HIGHLIGHT is a BASIC program designed to aid in textual analysis by highlighting desired portions of a text. This program has been found to have applications of value in a wide variety of language study situations. This article describes the original purpose for the program as well as some of these other applications. Professor Davison suggests we should be willing to experiment with programs that do not have an immediate general application, for it is in such experiments that new applications are discovered.

**KEYWORDS:** HIGHLIGHT, literary analysis, frequency counts, generalizing program applications, software, Atari, BASIC program.

One of the remarkable things about writing computer programs is that something designed for one purpose may unexpectedly be used for many other things. Often the programmer does not anticipate these other possibilities. This feature of programming is one of the things that makes working with computers so interesting and satisfying.

We have a tendency to be too practical in our approach to creating new programs. It is natural to seek designs that will have many applications, because a good program represents hard and time-consuming work. We must be wary, however, addressing only those areas and problems that promise immediate practical advantage. If we do so, we may limit the possibility of new discoveries. By taking some chances and occasionally pursuing a more esoteric goal we may discover that a resulting program, of little apparent utility outside a very narrow range, may, in fact, be easily adapted to much broader use.

**HIGHLIGHT**

The program offered here, is an example of a design, conceived originally as a tool for literary analysis, that proved to have unsuspected general applications.

Literary computing normally would not be considered an area of computer use that would have wide appeal or significance for the general reader. Nonetheless, simple programs, devised for specialized use in helping a literary researcher to understand a poem more fully, or to help a teacher of literature to communicate basic information about structures and features peculiar to literary language, can sometimes be used in far more general ways. The explanation is really rather simple.

The computer, large or small, is uniquely suited to identifying patterns. Its ability to scan a body of material of intense complexity and identify and track the repetition of selected elements has made it a tool suitable for exceptionally broad use.

It is self-evident that virtually all aspects of human knowledge, behavior, and learning have traditionally been described or understood in terms of patterns, patterns that are organized, managed, and categorized in order to abstract meaning, predict behavior of men or matter, and to pass on to others the essence and substance of experience. Numbers, mathematical sequences and relationships, sets, proportions, geographical realities, and neurological phenomena are random examples of such categorization.

In the instance of human verbal communication, the predominance of certain words, the composition or structure of phrases, configurations of speech sounds, and such, are equally susceptible to categorization. One can ascertain with accuracy the frequency, periodicity of occurrence, density, and clustering of a great variety of language elements. Such characteristics can be derived from transcriptions of spoken language or, more easily, from the printed form of artistic language—the poem, the narration, or the essay.

My own experimentation with literary analysis has focused almost exclusively on the isolation and description of verbal patterning. The purpose has been to ascertain relationships between basic elements of language—words, fragments of words, or larger units of composition. Syntactic features may also be isolated. On a more primitive level, the most elemental semantic structure of utterance, the phoneme or single speech sound, is also susceptible to
quantitative analysis and the description of its patterned occurrences. The computer is an ideal instrument for abstracting and ordering these various elements, and it will do it with much greater precision than was possible or reasonable by hand-analysis. We can see in the computer a tool that will not only make extrapolation of literary materials and elements more accurate but astonishingly more rapid. It has already save countless hours of tedious and painstaking effort.

The Significance of Patterning

The purposes of isolating verbal patterning in texts are fourfold. First, to better perceive the occurrence and repetition of elements, that is, to uncover with greater precision any patterning that might be present. Second, to describe and to chart the patterns discovered. Third, to analyze the patterns that have been isolated and described and to examine them statistically (e.g., frequency of occurrence relative to other elements, temporal or sequential spacing or intervals, density curves, configurations of contiguity, parallel or contrasting elements, and so on). Fourth, and last, to develop ways to make all this information—the presence of the pattern, its shape, its frequency—much easier to apprehend intellectually and experientially; and, to devise a way to translate this information into a medium, say a graphics display, immediately accessible and recognizable to anyone interested in literary structure, whether researcher, teacher, or student.

Small Versus Large

All of these objectives can be realized by using a large computer. The crux of the problem of getting broader participation in literary computing has been the difficulty of mainframe access and the cost of on-line and computing time.

The personal computer goes a long way towards eliminating these obstacles. It also enables us to use computing in simple analyses and tentative exploratory study in order to reinforce our traditional methods of criticism. The large computer, on the other hand, usually requires some interest in large, project-oriented studies, and in team efforts. The personal computer is much less threatening and, in many ways, less of an obstacle. It has therefore changed the atmosphere and way in which one may choose to work. Now, with its widespread popularity, it also allows us the opportunity to greatly expand the potential utility of any program we design.

