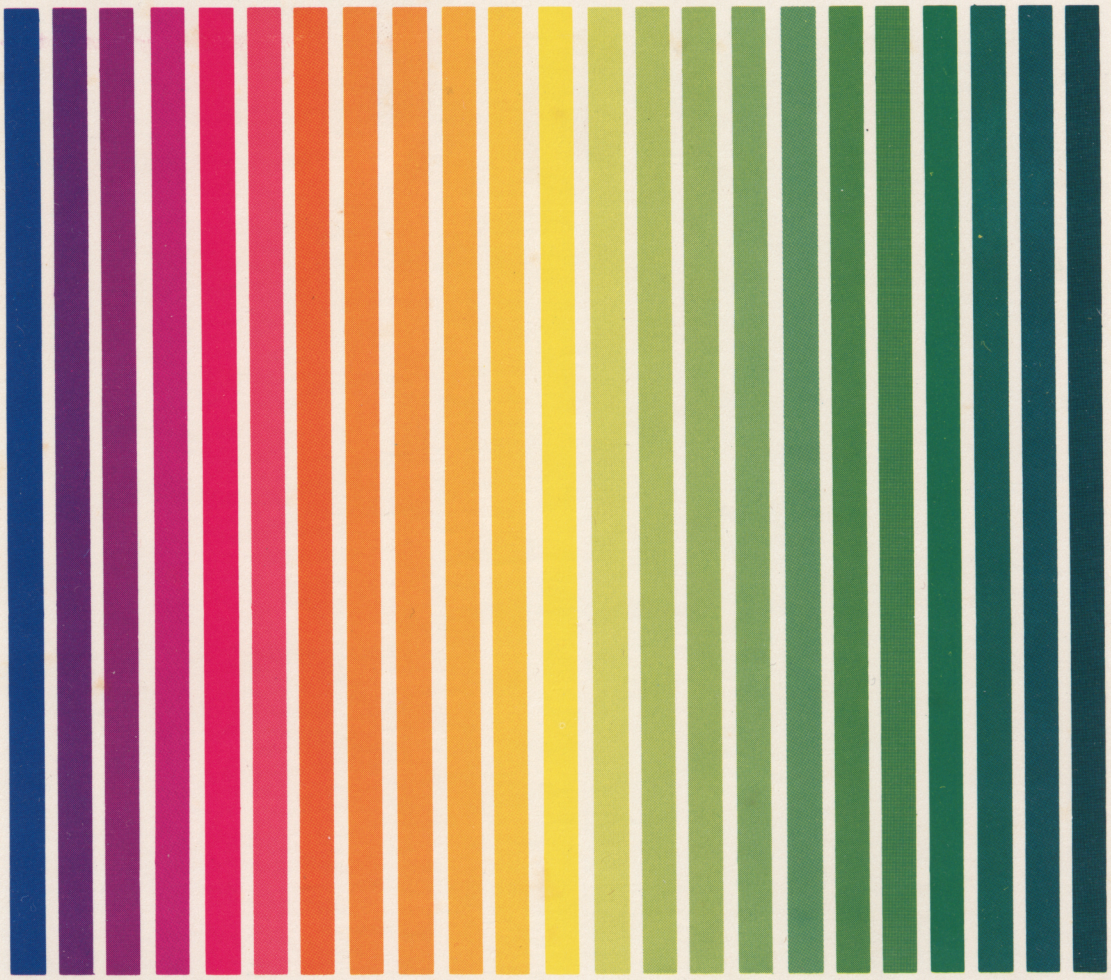
ATARI® PROGRAM EXCHANGE



WORD PROCESSING DISKETTE

ATARI Program-Text Editor(tm)
and Text Formatter (FORMS)

Diskette: 32K (APX-20076)

User-Written Software for ATARI Home Computers

WORD PROCESSING DISKETTE

ATARI Program-Text Editor(tm)
and Text Formatter (FORMS)

Diskette: 32K (APX-20076)

WORD PROCESSING DISKETTE

Distributed By

The ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

To request an APX Product Catalog, write to the address above, or call toll-free:

800/538-1862 (outside California)
800/672-1850 (within California)

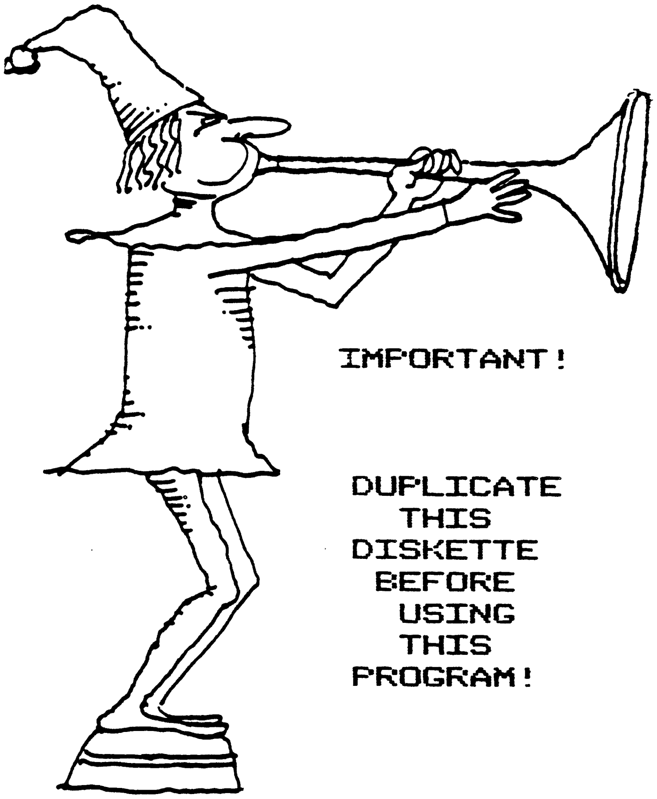
Or call our Sales number, 408/727-5603

Trademarks of Atari

The following are trademarks of Atari, Inc.

ATARI®
ATARI 400™ Home Computer
ATARI 800™ Home Computer
ATARI 410™ Program Recorder
ATARI 810™ Disk Drive
ATARI 820™ 40-Column Printer
ATARI 822™ Thermal Printer
ATARI 825™ 80-Column Printer
ATARI 830™ Acoustic Modem
ATARI 850™ Interface Module

Printed in U.S.A.



IMPORTANT!

DUPLICATE
THIS
DISKETTE
BEFORE
USING
THIS
PROGRAM!

This APX diskette is unnotched to protect the software against accidental erasure. However, this protection also prevents a program from storing information on the diskette. The program you've purchased involves storing information. Therefore, before you can use the program, you must duplicate the contents of the diskette onto a notched diskette that doesn't have a write-protect tab covering the notch.

To duplicate the diskette, call the Disk Operating System (DOS) menu and select option J, Duplicate Disk. You can use this option with a single disk drive by manually swapping source (the APX diskette) and destination (a notched diskette) until the duplication process is complete. You can also use this option with multiple disk drive systems by inserting source and destination diskettes in two separate drives and letting the duplication process proceed automatically. (Note. This option copies sector by sector. Therefore, when the duplication is complete, any files previously stored on the destination diskette will have been destroyed.)

PREFACE

This manual combines the two manuals accompanying the programs on the WORD PROCESSING DISKETTE. They are in the following order:

1. ATARI Program-Text Editor™

2. Text Formatter (FORMS)

Each manual is reproduced in its entirety, including separate Review Forms, so that you can send us your comments about the programs and manuals as you use them.

ATARI
PROGRAM-TEXT EDITOR™

ATARI®

A Warner Communications Company

Every effort has been made to ensure that this manual accurately documents this product of the ATARI Computer Division. However, because of the ongoing improvement and update of the computer software and hardware, ATARI, INC. cannot guarantee the accuracy of printed material after the date of publication and cannot accept responsibility for errors or omissions.

PRINTED IN U.S.A.

CONTENTS

ILLUSTRATIONS

TABLES

1

SYSTEM REQUIREMENTS

The ATARI® **Program-Text Editor™** (Model No. CX8121) requires:

- **ATARI 810™ Disk Drive**
- ATARI Blank Diskette (CX8100)

For information on your disk drive, refer to the *ATARI 810 Disk Drive Operators Manual.* Check the drive code setting to make certain that you have a disk drive designated as Drive 1. Because the Disk Operating System (DOS) II programs are included on the diskette containing the Program-Text Editor, you can easily load the editor software by inserting your diskette in Drive 1. Otherwise, you must have a copy of the DOS II Master Diskette, Model No. CX8101, inserted into Drive 1.

You must have at least 24K RAM in your ATARI Home Computer to operate the disk drive and the editor software. Although the software requires 24K memory, a total memory capacity of 32K is highly recommended and will result in increased program efficiency. For instructions on inserting additional **ATARI RAM Memory Modules™** into the **ATARI 800™ Computer**, refer to the *ATARI 800 Operators Manual.*

SETUP PROCEDURES

1. Verify that all power switches (console **and** disk drive) are turned to OFF.
2. Check that the computer console is properly connected to the television set and a standard wall outlet.
3. Place the disk drive at least 12 inches away from your television set and plug it into a standard wall outlet.
4. Connect the disk drive to either the computer console or another ATARI peripheral. Plug one end of the I/O Data Cord into the jack labeled I/O CONNECTORS on the back of the disk drive. Plug the other end into either the jack labeled PERIPHERAL on the computer console or one of the I/O CONNECTOR ports of another ATARI peripheral. If you connect your disk drive to another ATARI peripheral, verify that there is an I/O Data Cord plugged into the computer console.

TURNING ON THE SYSTEM

When you are ready to use the computer, proceed as follows:

1. Turn on the television set. Tune to Channel 2 or Channel 3, whichever has a weaker signal in your area. Make certain that the 2-CHAN.-3 switch on the computer console corresponds to your channel selection.

2. Turn on the disk drive. Notice that the BUSY and PWR ON light indicators are activated. Wait until the motor of the disk drive stops its activity and the BUSY light goes out before continuing to the next step.

3. Insert the diskette containing the Program-Text Editor into the disk drive designated as Drive 1.

   Note: DO NOT TOUCH THE EXPOSED PORTION OF THE DISKETTE.

4. Turn the computer console power switch to ON. This will activate the disk drive's loading procedure.

Note: *OPTIONAL*. To increase the RAM buffer size on a 48K system, before turning on your computer, remove any language cartridge that might be installed.

Take note of the following conditions to determine if you have successfully completed the power-on procedure. If you have a language cartridge inserted into the computer console, the screen displays the prompt applicable to that particular language. For example, the ATARI BASIC language prompt is the READY message; the ASSEMBLER EDITOR language prompt is the EDIT message. Otherwise, the DOS Menu should appear immediately upon the screen.

TURNING OFF THE SYSTEM

Warning: NEVER turn off the disk drive with a diskette in it. You may damage the information contained on the diskette and lose the ability to load your program.

When you are ready to end your editing session:

1. Use the exit command appropriate for your editing session.

2. Wait for the DOS II Menu display or the filename prompt to appear on the screen.

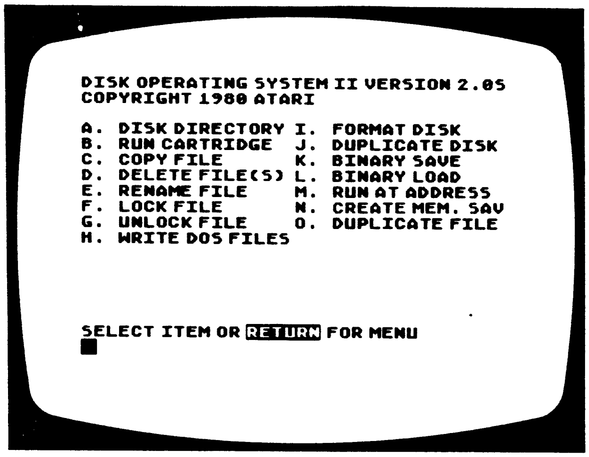
3. Remove the diskette from the disk drive and return it to the protective sleeve that was provided with the software.

4. You may turn off the television set, the computer, or the disk drive in any order.

2

OPERATIONAL PROCEDURES FOR THE EDITOR

You must load the editor through the DOS Menu. If the DOS Menu is not already displayed on your screen, type **DOS** and press RETURN. Refer to Figure 1. (The *DOS II Reference Manual* contains complete instructions for using the DOS II Menu options.)



```
DISK OPERATING SYSTEM II VERSION 2.0S
COPYRIGHT 1980 ATARI

A.  DISK DIRECTORY I.  FORMAT DISK
B.  RUN CARTRIDGE  J.  DUPLICATE DISK
C.  COPY FILE      K.  BINARY SAVE
D.  DELETE FILE(S) L.  BINARY LOAD
E.  RENAME FILE    M.  RUN AT ADDRESS
F.  LOCK FILE      N.  CREATE MEM. SAV
G.  UNLOCK FILE    O.  DUPLICATE FILE
H.  WRITE DOS FILES



SELECT ITEM OR RETURN FOR MENU
█
```

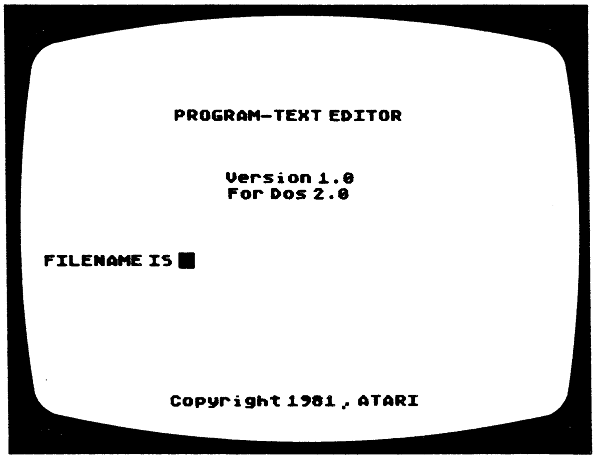
Figure 1   DOS II Menu

Because the editor program is included on a diskette that has been factory write-protected for software safety, you must prepare a diskette for your text files. For identification purposes, we refer to this diskette as a "data" diskette. With the DOS Menu displayed on the screen, remove the diskette containing the Program-Text Editor software. Refer to the *DOS II Reference Manual.* Format a blank diskette, then write new DOS files to it. Remove this diskette and reinsert the editor program diskette.

