exit to DOS, reboot the system, and re-load the program."

The challenge facing authors of self-help books is to engage in research that identifies the areas where naive users will stumble. In the tutorial sections, every possible contingency should be predicted, and heuristic strategies should be clearly marked as "Tips" and "Cautions."

Procedural Knowledge

Expressed in the form of a quick reference guide, procedural knowledge is a generalized form of tutorial knowledge—if the tutorials were properly conceived. It concisely restates the tutorial's steps, shorn of the specifics that tied the tutorial to a specific setting. Procedural mastery requires and assumes conceptual, tutorial, and heuristic knowledge. Thus, procedural knowledge should be expressed as concisely as possible

Customized Knowledge

The user learns how to manipulate the program's preferences, macros, style sheets, or other user-definable options well enough to customize the program effectively. The sections on customization should draw significantly on what has already been learned; the modifications should address significant shortcomings of the program, streamline difficult procedures, and allow for the expression of personal taste.

Application Knowledge

The supreme form of user mastery, application knowledge is achieved when the user can harness all other forms of computer knowledge at will to intuit, almost instantly, an approach to a specific problem. This knowledge requires mastery of concepts, experience with tutorials, knowledge of heuristics, memorization of procedures, and customization ability. Application mastery requires and assumes competence in all the other areas of computer knowledge. These sections should be written at the greatest level of abstraction. Here, breadth matters.

As these brief definitions illustrate, the six forms of knowledge are coupled in an outward spiral of user competence and empowerment: conceptual, tutorial, and heuristic knowledge serve as the foundation for procedural knowledge, while conceptual, tutorial, heuristic, and procedural knowledge serve as the foundation for application knowledge. It therefore follows that teaching this competence involves recursion along the spiral as each new level of competence is gained.

In what follows I sketch what I think this survey of computer knowledge suggests for the ideal way to organize computer "how-to" books.

I. Introductory tutorials

Conceptual, tutorial, and heuristic knowledge focus on the entities (menus, keys) and procedures (starting and quitting, saving and abandoning) that all must learn; such procedures should be organized along the lines of a typical work session.

II. Mastery of everyday skills

More conceptual, tutorial, and heuristic knowledge, spiraling outward from the introductory tutorials and expanding on all the basic procedures should follow and be procedurally organized (e.g., separate chapters on formatting, editing, etc.).

III. Intermediate/advanced procedures

The intermediate step should include still more conceptual, tutorial, and heuristic procedures which address intermediate and advanced procedures of interest to some, but not to all, readers. These exercises should focus on specific applications such as collaborative writing.

IV. User customization

Chapters on user customization should begin with conceptual, tutorial, and heuristic knowledge; separate chapters should address specific program shortcomings, how to overcome them, and how to streamline difficult or time-consuming procedures.

V. Applications

This section should begin with a tutorial (incorporating conceptual/tutorial/heuristic knowledge) that introduces a useful application. Additional applications should be discussed in more abstract terms.

VI. Quick reference

In very concise form, this section provides as complete a reference guide as possible, including procedures, command reference, and glossary

A computer manual or book written according to this organizational plan would succeed, I think, in its goal of "emplotting" the reader, as David Goodwin would put it [*Journal of Technical Writing and Communication* 21 (1991)]: attempting to "move the reader, step by step, through a sequence of actions to a desired outcome" (99).

To do so, this way of organizing a book tacitly embodies what can only be seen as a heroic narrative: the Hero (the reader) is naive at the outset, but through the mediation of the Helper (the author) and the computer's Power manages to overcome Obstacles (bugs, traps, unclear procedures, convoluted commands, confusing documentation).

In the end the reader attains the kind of competence and open-ended application mastery that frees the reader from the Helper. Yet well should we authors remember, as Goodwin admonishes us, that if we fail in our role as Helper, readers may liken us to the Obstacles.

* * * * * * * * * *

Bryan Pfaffenberger, an Associate Professor of Humanities at the School of Engineering and Applied Science, University of Virginia, teaches technical writing and the history and sociology of technology. The author of Que's *Computer User's Dictionary*, *PC Tools Version 7 Made Easy* (Osborne/McGraw-Hill), and *Microcomputer Concepts and Applications* (Harper/Collins), he is currently working on a social history of personal computing.

THE CLIPBOARD

Ian Richmond

Your Memory Not What It Used to Be?