Encouraged by this new promise of the household computer, we can now attempt to translate academically oriented concepts into microcomputer programs with the likelihood that hour work may be more widely used. Of several that I have recently written, this one, HIGHLIGHT, may be the most versatile for the average user. It was originally designed for a very simple purpose—to isolate on a screen display or in a printed copy any element or elements of a short literary text that the researcher or student wished to see set apart from the rest of the words and letters. The program has been built on the assumption that we can learn something about literary structure from such highlighting of selected elements. (I described a similar UNIVAC program in CREATIVE COMPUTING, Oct. 1, 1981.)

For example, figure 1 shows the printed result of isolations that demonstrate patterning in verse. The home computer is especially useful for this kind of program because it provides the user with more than one type of format in which to view the highlighted elements. Being able to invert the character display on the screen, for example, allows us to retain the full text and view the selected elements within their context, yet set apart, clearly and dramatically, in their black-on-white format. The hard-copy counterpart, however, retains only the highlighted elements.

Either way, the printout or the display, provides a schematic and easily apprehended statement on the patterning of the materials under study. Many things may be demonstrated: parallelisms, internal rhymes, thematically analogous words, and so on. The product is both a guide to the internal structuring and a cue to the semantic or aesthetic implications of such configurations. For example, I have found these isolations especially useful in identifying patterning of sounds and internal rhyme.

(There are some interesting variations in the sort of printout you may get depending on your printer. The Epson with Grafftrax, for example, will print inverse Atari characters as italicized. If you wanted a printed version with the selected elements in italics and the rest of the text in the normal font, you could simply eliminate the routine in HIGHLIGHT that converts unwanted text into blanks or underlines.)

Other Applications

A primary purpose of presenting HIGHLIGHT to the general reader is to suggest other applications of the program. First, any owner of an Atari could use HIGHLIGHT to prepare, for screen display or in printed form, special passages designed to teach reading, for example. (A word processing package could be used, in limited ways, for this as well.) Any level is possible, but let's assume it is to be for a youngster at home.

A simple story or poem could be chosen containing a number of new words to be introduced to the child. This text would then be entered, using any editor, or the DOS COPY command. The passage would then be called into HIGHLIGHT. The full text could then be printed out for the child to study, and subsequent texts could be printed that would omit whatever words the youngster was to study with particular care. I have used underlines as the replacement characters in order to provide a format that suggests filling in the blanks. (See figure 2a.) Children themselves could also use the program to experiment with highlighting whatever material might be of special interest. Teachers could also
print out simple quizzes or other study aids, or the program itself could be used with a large display monitor to call the students' attention to various elements of a text. Use in the study of foreign languages is similar. (See figure 2b.) The conjugation of verbs might be taught rather easily using a pre-prepared text that highlighted the stems or the various endings of the different tenses and persons. Many other sorts of language features could be examined or materials made for display on large monitors: suffixes and prefixes, for example. Or more ambitious types of grammatical analysis could very easily be set up. By selectively isolating, say, pronouns, their use, nature, function, and position (so important in foreign languages) could be illuminated. Also, adverbs, adjectives—the variety is almost limitless. Consider the English past participle and how a parent or teacher might use the program to clear confusions between it and the forms of the simple past tense.

The program enables the user to save each successive stage of the highlighting on disk for later recall. Depending on the age of the youngster, any or all stages of the uses I've described might be done entirely by the child. More advanced students, say in high school or college, could have access to sets of prepared texts that they could then load, display, or modify. Such activity might be more fruitful than the traditional computer managed teaching programs that we're familiar with. Any youngsters, or oldsters for that matter, might write original texts to work with.

I have mentioned ways that HIGHLIGHT could be used to learn about language. It could also be a tool for learning important historical dates, names of famous people, titles and authors of books, names of animals, etc. The list is limited only by our imagination.

We will make a mistake if we rely too heavily on stereotypical educational packages. There are many tools, not originally designed for learning—word processors and text editors—for example, that can be used to teach and to learn all sorts of subject matter and skills. We should take our cue from the terms commonly used to talk about the small computer. Its presence in the home is of capital significance and suggests its real potential as an aid to personal learning and knowledge. Its greatest promise lies in the maximization of personal choice and in the creative exploration of its use as free-form aid to intellectual and aesthetic growth.