Select the L-BINARY LOAD command. Answer the prompt, LOAD FROM WHAT FILE, with the name of the Program-Text Editor software, **MEDIT**. The program will automatically run after being loaded. Refer to Figure 2. Insert your data diskette into the disk drive at this time.

Caution: You may **not** change your data diskette once the editing session is started. Because the editor has built-in memory checks and free space allocation computations, a memory map of the diskette inserted at the time the editor performs its check is always retained. Therefore, even though the editor's workspace resides in RAM, the block-write command can result in an overwrite situation on any but the original diskette.

Note: Because the editor performs a free IOCB (Input/Output Control Block) check, you may receive the error message EDITOR CANNOT RUN - NO FREE IOCBs. PRESS START to return to DOS. Refer to the *ATARI Operating System Manual* (part number CO16555) for complete information on IOCBs and to the error messages on the back cover for an explanation of this condition.



```
PROGRAM-TEXT EDITOR

Version 1.0
For Dos 2.0

FILENAME IS █

Copyright 1981 , ATARI
```

Figure 2 Filename Prompt

Your Program-Text Editor is now ready to bring the file that you wish to edit into its workspace. At this point, there are several options available:

- Press the BREAK key to end the edit session and return control to DOS.
- Enter the filename of the program that you wish to edit.
- Create a new file under the editor by naming a file that does not exist. The editor will automatically create an empty file using the specified name.

The correct syntax for an acceptable filename is in the form:

D*n*:filename.extension,optional parameters separated by commas.

Example: D4:MYFILE.MAC,3,.ASM,D

The drive number ***n*** designation corresponds to the disk drive that contains your source program and must be between the numeric characters of one and eight. You may use a filename of from one to eight characters, either alpha characters A through Z or numeric characters 0 through 9.

Note: For a filename specification, an alpha character must be in the first character position. This rule does not apply to filename extensions.

Your optional extension may be from one to three characters long, using either alpha or numeric characters.

Remember the following specifications when answering the filename prompt.

- If no device is specified, the editor automatically assumes the use of the disk drive designated as Drive 1.
- Lowercase file specifications automatically convert to the correct uppercase syntax.
- If the file, its associated backup file, or its temporary file is locked (see the "Theory of Operation" section for further explanation), the editor displays the error message FILE LOCKED and reissues the filename prompt. Unlock any of these files through use of the DOS Menu. Refer to the *ATARI DOS II Reference Manual.*

Optional parameters may be entered in any order after the file specification:

,n OVERRIDE DESTINATION DRIVE. Unless otherwise specified, the default destination drive is the one on which the source file is located. You may move the destination file from the default drive by using this parameter. The value ***n*** is a numeric digit corresponding to the number of the destination disk drive.

Example: MYFILE,2

When you have more than one disk drive, use this optional parameter to edit large files or when there is not enough free space on the source diskette to allow you to save the edited file.

,D DELETE BACKUP FILE FLAG. If a backup file exists, this parameter tells the editor to erase it before beginning the editing session. Use of this parameter allocates free space at the cost of backup file protection.

Example: MYFILE,D

Note: If the source and destination drive are not the same, the editor automatically deletes a file with the same name on the destination drive.

,.ext OVERRIDE CUSTOMIZING FILE. Use of this parameter causes the editor to use the customizing file associated with the designated extension file. Unless this parameter is assigned, the editor defaults to use of the extension associated with the file specification being edited.

Example:

```
MYFILE,.PAS
MYFILE,.ASM
MYFILE,.BAS
```

Following are additional examples of valid filename prompt responses.

```
MYFILE
MYFILE.PAS
D3:MYFILE
D3:MYFILE,2
MYFILE.PAS,D,4
D2:MYFILE,.PAS,D,3
MYFILE,.ASM
d2:myfile,.pas,d,3
D4:MYFILE.BAS,3,.PAS,D
```

After receiving the filename specification, your Program-Text Editor checks the free space on the destination diskette and makes a comparison with the size of the file to edit. A minimum growth factor, considering the expansion of file storage capacity requirements because of additions or modifications, of **g** units is ascertained. (See the "Customizing the Editor" section.) If there is not enough room on the diskette for the edit file and the growth factor, the editor displays a warning message. You may choose to ignore the warning and continue with the editing session. Or you may abort the edit, exit from the editor, and return to the DOS Menu. If the editor determines that there is enough room on the diskette for the edit file and growth factor, the edit session begins.

Caution: If you ignore the warning message, be sure that you have as much free space as the size of your existing file plus room for any additions you will make during the editing session. If your calculations are not correct and you run out of free space on the diskette, you may lose all work completed in the current editing session.

Note: A minimum growth factor of **g** units is determined from the customizing file. If default factors are used, the minimum growth factor is 100 sectors of free space.

THEORY OF OPERATION

For efficiency and optimum protection, the Program-Text Editor uses a common **two-file** editing method. During the editing session, the original file remains intact while all modifications are made to a copy of the file. Therefore, this procedure allows for:

- Automatic backup copies of files to be edited
- Modification of the original file **only** after the editing session is terminated with a normal exit from the editor
- Use of sequential file access

A procedural outline of the two-file method is:

- Text is copied from the file to be edited into a memory buffer.
- When the buffer becomes full, data transfers to a **temporary** file.

Normal exit (Figure 3) from the editor causes the following sequence:

- The .BAK file is deleted.
- The edited file is renamed as the new .BAK file.
- The temporary file is renamed as the edited file.

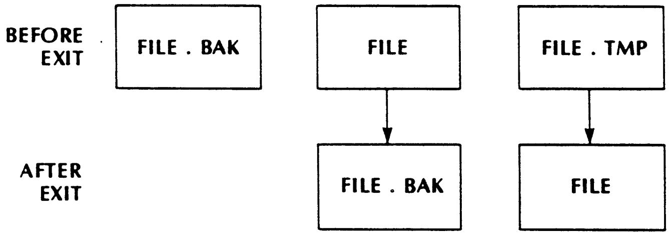


Figure 3 Normal Exit From the Editor

An abort exit (Figure 4) from the editor causes the following sequence:

- The temporary file is deleted.
- The original edited file and the .BAK file retain their integrity.

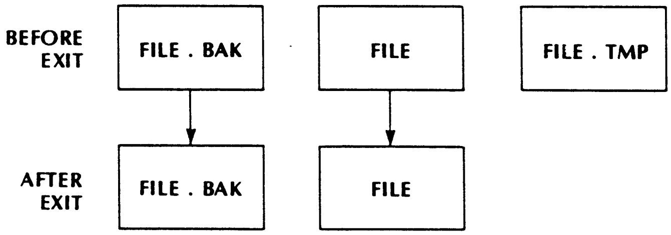


Figure 4 Abort Exit From the Editor

Your Program-Text Editor uses two modes of operation: immediate and command. Immediate mode operation is keyboard interactive. Command mode operation defers to a later time execution. All three windows and both operation modes are discussed at length in subsequent sections of this manual.

The Program-Text Editor is defined as a **source file editor**. A source file is a disk file consisting of ATASCII characters terminated by ATASCII EOLs (End-of-Line). Therefore, the editor functions with files containing the source code written for ATARI Computer programming languages. A line length default value of 114 columns can be changed to a maximum length of 200 columns by using the customizing file feature (see section titled "Customizing the Editor").

Two types of tabs are allowed: (1) regular tabulation as provided by the operating system in which blanks are substituted between tab stops or (2) expanding tabs. Expanding tabs only take one character in the file but are displayed as many columns of blanks. Set the type of tab by using the customizing file.

```
LINE OF TEXT-35 CHARACTERS
     LINE OF TEXT
     LINE OF TEXT
```

5 character displacement = 5 bytes of memory
using default value of 5, inserting blanks like the operating system

```
LINE  OF TEXT-35 CHARACTERS
▶....LINE OF TEXT
▶....LINE OF TEXT
```

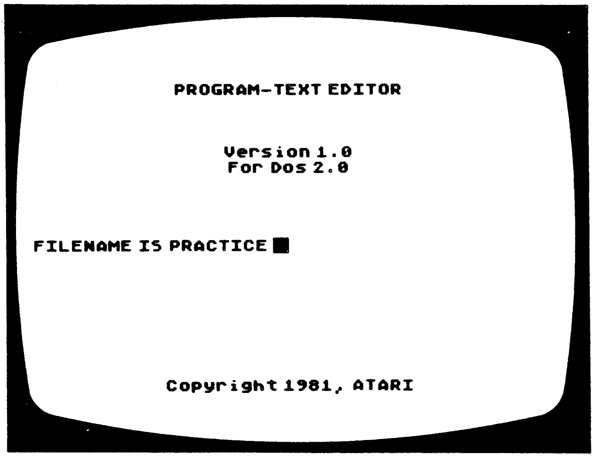
5 character displacement = 1 byte of memory
using expanding tabs

Figure 5 Illustration of Expanding Tabs

If you attempt to edit a file that does not meet source file definitions and customizing column limits, the editor truncates the lines in the file to conform to the set line length limits. Given this situation, the editor generates the LINE TOO LONG error message while reading the file either during initial entry to the editor or as an input command.

STARTING THE EDIT SESSION

Answer the filename prompt. For the purposes of demonstration, enter the filename **PRACTICE**. Refer to Figure 6.



```
PROGRAM-TEXT EDITOR

Version 1.0
For Dos 2.0

FILENAME IS PRACTICE █

Copyright 1981, ATARI
```

Figure 6 Answering the Filename Prompt

Notice the three windows displayed on the screen:

TEXT WINDOW — Appears at the top of the screen and consists of 20 lines.

ERROR WINDOW — Appears in inverse video and consists of a single line.

COMMAND WINDOW — Appears at the bottom of the screen and consists of three lines.

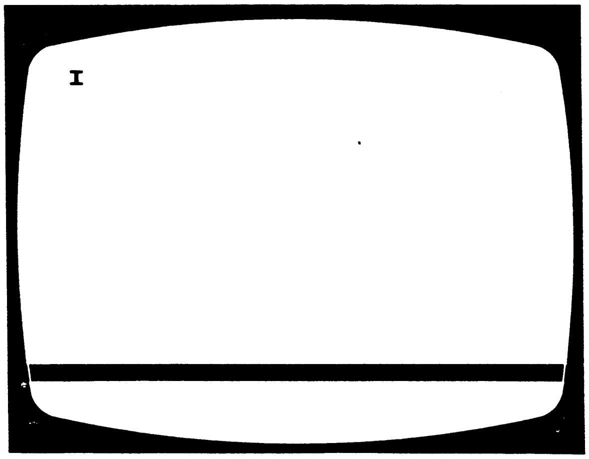
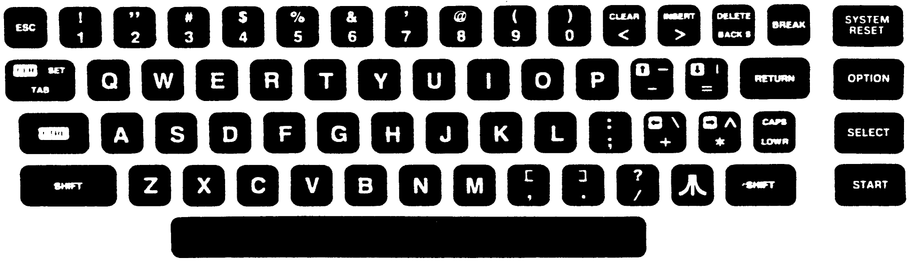


Figure 7 Windows

FAMILIARITY WITH THE KEYBOARD

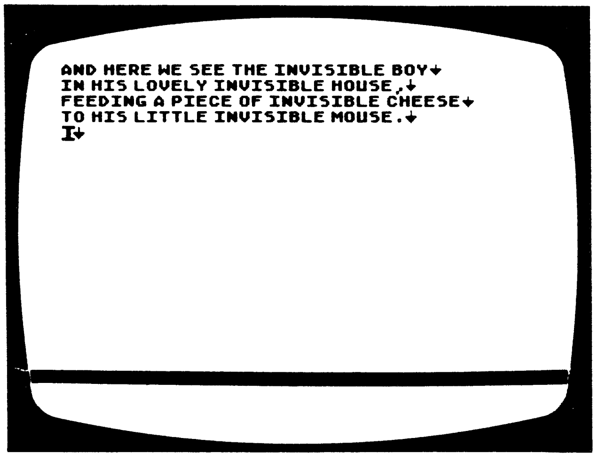
A summary of the immediate keystroke commands appears at the end of this section.



From the keyboard shown above, locate the following specific keys: CTRL, SHIFT, DELETE BACK S, CLEAR. Note that there are keys indicating directional arrows as well as arithmetic operators. Some keys serve a dual purpose, for example, the DELETE BACK S. As the operation of the SHIFT key on the computer keyboard is the same as the shift key of a typewriter, its use will select the function that appears on the top of the key.

Enter the following text onto your screen:

AND HERE WE SEE THE INVISIBLE BOY RETURN
IN HIS LOVELY INVISIBLE HOUSE, RETURN
FEEDING A PIECE OF INVISIBLE CHEESE RETURN
TO HIS LITTLE INVISIBLE MOUSE. RETURN



```
AND HERE WE SEE THE INVISIBLE BOY↓
IN HIS LOVELY INVISIBLE HOUSE,↓
FEEDING A PIECE OF INVISIBLE CHEESE↓
TO HIS LITTLE INVISIBLE MOUSE.↓
I↓
```

Figure 8 Example of Entered Text

After entering the lines, notice the following: every time you press the RETURN key, an ↓ appears on the screen. This figure indicates the carriage return action. Also, pay particular attention to the movement of the cursor. During execution of the keystroke entry, the cursor position indicates character placement by appearing immediately in front of the next entry. After any keystroke, the text window is updated to reflect the current state of the file, and the cursor moves accordingly. Look again at the above screen diagram and note the cursor positioning.

Using the table below, manipulate the cursor within your displayed text.

Keystroke	**Explanation**
CTRL ←	Moves cursor left
CTRL →	Moves cursor right
CTRL ↓	Moves cursor down one physical line
CTRL ↑	Moves cursor up one physical line
CTRL 2	Moves cursor to beginning of logical line
CTRL 3	Moves cursor to end of logical line

After you feel thoroughly acquainted with the movement produced by striking these keys, follow the procedure outlined below:

Position the cursor on the "A" of the first word in the first line of your text. Use the CTRL ← keystroke. Now use the CTRL ↑ keystroke. Note that both of these operations result in the warning message CURSOR AT END. The same error message will be displayed if you use a CTRL → if the cursor is in the far right position at the end of text.