My wife is always complaining of my absent-mindedness. Well, what can she expect, I ask her. All my appointments, deadlines and other such stuff are stored in my desktop computer, which beeps and flashes to jog my memory. Obviously, I tell her, the solution to my problem would be a nifty pocket computer to beep and flash at me when I'm not in the office. She never buys this argument, coldly ascribing it to a severe techno-addiction and cruelly observing that a 59-cent notebook and a nickel pencil will serve just as well. Oh well, maybe I could find a notebook that beeps, or a pencil with digital alarms.

It is some consolation to me to know that middle-aged profs aren't alone in finding that their memory just isn't what it used to be. Surprisingly enough, it happens to computers, too, despite their reputation for nearly infallible recall. Further, the cure for their absent-mindedness isn't always found in additional hardware.

My first computer was a ZX-81 with 2K of RAM for which I bought a whopping 16K memory expansion board and later upgraded to an incredible 64K. Believe it or not, the programs I ran on that computer had so much memory space that one could imagine them rattling around in it creating great cavernous echoes.

My first PC had 256K of RAM, a seemingly infinite capacity, in which *WordPerfect* 4.0, *Lotus 1-2-3*, and sundry other applications co-existed quite happily with the original memory-resident utility, *Sidekick*, and a 64K RAM disk. Alas, within a year the computer equivalent of absent-mindedness reared its ugly head: As I upgraded to newer and bigger versions of my software, I began receiving the infamous "Not enough memory" message.

Today, few computers come with less than 640K of RAM, but even that may not be quite enough. If you like to have a couple of memory-resident utilities on line along with your wordprocessor, you are almost sure to bump against your computer's memory limits.

The obvious solution to this problem, often known as "RAM cram" appears to be adding more memory to your computer. Certainly, adding a megabyte or two of extra memory is a great idea. Be aware, however, that this will only partially solve your problem. Even throwing out your old clunker to replace it with the latest 386 or 486 screamer with multimegs of RAM straining under the hood often won't solve RAM cram.

RAM cram, in fact, is less a problem of supply than of distribution. You can have ten or fifteen megabytes of RAM, but if your applications can't get at it, you are no better off than if you had only 640K, the most that MS-DOS directly accesses.

Why then is adding memory a good idea? Because a good deal of clever software has been written to take advantage of extra memory, even though it cannot be used directly by MS-DOS.

One type of program that can help enormously with RAM cram is a task switcher, such as *Desqview*, *Carousel*, or WordPerfect's *Office* 3.0. These programs allow you to load several applications into memory and to switch back and forth among them without having to exit and relaunch them each time. They accomplish this magic by switching programs in and out of memory as they are needed.

If you have memory beyond 640K (expanded or extended), most task switchers will move programs between the conventional RAM (the first 640K) and the extended or expanded memory. This is almost instantaneous and can eliminate the need for many memory resident programs. In effect, all your programs can run as though they are memory resident.

If you have no extra memory and not enough conventional RAM, the task switcher will normally use your hard disk as "virtual memory" by saving an image of the program in memory to the hard disk. Although this takes longer than switching entirely in memory, it is still faster than exiting one program and loading another. This is especially true since task switching preserves whatever you were doing in each program: when you return to a program you previously left, you pick up exactly where you left off.

If task switching is more than you need, perhaps a memory manager would help solve your prob-lem. These are utility programs, such as *QEMM 386*, *386Max*, and *QRAM* that redistribute your memory so that more of it is available to your programs and to MS-DOS.

Mostly, this type of program is designed to exploit the extended memory available on 386 and 486 computers. They can load DOS device drivers into the "high" memory regions (between 640K and one megabyte) and thrust memory-resident programs into the nether realms above one megabyte while ensuring that they still work as advertised. Of course, this management frees up acres of memory in the lower 640K that DOS uses directly.

Although this type of utility is usually written for computers with the newer 386 and 486 chips, there is one that can work considerable magic on more humble PC's and AT-class machines. QuarterDeck's *QRAM* (pronounced: "cram") can be a true lifesaver for those of us whose budgets don't run to the latest in hardware.

If you have an 8088 processor with an expanded memory board installed or an 80286 processor with expanded memory or with a Chips & Technologies NEAT chipset, *QRAM* can do many things that would otherwise require at least an 80386. For example, it can create a high RAM area into which it can load DOS device drivers, DOS buffers, and memory-resident programs. It can even load most of itself into high RAM so that it doesn't encroach on your precious 640K of DOS-accessible memory.