**The Program**

A brief explanation of the statements will be useful to those who may want to modify them to suit any personal needs or preferences. Lines:

- 170-380 produce the title and can be eliminated.
- 310-340 dimension the character strings, whose functions are: T$ is the string into which a text is read from the disk file; File $ is the filename of the text desired. TB$ is the string to which is output the result of the processing (the removal of unwanted text), and which is read for the printout. CLEAR$ is used to set the initial search. A1$, etc., are the elements (substrings) of the text that are searched and converted into inverse characters. Y$ and S$ are used for Y/N responses made by the user.
- 360 ends the program to the routine that picks up the text from disk and reads it, T$, for restructuring. This operation, which runs from 1680 to 1890, also has other functions. It is based on the assumption that the text has numbered lines, and it highlights automatically all numerals, colons, and asterisks (used to signal stanza or paragraph breaks). It also places a blank before all carriage returns to facilitate searches.

and how a parent or teacher might use the program to clear confusions between it and the forms of the simple past tense.
Once upon a time there was a little girl who lived in a big house near a busy street. Her mother told her to be very careful when she was playing outside because there were many cars racing back and forth on the roadway...

Figure 2a: Example of Child Education Text

Habia una vez una niña que vivia en una casa grande cerca de una calle de mucho tránsito. Su mama le dijo que tuviese mucho cuidado cuando jugaba afuera porque habia muchos carros que iban y venian a todo correr en la carretera.

Figure 2b: Example of Foreign Language Study Material.

680-820 execute the searches. The GOSUB routines called are identified by the remark statements that head each of them. The restructuring, or highlighting, of the text is carried out by the routine on lines:

550 The use of the program begins. The user choices are made from here on by response to queries. (A menu-driven format was discarded because of the multiple options that have to be in some instances sequentially linked. Also, there are relatively few choices—not enough to require a menu.) The option here allows the user to review the text, if so desired, and the program then moves on to:

600 where the strings to be highlighted are requested. With:

640-650, the queries are numbered and strings to be searched are requested. Up to five strings may be searched on one request.

680-820 execute the searches. The GOSUB routines called are identified by the remark statements that head each of them. The restructuring, or highlighting, of the text is carried out by the routine on lines:

1130-1220 where the string display mode is reversed from that of the original display text (T$). If T$ is in normal letters then the selected strings will be changed to inverse characters, if T$ is in inverse then they will revert to normal white-on-black. After the conversion process is completed, the user is then invited to:

830 review the now modified version of the text. This choice is then followed at line.

870 by the option to retain or discard the new version. A yes response offers a choice to print the highlighted version or to proceed to the highlighting of additional strings by branching back to the request for new elements to search. If a printout is chosen the user is asked whether a disk-save is also wanted. If so, the save routine, 1580-1660, will be invoked by line 1300. Then lines 1310-1490 execute the printout. A chance to reconsider the choice to save the highlighted version to disk is repeated after the printout has been completed (line 1500). This second chance has been added because the printed version will sometimes suggest that it is indeed worth saving, and the user can change his or her mind.

1530 allows the user to continue to another session of text preparation or to terminate.

The program as listed replaces all selected letters with an underline character. If you prefer a blank or some other symbol, you can change it in line 1270 by replacing the underline between the quotation marks at the end of the statement.

Author's Address

1373 Butler Avenue
Salt Lake City, UT 84102
Strings to isolate

Input 5 strings to be searched

TRAP

T$:?

IF AA=ASC("Y") OR AA=ASC("y") THEN ? CHR$(125):?

(28);CHR$(28):GOTO 550

(AA<>ASC("N") AND AA<>ASC("n") THEN ? CRS$

IF (AA<>ASC("Y") AND AA<>ASC("y")) AND

CLOSE #3:OPEN #3,4,0,"K:":GET #3,AA

POKE 752,1:? "Display text?"

? :? "           * *":? :? :? :?

? " in normal fashion."

? "blanks, when needed, are entered"

? "meaningful trailing blanks. Leading"

? "to allow for searches that include"

symbol, for example, (/). This is"

All strings must be closed with a"

queries."

? "All strings must be closed with a"

"symbol, for example, (/). This is"

? "to allow for searches that include"

? "meaningful trailing blanks. Leading"

? "blanks, when needed, are entered"

? "in normal fashion."

? "?

? "Simulate hit RETURN for remaining"

? "queries."