Note: When the cursor moves up and down a slight glitter of the screen may occur. Also, on occasion, you may notice the appearance of an additional line below the command window. These are normal operating conditions.

Notice that these cursor-movement keystrokes position the cursor but do not affect the entered text. Within the immediate mode operation, there are essentially two types of keystrokes: those that directly relate to cursor positioning and those that execute a change to the text itself. You must position the cursor at a precise point using the above key combinations. Refer to the table below for those keystrokes that will immediately edit entered text.

Keystroke	**Explanation**
SHIFT INSERT	Inserts a blank line above the current logical line
DELETE BACK S	Deletes character left of cursor
CTRL DELETE BACK S	Deletes character right of cursor
SHIFT DELETE BACK S	Deletes the logical line occupied by cursor
Regular keys	Insert character into text

Within the framework of this software and as a matter of convention, this manual introduces the terms ***logical line*** and ***physical line***. A logical line contains those characters entered between carriage returns. A physical line encompasses those characters contained in a straight line from the extreme left side to the extreme right side position of your television screen. A logical line can be one or more physical lines.

Return to your screen. You must use your cursor control keys to move your cursor during an edit session. Position the cursor so that it is over the "v" in the word "invisible." Use the DELETE BACK S key twice. (**Do not** press RETURN. Pressing the RETURN key at any time will introduce a carriage return figure, ↓ into your text.)

Take note of several unique conditions that might arise from operation of the DELETE BACK S and CTRL DELETE BACK S keys.

If the cursor is to the right of a carriage return, use either the DELETE BACK S key or the CTRL ← key to reposition the cursor. However, when the cursor is to the immediate right of a carriage return, use of the DELETE BACK S key deletes the carriage return itself. Similarly, if the cursor is to the left of a carriage return, use of the CTRL DELETE BACK S key repositions the cursor exactly as use of the CTRL → key. However, when the cursor is to the immediate left of a carriage return, use of the CTRL DELETE BACK S key removes the carriage return itself. Concatenation follows the carriage return deletion. If the maximum line length is exceeded, the editor:

- Restores the deleted carriage return
- Aborts the command line
- Displays the error message LINE TOO LONG
- Returns to immediate mode operation

Other specific conditions that result when the cursor is positioned:

Within an expanding tab	Use of either keystroke deletes the entire tab.
At the beginning of the buffer	Use of the DELETE BACK S key results in no operation and generates the error CURSOR AT END.
Above the text window	Use of the DELETE BACK S key causes an automatic scroll that pulls down the previous line.
At the end of the buffer	Use of the CTRL DELETE BACK S key results in no operation and generates the error CURSOR AT END.

Note: Attempted deletion of the last carriage return in the buffer is illegal and results in the CURSOR AT END error message. Use a delete line operation to successfully remove this last carriage return.

Follow the same procedure to acquaint yourself with the use of the other keystrokes outlined in the table. Use cursor control keystrokes to position the cursor. Select the appropriate key to accomplish the desired change. Use cursor control keystrokes to remove the cursor from the logical line.

On the ATARI Computer keyboard, locate the ESC key. Use this key in conjunction with control graphics keys to print specific graphics characters. Refer to Figure 9 for keystroke combinations to produce a chosen graphics display.

Press the ESC key and then press:

Press the ESC key and then press the CTRL key simultaneously with:

Press the ESC key and then press the SHIFT key simultaneously with:

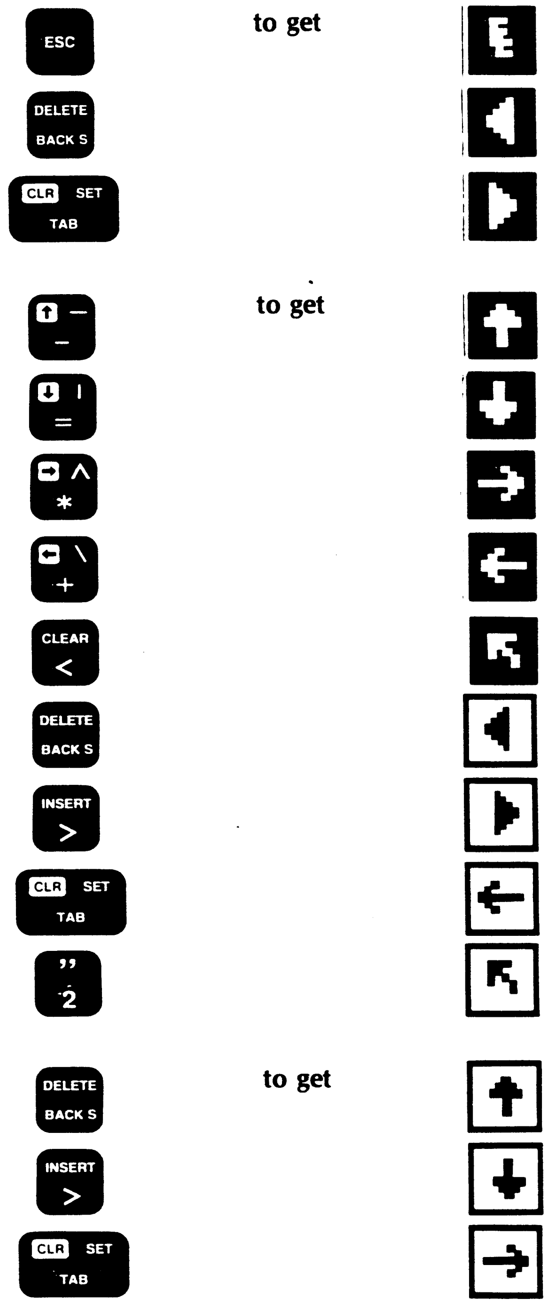


Figure 9 Escape Sequence Characters

If the cursor is within an expanding tab or to the right of a carriage return when a character is inserted into text, the editor automatically repositions the tab or carriage return to the right of the cursor.

Additional cursor movement keystrokes:

Keystroke	Explanation
CTRL 8	Displays previous screen
CTRL 9	Displays next screen

Use the keystrokes above to respectively display either 20 physical lines above or below the text window. Additional reserved keystrokes include:

Keystroke	Explanation
CLEAR SET TAB	Tabs to next tab stop
CTRL RETURN	Returns and auto-indents to same level
CTRL CLR SET TAB	Toggles visible-tab mode
SHIFT CLR SET TAB	Toggles visible-carriage return mode

Use the CLR SET TAB key to position the cursor. Space tabs insert a selected number of blanks between tab stops, and the cursor positions itself accordingly. Expanding tabs, however, insert a character into the text that indicates the tab function. By using the customizing file, you can display the expanding tab character as either blanks or a right triangle followed by periods. Set your default choice within the parameters of the customizing file. If you have chosen the expanding tab option, use the CTRL CLR SET TAB immediate mode keystroke command to display the alternate character choice.

Carriage returns can be displayed as blanks or down-arrows. Default choice is set within the parameters of the customizing file. Use the SHIFT CLR SET TAB immediate mode keystroke to display the alternate character choice.

Auto-indention allows you to reposition the cursor to return to an automatic tab stop on the next logical line. Press the CTRL and RETURN keys simultaneously. The indention of the logical line containing the cursor determines the position of the automatic tab.

Keystroke	Command
CTRL CLEAR	Erases the error window
START	Executes command window
SELECT	Selects the alternate command line
OPTION	Changes mode
BREAK	Aborts command being executed

Error messages displayed within the error window are cleared in three ways:

- Pressing the CTRL CLEAR keys will clear the error.
- If a syntax error occurs, the window clears when the command is corrected.
- After four seconds of elapsed time, the error window automatically clears with any keystroke entry.

Use the OPTION key to change operation modes. In immediate mode operation, use of OPTION enters command mode. Switching these operation modes automatically clears the current command window. To avoid this erasure, use the SHIFT OPTION combination keystroke. The current command line remains intact, and the cursor positions itself at the end of the command line.

Within the command mode, all keystrokes enter text into the command window. All immediate and reserved keystrokes, with the exception of DELETE BACK S, can still be executed. Use of the DELETE BACK S key deletes the last character typed into the command window. Pressing OPTION twice while in command mode deletes the entire command line.

During execution of the command window, the editor is in command mode. Notice that the cursor remains in the command window while the command is being executed. After successful completion of the command execution, the cursor disappears from the command window and the editor returns to immediate mode operation. Use the SELECT key to rotate displays of the command line and any alternate entry. Touch the BREAK key during execution of the command line to discontinue processing. As soon as the current command execution is completed, a BREAK KEY ABORT message appears in the error window, and the editor returns to immediate mode operation. Touching BREAK at any other time has no effect.

In command mode, the use of OPTION returns control to immediate mode. The command line remains in the command window for later execution. Use the START key to execute commands within the command window. A NOT COMPLETE error message results when a command contains a syntax error. The editor remains in command mode so that correction can be made. Executing a blank command or an empty display window returns control to immediate mode.

COMMAND MODE OPERATION

The command window accepts and displays all keystroke entries made in command mode operation. With the exception of the DELETE BACK S key, all immediate reserved keystrokes function identically within either operation mode. The command window is three physical lines long and allows a single command line that is made up of one or more commands. You may enter spaces between commands for better readability, and use either upper- or lowercase. Within the command window, a carriage return is displayed as the inverse E escape sequence character. A mini-interpreter checks each keystroke for valid syntax. The following syntax error messages may be displayed:

- UNRECOGNIZED COMMAND
- DELIMITER ERROR
- NUMBER TOO BIG

If a syntax error occurs, the editor ignores all keystrokes until you delete the offending character from the command window. Manipulation of the command window is as follows:

- OPTION key returns the editor to immediate mode operation.
- OPTION key pressed twice erases the entire command window.
- DELETE BACK S key deletes the last character entered into the command window.
- START key executes the command line if the syntax is correct and complete.
- SELECT key swaps the command line displayed in the command window with an alternate command line.

After execution of the command line, the editor returns to immediate mode operation. The command line is not erased and may be reexecuted by pressing START.

EXITING THE EDITOR

Depending upon your desired end result, choose one of the following options to exit from the editor:

Command	**Explanation**
EXIT	Use this command to exit from the editor and return to DOS. All changes made during the edit session are retained.
EXIT2	Use this command to exit from and restart the editor. In effect, this command duplicates the action of EXIT followed by the DOS "L" (load) command, and you will receive the editor sign-on filename prompt.
ABORT	Use this command to exit from the editor without incorporating any changes made during the edit session and return control to DOS.
ABORT2	Use this command to exit without incorporating any changes made during the edit session and restart the editor. In effect, this command duplicates the action of ABORT and DOS "L" (load) commands. You will receive the editor sign-on filename prompt.
REOPEN	Use this command to exit from the editor and automatically reenter the same file. In effect, this command duplicates the action of EXIT2 and answering the filename prompt with the specification of the file you are editing. See "Specialized Commands" within this section for specific details.

Note: The editor accepts the exiting commands in the form EXIT***n*** and ABORT***n*** as valid syntax. However, at execution time, the error message NUMBER TOO BIG is generated if ***n*** is greater than 2.

CURSOR MOVEMENT COMMANDS

You may manipulate the cursor through command mode operation. This method lets you quickly move the cursor to where you want it. To use the following table effectively, you must be familiar with two terms: ***buffer*** and ***file***. In this particular software application, text is copied from the file to be edited into a memory buffer where modification is achieved. When the memory buffer becomes full, it is written to a temporary file. This process is repeated continuously until all text has been copied from the edited file into a temporary file. As you can determine, the contents of the edited file and the memory buffer can differ.

Note: Take care in planning your editing session. You cannot easily edit the portion of the file that has been written out of the buffer. Make your modifications from the beginning to the end of the file. To edit a part of the file that has already been written out of the buffer, use the REOPEN command (see "Specialized Commands" contained within this section) or reenter the editor. Both of these methods require lengthy disk access.

Command	Explanation
CL***n***	Moves cursor left ***n*** characters
CR***n***	Moves cursor right ***n*** characters
CU***n***	Moves cursor up ***n*** logical lines
CD***n***	Moves cursor down ***n*** logical lines
CBB	Moves cursor to beginning of buffer
CEB	Moves cursor to end of buffer
CBF	Moves cursor to beginning of file
CEF	Moves cursor to end of file
CBL	Moves cursor to beginning of the logical line
CEL	Moves cursor to end of the logical line
CC***n***	Moves cursor to column ***n*** (range 1-200)

Note: The notation ***n*** signifies an optional numeric argument, which usually acts as a repeat counter, with a range of 1-65535. With the exception of margin values, if ***n*** is omitted, the editor assumes a value of 1.

The error message CURSOR AT END is generated each time you attempt to position the cursor:

- Left, before the beginning of the buffer
- Right, past the end of the buffer
- Up, before the beginning of the buffer
- Down, past the end of the buffer

Note: Each time the editor generates this error message, it aborts the command line and enters immediate mode operation.

Use the cursor control movements to position the cursor at strategic locations to implement the more sophisticated commands available in the editor.

SEARCH COMMANDS

In the following commands, delimiters must be used to separate the string from the search command notation. You may either use the slash mark, /, or a set of quotation marks as delimiter characters. As an example, the SB/-/***n*** command explained below can also be entered as SB"-"***n***. You can use "wild cards" as a substitution for characters in a search string. The editor recognizes the inverse video question mark ? as a wild card that will match any character while searching. (To display any inverse video characters from the ATARI 800 keyboard, use the Λ key.)

Command	**Explanation**
SB/-/***n***	Search for ***n***th occurrence of string in buffer
SF/-/***n***	Search for ***n***th occurrence of string in file
SRB/-/-/***n***	Search and replace ***n*** times in buffer
SRF/-/-/***n***	Search and replace ***n*** times in file
SRVB/-/-/***n***	Search and replace with verify ***n*** times in buffer
SRVF/-/-/***n***	Search and replace with verify ***n*** times in file

Note: The notation ***n*** signifies an optional numeric argument, which usually acts as a repeat counter, with a range of 1-65535. With the exception of margin values, if ***n*** is omitted, the editor assumes a value of 1.