On my 80286 with a NEAT chipset and two megabytes of expanded memory, *QRAM* creates 128K of high memory into which it loads my mouse driver, my DOS buffers, my ANSI.SYS driver, my foreign keyboard definition, and the 32K of *Memory Mate* that doesn't load into expanded memory. All this gives me about 75K more conventional RAM than I would otherwise have. This may not sound like much, but it is about 20 extra pages of a *WordPerfect* document that can be kept in memory instead of spilling over onto the infinitely slower hard disk.

For real power, you can combine a memory manager with a task switcher for even better memory control. *Desqview*, for example, can be used along with *QRAM* or *QEMM 386* and will load a good part of itself into high memory. Even using a memory manager with a program such as the WordPerfect *Office*, which takes up 50K of RAM, gives you extra memory to speed up task switching and applications.

Of course, the grandaddy of all these programs, albeit a rather young grandpa, is *Windows* 3.0. *Windows* combines memory management, task switching, and other goodies all into one program and makes your memory get up and boogie. The only drawback is that you must have the 80386 or 486 platform with the 4 or 5 megabytes of mem-ory that *Windows* needs to really strut its stuff.

If that level of hardware is not in your budget or your old AT clone just isn't ready for the junk pile, you really should look at the DOS task switching and memory management programs. They'll give your computer's memory a new lease on life, and—who knows?—the ability to call up data at the touch of a key might even help your own absent-mindedness.

* * * * * * * * * *

Ian M. Richmond is a member of the Department of French at the University of Western Ontario, London, Ontario, Canada N6A 3K7. He may be reached at 519-661-2163, Ext 5703, as well as BITNET: IMR@UWOVAX.BITNET

TEXTechography

Arthur A. Molitierno

Abbreviations:

Countries	
AU	Austria
CZ	Czechoslovakia
GR	Germany
JP	Japan
NL	Netherlannds
SP	Spain
SW	Switzerland
UK	United Kingdom

Terms	
DTP	Desktop Publishing
np	no page
n.	number

Months	
JA	January
FE	February
MR	March
AP	April
MY	May
JE	June
JL	July
AU	August
SE	September
OC	October
NO	November
DE	December

[*Editor's Note:* TEXT Technology *welcomes bibliographic items of interest from the entire community of text producers. Send items to Arthur A. Molitierno.*

Each issue will feature a bibliographic form employed by a specific orgainzation, association, or journal. This issue features the format for the Journal of Ecology.

Although there will be some notable differences from the original style (italicizing the names of software or journals, for instance), "TEXTechography" will present the featured style as close as possible to the original's appearance.]

Anonymous. (1990). Computer teachers respond to Halio. *Computers and Composition*, **7** (3), 73-79. [comments on the work of M. P. Halio, who reported in the January 1990 issue of *Academic Computing* that first year writing students using Macs produced poorer quality work than their counterparts using DOS-based computers]

———. (1990). Electronic Publishing on a PC. *Infografik [GR]*, **5**, 16-20. [in German, report on Page Express, software designed to fill the gap between DTP and pre-press systems, includes modules for input, processing, layout, outputting, color correction, among others]

———. (1990). Electronic publishing on a PC. *Infografik [GR]*, **5**, 16-20. [in German, description of Page Express, program activate Postscript language for DTP and printers; resolution to 800x400 dpi]

———. (1990). Greater value—smaller price. *Electronikschau [AU]*, **9**, 56-57. [in German, describes features of Data Beckers Golden Series of PC shareware: PC-TEXT 2.0, Lighting Press (DTP), LHarc 1.13, Formula 1.2, Daisy 5.0, Top-20, Font Editor 1.2, NYET II]

———. (1990). North Atlantic Publishing Systems: linking XyWrite and Xpress. *Seybold Report on Publishing Systems*, **20** (2), 3-9. [discusses copy routing in multi-user environments, with attention to XyWrite and Xpress; batch processing for layout; article builds upon Quark Xtensions from North Atlantic Publishing Systems: CopyFlow, CopyFlow Reports, and CopyBridge]

———. (1990). Quicker success using WordPerfect 5.0/5.1. *Chip [GR]*, **10**, 356-58, 360, 362, 364, 366,368, 370. [in German, tutorial and explanation of how WordPerfect's graphics features may be used; includes stp-by-step illustrations of procedures]

———. (1990). The graphical user interface. *Wharton Report [UK]*, **146**, 1-8. [discussion of GUI (graphical user interface), concentrating on Windows 3.0, applications, and related experiences of users in England and the USA]

———. (1990). The Microsoft Windows 3.0 graphics system. *Eletronik Praxis [GR]*, **25** (16), 98-102. [in German, review of Windows and its applicability to PCs]