? "All strings must be closed with a"

? "symbol, for example, (/). This is"

? "to allow for searches that include"

? "meaningful trailing blanks. Leading"

? "blanks, when needed, are entered"

? "in normal fashion."

? "?

550 POKE 752,1:? "Display text?"

560 CLOSE #3:OPEN #3,4,0,"K:":GET #3,AA

570 IF (AA<>ASC("Y") AND AA<>ASC("y")) AND

(AA<>ASC("N") AND AA<>ASC("n")) THEN ? CR$(?/>

(28);CHR$(28):GOTO 550

580 IF AA=ASC("Y") OR AA=ASC("y") THEN ? CHR$(125):?

T$:?

590 TRAP 600

600 "Use slash (/) to close string"

610 "Input 5 strings to be searched"

620 "Strings to isolate"
1170 IF TS(L,1)=" " THEN GOTO 1190
1180 TS(L,1)=CHR$(ASC(TS(L,1))+128)
1190 I=I+1:N=N+1
1200 NEXT R
1210 I=I-1
1220 RETURN
1230 ? CHR$(125):I=1:TB$=TS:K=0:REM REMOVES ALL BUT HIGHLIGHTED TEXT
1240 FOR D=1 TO LEN(TB$)-1
1250 IF TB$(L,1)=CHR$(155) THEN POSITION 10,10:? "Processing text: ";:CHR$(128):K=K+1:GOTO 1280
1260 IF ASC(TB$(L,1))=160 THEN TB$(L,1)= " ":GOTO 1280
1270 IF ASC(TB$(L,1))<170 THEN TB$(L,1)= " "
1280 I=I+1
1290 NEXT D
1300 IF SS=ASC("Y") OR SS=ASC("y") THEN POP: GOSUB 1580:GOTO 1000
1310 ? ? ? ? " CHECK YOUR PRINTER":FOR HOLD=0 TO 100: NEXT HOLD: GOTO 1320
1320 ? CHR$(125):" PAPER SET IN PRINTER?":? ? "Print text?"
1330 CLOSE #3:OPEN #3,4,0,"K:":GET #3,AA:IF AA=ASC("N") OR AA=ASC("n") THEN ? CHR$(125):GOTO 1500
1340 IF (AA<>ASC("Y") AND AA<>ASC("y")) OR (AA<>ASC("N") AND AA<>ASC("n")) THEN GOTO 1320
1350 TB$(LEN(TB$)+1)=CHR$(155)
1360 TB$(LEN(TB$)+1)=CHR$(155)
1370 TB$(LEN(TB$)+1)="Filename "
1380 IF FILE$(3,3)=":" THEN TB$(LEN(TB$)+1)=FILE$(4):GOTO 1400
1390 TB$(LEN(TB$)+1)=FILE$(3)
1400 CLOSE #2:TRAP 1310
1410 IF TYPE=ASC("2") THEN OPEN #2,8,0,"P:"GOTO 1450
1420 OPEN #2,8,0,"R2:
1430 XIO 34,#2,192,0,"R2:
1440 XIO 36,#2,0,4,"R2:
1450 ? #2:TB$?
1460 ? CHR$(28); "Another copy?"
1470 CLO 3:OPEN #3,4,0,"K":GET #3,AA:IF AA=ASC("Y") OR AA=ASC("y") THEN #2; CHR$(138):CHR$(138):GOTO 1450
1480 IF (AA<>ASC("Y") AND AA<>ASC("y")) OR (AA<>ASC("N") AND AA<>ASC("n")) THEN ? CHR$(28):GOTO 1460
1490 CLOSE #2:IF FIRST<>0 THEN GOTO 420
1500 ? ? "Save isolations on disk?"
1510 CLOSE #3:OPEN #3,4,0,"K":GET #3,AA:IF AA=ASC("Y") OR AA=ASC("y") THEN POSITION 10,10:? "Processing text: ";:CHR$(128):CHR$(138):GOTO 1500
1520 IF (AA<>ASC("Y") AND AA<>ASC("y")) AND (AA<>ASC("N") AND AA<>ASC("n")) THEN ? CHR$(28):GOTO 1460
1530 ? CHR$(125):" Do you wish to continue the session?"
1540 CLO 3:OPEN #3,4,0,"K":GET #3,AA:IF AA=ASC("Y") OR AA=ASC("y") THEN TB$=TS?:CHR$(125):CHR$(129):GOTO 550
1550 IF (AA<>ASC("Y") AND AA<>ASC("y")) AND (AA<>ASC("N") AND AA<>ASC("n")) THEN GOTO 1530
1560 GOTO 1080
1570 ? "DISK FULL?":CLOSE #1:TRAP 40000:GOTO 1590:REM FOR RECORDING ISOLATIONS
1580 TRAP 1570:CHR$(125):? ? ? ? " Name of file (Use D): ";:INPUT FILES:IF FILES(1,1)<"D" THEN POKE 752,0:GOTO 1590
1600 OPEN #1,8,0,FILES
1610 PRINT #1,TS
1620 CLOSE #1
1630 POKE 752,1
1640 ? ? "TEXT RECORDED":? ?
1650 FOR G=0 TO 100:NEXT G
1660 RETURN
1670 ? "NOT ON DISK":CLOSE #1:POKE 752,0:GOTO 1700