In general, all string searches begin after the current cursor location. In successful file and buffer searches, the cursor is positioned after the ***n***th occurrence of the string. The logical line containing the cursor is displayed as the first line of the text window.

In unsuccessful buffer searches, the editor:

- Retains the cursor in its original position
- Generates a SEARCH FAILED error message
- Aborts the command line
- Returns to immediate mode operation

In unsuccessful file searches, the editor repeatedly writes out the current buffer and reads a new buffer. If the end-of-file is reached before finding the ***n***th occurrence of the string, the search fails. Then, the editor:

- Retains the cursor in its original position **if** the last line in the file was already in the buffer before the search began **or**
- Positions the cursor to the beginning of the last buffer read **if** new lines were introduced from the file to the buffer area
- Follows the same procedure outlined above for unsuccessful buffer searches

In general, the **search and replace** commands, perform a search for the first string and replace it with the designated second string for the specified ***n*** times. The replacement string may be null or have a different amount of characters than the search string. Care should be taken to avoid the following conditions resulting in error messages. As with all error conditions, the editor aborts the command line and returns to immediate mode operation.

LINE TOO LONG

Could result if the insertion of a large replacement string into a text line exceeds the maximum line length.

Result of Operation: Only the first part of the replacement string would be inserted into the text.

LINE TOO LONG

Could result if the search string contains carriage returns. When a carriage return is deleted and the lines are concatenated, the resulting new line could exceed the maximum line length.

Result of Operation: The cursor is located to the right of a partial search string and that logical line is displayed as the first line of the text window.

CURSOR AT END

Could result if the search string terminates with a carriage return and is found on the last line of the buffer. Because the editor does not allow the last carriage return in the buffer to be deleted (except with a delete-line command), this search results in the given error message.

Result of Operation: The editor will find the string but abort the command line, resulting in no replacement.

In unsuccessful buffer searches, the editor:

- Retains the cursor in its original position if no replacement has been made **or**
- Positions the cursor after the last successful replacement **and**

  - generates a SEARCH FAILED error message
  - aborts the command line
  - returns to immediate mode operation

In unsuccessful file searches, the editor repeatedly writes out the current buffer and reads in a new one. If the end-of-file is found before the ***n***th occurrence of the search string, the command fails. Then, the editor follows the same procedure as outlined above for unsuccessful buffer searches.

Search and replace with verify commands for buffer and file use the same procedure as those respective commands without verification. Additionally:

- Before each replacement, the editor moves the cursor after the found search string and displays that logical line as the first line of the text window.
- A prompt question appears in the error window

  R Signifies replacement of the search string
  S Signals a "skip" of this occurrence
  Q Terminates or prematurely "quits" the search and replace command.

You may type your response in upper- or lowercase letters. If the response is valid, the editor clears the error window and completes the operation. If the response is invalid, your typed character is displayed; the cursor appears in the error window with a question mark.

BLOCK COMMANDS

You can manipulate a section of text lines by placing them within a defined **block**. To do this, you must precede and follow the designated text with a **block marker** that flags the attention of the editor and signals the beginning and end of the block. In the case of having more than two markers, the block is defined to be the group of text between the first encountered set of markers within the buffer.

Note: Do not use a line within your file that matches the block marker text designation.

Block markers are:

- A special text line displayed as follows

  ```
  ****BLOCK MARKER****↓
  ```

  that must be the **Only** text on the line itself
- Converted to a regular text line in the file by adding, deleting, or changing characters within the marker
- Automatically deleted from any text written out of the buffer

Although block commands offer timely execution of lengthy procedures, consider the limitations imposed upon their function by the system, itself. For example, error conditions can result if you attempt to move blocked text from the top of memory through insufficient available RAM. Also, exercise caution when you assign a filename specification within the parameters of a block-read or block-write operation so that you do not attempt to read or write to an already open file. Remember that the .BAK and .TMP file extensions are reserved for internal use by the editor. Also, if you wish to specify a drive number within a block-read or block-write operation, use the drive number D*n* designation within your filename string.

Command		Explanation
MS	Marker Set	Marker Set Inserts a block-marker text line **Before** the logical line containing the cursor
MC	Marker Clear	Marker Clear Removes all block-marker text lines and repositions the cursor to the beginning of the buffer

After setting the block markers, use the following commands to perform your intended operation.

Command		Explanation
BC	Block Copy	Copies the marked text before the line on which the cursor is positioned. The block markers are not copied with the text.
BM	Block Move	Moves a marked block of text before the logical line containing the cursor. The block markers are also moved.
BD	Block Delete	Deletes a marked block of text. The block markers are also deleted.
BP	Block Print	Prints the marked block on the system printer (P:). Expanding tabs and carriage returns are displayed as blanks.
BW/-/	Block Write	Writes the marked block to a disk file named within the delimiters. The block markers are not written to the file.
BR/-/	Block Read	Reads the disk file named within the delimiters and inserts that block before the logical line containing the cursor. Automatic paging will occur to read in the entire file if memory becomes full.

In general, the use of these commands:

- Positions the cursor at the beginning of the current line.
- Scrolls the screen until the cursor is on the first line of the text window.

Error messages that could be generated from an attempt of the above commands are:

MEMORY FULL	Results if there is not enough free memory to hold the entire block on a move or copy command.

Procedure for recovery from this condition is to:

- Use a BW/-/ command
- Position the cursor at the desired location for the block operation
- Use a BR/-/ command

An alternate recovery method is to:

- Use the REOPEN command
- Reposition the cursor at the desired location for the block operation
- Repeat the block command

I/O ERROR ***nnn***

May result during a print or write command. A standard ATARI operating system error number is given to aid you in isolating the problem.

In a print operation, the editor:

- Aborts the command line
- Returns to immediate mode operation.

In a write operation, the editor: closes the file.

INSERTING AND DELETING COMMANDS

Command	**Explanation**
IT/-/***n***	Inserts text string at the cursor location ***n*** times. If the cursor is past the last line in the buffer, the editor inserts a carriage return to the right of the cursor before inserting the text.
DB***n***	Deletes ***n*** characters before the cursor.
DA***n***	Deletes ***n*** characters after the cursor.
DF***n***	Deletes every character between the beginning of the logical line and the current cursor location. When the cursor position is immediately past a carriage return, the entire logical line is deleted except for the carriage return, itself. After this occurrence, the cursor moves to the beginning of this null line.

DL***n***

Deletes the logical line containing the cursor. After deletion the cursor moves before the first character of the next logical line.

RL

Inserts the text stored in the "recover-line" buffer in front of the line containing the cursor. Use this command to recover from accidental deletion of a line or to achieve a simple one-line move. You can insert text into the recover-line buffer by using a command or an immediate keystroke to delete a logical line.

Note: The notation ***n*** signifies an optional numeric argument, which usually acts as a repeat counter, with a range of 1-65535. With the exception of margin values, if ***n*** is omitted, the editor assumes a value of 1.

Error messages that could be generated from an attempt to use the above commands are:

MEMORY FULL

Results if too little free memory exists to allow for complete input of the string argument.

Result of Operation:

- Inserts either none, or only a part, of the string
- Aborts the command line
- Returns to immediate mode operation

LINE TOO LONG

Results from a deletion command if the editor deletes a carriage return and attempts to concatenate lines that will exceed the current line length limits. Also, this error condition results from text string insertion that causes maximum line length limits to be exceeded.

CURSOR AT END

Results from a deletion command if the cursor is at the **beginning** of the buffer when the editor attempts a deletion of characters **before** the cursor or if the cursor is at the **end** of the buffer when the editor attempts a deletion of characters **after** the cursor.

Note: To delete the last carriage return in the buffer, use the delete-line command.

SPECIALIZED COMMANDS

LM*n* and **RM*n***: Left and Right Margin Set Commands. If your television set needs adjustment to avoid cutting columns off of the display, change the left and right margins respectively by using these commands. Both margins are set a designated number of spaces dependent upon the value of ***n***. If you omit the designation for ***n***, the editor assumes a value of 1 for the left margin and a value of 40 for the right margin. The rule for setting the margin values is that the left margin must be greater than or equal to 1, but less than the value of the right margin. The right margin must be less than or equal to 40, but greater than the value of the left margin.

If you attempt a designation for ***n*** that is not in conformance with the margin rule, the editor generates the error message MARGIN VALUE ERROR, aborts the command line, and returns to immediate mode operation. This error condition also occurs if a new margin value causes existing command lines to exceed margin boundaries. Set automatic default values for both margins by using the customizing file.

CTS*n*: Convert Tabs to Spaces Command. Use this command to convert expanding tabs into spaces for a specified ***n*** of logical lines. If you omit the designation for ***n***, the editor assumes a value of 1. Error conditions can occur in two instances:

MEMORY FULL — This error is generated when the editor runs out of free memory during the conversion. A partially converted line may appear above the line that is being executed at the time of the error condition.

CURSOR AT END — This error is generated when the editor runs out of lines to convert in the buffer.

Error conditions cause the editor to abort the command line and return to immediate mode operation.

REOPEN: Reopen Editor With Same File. Use the REOPEN command to exit normally from the editor. The editor automatically reenters the same file, retains the original command line, and positions the cursor to the beginning of the file. Minimum growth factor determinations are made by the editor. The editor displays a warning message if the recalculated disk free space shows a limitation. You can choose to leave the editor or ignore the warning and continue with your editing session. When you reenter, the editor ignores all commands past the REOPEN on the current command line and empties both command line entries.

Use this command as a safety factor and backup procedure. Consistent and frequent implementation of the REOPEN command assures you the retention of your most current work in the event of an unforeseeable occurrence such as a power failure. Fifteen-minute interval "saves" are a common data processing practice.

Note: If you are using more than one disk drive, the editor switches source and destination drives each time you execute the REOPEN command.

PL*n*: Print ***n*** Lines on the System Printer. Use this command to print a specified number of lines on the system printer (P:). If ***n*** is not assigned, the editor assumes a value of 1. If you assign a value to ***n*** that is larger than the number of lines currently residing in the buffer, the editor automatically writes out the buffer and reads in a new one. If the editor encounters an end-of-file before the assigned number of ***n*** lines has been printed, the CURSOR AT END error message results. The editor aborts the command line and returns to immediate mode operation.

Printing starts from the logical line containing the cursor. Before printing the lines, all carriage returns and tab fields are changed to blanks.

Cursor positioning remains stationary unless the buffer is written out. The cursor moves to the beginning of any newly read buffer.

WL/-/*n*: Write ***n*** Lines to Disk File. Use this command to write a specified number of lines to the disk file designated within the delimiters. If ***n*** is not assigned, the editor assumes a value of 1. If you assign a value to ***n*** that is larger than the number of lines currently residing in the buffer, the editor automatically writes out the buffer and reads in a new one. If the editor encounters an end-of-file before the assigned number of ***n*** lines has been written, the CURSOR AT END error message results. The editor aborts the command line and returns to immediate mode operation.

Writing starts from the logical line containing the cursor. Cursor positioning remains stationary unless the buffer is written out. The cursor moves to the beginning of any newly read buffer.

Caution: Remember the editor reserves .BAK and .TMP extender designations. Do not attempt a write or read operation to an already open file.

LARGE FILE COMMANDS

You can edit a file that is too large to fit into available free RAM space by using two specialized commands formulated specifically for this purpose.

IH	inputs half the available RAM from the file
OC	outputs text up to the current position of the cursor

When the editor receives the IH command, its immediate response is to calculate available memory and input approximately half of that amount from the file into the buffer. After receiving the OC command, the editor outputs text from the beginning of the buffer up to the logical line containing the cursor. Thereafter, that logical line becomes the first line in the buffer. With combined use of these two commands, you can obtain free memory to successfully edit files larger than will fit into current memory. Error or warning conditions that can occur include:

INPUT EOF	The editor reaches the end of the specified input file.
I/O ERROR ***nnn***	A fatal disk or printer error occurs.
LINE TOO LONG	The editor encounters a line that exceeds the maximum line length set by the customizing file.

CANNOT-PREVIOUS DISK I/O ERROR	The editor cannot perform an intended function because of a previous error condition.

TABLE 1 — IMMEDIATE MODE RESERVED KEYSTROKES

CTRL ←	Move cursor left (skip across expanding tabs)
CTRL →	Move cursor right (skip across expanding tabs)
CTRL ↓	Move cursor down one physical line
CTRL ↑	Move cursor up one physical line
CTRL 2	Move cursor to beginning of logical line
CTRL 3	Move cursor to end of logical line
CTRL 8	Display previous screen of characters
CTRL 9	Display next screen of characters
regular keys	Insert character into text
SHIFT INSERT	Prepare to insert new line(s)
TAB	Tab to next tab stop
CTRL RETURN	Return with auto indent to same level
DELETE BACK S	Delete character left of cursor
CTRL DELETE BACK S	Delete character right of cursor
SHIFT DELETE BACK S	Delete logical line containing cursor
CTRL TAB	Toggle visible-tab mode (if expanding tab option selected)
SHIFT TAB	Toggle visible-carriage return mode
SHIFT CLEAR	Clear error window
START	Execute command window
SELECT	Select alternate command line
OPTION	Change mode
BREAK	Abort command being executed

TABLE 2 — COMMAND MODE INSTRUCTIONS

EXIT	Exit normally from edit - return to DOS
EXIT2	Exit normally from edit - restart editor
ABORT	Exit without saving changes - return to DOS
ABORT2	Exit without saving changes - restart editor
SB/-/***n***	Search for ***n***th occurrence of string in buffer
SF/-/***n***	Search for ***n***th occurrence of string in file
SRB/-/-/***n***	Search and replace ***n*** times in buffer
SRF/-/-/***n***	Search and replace ***n*** times in file
SRVB/-/-/***n***	Search and replace with verify ***n*** times in buffer
SRVF/-/-/***n***	Search and replace with verify ***n*** times in file
MS	Marker set
MC	Marker clear
BC	Block copy
BM	Block move
BD	Block delete
BP	Block print
BW/-/	Block write to disk file
BR/-/	Block read from disk file
CL***n***	Move cursor left ***n*** characters
CR***n***	Move cursor right ***n*** characters
CU***n***	Move cursor up ***n*** logical lines
CD***n***	Move cursor down ***n*** logical lines
CBB	Move cursor to beginning of buffer
CEB	Move cursor to end of buffer
CBF	Move cursor to beginning of file
CEF	Move cursor to end of file
CBL	Move cursor to beginning of logical line
CEL	Move cursor to end of logical line
CC***n***	Move cursor to column ***n***
IT/-/***n***	Input string ***n*** times at cursor position
DB***n***	Delete ***n*** characters before cursor
DA***n***	Delete ***n*** characters after cursor
DF	Delete first part of logical line
DR	Delete remainder of logical line
DL***n***	Delete ***n*** logical lines
RL	Recover last deleted line
IH	Input half of available RAM from file
OC	Output text to file up to line containing cursor

REOPEN	Reopen editor with same file
PL***n***	Print ***n*** lines on system printer
WL/-/***n***	Write ***n*** lines to disk file
CTS***n***	Convert expanding tabs to spaces for ***n*** lines
LM***n***	Set left margin to width ***n***
RM***n***	Set right margin to width ***n***

Note: ***n*** is an optional numeric argument, which usually acts as a repeat counter, with a range of 1-65535. With the exception of margin values, if ***n*** is omitted, the editor assumes a value of 1.