———. (1990). Universal design tool. *Infografik [GR]*, **5**, 4-5. [in German, report of Pictures 2.0, a universal 2D/3D CAD program for all MS-DOS PCs]

Baird, B., Blevins, D. & Zahler, N. (1990). The artificially intelligent computer performer: the second generation. *Interface [NL]*, **19** (2-3), 197-204. [discusses AICP (artificially intelligent computer performer), a program designed to equip the computer with more human-like performance characteristics]

Barker, P.. (1990). Intelligent electronic books. *Journal of Artificial Intelligence in Education*, **2** (1), 7-13. [discussion of interactive and intelligent electronic books and the use of CD-ROM and multi-media models for electronic books]

Bolton, W. (1990). Lines, Boxes, Etc. *Computers and the Humanities [NL]*, **24** (4), 308-10. [software review of Lines. Boxes, Etc. (LBE), a program for enhancing the line and box drawing of WordPerfect; Etc. supports the 1-10 character sets of WP Compose function; package includes added foreign characters and special figures for WP]

———. (1990). The bard in bits: electronic editions of Shakespeare and programs to analyze them. *Computers and the Humanities [NL]*, **24** (4), 275-87. [discussion and comparison of 3 current electronic editions of Shakespeare; includes discussion of 3 commercial programs for text analysis using microcomputers]

Bonitz, M. (1990). Science Citation Index on CD-ROM: the largest system in the world. *International Forum on Information and Documentation [USSR]*, **15** (3), 9-12. [an explanation of the manner by which bibliographically compatible papers are retrieved on Science Citation Index on CD-ROM]

Cabezas, R. (1990). Computer graphics: the reality of the virtual. *Novatica [SP]*, **16** (86), 6-18, 33-36. [in Spanish, comprehensive review of computer graphics, with discussions of the graphics environment, modelling, rendering, and animation; briefly indicates future developments]

Cabiro, I. (1990). Introduction to hypertext as a general information tool. *Revista Espanola de Decumentacion Cientifica [SP]*, **13** (2), 685-709. [in Spanish, general historical review of the concept and applications of hypertext, beginning with the developers, Englebart (1963) and Nelson (1967), and including descriptions of the most important developments and features of systems and research concepts]

Carlson, P. (1990). Cognitive tools and computer-aided writing. *AI Expert*, **5** (10), 48-50, 52-55. [discussion of CAW (computer-aided writing) and the development of prototypical tools for assessing prose—including stylistic features which asses the tone and voice of a passage]

Catarci, T. & Batini, C. (1989). *Proceedings of the 8th international conference on entity-relationship, Toronto, Ontario, Canada*, pp. 157-58, 18-20 OC [examination of QL (query language), fourth generation languages, and how users may use QL to eliminate drawbacks of some information systems]

Cobb, S. (1990). Power start. *What Micro [UK]*, OC, 46-52. [review of Wordstar 6.0, Displaywrite 5.0, Multimate 4.0, Word for DOS, Sprint, and Wordperfect 5.1]

Crawford, W. (1990). Beyond Courier: good writing deserves good typography. *Library Hi Tech*, **18** (3), 93-100. [review of Swfte Glyphix, program for generating typefaces of any size, working under Microsoft Word or WordPerfect]

Deegan, M. (1990). Categorical grammar, generative phonology, and the morphology of Old English. *Literary and Linguistic Computing [UK]*, **5** (1), 70-76. [description of how a learning program may be generated by using Prolog programming language and categorical grammar formalism to generate a rule for strong noun inflections of Old English; use of generative phonology helps eliminate some rules]

DiPardo, A. & DiPardo, M. (1990). Towards the metapersonal essay: exploring the potential of hypertext in the composition class. *Computers and Composition*, **7** (3), 7-22. [explores the use of HYPERCARD and explains the function of stackware for the authors' design of a hypertext program for composition class]

Dolak, F. (1990). The best keeps getting better: WordPerfect 5.1 rates a 10. *Library Software Review*, **9** (4), 234-49. [review of WordPerfect 5.1 which accesses expanded memory; major features of program considered]

Elmiger, T. (1990). Much is learnt from a bad case of "desk top publishing." *Sysdata [SW]*, **21** (10), 39-41. [in German, an account of unfortunate individual who without adequate preparation attempted to produce publicity material with A5 but was unprepared for the realities of pictures, layout, format, and color]

Ester, M. (1990). Image quality and viewer perception. *Leonardo [UK]*. [an assessment of image quality through a study of the responses of a group of art historians' reactions to digitalized test images]