1680 REM ROUTINE FOR INPUT OF SAVED TEXT. FORMATS TEXT FOR SEARCHES
1690 TRAP 1670
1700 POSITION 2,5:" Name of file (USE D): ";:INPUT FILES:IF FILES(1,1)<"D" THEN POKE 752,0:GOTO 1700
1710 TS=" ":POKE 752,1
1720 ? :? " Input text in INVERSE LETTERS = 1": ? "STANDARD LETTERS = 2"
1730 CLOSE #3:OPEN #3,4,0,"K":GET #3, INVERSE
1740 IF INVERSE<>ASC("1") AND INVERSE<>ASC("2") THEN GOTO 1720
1750 :? :? " Loading text, one moment please."
1760 IF INVERSE=ASC("1") THEN GOTO 1900
1770 OPEN #1,4,0,FILE$?
1780 I=0
1790 I=I+1
1800 GET #1,T
1810 IF T=155 THEN TB$(I,1)=" ":I=I+1:REM PUTS SPACE BEFORE OR TO ENABLE SEARCH
1820 IF T>47 AND T<59 OR T=42 THEN T=T+128:GOTO 1840:REM HIGHLIGHTS LINE NUMBERS AND ASTERISK
1830 IF (T>32 AND T<48) OR (T>57 AND T<65) THEN TB$(I,1)=": ":I=I+1:REM PUTS SPACE BEFORE PUNCTUATION
1840 TRAP 1880
1850 TS(L,1)=CHR$(T)
1860 IF T=155 THEN TB$(I,1)=": ":REM PUTS SPACE AFTER CR TO ENABLE SEARCH
1870 GOTO 1790
1880 CLOSE #1
1890 RETURN
1900 OPEN #1,4,0,FILES
1910 I=0
1920 I=I+1
1930 GET #1,T
1940 IF T=155 THEN TB$(I,1)=": ":I=I+1:REM PUTS SPACE BEFORE OR TO ENABLE SEARCH
1950 IF T>47 AND T<59 OR T=42 THEN T=T+128:GOTO 1840:REM HIGHLIGHTS LINE NUMBERS AND ASTERISK
1960 IF (T>32 AND T<48) OR (T>57 AND T<65) THEN TB$(I,1)=": ":I=I+1:REM PUTS SPACE BEFORE PUNCTUATION
1970 TRAP 1910
1980 TS(L,1)=CHR$(T)
1990 IF T=155 THEN TB$(I,1)=": ":REM PUTS SPACE AFTER CR TO ENABLE SEARCH
2000 GOTO 1920
2010 CLOSE #1
2020 RETURN
2030 REM SUBSTRING SEARCH ROUTINE
2040 REM courtesy of Edward C. Smith
2050 A1=A1+1
2060 GOSUB 2110:REM machine search
2070 IF A=0 THEN RETURN
2080 BEG=A1+1:FIN=A1+A-1+LEN(A1)-1
2090 IF TS(BEG,FIN)=A1 THEN I=BEGIN:FIN=FOCUS 1130
2100 GOTO 2050
2110 LY=LEN(TS(L,1)):LX=LEN(A1):POKE 207,LX
2120 GOSUB 2170:IF A=0 THEN 2160
2130 B=LY-LX-A1+3
2140 Z=USR(1664,ADR(TS(A1)),ADR(A1),B)
2150 IF A=0 THEN A1=1
2160 RETURN
2170 REM CORRECT START LOCATION A1 IF ENTERED VALUE IS OUT OF RANGE
2180 A1=A1+LY-LX+1 OR LX>LY THEN A=0:A1=1
2200 RETURN

CALICO Journal, Volume 2 Number 1 19