Note: "/-/" is a required character string delimited by either a pair of slashes or a pair of quotes. "/-/-/" is a pair of required strings delimited by either a triplet of slashes or a triplet of quotes.

3

CUSTOMIZING THE EDITOR

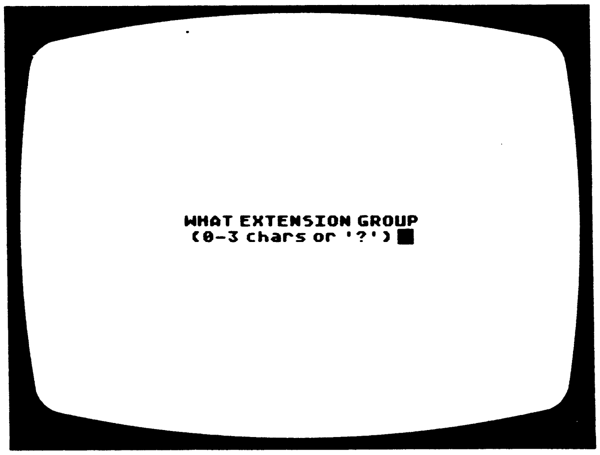
You can use the editor to full advantage by establishing specific parameters to handle distinct file extensions. For example, you may wish to turn off the auto-indention feature on all file extensions except for languages similar to PASCAL. Maximum line lengths for .ASM files are different than, for instance, .BAS and should be altered from the default value. By setting a customizing screen color you can visually determine the nature of your editing file.

Customizing file alterations use the BASIC programming language. You must have an ATARI BASIC language cartridge inserted into the left slot of your computer console. Refer to the *ATARI 800 Operators Manual* for instruction in installing a cartridge.

To load the customizing file:

1. Place the diskette containing the editor program into your disk drive.
2. Turn on the computer. Wait for the READY message prompt with the cursor to appear on the screen.
3. Type **RUN "D:MEDITCM.BAS"** and press RETURN.
4. Remove the program diskette and insert your data diskette.

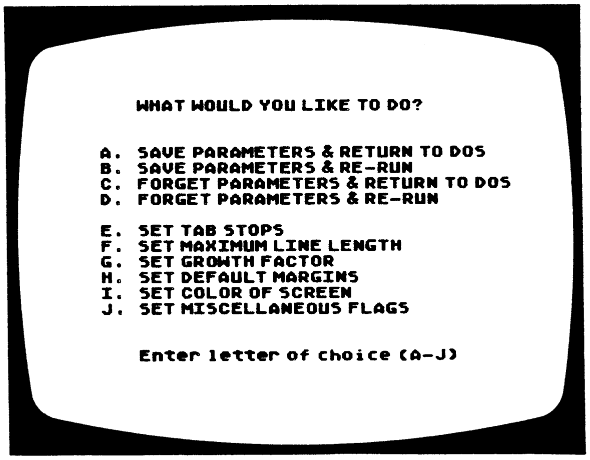
The Customizing File Menu that appears will allow you to select the area in which you wish to change the default values. Most of the selections are self-documented. You can reference the instructions included in the software program or you can type **N** in response to the WOULD YOU LIKE INSTRUCTIONS? (Y/N) query and use this manual.



```
WHAT EXTENSION GROUP
(0-3 chars or '?') ■
```

Figure 10 Extension Group Prompt

Refer to Figure 11. By answering the WHAT EXTENSION GROUP query, you establish the filename specification extension that you wish to customize. Enter the ? to return to the instructions for use of the file.



```
WHAT WOULD YOU LIKE TO DO?

A. SAVE PARAMETERS & RETURN TO DOS
B. SAVE PARAMETERS & RE-RUN
C. FORGET PARAMETERS & RETURN TO DOS
D. FORGET PARAMETERS & RE-RUN

E. SET TAB STOPS
F. SET MAXIMUM LINE LENGTH
G. SET GROWTH FACTOR
H. SET DEFAULT MARGINS
I. SET COLOR OF SCREEN
J. SET MISCELLANEOUS FLAGS

Enter letter of choice (A-J)
```

Figure 11 Customizing File Menu

A-D—Parameters

The Customizing File Menu appears as soon as you have answered the extension group prompt. Depending upon your choice of parameters, all changes that you enter into the customizing file will be retained as new values for your selected group. Use the first four fields of the menu, A-D, to establish the changes or to disregard them. Selection of **A** or **C** returns control to DOS for easy access into the editor. Selection of **B** or **D** reruns the customizing file.

E—Set Tab Stops

Selection **E** allows you to set your tab stop values. As the software instructions indicate, tab stop values cannot be changed during an editing session. The screen displays default and current tab stop values. A (Menu, Set, Clear) Select Item prompt appears on the screen.

M Reruns the Customizing File Menu

S Brings **(2-199) What column to set** onto the screen. Choose the column tab stop by pressing the number combination followed by RETURN. All current tab stop values will be redisplayed. Note the inclusion of your new value.

C Brings **(2-199 or *) What column to clear** onto the screen. Choose the column tab stop by pressing the number combination followed by RETURN. All current tab stop values will be redisplayed. Note the exclusion of your new value.

Press the "*" key to clear all tab stop values. Wait for the Select-Item prompt to appear. Use the Set command to enter new values.

F—Set Maximum Line Length

Maximum line length defaults to 114 characters or 3 physical lines in conformance with the ATARI Computer's built-in operating system screen editor. The editor allows from 2 to 200 characters per logical line. Enter your chosen value and press RETURN. Control automatically returns to the Customizing File Menu.

G—Set Minimum Growth

Use this command to determine your space allocation before receiving an I/O ERROR 162 (disk full) error message. You can ignore the minimum growth check warning and proceed with your editing session. However, be mindful of its usefulness as a warning device. Enter your chosen value and press RETURN. Control automatically returns to the Customizing File Menu.

H—Set Default Margins

If display columns are being cut off at the sides of your television screen, you can change the left and right margins.

I—Set Color of Screen

Using the customizing file, you can alter three variables that precisely determine the color display. The first variable, indicated as COLOR, controls the background color selection. Refer to the following table for numbers corresponding to the color of your choice.

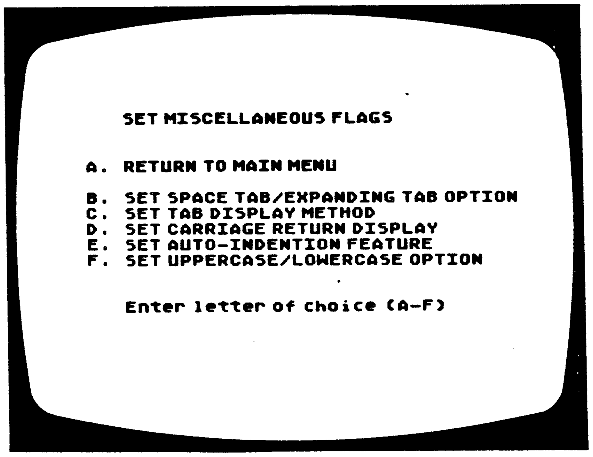
TABLE 3—THE ATARI COLORS AND NUMBERS

BACKGROUND COLORS	**CORRESPONDING NUMBERS**
GRAY	0
LIGHT ORANGE (GOLD)	1
ORANGE	2
RED-ORANGE	3
PINK	4
PURPLE	5
PURPLE-BLUE	6
AZURE BLUE	7
SKY BLUE	8
LIGHT BLUE	9
TURQUOISE	10
GREEN-BLUE	11
GREEN	12
YELLOW-GREEN	13
ORANGE-GREEN	14
LIGHT ORANGE	15

Note: Colors will vary with type and adjustment of television or monitor used.

The second variable, B-lum, controls the luminance of the background color on the screen. The third variable, C-lum, controls the character luminance. Luminance is changed on every **even** number: 0, 2, 4, 6, 8, 10, 12, and 14. Follow certain rules when assigning luminance numbers to ensure a useable combination. To obtain the best clarity and avoid the occurrence of a blank screen:

- Do not equate the luminance values for the two variables, B-lum and C-lum.
- The two luminance values must be greater or less than each other by a factor of 8.



```
     SET MISCELLANEOUS FLAGS

A.   RETURN TO MAIN MENU

B.   SET SPACE TAB/EXPANDING TAB OPTION
C.   SET TAB DISPLAY METHOD
D.   SET CARRIAGE RETURN DISPLAY
E.   SET AUTO-INDENTION FEATURE
F.   SET UPPERCASE/LOWERCASE OPTION

     Enter letter of choice (A-F)
```

Figure 12 Customizing File Submenu J

J—Set Miscellaneous Flags

A—Return to Main Menu

Return to the main menu after choosing your new values. You may then make a selection to retain or disregard the parameters you have selected.

B—Set Type of Tab

Note: Not all ATARI software recognizes expanding tabs.

Space tabs insert a selected number of blanks between tab stops, and the cursor positions accordingly. Expanding tabs, however, insert a character into the text that indicates the tab function.

C—Set Tab Display Method

Expanding tabs can be conventionally displayed as spaces or usefully displayed as right triangles followed by periods. The value entered into this parameter can be displaced by an immediate mode keystroke.

D—Set Carriage Return Display

A carriage return can either be displayed as a space or a down arrow. The value entered into this parameter can be displaced by an immediate mode keystroke.

E—Auto-Indention Feature

Auto-indention allows you to reposition the cursor to an automatic tab stop on the next logical line. To activate auto-indention, you press the CTRL and RETURN keys simultaneously. Use the customizing file to disengage this feature.

F—Set Shifting Caselock

After you answer the filename prompt and begin the edit of your specified file, this option comes into effect. Set a shift-lock for uppercase designation or a no-lock for upper- and lowercase. The parameter value entered may be displaced by using the CAPS LOWR key during an editing session.

EDITOR MESSAGES

Warnings

USING DEFAULTS
No customizing file was found that matched the extension of the filename, so the editor uses its built-in defaults.

NEW FILE
The file named to be edited does not exist; therefore, the editor creates a new file using the specified name given at the prompt.

INPUT EOF
The end of file has been reached on the input file.

CANNOT-PREVIOUS DISK I/O ERROR
Refer to a previous execution for cause of error. Use of EXIT, IH, or OC commands may be restricted.

CURSOR AT END
Occurs whenever the cursor tries to move past either end of the text buffer.

Prompt Messages

VERIFY(REPLACE, SKIP,QUIT)?
Displays in the error window before each replacement while executing a search-and-replace-with-verify command.

Error Messages

MARGIN VALUE ERROR
Occurs when a designation for ***n*** is not in conformance with the margin rule.

LINE TOO LONG
Occurs whenever the addition of text to the current line causes it to exceed the maximum line length set by the customizing file.

MEMORY FULL
Means that there is not enough free RAM in the buffer to carry out the operation.

ILLEGAL DEV:FILE.EXT
Tells you that the last filename prompt was incorrectly answered.

DELIMITER ERROR
Tells you that the command being entered into the command window requires a slash (/) or double quotation mark (") for proper syntax.

SEARCH FAILED
Occurs when a search command was executed and the search string could not be found.

NOT COMPLETE
Occurs when you try to execute the command window when an incomplete command line exists there.

UNRECOGNIZED COMMAND
Occurs when you type an invalid character into the command window.

BREAK KEY ABORT
Acknowledges that you have pressed the BREAK key during execution of the command window.

I/O ERROR ***nnn***
Tells you that a fatal disk or printer error has occurred. ***nnn*** is an error number generated by the operating system. Refer to the *ATARI Disk Operating System II Reference Manual.*

NUMBER TOO BIG
Tells you that the argument ***n*** given in the command window is too large for the command specified or the current line length limit.

CANNOT FIND MARKED BLOCK
Means that the editor could not find a marked block of text while executing a BC, BD, BM, BP, BW/-/, or BR/-/ command.

CANNOT FIND FILE
Means that the editor could not find the file requested in a BR/-/ command.

EDITOR IS CONFUSED
Occurs when internal editing pointers have been damaged. Try immediate mode keystrokes CTRL **2** and CTRL ← until you no longer receive this error message. (If this error should occur, it would be helpful to us if you could find a repeatable sequence of events that reproduces it and report to ATARI Customer Service.)

FILE LOCKED
Means that the file you requested to edit is locked or the associated .BAK or .TMP file is locked.

EDITOR CANNOT RUN-NO FREE IOCBs
Occurs if you have attempted to use any other DOS but 2.0S; or if you have called the editor directly, and at least three available IOCBs do not exist.

Note: *n* is an optional numeric argument, which usually acts as a repeat counter, with a range of 1-65535. With the exception of margin values, if ***n*** is omitted, the editor assumes a value of 1.

LIMITED WARRANTY ON MEDIA AND HARDWARE ACCESSORIES.

We, Atari, Inc., guarantee to you, the original retail purchaser, that the medium on which the APX program is recorded and any hardware accessories sold by APX are free from defects for thirty days from the date of purchase. Any applicable implied warranties, including warranties of merchantability and fitness for a particular purpose, are also limited to thirty days from the date of purchase. Some states don't allow limitations on a warranty's period, so this limitation might not apply to you. If you discover such a defect within the thirty-day period, call APX for a Return Authorization Number, and then return the product along with proof of purchase date to APX. We will repair or replace the product at our option.