Gold, S. (1990). Flexible access. *Micro Decision [UK]*, **112**, 105. [review of XTree Pro Gold, menu driven utilities program for DOS computers]

Graf, J. (1990). Good-value quintet. *Chip [GR]*, **11**, 204-06, 208, 210-11. [in German, reviews 5 DTPs (Desktop Publishing Program): Byline, Finesse 3.1, PFS First Publisher, Fontasy 3, Timeworks; author compares about 60 features in a convenient table and supplies some sample screens]

Hampshire, N. (1990). Set menus. *Micro Decision*, **112**, 109. [review of Clarity, wordprocessing package which features structured forms for reports, manuals, and books; uses document style templates which may be used to create DTP quality work]

Haring, G., Penz, F. & Weichselberger, G. (1990). A comparative evaluation of graphical user-interfaces. *SIGCHI Bulletin*, **22** (1), 12-15. [review of Microsoft's Windows and Digital's GEM based on questionnaire submitted to users]

Harrison, M. A. & Munson, E. V. (1989). On integrated bibliography processing. *Electronic Publishing: Origination, Dissemination and Design [UK]*, **2** (4), 193-209. [discusses the function of integrated editing environment for processing of bibliographical data, problems with sharing databases among multiple authors, need for new features; discusses these matters as they relate to the GNU Emacs BIBTEX-Mode and TEX-mode integrated editing environment]

Hayne, S. & Purdin, T. (1990). A distributed group tool for issue analysis. *Proceedings of the 1990 symposium on applied computing, Fayetteville, Arizona*, pp. 325-27, 5-6 AP [description of how GIA (Graphic Issue Analysis) system is used to promote sharing of ideas in real time through local networks; system allows a group to present an idea graphically in order to gather further ideas or resolve conflicts and reach consensus]

Hetherington, A. (1990). Importing Hyper-Card graphics to Works. *Microsoft Works in Education*, **2** (2), 3-4, 6-7. [tutorial for the MAC explaining how to assimilate clip-art files into Works by using Hyper-Card as a graphics source; two approaches producing the same result include the use of the Scrapbook and the Clipboard]

Horvath, F. (1990). World exploits computers to overcome a language barrier. *Knizice a Vedecke Informacie [CZ]*, **22** (6), 261-67. [in Slovak, review of how machine translation is being employed throughout the world—Japan, eastern Europe, Canada, USA, China; covers such topics as lexical, syntactical, and morphological analysis]

Jackson, P. (1990). A visual approach. *What Micro [UK]*, OC, 56-60. [reviews 7 graphics-based wordprocessors; 3 Windows-based: Word for Windows,1.0, Ami Professional 1.2, Guide 3.0; Describe 1.1 for OS/2 Presentation Manager; 3 Mac-based: Word 4.0, Nisus 2.11, and Macwrite II 1.1]

Jenkins, C. (1990). Stop your business drowning . *Which Computer? [UK]*, **13** (12), 46-47, 51, 54, 56, 58, 63. [70% of paper documents filed are not read again; indicates ways for effectively managing paper in an office environment]

Joram, E., Woodruff, E., Lindsay, P. & Bryson, M. (1990). Students' editing skills and attitudes toward word processing. *Computers and Composition*, **7** (3), 55-72. [examination of student attitudes concerning text editing and preferences concerning the method of revision—wordprocessor or paper and pencil; indicates students' preferences differ depending on the task of either composing or revising; graduate students and professional writers also interviewed]

Kempen, G. (1990). Language technology and the future of text automation. *Informatie [NL]*, **32** (9), 724-27. [in Dutch, discussion of problems when transporting text from one system to another; includes review of text coding]

Kluepfel, G. A. (1990). Samma Word IV Plus 2.02. *Computers and the Humanities [NL]*, **24** (4), 303-07. [software review; package includes wordprocessor, spreadsheet, and textbase; includes discussion of ASCII merging]

Lancaster, D. (1990). High-performance PostScript. *BYTE*, **15** (8), 297-300. [discussion of how PostScript allows a printer to achieve high-performance and typesetting-machine quality printing]

Lanker, E. (1990). The Middle Ages in the Computer Age. *Sysdata [SW]*, **21** (12), 63-65. [in German, description of AGENDA, a PC program which can aid in historical research; program analyzes category hierarchies from notes, key words, and source indicators to form explicit and implicit cross references for locating important historical data]