You void this warranty if the APX product: (1) has been misused or shows signs of excessive wear; (2) has been damaged by use with non-ATARI products; or (3) has been serviced or modified by anyone other than an Authorized ATARI Service Center. Incidental and consequential damages are not covered by this warranty or by any implied warranty. Some states don't allow exclusion of incidental or consequential damages, so this exclusion might not apply to you.

DISCLAIMER OF WARRANTY AND LIABILITY ON COMPUTER PROGRAMS.

Most APX programs have been written by people not employed by Atari, Inc. The programs we select for APX offer something of value that we want to make available to ATARI Home Computer owners. To offer these programs to the widest number of people economically, we don't put APX products through rigorous testing. Therefore, APX produts are sold "as is", and we do not guarantee them in any way. In particular, we make no warranty, express or implied, including warranties of merchantability and fitness for a particular purpose. We are not liable for any losses or damages of any kind that result from use of an APX product.

ATARI®

PRINTED IN U.S.A.

A Warner Communications Company

APX ATARI® PROGRAM EXCHANGE
P.O. Box 3705
Santa Clara, CA 95055

Review Form

We're interested in your experiences with APX programs and documentation, both favorable and unfavorable. Many of our authors are eager to improve their programs if they know what you want. And, of course, we want to know about any bugs that slipped by us, so that the author can fix them. We also want to know whether our instructions are meeting your needs. You are our best source for suggesting improvements! Please help us by taking a moment to fill in this review sheet. Fold the sheet in thirds and seal it so that the address on the bottom of the back becomes the envelope front. Thank you for helping us!

1. Name and APX number of program.

____

____

2. If you have problems using the program, please describe them here.

____

____

____

3. What do you especially like about this program?

____

____

____

4. What do you think the program's weaknesses are?

____

____

____

5. How can the catalog description be more accurate or comprehensive?

____

____

6. On a scale of 1 to 10, 1 being "poor" and 10 being "excellent", please rate the following aspects of this program:

_______ Easy to use
_______ User-oriented (e.g., menus, prompts, clear language)
_______ Enjoyable
_______ Self-instructive
_______ Useful (non-game programs)
_______ Imaginative graphics and sound

7. Describe any technical errors you found in the user instructions (please give page numbers).

8. What did you especially like about the user instructions?

9. What revisions or additions would improve these instructions?

10. On a scale of 1 to 10, 1 representing "poor" and 10 representing "excellent", how would you rate the user instructions and why?

11. Other comments about the program or user instructions:

From

STAMP

ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

[seal here]

TEXT FORMATTER (FORMS)

by

Dale Yocum

CONTENTS

INTRODUCTION

OVERVIEW

Welcome to the world of word processing! Once you experience it, you'll never want use a typewriter again. In fact, you might as well give yours to the kids for keyboard practice so they can use your computer when they're older. Or, you can donate it to your favorite charity, if you can find a charity that still uses typewriters.

Now you can alter your layout as much and as often as you like without having to retype your text. Such a tool might even remove an occasional writer's block!

With FORMS and either a line-oriented or full-screen text editor, you can create reports, letters, novels, or any other printed matter as quickly as your thoughts spill out on the keyboard. (These instructions explain how to use both kinds of text editors with FORMS.) And if you're like many users, you'll even start enjoying formerly boring or formidable writing tasks.

The procedure is simple. First, you load your text editor and create your input file of text. You intersperse FORMS control words in this file to indicate whatever formatting you want. For example, you enter the control word .I5 on a line to indent subsequent text five spaces from the left-hand margin. Next, you store your file on diskette or cassette. Then, to print a formatted copy of your file, you load FORMS and specify the name of your input file in response to a prompt. That's all there is to it! FORMS takes over and prints your formatted text. Whenever you want to revise your input file, you follow the same procedure, except that you load in your existing file after loading in your text editor.

All FORMS control words consist of a period in column one, followed by an uppercase or lowercase letter (FORMS accepts both), and sometimes by a number. You enter one control word per line, and usually your control words are on separate lines from your text. Some of the formatting possible via FORMS incudes specifying page height, line spacing, line length, left-hand margin, paragraph indention, running top title, text centering, and escape codes to change character sets on the ATARI 825 Printer. You can also change many of these settings at any point in your text. Thus, for example, you can double-space the body of your text but indent and single-space quotations.

At the point of printing a copy, FORMS asks you whether you want to print your entire document (data file) or only selected pages. This feature is very convenient when you revise only one section or a few pages of a large document. Another control word permits FORMS to support both separate sheets and continuous forms.

Experience has shown that, although users start out using very simple formatting to become familiar with a text formatter, sooner or later they turn to some of the formatter's fancier features. FORMS has enough control words to give you lots of ways to experiment! FORMS together with a text editor constitutes a good all-purpose word processing package. These user instructions were formatted through FORMS and printed on an ATARI 825 80-column printer.

REQUIRED ACCESSORIES

16K RAM for cassette version
24K RAM for diskette version
ATARI BASIC Language Cartridge
A compatible text editor
ATARI 410 Program Recorder for cassette
ATARI 810 Disk Drive for diskette
ATARI printer or equivalent printer

A SAMPLE SESSION

Before describing the FORMS control words and features in detail, let's run through a sample session. We'll create a text file using the ATARI Assembler Editor cartridge and store the file on diskette.

After turning on our disk drive, we insert the Assembler Editor cartridge, power up our computer, and insert the diskette on which we want to store our file at the end of the session.

The Assember Editor program uses a line-oriented text editor, and so we'll type NUM to cause the editor to generate a line number for each line of text. Next, we key in our text. The sample appears on the next page.

SAMPLE INPUT FILE

```
10 .W75
20 .BTEST FILE
30 .P
40 The text file, created using a
50 separate editor, contains both textual
60 material, plus special control words to
70 the FORMS program.
80 A line beginning with a period is
90 assumed to be a control word and must match
100 one of the control words recognized by FORMS.
110 An example is:
120 .L1
130 .I15
140 Indent this line 15 spaces
150 .L1
160 .I0
170 and then return to no indention.
```

The output file would then look like this:

```
                              TEST  FILE

     The text file, created using a separate editor, contains both textual
material, plus special control words to the FORMS program. A line
beginning with a period is assumed to be a control word and must match one
of the control words recognized by FORMS. An example is:

               Indent this line 15 spaces

and then return to no indention.
```

In this sample, we used the control word .W to set the right margin at column 75 and .B to center a title using an elongated character set available on the ATARI 825 printer. Next, we used the control word .P to create a paragraph that skips one line and indents five spaces. Then, midway through the paragraph, we used the control word .L1 to skip 1 line and the control word .I15 to indent some text. After the indented line, we skipped another line with .L1 and we used .I0 to reset our regular left-hand margin to its former column position.

If this were all we wanted to write today, we would then store it on cassette or diskette. Since we're using the Assembler Editor cartridge, we'll use the command LIST#D:SAMPLE to name and store our sample file on diskette.

We're now ready to print a formatted copy of the file. We turn on our printer and interface module. Because we have the diskette version of FORMS, we load it into RAM by typing RUN "D:FORMS" and pressing the RETURN key. FORMS first prompts us for the name of our text file:

```
TEXT FILE NAME?
```

We respond by entering the name, SAMPLE in this case. Next, FORMS asks us on what printed page number we want to begin printing:

```
          WHAT PAGE SHALL I START PRINTING
          WITH (HIT RETURN TO PRINT ENTIRE
          DOCUMENT)?
```

Obviously, our sample will fill only part of the first page, so we can simply press the RETURN key to cause FORMS to print the entire file. FORMS then formats and prints our input file. You'll see the input file scroll on your video screen as FORMS reads through your file.

At the end of the video display, FORMS lists the number of SOURCE ERRORS encountered in the file. These are lines that began with a period, alerting FORMS to a control word, but that FORMS couldn't understand or carry out. In addition, if our text contains text formatting errors, in each instance FORMS prints (and also displays on the video screen) the last control word it read and then prints the SOURCE ERROR message. For example, if you type .IO (the letter "O") instead of .I0 (the number zero) to reset your left-hand margin, then FORMS displays the message:

```
          *** SOURCE ERROR, LAST COMMAND: ***
          *** .IO ***
```

Those are the steps for using FORMS with a line-oriented text editor. Using a full-screen text editor involves the same steps, except your text lines don't begin with line numbers. For more information about these two types of text editors, see the section titled "USING LINE-ORIENTED AND FULL-SCREEN TEXT EDITORS" later in this manual.

SUMMARY OF STEPS--TEXT EDITING

USING THE ATARI ASSEMBLER EDITOR CARTRIDGE

These are the steps you normally follow to create or revise a text file with the text editor on the ATARI Assembler Editor Computing Language Cartridge (CXL4003). If you're using another text editor, then skip this section.

1 LOAD YOUR TEXT EDITOR INTO RAM

a. If you're planning to store your text file on diskette, turn on your disk drive and insert the diskette

b. Insert the Assembler Editor cartridge in the (Left Cartridge) slot of your computer.

c. Power up your computer and turn on your video screen. You'll be in EDIT mode--ready to create a text file or load one in from diskette or cassette.

d. See the Assembler Editor User's Manual (Part No. C014189-03 REV. 1), 1981, Chapter 3, "Using the Editor", pp. 15 - 23, for more information about using this editor.

2 LOAD YOUR TEXT FILE INTO RAM (to work on an existing file)

Type ENTER#D:filename to load a diskette file (replacing "filename" with the name of your text file in upper case) or ENTER#C: to load a cassette file (making sure you have the tape rewound to the beginning.)

3 CREATE/REVISE YOUR TEXT FILE

a. Consult the user manual for for the editing functions available to create and revise your text file.

4 STORE YOUR TEXT FILE

Use the command LIST#D:filename to store your text file on diskette, or use the command LIST#C: to store your text file on cassette (be sure you rewind the tape to the beginning).

USING THE ATARI PROGRAM TEXT EDITOR

These are the steps you normally follow to create or revise a text file with the ATARI Program Text Editor (under development). If you're using another text editor, then skip this section.

1 LOAD YOUR TEXT EDITOR INTO RAM

a. Turn on your disk drive and insert the Program Editor diskette.

b. Power up your computer and turn on your video screen.

c. Call DOS and use menu selection L (Binary Load) to load the text editor into RAM.

2 LOAD YOUR TEST FILE INTO RAM (to work on an existing file)

Enter the name of your text file on diskette in response to the FILENAME? prompt (e.g. SAMPLE). If you're using more than one disk drive, include the device initial (D) and the number of the drive containing your text file, in the format D2:SAMPLE .

3 CREATE/REVISE YOUR TEXT FILE

Consult the user manual for the editing functions available to create and revise your text file.

4 STORE YOUR TEXT FILE

The EXIT command automatically causes the editor to store your text file on diskette under the name you specified when you entered the editor.

USING ANOTHER TEXT EDITOR

These are the steps you normally follow to create or revise a text file with a text editor other than the editor on the ATARI Assembler Editor Computing Language Cartridge or the ATARI Program Text Editor (under development). If you're using on the these latter text editors, then skip this section.

1 LOAD YOUR TEXT EDITOR INTO RAM

For loading instructions, consult the user manual for your text editor.

2 LOAD YOUR TEXT FILE INTO RAM (to work on an existing file)

For text file loading instructions, consult the user manual for your text editor.

3 CREATE/REVISE YOUR TEXT FILE

a. Consult the user manual for your text editor for the editing functions available to create and revise your text file.

4 STORE YOUR TEXT FILE

For instructions on how to store text files, consult the user manual for your text editor.

SUMMARY OF STEPS--USING FORMS

DISKETTE FILES

These are the steps you normally follow to print a formatted version of your text file when FORMS and your input file are on diskette. If you're using cassettes, then skip this section.

5 LOAD THE TEXT FORMATTER (FORMS) INTO RAM

a. Insert the ATARI BASIC Language Cartridge in the (Left Cartridge) slot of your computer.

b. Turn on your printer. If you're using an ATARI 825 Printer, then also turn on your ATARI 850 Interface Module.

c. Turn on your disk drive and insert the FORMS diskette.

d. Power up your computer and turn on your video screen (if these aren't already on).

e. When the READY prompt displays, type RUN "D:FORMS" and press the RETURN key. If you're using more than one disk drive, remember to follow the device initial (D) with the number of the drive containing FORMS (e.g., RUN "D2:FORMS" for disk drive two). FORMS will load into RAM and start.

6 PREPARE YOUR TEXT FILE FOR AUTOMATIC LOADING BY FORMS

a. If you're using only one disk drive, remove the FORMS diskette after it has loaded and insert your text file diskette. Otherwise, insert your text file in another disk drive.

7 ANSWER THE FORMS PROMPTS FOR FILENAME AND STARTING PAGE

a. When you start the FORMS program, you'll see the prompt TEXT FILE NAME? . Enter the file name (e.g., SAMPLE). If your diskette is not on disk drive one, include the device initial (D) and the number of the disk drive containing your file (e.g., D2:SAMPLE for disk drive two).

b. The second prompt asks you on what printed page number you want FORMS to begin printing. The prompt is WHAT PAGE SHALL I START PRINTING WITH (HIT RETURN TO PRINT ENTIRE DOCUMENT)? . To print the entire file, simply press the RETURN key. To begin printing on another page, enter the page number and press the RETURN key (e.g. 4 <CR>).

If you specify a number, FORMS will also ask you the output page number after which it is to stop printing. The prompt is WHAT IS THE LAST PAGE YOU WANT TO PRINT? . To print to the end of the file, simply press the RETURN key. To stop printing after a particular page, enter the number of the last page you want printed and press the RETURN key.

CASSETTE FILES

These are the steps you normally follow to print a formatted version of your text file when FORMS and your input file are on cassette. If you're using diskettes, then skip this section.

5 LOAD THE TEXT FORMATTER (FORMS) INTO RAM

a. Insert the ATARI BASIC Language Cartridge in the (Left Cartridge) slot of your computer.

b. Insert the FORMS cassette in the program recorder, press REWIND, and then press PLAY.

c. Type CLOAD and press the RETURN key twice.

d. After FORMS loads into RAM, you'll see the READY prompt. Type RUN and press the RETURN key.