Lansky, T. (1990). Not only for specialists. *Chip [GR]*, **12**, 380-88. [report on 5 specialized wordprocessing programs: Wi-Tex 3.0—text layout; Lex-WP—multi-user; Chi-writer—for scientists and linguists; Layertext—creating & handling forms; Formula Manager Plus—creating & handling forms]

Laurma, T. (1990). Desktop publishing on the mainframe: integrating APL2 and Ventura Publisher. *APL Quote Quad*, **20** (4), 233-38. [a paper describing how APL applications may be provided with high quality output through integrating the mainframe APL2 environment and a PC desktop publishing package]

Leslie, M. (1990). The Hartlib Papers: text retrieval with large datasets. *Literary and Linguistic Computing [UK]*, **5** (1), 58-69. [explanation of the Hartlib Papers—one of the greatest archives for seventeenth century intellectual history—and how this archival collection is being processed for text retrieval]

Lombardi, G. (1990). Integrated systems for testing and grading. *Computer in Life Science Education*, **7** (9), 65-67. [discussion of programs which allow for interactive test production, administration, and grading]

Loveria, G. & Kinstler, D. (1990). Multimedia: DVI arrives. *BYTE*, **15** (11), 105-08. [discussion of DVI (digital video interactive) technology and how this technology is used in commercial and educational applications; recent developments include use of DVI for tough-screen kiosks for museum visitors]

Marmion, D. (1990). Windows 3.0: confessions of a convert. *Computers in Libraries*, **10** (9), 21-25. [discussion of the environment of Windows 3.0, with material on hardware requirements, pointing devices, operating modes, and various procedures]

———. (1990). Windows 3.0: confessions of a convert. *Computers in Libraries*, **10** (10), 18-21. [discusses advantages of CUA (common user access, which means different applications will have a common "look" because of the same styled menus bars, for example); considers installation and .INI file features]

Martin, J. (1990). *Hyperdocuments and how to create them*. Prentice Hall, Englewood Cliffs, N.J. [text explains the major functions of hyperdocuments, how to create them, and the various software packages to generate such documents; features step-by-step guidelines and procedures]

Moad, J. (1990). Windows 3.0: the corporate standard. *Datamation*, **36** (9), 30-32, 36. [review which emphasizes Windows' memory management and user interface]

Mones-Hattal, B., O'Connell, K. & Sokolove, D. (1990). Guidelines for curricula in computer graphics in the visual arts. *Computer Graphics*, **24** (3), 78-113. [preliminary report drafted by the ACM SIGGRAPH education committee to meet the needs of such accrediting agencies as the National Association of Schools of Art and Design]

Nakabayashi, A. (1990). Computerized typesetting of the newspapers. *Journal of the Institute of Electronics, Information and Communication Engineers [JP]*, **73** (5), 526-29. [in Japanese, discusses CTS (computerized typesetting systems); the use of JPS (Japanese Publishing System) developed by IBM and Press/F (Progressive Editing Support System by FACOM); effect of CTS on newspaper companies in Japan]

O'Brien, A. M. (1990). Graphics link. *What Micro [UK]*, OC, 97. [reviews TGL+, graphics file conversion package which allows user to format files from one format into another]

Peze, R. (1990). What does a graphics screen do for text processing? *Sysdata [SW]*, **21** (12), 61-62. [in German, considers conventional programs for text processing superior to such graphics managers as Windows]

Philips, T., Hendy, P. & Jackson, P. (1990). Visual impact reviews. *Micro Decision [UK]*, **112**, 49, 51, 53, 55, 57, 61, 63, 65, 67, 69. [review of 9 popular presentation graphics packages for business]

Prakel, D. (1990). Mac DTP. *What Micro [UK]*, NO, 60-62, 64. [reviews 4 second generation DTP programs for the Apple Macintosh: Letraset Design Studio 1.01; Multi-Ad Creator 2.1; Aldus Pagemaker 4.0; Quark XPress 3.0]

Rehr, D. (1990). Desktop publishing: how can it be kept simple? *Office*, **112** (5), 61-62. [consideration of "font frenzy"—the unjustified need for innumerable fonts; indicates 3 basic fonts quite adequate for most purposes: Courier, Times Roman, Helvetica; some information of line spacing and line length included]

Reviews. (1990-91). Adobe Type Manager for Windows 1.0 Reviews [Font Generation]. [1) *Infoworld* (FE 4, 1991) G. Gruman, 68, 72, 76; 2) *PC Magazine* (DE 11, 1990) 43; 3) *PC Publishing* (JA 1991) D. Dean, 15-16; 4) *PC World* (JA 1991) G. Campbell, 124-26]

———. (1990-91). Adobe Illustrator 3.0 Reviews [Drawing]. [1) *BYTE* (FE 1991) T. Thompson, 178, 180; 2) *Infoworld* (DE 1990) b. Barbante, 64; 3) *Infoworld* (JA 21, 1991) D. and D. Green, 67-69; 4) *Macuser* (FE 1991) 50-51; 5) *Publish!* (NO 1990) J. Schmal, 31-32]

———. (1990-91). Ashlar Vellum 1.0 [Macintosh] 3.0 [IBM/Compatibles] Reviews [CAD]. [1) *Infoworld* (AP 30, 1990) K. Milburn, 73, 76-77; 2) Macuser (MY 1990) 98, 100, 104; *3) PC World* (FE 1991) C.J. Delucchi, 156]

———. (1990-91). Corel Draw! 2.0 Reviews [Drawing]. [1) *Computers in Education* (DE 1990/JA 1991) S. Rimmer, 30, 32-33; 2) *PC Magazine* (JA 29, 1991) 33-34; 3) *Physicians & Computers* (DE 1990) N. Sondak and V. Sondak, 17-18; 4) *Step-By-Step Electronic Design* (FE 1991) S. and E. Ihrig, 7-9]

———. (1990). Digi-View Gold 4.0 Reviews [Video Digitizer]. [1) *Amiga Resource* (AP 1990) S. Jacobs, 48, 50; 2) *Computer Shopper* (JL 1990) D. Johnson, 564, 570]

———. (1990-91). DisplayWrite 5.0 Reviews [Wordprocessing]. [1) *Infoworld* (JA 1991) 54-67; 2) *PC Magazine* 132, 134, 137; 3) *PC World* (AU 1990) S. Lusty, 96]

———. (1990-91). Express Publisher 2.0 Reviews [DTP]. [1) *New York Times* (NO 6, 1990) L.R. Shannon; 2) *PC/Computing* (NO 1990) 80; 3) *PC Publishing* (JA 1991) R. Mueller, 37-38; 4) *PC World* (JA 1991) M. Hendricks, 98]

———. (1990-91). Grammatik Mac 2.0 Reviews [Grammar, Style Checker]. [1) *Computer Shopper* (JA 1991) J. Quaraishi, 590, 595; 2) *Infoworld* (SE 17, 1990) Y. Lee, 68; 3) *Personal Publishing* (FE 1991) P. Bishop, 62, 63]

———. (1990). Microsoft Word for the Macintosh 4.0 Reviews [Wordprocessing]. [1) *Consumer Guide: Computing Buying Guide* (OC 1990) 300-302; 2) *Infoworld* (AU 13, 1990) G. Gruman, J. Lombardi, E. Azinger, and J. Eckert, 59-69]

———. (1990-91). PageMaker 4.0 Reviews [DTP]. [1) *Infoworld* (NO 5, 1990) B. Assadi and G. Gruman, 72, 77, 81, 85, 89; 2) *Macworld* (FE 1991) S. Roth, 142-151; 3) *Publish!* (OC 1990) D. Burns, 58-63]

———. (1990). Publisher's Powerpak 1.2 Reviews [Font Generation]. [1) *Infoworld* (MR 1990) S. Timacheff, 77; 2) *PCM* (AU 1990) L. Crawley, 83-84]

———. (1990-91). QuarkXPress 3.0 Reviews [DTP]. [1) *Infoworld* (NO 5, 1990) B. Assadi and G. Gruman, 72, 77, 81, 85, 89; 2) *Macworld* (FE 1991) S. Roth, 142-51; 3) *Publish!* (OC 1990) D. Burns, 58-63; 4) *Step-By-Step Electronic Design* (FE 1991) K. Tinkel, 11, 15]

———. (1990-91). WordStar 6.0 Reviews [Wordprocessing]. [1) *Computer Shopper* (JA 1991) S. Vaughan-Nichols, 374, 378, 381; 2) *Infoworld* (JA 7, 1991) 54-67; 3) *PC Magazine* (DE 11, 1990) 188, 191-92]

Rosenberger, R. (1990). Software: conductor of orchestrated solutions. *IMC Journal*, **26** (5), 12-14. [discussion of software that governs document management; current trend is to examine application and system software to solve document management problems]

Schrodl, K. (1990). The "RightPlus" spell checker. *MC Magazin fuer Computer praxis [GR]*, **11**, 168-71. [in German, review of software for correcting typing mistakes; package includes German glossary of 25,000 words; extension of glossary to 230,000 words is available]