6 PREPARE YOUR TEXT FILE FOR AUTOMATIC LOADING BY FORMS

a. Remove the FORMS cassette, insert the text file cassette, press REWIND, and then press PLAY.

7 ANSWER THE FORMS PROMPTS FOR FILENAME AND STARTING PAGE

a. When you start the FORMS program, you'll see the prompt TEXT FILE NAME? . Type C: in response to the prompt and press the RETURN key twice.

b. The second prompts asks you on what printed page number you want FORMS to begin printing. The prompt is WHAT PAGE SHALL I START PRINTING WITH (HIT RETURN TO PRINT ENTIRE DOCUMENT)? . To print the entire file, simply press the RETURN key. To begin printing on another page, enter the page number and press the RETURN key (e.g. 4 <CR>). Answer these prompts quickly, as the tape is rolling.

If you specify a number, FORMS will also ask you the output page number after which it is to stop printing. The prompt is WHAT IS THE LAST PAGE YOU WANT TO PRINT? .
To print to the end of the file, simply press the RETURN key. To stop printing after a particular page, enter the number of the last page you want printed and press the RETURN key.

LINE-ORIENTED AND FULL-SCREEN TEXT EDITORS

Most text editors are either line-oriented or full-screen. By line-oriented, we mean all the lines of your text file begin with numbers and you sometimes use these numbers with editing commands to revise your text. By full-screen, we mean your lines of text contain no initial line numbers, and generally you use directional arrow keys to move around the screen to add and revise your text.

HOW YOUR TEXT EDITOR INFLUENCES FORMS

FORMS assumes you're using a line-oriented text editor. In this mode, FORMS reads the number beginning each line until it reaches a space or a character other than a digit. It then skips over the line number and one space, if present, and interprets the rest of each line for control words and text. If more than one space follows a line number, FORMS preserves these additional spaces. If your text immediately follows the line number (with no intervening space), FORMS retains the first text character.

If FORMS doesn't encounter a number at the beginning of the first input line, it assumes your text editor is full-screen and it uses everything on the line.

RESETTING THE TEXT EDITOR FLAG

FORMS sets a flag when it determines the mode of your text editor. The flag is preset to line mode. However, the first time it encounters a line without a beginning number, the flag flips over permanently to full-screen mode and FORMS thereafter uses everthing on each line. In the same manner, if your first line doesn't begin with a number, then FORMS assumes you're using a full-screen editor and it permanently flips over the flag to full-screen mode at the start of your input file.

STARTING YOUR INPUT FILE WITH A NUMBER WHEN USING A FULL-SCREEN TEXT EDITOR

Suppose you're using a full-screen editor, but you want to begin your text with a number (a date, for example). You don't want FORMS to interpret this number as a line number, and therefore ignore it. To signal FORMS to print this number, imbed the command .L as the first input line (this command tells FORMS to start a new line) and enter your number as the second input line. For example, to begin your text with the date 5/01/81, enter it this way:

```
.L
5/01/81
```

GENERAL NOTES ON USING FORMS

ENTERING FORMS CONTROL WORDS

All FORMS control words begin with a period in column one, followed by one letter, which can be upper case or lower case. For example, .S is the control word to tell FORMS to begin a new page. You can also enter this as .s . Many control words also contain a number. For example, .I5 tells FORMS to indent the text on subsequent lines five spaces from the left-hand margin. The control word descriptions indicate which control words use numbers.

You enter these control words right along with your text, at whatever point you want to change the layout. You place some control words on the same line as the affected text. For example, to center a line of text (the control word is .C), you enter the control word and the text to be centered as one line:

```
.CCENTER THIS LINE
```

In other cases, you place a control word on a line separate from the text it controls. For example, to tell FORMS to move to column 20 of the current line and print text, you use .M20 and you place on the next line the text to print starting in column 20:

```
.M20
TITLES/CATEGORY
```

The format of each control word in the descriptions indicates whether the control word belongs by itself or with accompanying text.

Enter one control word per line. Thus, if you want to skip a line with the .L1 control word and indent text five spaces with the .I5 control word, you enter each control word on a separate line:

```
.L1
.I5
```

LINE FEEDS, LINE FILL, AND BREAKS

FORMS fills up a line to the default or user-set line length by adding successive words from the input file until one more word would cause FORMS to exceed the right-hand margin. However, some control words cause an automatic break in filling up lines, such that FORMS starts a new line with the text following these control words regardless of the space remaining in the current line. For example, the control word to start a new line, .L , forces FORMS to end the current line and begin printing subsequent text on a new line. The individual descriptions indicate when a control word causes a break.

TELLING FORMS TO START A NEW LINE

Sometimes you'll want to start a new line at a specific point in your text, even though room remains on the current line. To tell FORMS to break and begin a new output line, use the control word .L and then continue your text on the next line. For example, to end the line containing the text "looks like the following:" and tell FORMS to put the next two input lines (for example, (1) ENTER TEXT and (2) WHEN READY) on two separate output lines, you would enter your text as follows:

```
looks like the following:
.L
ENTER TEXT
.L
WHEN READY
```

Your output text will be:

```
looks like the following:
ENTER TEXT
WHEN READY
```

Alternatively, you can begin each line with a comma that you want to print on a new line. With this method, the same input text would look like this:

```
looks like the following:
,ENTER TEXT
,WHEN READY
```

UNDERLINING

A nice feature of ATARI 825 printers is their ability to handle underlining. FORMS doesn't control underlining, but we mention the method here because underlining enters into many aspects of text formatting.

To underline any block of text, use the CTRL-O (the letter "O") code at the beginning of the block and the CTRL-N code at the end of the block. Sometimes you'll want to underline indented text that takes up more than one output line. To prevent the printer from extending the underline from the left-hand margin of these subsequent lines, break your text into blocks comprising one line each and place the CTRL-O and CTRL-N codes around each block. (Usually you'll need to print at least one draft to determine these blocks and then modify these lines for the next printing.)

CONTROL WORD DESCRIPTIONS

CENTER TEXT IN BOLDFACE

Format: .B<text>

Example: .BDESCRIPTIONS

Use this control word to center the text immediately following it, using an elongated character set, which resembles boldface. This control word is usable only on the ATARI 825 Printer, which supports several character sets. The text will center between the current indent value (.I) and the current maximum line length (.W). This control word causes a break.

Remember that you may use only half as many characters per line with an elongated character set as you're using in your normal character set. For more information about elongated character sets, see the ATARI 825 80-Column Printer Operator's Manual, 1980.

The example would print as:

DESCRIPTIONS

CENTER TEXT

Format: .C<text>

Example: .CTable of Contents

Use this control word to center text within the current indent value (.I) and maximum line length value (.W). The text following .C may be no longer than the number of spaces between these two values. For example, if your current indent value is 10 and your line length is 80, then the line of text can't exceed 70 characters. This control word causes a break.

In the example, the text Table of Contents would appear as follows:

Table of Contents

TURN OFF LINE FILL MODE

Format: .D

Use this control word to turn off line fill temporarily so that the end of each output line corresponds exactly to the end of each input line. The default mode is to fill lines (.DO). This control word affects only line filling. FORMS still uses any indent value and centering control words you enter. This control word causes a break.

Because creating tables with multiple columns is difficult in FORMS, you might want instead to use this control word to turn off line filling and enter your columns as you want them to appear on output. To turn the line fill function back on, use .D0 , described below.

Note. Beginning users tend to use .D instead of using other control words that would save time and keystrokes. This control word is seldom necessary.

TURN ON LINE FILL MODE

Format: .D0

Use this control word to turn line fill mode back on again after having turned it off with the .D control word. (Note. The character following the .D is a zero, not the letter "O".) Line fill is the default mode. This control word causes a break.

ESCAPE SEQUENCE FOR PRINTER CONTROL CODE

Format: .E#

Examples: .E14

Use this control word to send a control code to your printer. The control word consists of an escape character followed by the ATASCII character represented by the decimal number, #. These codes access special features like character sets and line feeds. Table 2 (Printer Control Codes) of the ATARI 825 80-Column Printer Operator's Manual lists the codes for the ATARI 825 printer. For other printers, refer to the user manual to determine whether you can use this control word. Some of the most useful codes for the ATARI 825 printer are:

.E14	Start elongated print (twice normal width)
.E15	Stop elongated print
.E17	Selected proportional character set
.E19	Select standard character set
.E20	Select condensed character set
.E28	Advance half line (for subscripts, etc.)
.E30	Reverse half line (for superscripts, etc.)

Most of these codes remain in effect until you enter another control code of the same type (i.e., one character set control code to replace another one). In addition, these codes remain in effect as long as the printer is turned on. FORMS doesn't reset the printer each time you run the program.

LINE FEEDS BETWEEN PAGES

Format: .F#

Example: .F3

Use this control word to alter the default setting of 6 line feeds between pages. This number of lines, when combined with the default page height (.H) setting of 59 lines per page, comprises the number of lines available per 11-inch sheet: 65 lines. (FORMS uses the remaining line to print the page number.) Change the line feed setting to accommodate different paper lengths. This control word has no upper limit. Remember to adjust your page height value so that the two values total to the number of lines per page. For example, if you want to print 62 lines (.H62), then adjust line feed to three lines (.F3).

The control word .F0 issues a form-feed between pages instead of just blank lines, for printers with a form-feed capability.

RESUME READING FROM INPUT FILE

Format: .G

Use this control word at your computer keyboard for either of two occasions. At the time FORMS prompts you for the name of your file, it also displays this message at the bottom of the screen:

```
(FOLLOW NAME WITH * FOR MANUAL MODE)
```

If you enter your file name and append an asterisk, FORMS enters manual mode. See the manual mode control word (.K) for more information. You also have the option of entering manual mode at any point(s) during which FORMS is reading your file by entering the .K control word in your input file.

When you complete your manual mode activities, use control word .G to tell FORMS to resume reading your input file. The prompt MANUAL INPUT MODE (INPUT '.G' WHEN DONE) displays on your video screen when you enter manual mode to remind you of the control word to use to resume input file mode.

SET PAGE HEIGHT

Format: .H#

Example: .H57

Use this control word to change the default page height of 59 lines. You may use any value over 9. Lines are added or removed from the bottom of the page. This value, together with the line feed value (.F) comprises the number of lines available on 11-inch paper. Change this value if you want more or fewer lines per page. Remember to adjust your line feed value so that the two values add up to the number of lines per page. For example, if you want to change your line feed to 8 lines per 11-inch page (.F8), then adjust your page height setting to 57 lines (.H57).

INDENT TEXT

Format: .I#

Example: .I4

Use this control word to end the current output line and start a new line, with the left margin moved to the column position indicated by the number following .I . This value can be from 0 to the maximum line width (specified by .W#), minus 1. For example, .I4 indents subsequent lines of text four columns from the left margin. The control word .I0 cancels the previous .I# control word and positions subsequent lines of text at the left margin. The default value is .I1 . (Note. .I0 and .I1 are equivalent; they both refer to the left-hand margin.) This control word has no effect on the right margin. (To indent the right margin, use the maximum line length control word, .W#). This control word causes a break.

JUMP TO NEXT PAGE

Format: .J#

Example: .J8

Use this control word to start a new page if the number of output lines following .J can't fit on the current page. This control word is useful for assuring that a block of text remains together on a page and for leaving room for tables, figures, or illustrations. For example, to avoid breaking up a figure occupying 8 output lines, enter the control word .J8 prior to this block of text. If at least 8 lines remain on the current output page, FORMS will print the figure on that page; otherwise, FORMS leaves the remainder of the current page blank and prints the figure beginning at the top of the next page.

MANUAL INPUT MODE

Format: .K

Use this control word whenever you want FORMS to stop reading your input file temporarily and instead read control words and text you type at your keyboard at the time of printing. This control word causes a line break. Use the same procedures for entering control words and text in manual mode as you do in input file mode.

You can accomplish the same thing when FORMS starts to print your file by answering the TEXT FILE NAME? prompt with the name of your input file followed by an asterisk (e.g., SAMPLE*). FORMS will immediately go into manual mode and wait for your keyboard input.

Use the .G control word to tell FORMS to resume reading your input file.

Manual mode is useful when you want to try a variety of formatting techniques fairly quickly. You can bypass having to load your text editor and source file, revise your file, save it, and then load FORMS. Instead enter a .K in your input file wherever you want to experiment, load the FORMS program, and enter control words and/or text from the keyboard. After seeing the results, simply reload FORMS and enter other control words and/or to try out other approaches.

This mode is also useful if you want to change the layout of a block of text, depending on the purpose of the printing. For example, you might want to use one character set and indent and line length values for one purpose and another character set and indent and line length values for another purpose. This set of control words (.K and .G) let you specify such values at the point of printing.

NEW LINE

Format: .L

Use this control word to tell FORMS to end the current output line and start a new one with the text following .L . You can also cause a break at any time by placing a comma in column one of the text starting the new line. Thus, the following input lines both cause a line break:

```
1).L
  this text starts a new line

2) ,this text starts a new line
```

SKIP LINES

Format: .L#

Example: .L4

Use this control word to insert as many blank output lines as you indicate in the number following .L . The value can be 1 or greater. This control word causes a break. For example, .L4 tells FORMS to insert four blank output lines before printing the next line of text.

MOVE TO SPECIFIED COLUMN

Format: .M#

Example: .M20

Use this control word to tell FORMS to move to a particular column in the current output line and resume printing from there. If the current output line has printed beyond that value, then FORMS prints the text on the next line, starting in that column. For example, .M20 tells FORMS to stop printing, move to column 20 of the current output line and resume printing the next text (or to move to column 20 of the next output line if FORMS has already passed column 20 of the current output line). This control word does not cause a break and it applies only to the current line. (See .I for a control word that does cause a break and continues to print starting in a specified column in every line until turned off.)

Use this control word to create columns of data, tables, and other tabular text. Note. When you use a proportional character set available on the ATARI 825 printer, columns will not align with this command. Either align them manually, or use a monospaced character set for your tabular data.

PAGE NUMBER

Format: .N#

Example: .N5

Use this control word to change the current output page number to the value you specify. For example, entering .N5 changes the current output page number to 5. This number prints at the bottom of each page in the center, in the format:

-5-

To turn off page numbering, set this value to zero (.N0). You can enter these control words anywhere on the page on which you want them to take effect. The default is not to print the page number on the first output page, but to number all other pages. To cause the number to print on the first page, use .N1 .