Simonton, D. (1990). Lexical choices and aesthetic success : a computer content analysis of 154 Shakespeare[an] sonnets. *Computers and the Humanities [NL]*, **24** (4), 251-64. [a combination of aesthetic paradigm and computer content analysis used to determine patterns in the sonnets; shows how Shakespeare modified his vocabulary in the ending couplets in his best—as defined by the computer—sonnets]

Smart, H. (1990). The bargain basement. *What Micro [UK]*, OC, 66-70. [reviews of six inexpensive wordprocessors aimed at novice users: Topcopy Plus, Volkswriter 2, Letterperfect, Locoscript PC, PC Type+, Eight in One]

Smith, B. (1990). Microsoft Word brings PC-style word processing to Unix. *BYTE*, **15** (13), 209-10. [highlights features included in and excluded from Word 5.0]

Stopford, C. (1990). Desk top type: tradition and technology. *Communicator [UK]*, **2** (6-7), 3-7. [discussion of the problem of typefaces and the use of desktop publishing techniques; indicates typographic literacy is required for use of current programs]

Tadashi, S. (1990). Applications of text processing using natural processing in printer. *Joho Kanri [JP]*, **33** (5), 425-33. [in Japanese, description of INDEXER, a natural language processor which automatically indexes and sorting kana characters to kanji characters; system may be employed for address books, name lists, books, and CD-ROM indexing]

Tai, T. (1990). Quicker success using Wordperfect 5.0/5.1. *Chip [GR]*, **12**, 356-64. [in German, describes making of tables and writing of mathematical formulae; includes procedures for making of tables within blocks of text, producing lines and shading, performing calculations within tables, and how to manage symbols]

———. (1990). Quicker success with WordPerfect 5.0/5.1. *Chip [GR]*, **11**, 384, 386, 390, 392, 394. [in German, step by step description of such features as: footnotes and endnotes, lists of contents and key-words, word splitting, indexing]

Takahashi, Z. & Yoshida, T. (1990). Development and practice of computer manual refinement and proofreading system MAPLE. *Transactions of the Information Processing Society of Japan [JP]*, **31** (7), 1051-62. [in Japanese, extensive review of the features of MAPLE (Manual Publishing Adviser), including quantifying expressions and sentences with complex modification; additional discussion of both ATLAS and ESHELL/X]

Thompson, D. (1990). Electronic bulletin boards: a timeless place for collaborative writing projects. *Computers and Composition*, **7** (3), 43-53. [explains one solution to the problems of time-consuming collaborative computer writing: use of electronic bulletin boards to allow students to share text and in the process communicate with each other through messages indicating current research]

Vieten, M. (1990). A new friend for people who write a lot. *Chip [GR]*, **12**, 184-86. [in German, test of wordprocessing software: Beckertext II for the Commodore Amiga PC; test includes such features as: preview, hyphenating, searching, and indexing]

Articles & Reviews Welcome

TEXT Technology welcomes submissions that pertain to wordprocessing, text-analysis, and research applications in professional writing, either corporate or academic. Also, hardware and software reviews are encouraged, but please contact Jim Schwartz before submitting them—either call Jim at 419-586-2365 or send him a note at the following BITNET address

JSCHWARTZ@WSU

Manuscripts may be submitted on MS-DOS 5¼" or 3½" floppy disks, through BITNET (in ASCII), or in hardcopy format.

New Subscriber Information

To academic and corporate writers and teachers of writing, *TEXT Technology* brings analyses of microcomputer hardware and software, discussions of programming techniques (both in languages and in applications), book reviews, updates of significant events in computing around the world, bibliographic citations, and much more.

TEXT Technology, created by the editor of the *Research in Word Processing Newsletter*, also will become an information clearinghouse for subscribers' opinions and queries about personal computing during the '90s—and beyond.

Subscription rates for one year (6 bi-monthly issues—16 pages) of *TEXT Technology* are as follows:

US	$20
Canada	$27
Foreign	$35

All prices are in US funds

Please fill out a copy of the form below and send it with your check or purchase order to

Subscriptions Department
TEXT Technology
Wright State University—Lake Campus
7600 State Rte. 703
Celina, Ohio, USA 45822-2921

Name ______________________

College or Corporation ______________________

Street ______________________

City ______________________

State or Province ______________________

Zip Code or Postal Code ______________________

BITNET ______________________

Affiliation: ACH ____ ALC ____ ALLC ____ 09/01/91