This control word is useful for turning off page numbering, which is always in arabic form, on introductory pages like the preface, table of contents, and the verso of the title page. For these pages, enter .N0 until you want to start numbering. Then use .N1 (or whatever value you want to assign the first page) to tell FORMS to print page numbers for the body of your text.

Important Note. FORMS prompts you at the time of printing for the page number at which you want FORMS to begin printing. FORMS uses the printed page number as its guide. Therefore, to use this feature, don't suppress page numbering.

NEW PARAGRAPH

Format: .P

Use this control word to start a new paragraph. It tells FORMS to end the current output line (i.e., to break), skip one line, and indent the beginning of the next line five spaces (or the number of spaces you set with the .P# command, described below).

USER-DEFINED NEW PARAGRAPH

Format: .P#

Example: .P3

Use this control word to start a new paragraph for which you specify the number of spaces to indent. As with .P , FORMS will end the current output line, skip one line, and resume printing on the next line. For example, to indent your paragraphs three spaces, enter .P3 once. Thereafter, FORMS indents all your paragraphs three spaces whenever you use .P , until you change the value again. If you've indented your left-hand margin, FORMS will indent your paragraphs starting from this value rather than from the left-hand margin. You may use any value between 2 and the maximum line lenth (.W), minus 1. Thus, if your line length is 85 characters, you may indent the first line of each paragraph as far as 84 characters! The default is .P5 . (Note. The values .P0 and .P1 both default to the left-hand margin.)

PAUSE AT END OF EACH PAGE

Format: .Q

Use this control word to tell FORMS to pause at the end of each page so that you can change paper in your printer. When you have the paper positioned, press the RETURN key to resume printing. Use .Q0 to turn off this option. The default is .Q0 .

PRINT CONTINUOUSLY

Format: .Q0

Use this control word to turn off the paper pause control word (.Q) you entered earlier in your input file. FORMS will then resume printing continuously to the end of your input file or to the page you specified in response to the WHAT IS THE LAST PAGE YOU WANT? prompt at the time of printing. PRINT CONTINUOUSLY is the default and you needn't enter .Q0 unless you've used .Q elsewhere.

REMARK

Format: .R<text>

Example: .RThis file name is FORMS.APX

Use this control word to enter comments in your input file that you don't want printed in your formatted output. Precede each continuation line of a comment with this control word.

START NEW PAGE

Format: .S

Use this control word to tell FORMS to stop printing on the current page and print the text following this control word beginning at the top of the next page. This control word causes a break, and FORMS leaves the rest of the current page blank.

TOP TITLE

Format: .T<text>

Example: .TFORMS User Instructions

Use this control word to tell FORMS to print the string following .T centered in the top margin area of each page. See the top of this page for an example. This string can be as long as 80 characters. The top title uses current indent value (.I) and line length (.W) to center its title.

(Note. If you use any of the proportional character sets, the top title will often be off-center. In addition, the top title prints in whatever character set it is currently using via the .E control word at the time it line feeds to a new page.)

To turn off this string, enter .T somewhere on the page preceding the page on which you don't want the string to print. To change the string, enter another string in the same manner. For example, to change the top title from FORMS User Instructions to Summary of FORMS Control Words, enter .TSummary of FORMS Control Words. The default is .T (no top title).

LINE SPACING

Format: .V#

Example: .V1

Use this command to tell FORMS how many lines to skip between output lines of text. You can change this number anywhere in your input file. The default is .V0 , which is single spacing. To double space your text, use .V1 ; to triple space, use .V2 . This control word causes a break.

RIGHT-HAND MARGIN SETTING

Format: .W#

Example: .W100

Use this control word to set the right-hand margin. You can use it anywhere in your input file to change the margin as desired. This control word causes a break. FORMS will print as many words and other concatenated strings on a line as it can fit within this range. The default value is 78, which suits the default 10 cpi, monospaced character set on the ATARI 825 printer. This value can range from 1 to 200, but it must be at least one greater than your current indent (.I) setting. For example, if you set an indent value of 20 (.I20), then .W must be at least 21 (.W21). In this case, your text would print one character per line!

Note. When you change character sets, you'll need to adjust this value. For example, a value between .W90 and .W100 is about right for a full width line using the proportional character set (via the .E17 control word). Be advised that FORMS doesn't handle proportional spacing well. Some lines containing many capital letters will be very long, whereas other lines will be quite short. The effect is quite ragged.

PROGRAM WARNINGS

SOURCE ERRORS

When FORMS sends a SOURCE ERROR message, it includes the last FORMS control word it read. Some common errors are immediately identifiable in this way. For example, sometimes users unintentionally begin a line with a period that is part of their text rather than the beginning of a control word. But FORMS automatically interprets the next character(s) as a control word. Thus, if FORMS doesn't recognize the characters as valid control words, it prints a SOURCE ERROR message. Another mistake easily identified is using an uppercase O rather than a zero as the numeric value part of a control word.

However, you can't always count on the control word accompanying the SOURCE ERROR message as being the source of the problem. For example, suppose you set your right-hand margin to 60 (.W60) and later set an indent value of 65 (.I65). FORMS will send a SOURCE ERROR message when it tries to reset the indent value to 65 and discovers the value exceeds the right-hand margin. In this case, it would display the indent control word as part of the SOURCE ERROR message, but the real problem is the incompatibility of the two values. The solution is to reset either the right-hand margin or the indent value. Thus, some kinds of errors will take some analysis when the control word accompanying the SOURCE ERROR message is valid.

CENTERING

Keep in mind that FORMS uses any current indent value and the right-hand margin to center text via the .C control word. In some cases, you might want to reset the indent value to .I0 temporarily so as to center a string correctly across the whole line length.

In addition, when you imbed control codes (e.g., CTRL-O and CTRL-N to underline) in a string of text you want centered (using .C for text or .T for the centered top title), FORMS counts each code as a character in the string to be centered and therefore changes the true length of the string to be longer than it really is. To compensate for the offset, add extra space(s) between the .C control word and the text to be centered. You'll need to experiment to determine the correct number of spaces to add, since this number depends on whether the number of characters in your string is even or odd and on the nature of the control codes you're imbedding.

PROPORTIONAL SPACING

Remember that when you use a proportional character set, the right-hand margin is quite ragged. This is especially apparent in a line containing many uppercase letters, which often extends far beyond other lines. You can control the appearance of such lines by forcing FORMS to break after a specific word (see the control word .L).

ADVANCED HINTS

POSITIONING PAPER IN THE PRINTER

If you're using an ATARI 825 printer, position the paper so that the top is even with the top of the printhead. For other printers, you might have to experiment to determine the correct initial position.

After FORMS finishes printing a file, it advances the paper a few additional lines for ease in tearing off perforated paper. Remember to reposition the top of the paper to line up with the top of the printhead before using FORMS again.

EXPERIMENTING WITH FORMS WITHOUT LOADING A TEXT FILE

You might want to do some quick experimentation with control words and text before imbedding control words in your text file. You can do some fast testing without loading a text file into RAM by running FORMS as usual, but answering the TEXT FILE NAME? prompt with E: instead of with the name of a text file. Respond to the prompt asking for the starting page by pressing the RETURN key. The E: command turns control over to the keyboard. You can then key in control words and text. FORMS reads these lines as it would if you had used the control word .K to go into manual mode. FORMS echoes the lines on your video screen.

Sometimes nothing happens immediately after you enter a control word or text and press the RETURN key. The reason can be either that the line buffer FORMS fills before printing the output line is not yet full, or that the control word is such that it doesn't affect the immediate output line. Examples of the latter case are changing the page number, (.N#) or the top title (.T<text>) midway on an output page. The change will show up in the next occurrence of that action.

Use CTRL-3 to simulate an END OF FILE command to cause FORMS to print the SOURCE ERROR summary and to form-feed the paper. You then also return to the READY prompt in BASIC.

SAMPLE INPUT FILE FOR TITLE AND CONTENTS PAGES

```
.W75
.L4
.BTEXT FORMATTER (FORMS)
.L2
.Cby
.12
.cDale Yocum
.110
.cProgram and Manual Contents   1981 Atari, Inc.
.11
.e17
.w100
.i5
Copyright and right to make backup copies.
On receipt of this computer program and
associated documentation (the software),
ATARI grants you a nonexclusive
license to execute the enclosed software
and to make backup or archival copies of
the computer program for your personal
use only, and only on the condition that
all copies are conspicuously marked
with the same copyright notices that appear
on the original.
This software is copyrighted.
You are prohibited from reproducing,
translating, or distributing this
software in any unauthorized manner.
.i0
.s
.n0
.BCONTENTS
.L2
.I5
INTRODUCTION . . . 1
.L1
.i10
Overview . . . 1
,Required accessories . . . 2
.l1
.i5
A SAMPLE SESSION . . . 3
.L1
SUMMARY OF STEPS--TEXT EDITING . . .6
.L1
.I10
Using the ATARI Assembler Editor Cartridge . . . 6
,Using the ATARI Program Text Editor . . . 7
,Using another text editor . . . 8
.l1
.i5
SUMMARY OF STEPS--USING FORMS . . . 9
.l1
```

```
.i10
Diskette files . . . 9
,Cassette files . . . 10
.L1
.I5
LINE-ORIENTED AND FULL-SCREEN EDITORS . . . 11
.l1
.i5
GENERAL NOTES ON USING FORMS . . . 12
.L1
.I10
Entering FORMS control words . . . 12
,Line feeds, line fill, and breaks . . . 12
,Telling FORMS to start a new line . . . 13
,Underlining . . . 13
.l1
.i5
CONTROL WORD DESCRIPTIONS . . . 14
.L1
.I10
.E19
.D
 .B<text> Center text in boldface . . . 14
 .C<text> Center text . . . 14
 .D       Turn off line fill mode . . . 14
 .D0      Turn on line fill mode . . . 15
 .E#      Escape sequence for printer control . . . 15
 .F#      Line feeds between pages . . .16
 .G       Resume reading input file . . . 16
 .H#      Set page height . . . 16
 .I#      Indent text . . . 17
 .J#      Jump to next page . . . 17
 .K       Manual input mode . . . 17
 .L       New line . . . 18
 .L#      Skip lines . . . 18
 .M#      Move to specified column . . . 18
 .N#      Page number . . . 19
 .P       New paragraph . . . 19
 .P#      User-defined new paragraph . . . 20
 .Q       Pause at end of each page . . . 20
 .Q0      Print continuously . . . 20
 .R<text> Remark . . . 20
 .S       Start new page . . . 21
 .T<text> Top title . . . 21
 .V#      Line spacing . . . 21
 .W#      Right-hand margin setting . . . 22
.l1
.i5
.e17
PROGRAM WARNINGS . . . 23
.L1
.I10
Source errors . . . 23
,Centering . . . 23
,Proportional spacing . . . 23
.l1
.i5
ADVANCED HINTS . . . 24
.L1
.I10
Positioning paper in the printer . . . 24
,Experimenting with FORMS without loading a text file . . . 24
.l1
.i5
SAMPLE INPUT FILE FOR TITLE AND CONTENTS PAGES . . . 25
```

These are 1 space in from the left-hand margin to prevent FORMS from reading them as contol words tobe executed. See the resulting list on the CONTENTS page.

Limited Warranty on Media and Hardware Accessories. We, Atari, Inc., guarantee to you, the original retail purchaser, that the medium on which the APX program is recorded and any hardware accessories sold by APX are free from defects for thirty days from the date of purchase. Any applicable implied warranties, including warranties of merchantability and fitness for a particular purpose, are also limited to thirty days from the date of purchase. Some states don't allow limitations on a warranty's period, so this limitation might not apply to you. If you discover such a defect within the thirty-day period, call APX for a Return Authorization Number, and then return the product along with proof of purchase date to APX. We will repair or replace the product at our option.

You void this warranty if the APX product: (1) has been misused or shows signs of excessive wear; (2) has been damaged by use with non-ATARI Home Computer products; or (3) has been serviced or modified by anyone other than an Authorized ATARI Computer Service Center. Incidental and consequential damages are not covered by this warranty or by any implied warranty. Some states don't allow exclusion of incidental or consequential damages, so this exclusion might not apply to you.

Disclaimer of Warranty and Liability on Computer Programs. Most APX programs have been written by people not employed by Atari, Inc. The programs we select for APX offer something of value that we want to make available to ATARI Home Computer owners. To offer these programs to the widest number of people economically, we don't put APX products through rigorous testing. Therefore, APX products are sold "as is," and we do not guarantee them in any way. In particular, we make no warranty, express or implied, including warranties of merchantability and fitness for a particular purpose. We are not liable for any losses or damages of any kind that result from use of an APX product.

For the complete list of current
APX programs, ask your ATARI retailer
for the APX Product Catalog

APX ATARI*
PROGRAM
EXCHANGE

P.O. Box 3705
Santa Clara, CA 95055

Review Form

We're interested in your experiences with APX programs and documentation, both favorable and unfavorable. Many of our authors are eager to improve their programs if they know what you want. And, of course, we want to know about any bugs that slipped by us, so that the author can fix them. We also want to know whether our instructions are meeting your needs. You are our best source for suggesting improvements! Please help us by taking a moment to fill in this review sheet. Fold the sheet in thirds and seal it so that the address on the bottom of the back becomes the envelope front. Thank you for helping us!

1. Name and APX number of program.

____

____

2. If you have problems using the program, please describe them here.

____

____

____

3. What do you especially like about this program?

____

____

____

4. What do you think the program's weaknesses are?

____

____

____

5. How can the catalog description be more accurate or comprehensive?

____

____

6. On a scale of 1 to 10, 1 being "poor" and 10 being "excellent", please rate the following aspects of this program:

________ Easy to use
________ User-oriented (e.g., menus, prompts, clear language)
________ Enjoyable
________ Self-instructive
________ Useful (non-game programs)
________ Imaginative graphics and sound

7. Describe any technical errors you found in the user instructions (please give page numbers).

____

____

____

8. What did you especially like about the user instructions?

____

____

____

9. What revisions or additions would improve these instructions?

____

____

____

10. On a scale of 1 to 10, 1 representing "poor" and 10 representing "excellent", how would you rate the user instructions and why?

____

____

11. Other comments about the program or user instructions:

____

____

____

From

____

____

____

STAMP

ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

[seal here]